

Queens College
Summer 2020

PROBLEMS IN COMPUTATION THEORY

Bojana Obrenić

Contents

1 Counting	1
2 Regular Expressions	17
3 Context-Free Grammars	25
4 Finite Automata	104
5 Push-Down Automata	200
6 Turing Machines & General Algorithms	251
A Sample Diagonalization Proofs	371
B Sample Construction Correctness Proof	372
C Algorithm Checklist	373

1 Counting

1 Problem 1 Let sets A and B be defined as follows:

$$\begin{aligned}A &= \{a, b, c, d, e\} \\B &= \{0, 1, 2\}\end{aligned}$$

(a) How many total functions are there from set A to the set B ? Justify briefly your answer.

Answer: There are $|B|^{|A|} = 3^5 = 243$ total functions from set A to set B .

(b) Construct a total function $f_1 : A \rightarrow B$. If such a function does not exist, explain why.

Answer:

$$f_1 = \{(a, 0), (b, 0), (c, 0), (d, 0), (e, 1)\}$$

(c) Construct a total injective function $f_2 : A \rightarrow B$. If such a function does not exist, explain why.

Answer: Since $|A| = 5 > 3 = |B|$, there does not exist a total injection from set A to set B .

(d) Construct a total surjective function $f_3 : A \rightarrow B$. If such a function does not exist, explain why.

Answer:

$$f_3 = \{(a, 0), (b, 1), (c, 2), (d, 0), (e, 1)\}$$

Problem 2 Let N be the set of natural numbers and let \mathcal{F} be the set of total functions from set N to set $\{0, 1\}$.

(a) Construct a total injective function $g_1 : N \rightarrow \mathcal{F}$. If such a function does not exist, explain why.

Answer: Let the image of element $i \in N$ be the function $c_i \in \mathcal{F}$:

$$g_1(i) = c_i$$

where c_i is defined as follows:

$$c_i(x) = \begin{cases} 1 & \text{if } x = i \\ 0 & \text{if } x \neq i \end{cases}$$

(b) Construct a total injective function $g_2 : \mathcal{F} \rightarrow N$. If such a function does not exist, explain why.

Answer: Since

$$|\mathcal{F}| = |\{0, 1\}|^{|N|} = 2^{\aleph_0} > \aleph_0 = |N|$$

there does not exist a total injection from set \mathcal{F} to set N .

Problem 3 Let sets A and B be defined as follows:

$$\begin{aligned}A &= \{a, b, c, d, e\} \\B &= \{0, 1, 2\}\end{aligned}$$

and let N be the set of natural numbers.

(a) Construct a total injective function f_1 from B to $A \times B$. If such a function does not exist, explain why.

Answer: Since $|A \times B| = 5 \times 3 = 15$, there are exactly $15 \times 14 \times 13 = 2730$ different total injections from B to $A \times B$. Here is one of them:

$$f_1(x) = (d, x)$$

(b) Construct a total surjective function f_2 from A to a proper subset of N . If such a function does not exist, explain why.

Answer: Choose set $\{8\} \subset N$ to be the range of f_2 and define f_2 as follows:

$$f_2(x) = 8$$

(c) Construct a total injective function f_3 from A to $\mathcal{P}(B)$. If such a function does not exist, explain why.

Answer: Since $|\mathcal{P}(B)| = 2^3 = 8$, there are exactly $8 \times 7 \times 6 \times 5 \times 4 = 6720$ different total injections from A to $\mathcal{P}(B)$. Here is one of them:

x	$f_3(x)$
a	\emptyset
b	$\{0\}$
c	$\{1\}$
d	$\{2\}$
e	$\{0, 1\}$

(d) Construct a proper subset S_1 of $\mathcal{P}(B)$ which is infinite and countable. If such a set does not exist, explain why.

Answer: Impossible— $\mathcal{P}(B)$ is a finite set (with 8 elements) and cannot have an infinite subset.

(e) Construct a proper subset S_2 of N which is infinite and countable. If such a set does not exist, explain why.

Answer: For example:

The set of even natural numbers:

$$S_2 = \{x \mid (\exists y \in N)(x = 2y)\}$$

The set of odd natural numbers:

$$S_2 = \{x \mid (\exists y \in N)(x = 2y + 1)\}$$

The set of powers of 2:

$$S_2 = \{x \mid (\exists y \in N)(x = 2^y)\}$$

The set of perfect squares:

$$S_2 = \{x \mid (\exists y \in N)(x = y^2)\}$$

Problem 4 Let N be the set of natural numbers and let \mathcal{F} be the set of total functions from set N to set $\{0, 1\}$.

(a) Construct a total injective function:

$$g_1 : N \rightarrow \mathcal{F}$$

If such function does not exist, explain why.

(b) Construct a total surjective function:

$$g_2 : N \rightarrow \mathcal{P}(N)$$

If such function does not exist, explain why.

(c) Construct a total bijective function:

$$g_3 : \mathcal{P}(N) \rightarrow \mathcal{F}$$

If such function does not exist, explain why.

Problem 5 (a) Construct an injective function from $N \times N$ to $N \times N \times N$. Justify your answer briefly. If such a function does not exist, explain why.

Answer: There are infinitely many such functions. A simple solution is:

$$f : (x, y) \rightarrow (x, y, 0)$$

This function is injective since it is essentially the identity function in $N \times N$.

(b) Construct an injective function from $N \times N \times N$ to $N \times N$. Justify your answer briefly. If such a function does not exist, explain why.

Answer: There are infinitely many such functions. A simple solution is:

$$f : (x, y, z) \rightarrow (2^x \cdot 3^y \cdot 5^z, 0)$$

Observe that the second component of the target pair is always equal to 0. By the properties of the prime-factor decomposition of integers, two distinct triplets are mapped to two pairs that differ in their first component, whence the injectiveness.

(c) Construct a surjective function from $N \times N \times N$ to $N \times N$. Justify your answer briefly. If such a function does not exist, explain why.

Answer: There are infinitely many such functions. A simple solution is:

$$f : (x, y, z) \rightarrow (x, y)$$

This function is surjective since it is a projection of the identity function in $N \times N \times N$. Exactly \aleph_0 elements of $N \times N \times N$ are mapped into each element of $N \times N$, precisely all those triplets that agree in the first two components and differ in the third component.

Problem 6 Let $N = \{0, 1, \dots\}$ be the set of natural numbers. Construct five different injective functions from N to $N \times N$. If such functions do not exist, explain why.

Answer: There are infinitely many correct answers. For example:

$$\begin{aligned} f_1(x) &= (0, x) \\ f_2(x) &= (x, 1) \\ f_3(x) &= (17, 5x) \\ f_4(x) &= (2^x, 12) \\ f_5(x) &= (x^2, x^3) \end{aligned}$$

Problem 7 Let $N = \{0, 1, \dots\}$ be the set of natural numbers.

(a) Construct an injective function from $N \times N$ to N . Justify your answer briefly. If such a function does not exist, explain why.

Answer: There are infinitely many correct answers. A simple one:

$$f : (x, y) \mapsto 2^x \cdot 3^y$$

To see that f is injective, we have to show that

$$f(x, y) = f(v, w)$$

is impossible unless $(x, y) = (v, w)$. Observe that 2 and 3 are prime. Hence:

$$2^x \cdot 3^y = 2^v \cdot 3^w$$

is possible only if:

$$x = v \text{ and } y = w$$

which means:

$$(x, y) = (v, w)$$

whence the claim.

(b) Construct a non-injective function from $N \times N$ to N . Justify your answer briefly. If such a function does not exist, explain why.

Answer: There are infinitely many correct answers. A trivial one:

$$f : (x, y) \mapsto 0$$

Since f sends the entire set $N \times N$ to a single element of N , f is trivially non-injective.

For a less trivial example, let:

$$g : (x, y) \mapsto x$$

g is surjective, and every element of N is the image of infinitely many elements of $N \times N$. Precisely, all pairs of the form:

$$(a, y), \text{ for } y = 0, 1, 2, \dots$$

are mapped to a under f .

Problem 8 Let:

$$\Sigma = \{a, b, c, d, e\}$$

and let $N = \{0, 1, \dots\}$ be the set of non-negative integers.

(a) Describe precisely a procedure for assigning a distinct non-negative integer to every total function from N to Σ . Justify your answer briefly. If such a procedure does not exist, explain why.

Answer: Such procedure does not exist. Observe that the set of total functions from N to Σ contains the set of total functions from N to $\{a, b\}$ as a proper subset. The latter is known to be uncountable, by the Cantor's

diagonalization argument. An uncountable set cannot be mapped injectively into N .

(b) Describe precisely a procedure for assigning a distinct non-negative integer to every total function from Σ to N . Justify your answer briefly. If such a procedure does not exist, explain why.

Answer: Every total function $f : \Sigma \rightarrow N$ corresponds to a quintuple:

$$\langle n_1, n_2, n_3, n_4, n_5 \rangle$$

such that:

$$\begin{aligned} n_1 &= f(a) \\ n_2 &= f(b) \\ n_3 &= f(c) \\ n_4 &= f(d) \\ n_5 &= f(e) \end{aligned}$$

Hence, every total function from Σ to N corresponds to a point in N^5 . N^5 is known to be countable, and there are infinitely many injections from N^5 to N . A simple one relies on the uniqueness of prime factorization:

$$f(\langle n_1, n_2, n_3, n_4, n_5 \rangle) = 2^{n_1} \cdot 3^{n_2} \cdot 5^{n_3} \cdot 7^{n_4} \cdot 11^{n_5}.$$

Problem 9 Let $\Sigma = \{0, 1\}$. Let N be the set of natural numbers.

State the cardinality of each of the following sets. For a finite set, state the exact number. For an infinite set, specify if it is countable or not.

Answer:

1. Set of all subsets of Σ has 4 elements.
2. Set of all non-empty subsets of Σ has 3 elements.
3. Set of all strings that belong to Σ^* is infinite and countable.
4. Set of all non-empty strings that belong to Σ^* is infinite and countable.
5. Set of all subsets of Σ^* is infinite and uncountable.
6. Set of all non-empty subsets of Σ^* is infinite and uncountable.
7. Set $\Sigma \times \Sigma$ has 4 elements.
8. Set of all subsets of $\Sigma \times \Sigma \times \Sigma$ has $2^8 = 256$ elements.
9. Set of all (total) functions from Σ to $\Sigma \times \Sigma \times \Sigma$ has $8^2 = 64$ elements.
10. Set of all (total) functions from N to $\Sigma \times \Sigma \times \Sigma$ is infinite and uncountable.
11. Set of all (total) functions from N to Σ is infinite and uncountable.
12. Set of all regular expressions over Σ is infinite and countable.

13. Set of all context-free grammars over Σ is infinite and countable.
14. Set of all languages over Σ is infinite and uncountable.
15. Set of all finite languages over Σ is infinite and countable.
16. Set of all infinite languages over Σ is infinite and uncountable.
17. Set of all regular languages over Σ is infinite and countable.
18. Set of all context-free languages over Σ is infinite and countable.

Problem 10 Let:

$$\Sigma = \{a, b, c\}$$

and let:

$$N = \{0, 1, \dots\}$$

be the set of natural numbers. State the cardinality of each of the following sets. (For a finite set, state the exact number. For an infinite set, specify if it is countable or not.)

Answer:

1. $|\mathcal{P}(\Sigma)| = 2^3 = 8$
2. $|\Sigma^*| = \aleph_0$
(countable)
3. $|\text{set of total functions from } \Sigma \text{ to } \{0, 1\}| = 2^3 = 8$
4. $|\text{set of total functions from } \Sigma^* \text{ to } \{0, 1\}| > \aleph_0$
(uncountable)
5. $|\mathcal{P}(\Sigma^*)| > \aleph_0$
(uncountable)
6. $|\text{set of total functions from } N \text{ to } \Sigma| > \aleph_0$
(uncountable)
7. $|\Sigma \times \Sigma| = 3 \times 3 = 9$
8. $|\Sigma^* \times \Sigma^*| = \aleph_0$
(countable)
9. $|\text{set of all finite subsets of } N| = \aleph_0$
(countable)
10. $|\mathcal{P}(N)| > \aleph_0$
(uncountable)

Problem 11 Let:

$$\begin{aligned}\Sigma &= \{a_1, a_2, \dots, a_k\} \\ V &= \{A_1, A_2, \dots, A_m\}\end{aligned}$$

and let:

$$N = \{0, 1, \dots\}$$

be the set of natural numbers. State the cardinality of each of the following sets. (For a finite set, state the exact number. For an infinite set, specify if it is countable or not.)

Answer:

1. $\Sigma \times V$ has km elements.
2. V^* is infinite and countable.
3. $(\Sigma \times V)^*$ is infinite and countable.
4. $\mathcal{P}(\Sigma)$ has 2^k elements.
5. $\mathcal{P}(V^*)$ is infinite and uncountable.
6. set of total functions from V to Σ has k^m elements.
7. set of total functions from N to V is infinite and uncountable.
8. set of all context-free grammars of the form (V, Σ, P, S) is infinite and countable.
9. set of all regular expressions over Σ is infinite and countable.
10. language defined by the regular expression $(a_1 \cup a_2)(a_2 a_3)^*$ is infinite and countable.
11. $N \times \Sigma$ is infinite and countable.
12. set of all languages over Σ is infinite and uncountable.

Problem 12 Let:

$$\begin{aligned}\Sigma &= \{a, b, c, d, e\} \\ V &= \{A, B, D, E, F, H, J, K\}\end{aligned}$$

State the cardinality of each of the following sets. (For a finite set, state the exact number. For an infinite set, specify if it is countable or not.)

Answer:

1. $\Sigma \cup V$ has $5 + 8 = 13$ elements.
2. $\Sigma^* \cup V^*$ is infinite and countable.
3. $\Sigma \cap V = \emptyset$, hence it has 0 elements.
4. $(\Sigma \cap V)^* = \emptyset^* = \{\lambda\}$, hence it has 1 element.
5. $\mathcal{P}(V)$ has $2^8 = 256$ elements.
6. $\mathcal{P}(\Sigma^*)$ is infinite and uncountable.
7. set of all regular expressions over Σ is infinite and countable.
8. set of all languages over Σ is infinite and uncountable.
9. set of total functions from Σ to V has 8^5 elements.

10. set of all context-free grammars of the form (V, Σ, P, S) is infinite and countable.
11. $V \times \Sigma$ has $8 \times 5 = 40$ elements.
12. set of all languages over V is infinite and uncountable.

Problem 13

1. Which sets are the domain and the codomain of the Gödel numbering function?

Answer: The domain of the Gödel numbering is the set of all non-empty finite sequences of natural numbers:

$$\bigcup_{k=1}^{\infty} N^k$$

and its codomain is the set N of natural numbers.

2. Calculate the image of the tuple $\langle 0, 1, 0 \rangle$ under Gödel numbering and show your work. If this image does not exist, prove it.

Answer:

$$\begin{aligned}\mathcal{G}(\langle 0, 1, 0 \rangle) &= 2^{1+0} \cdot 3^{1+1} \cdot 5^{1+0} \\ &= 2 \cdot 9 \cdot 5 = 90\end{aligned}$$

3. Calculate the pre-image of the number 2100 under Gödel numbering and show your work. If this pre-image does not exist, prove it.

Answer: The prime factorization of 2100 is obtained as follows:

$$\begin{aligned}2100 &= 2 \cdot 1050 \\ &= 2 \cdot 2 \cdot 525 \\ &= 2 \cdot 2 \cdot 3 \cdot 175 \\ &= 2 \cdot 2 \cdot 3 \cdot 5 \cdot 35 \\ &= 2 \cdot 2 \cdot 3 \cdot 5 \cdot 5 \cdot 7 \\ &= 2^2 \cdot 3 \cdot 5^2 \cdot 7 \\ &= 2^{1+1} \cdot 3^{1+0} \cdot 5^{1+1} \cdot 7^{1+0}\end{aligned}$$

Hence:

$$(\mathcal{G})^{-1}(2100) = \langle 1, 0, 1, 0 \rangle$$

4. Calculate the image of the tuple $\langle 3, 0, 0, 0 \rangle$ under Gödel numbering and show your work. If this image does not exist, prove it.

Answer:

$$\begin{aligned}\mathcal{G}(\langle 3, 0, 0, 0 \rangle) &= 2^{1+3} \cdot 3^{1+0} \cdot 5^{1+0} \cdot 7^{1+0} \\ &= 16 \cdot 3 \cdot 5 \cdot 7 = 48 \cdot 5 \cdot 7 = 240 \cdot 7 \\ &= 1680\end{aligned}$$

5. Calculate the pre-image of the number 355 under Gödel numbering and show your work. If this pre-image does not exist, prove it.

Answer: This pre-image does not exist. 355 is odd—the first prime number, which is 2, is not present in its factorization. In contrast, every Gödel number is a product in which each prime in the sequence of a certain number of initial primes features as a factor with non-zero multiplicity.

Problem 14 (a) Calculate the image of the tuple $\langle 3, 1, 1, 0 \rangle$ under Gödel numbering and show your work. If this image does not exist, prove it.

Answer:

$$\begin{aligned}\mathcal{G}(\langle 3, 1, 1, 0 \rangle) &= 2^{3+1} \cdot 3^{1+1} \cdot 5^{1+1} \cdot 7^{0+1} \\ &= 16 \cdot 9 \cdot 25 \cdot 7 = 25200\end{aligned}$$

(b) Calculate the pre-image of the number 14700 under Gödel numbering and show your work. If this pre-image does not exist, prove it.

Answer: The prime factorization of 14700 is obtained as follows:

$$\begin{aligned}14700 &= 2 \cdot 7350 \\ &= 2 \cdot 2 \cdot 3675 \\ &= 2 \cdot 2 \cdot 3 \cdot 1225 \\ &= 2 \cdot 2 \cdot 3 \cdot 5 \cdot 245 \\ &= 2 \cdot 2 \cdot 3 \cdot 5 \cdot 5 \cdot 49 \\ &= 2 \cdot 2 \cdot 3 \cdot 5 \cdot 5 \cdot 7 \cdot 7 \\ &= 2^2 \cdot 3 \cdot 5^2 \cdot 7^2 \\ &= 2^{1+1} \cdot 3^{0+1} \cdot 5^{1+1} \cdot 7^{1+1}\end{aligned}$$

Hence:

$$(\mathcal{G})^{-1}(14700) = \langle 1, 0, 1, 1 \rangle$$

Problem 15 Calculate the pre-image of 6720 under Gödel numbering and show your work. If this pre-image does not exist, prove it.

Advice for Answer: $\langle 5, 0, 0, 0 \rangle$.

Problem 16 (a) Is 105 a Gödel number? Prove your answer. If your answer is “yes”, write the sequence encoded by this Gödel number.

Answer: No. Every Gödel number is of the form:

$$\prod_{i=0}^n (p^{\#}(i))^{x_i+1}$$

where $p^{\#}(i)$ is the i th prime, and all x_i are natural numbers. However:

$$105 = 3 \times 5 \times 7$$

Thus, there is no natural number x_0 such that $2^{x_0+1} = (p^{\#}(0))^{x_0+1}$ is a factor of 105, whereas $p^{\#}(1) = 3$, $p^{\#}(2) = 5$, and $p^{\#}(3) = 7$ appear in the factorization of 105.

(b) Is 24 a Gödel number? Prove your answer. If your answer is “yes”, write the sequence encoded by this Gödel number.

Answer: Yes.

$$24 = 2^3 \times 3^1 = 2^{2+1} \times 3^{0+1}$$

whence the sequence encoded by 24 is $\langle 2, 0 \rangle$.

Problem 17 (a) Is 6 a Gödel number such that 4 occurs in the sequence encoded by it? Prove your answer. If your answer is “yes”, write the entire sequence encoded by this Gödel number.

Answer: No. Although 6 is a Gödel number, 4 does not occur in the sequence encoded by it. Every Gödel number is of the form:

$$\prod_{i=0}^n (p^\#(i))^{x_i+1}$$

where $p^\#(i)$ is the i th prime, and all x_i are natural numbers. However:

$$6 = 2 \times 3 = 2^{0+1} \times 3^{0+1}$$

whence the sequence encoded by 6 is $(0, 0)$, which does not contain 4.

(b) Is 18 a Gödel number such that 1 occurs in the sequence encoded by it? Prove your answer. If your answer is “yes”, write the entire sequence encoded by this Gödel number.

Answer: Yes.

$$18 = 2 \times 3^2 = 2^{0+1} \times 3^{1+1}$$

whence the sequence encoded by 18 is $(0, 1)$.

Problem 18 **(a)** Calculate the image of the tuple $\langle 4, 1, 1 \rangle$ under Gödel numbering and show your work. If this image does not exist, explain why.

Answer:

$$\mathcal{G}(\langle 4, 1, 1 \rangle) = 2^{4+1} \cdot 3^{1+1} \cdot 5^{1+1} = 32 \cdot 9 \cdot 25 = 7200$$

(b) Calculate the pre-image of the number 13860 under Gödel numbering and show your work. If this pre-image does not exist, explain why.

Answer: Since the prime factorization of 13860 is:

$$13860 = 2^2 \cdot 3^2 \cdot 5 \cdot 7 \cdot 11$$

its pre-image under Gödel numbering is:

$$(\mathcal{G})^{-1}(13860) = \langle 1, 1, 0, 0, 0 \rangle$$

(c) Construct a finite sequence of positive integers that does not have an image under Gödel numbering, and explain your construction briefly. If such a sequence does not exist, explain why.

Answer: Impossible—Gödel numbering is a total (injective) function from the set of all finite sequences of natural numbers to natural numbers. Hence, Gödel number is defined for every such sequence.

(d) Construct an integer greater than 5 that does not have a pre-image under Gödel numbering, and explain your construction briefly. If such an integer does not exist, explain why.

Answer: For instance: 7. Recall that for every Gödel number there exists a value such that this Gödel number is a product composed exactly of all primes not greater than the given value, where each prime appears with a positive exponent. Since 7 contains the prime number 7 (with exponent of 1) but does not contain a smaller prime number (say, 2 or 3 or 5) with a positive exponent, we conclude that 7 cannot be a Gödel number.

Problem 19 **(a)** Define precisely the domain and the codomain of the Gödel numbering function.

Answer: The domain of Gödel numbering is the set of all non-empty finite sequences of natural numbers:

$$\bigcup_{k=1}^{\infty} N^k$$

The codomain of Gödel numbering is the set N of natural numbers.

(b) Calculate the image of the tuple $\langle 2, 1, 0, 0 \rangle$ under Gödel numbering and show your work. If this image does not exist, prove it.

Answer:

$$\begin{aligned} \mathcal{G}(\langle 2, 1, 0, 0 \rangle) &= 2^{2+1} \cdot 3^{1+1} \cdot 5^{0+1} \cdot 7^{0+1} \\ &= 8 \cdot 9 \cdot 5 \cdot 7 = 2520 \end{aligned}$$

(c) Calculate the pre-image of the number 3300 under Gödel numbering and show your work. If this pre-image does not exist, prove it.

Answer: The prime factorization of 3300 is obtained as follows.

$$\begin{aligned} 3300 &= 2 \cdot 1650 \\ &= 2 \cdot 2 \cdot 825 \\ &= 2 \cdot 2 \cdot 3 \cdot 275 \\ &= 2 \cdot 2 \cdot 3 \cdot 5 \cdot 55 \\ &= 2 \cdot 2 \cdot 3 \cdot 5 \cdot 11 \end{aligned}$$

The prime factorization of a Gödel number that corresponds to a length- k sequence contains non-zero powers of the first k primes. However, the factorization of 3300 contains a non-zero power of 11, but does not contain such a power of 7. We conclude that 3300 cannot be a Gödel number.

Problem 20 **(a)** Define precisely the domain and the codomain of the Gödel numbering function.

Answer: The domain of Gödel numbering is the set of all non-empty finite sequences of natural numbers:

$$\bigcup_{k=1}^{\infty} N^k$$

The codomain of Gödel numbering is the set N of natural numbers.

(b) Calculate the image of the tuple $\langle 4, 2, 1 \rangle$ under Gödel numbering and show your work. If this image does not exist, explain why.

Answer:

$$\mathcal{G}(\langle 4, 2, 1 \rangle) = 2^{4+1} \cdot 3^{2+1} \cdot 5^{1+1} = 32 \cdot 27 \cdot 25 = 21600$$

(c) Construct a finite sequence of positive integers that does not have an image under Gödel numbering, and explain your construction (briefly.) If such a sequence does not exist, explain why.

Answer: Impossible—Gödel numbering is a total (injective) function from the set of all finite sequences of natural numbers to natural numbers. Hence, Gödel number is defined for every such sequence.

(d) Calculate the pre-image of the number 6930 under Gödel numbering and show your work. If this pre-image does not exist, explain why.

Answer: Since the prime factorization of 6930 is:

$$6930 = 2 \cdot 3^2 \cdot 5 \cdot 7 \cdot 11$$

its pre-image under Gödel numbering is:

$$(\mathcal{G})^{-1}(6930) = \langle 0, 1, 0, 0, 0 \rangle$$

(e) Construct an integer greater than 25 that does not have a pre-image under Gödel numbering, and explain your construction (briefly.) If such an integer does not exist, explain why.

Answer: For instance: 26. Recall that for every Gödel number there exists a value such that this Gödel number is a product composed exactly of all primes not greater than the given value, where each prime appears with a positive exponent. Now, 26 contains the prime number 13 (with exponent of 1, since $26 = 2^1 \cdot 13^1$) but does not contain a smaller prime number (precisely, any of the numbers 3, 5, 7, 11) with a positive exponent. Hence, 26 cannot be a Gödel number.

Problem 21 (a) Calculate the image of the tuple $\langle 4, 0, 1 \rangle$ under Gödel numbering and show your work. If this image does not exist, prove it.

Answer:

$$\mathcal{G}(\langle\langle 4, 0, 1 \rangle\rangle) = 2^{4+1} \cdot 3^{0+1} \cdot 5^{1+1} = 32 \cdot 3 \cdot 25 = 2400$$

(b) Calculate the pre-image of the number 4620 under Gödel numbering and show your work. If this pre-image does not exist, prove it.

Answer: Since the prime factorization of 4620 is:

$$4620 = 2^2 \cdot 3 \cdot 5 \cdot 7 \cdot 11$$

its pre-image under Gödel numbering is:

$$(\mathcal{G})^{-1}(4620) = \langle 1, 0, 0, 0, 0 \rangle$$

Problem 22 (a) Calculate the pre-image of the number 1200 under Gödel numbering and show your work. If this pre-image does not exist, prove it.

Answer: The prime factorization of 1200 is obtained as follows:

$$\begin{aligned} 1200 &= 2 \cdot 600 \\ &= 2 \cdot 2 \cdot 300 \\ &= 2 \cdot 2 \cdot 2 \cdot 150 \\ &= 2 \cdot 2 \cdot 2 \cdot 2 \cdot 75 \\ &= 2 \cdot 2 \cdot 2 \cdot 2 \cdot 3 \cdot 25 \\ &= 2 \cdot 2 \cdot 2 \cdot 2 \cdot 3 \cdot 5 \cdot 5 \\ &= 2^4 \cdot 3 \cdot 5^2 \\ &= 2^{1+3} \cdot 3^{1+0} \cdot 5^{1+1} \end{aligned}$$

Hence:

$$(\mathcal{G})^{-1}(1200) = \langle 3, 0, 1 \rangle$$

(b) Does there exist an algorithm that solves the following problem:

INPUT: An arbitrary finite sequence of arbitrary integers.

OUTPUT: A sticker bearing an integer label, so that no two different sequences receive the same label?

Explain your answer briefly.

Answer: Yes. Let \mathbb{Z} be the set of all integers. We are looking for an injection, say f , which maps the set

$$\bigcup_{k=1}^{\infty} \mathbb{Z}^k$$

into the set N of natural numbers.

First, we construct an injection $g: \mathbb{Z} \rightarrow N$ as follows:

$$g(x) = \begin{cases} 2x & \text{if } x \geq 0 \\ -2x - 1 & \text{if } x < 0 \end{cases}$$

Injection g maps integers into natural numbers; it is straightforwardly extended to map every finite sequence of arbitrary integers into a finite sequence of natural numbers. To complete the construction, we compose g with the Gödel numbering \mathcal{G} , which maps injectively the set of all non-empty finite sequences of natural numbers:

$$\bigcup_{k=1}^{\infty} N^k$$

into the set N of natural numbers.

Finally, for any finite sequence of integers w :

$$f(w) = \mathcal{G}(g(w))$$

Problem 23 (a) Define precisely a procedure that assigns a distinct label, which is a positive integer, to every element of the language:

$$a^* b^* c^*$$

If such procedure does not exist, explain why (and ignore part (b).)

Answer: Observe that every string in the given language is of the form:

$$a^j b^k c^n$$

whence the obvious bijection between $a^* b^* c^*$ and N^3 :

$$a^j b^k c^n \mapsto \langle j, k, n \rangle$$

To map N^3 injectively into N , employ Gödel numbering:

$$\langle j, k, n \rangle \mapsto 2^{j+1} \cdot 3^{k+1} \cdot 5^{n+1}$$

The required mapping is defined as follows:

$$f(a^j b^k c^n) = 2^{j+1} \cdot 3^{k+1} \cdot 5^{n+1}$$

(b) Apply the procedure that you defined in part (a) to calculate the labels assigned to strings:

$$a, b, cc, aaab, aabbc$$

Answer:

$$\begin{aligned}f(a) &= f(a^1 b^0 c^0) = 2^{1+1} \cdot 3^{0+1} \cdot 5^{0+1} = 60 \\f(b) &= f(a^0 b^1 c^0) = 2^{0+1} \cdot 3^{1+1} \cdot 5^{0+1} = 90 \\f(cc) &= f(a^0 b^0 c^2) = 2^{0+1} \cdot 3^{0+1} \cdot 5^{2+1} = 750 \\f(aaab) &= f(a^3 b^1 c^0) = 2^{3+1} \cdot 3^{1+1} \cdot 5^{0+1} = 720 \\f(aabbc) &= f(a^2 b^2 c^1) = 2^{2+1} \cdot 3^{2+1} \cdot 5^{1+1} = 5400\end{aligned}$$

Problem 24 Let n be the Gödel number of the sequence:

$$\langle x_1, x_2, x_3, x_4 \rangle$$

Express the Gödel number of the sequence:

$$\langle x_1 + 3, x_2, x_3 + 1, x_4 \rangle$$

in terms of n .

Answer: Let m be the Gödel number of the sequence $\langle x_1 + 3, x_2, x_3 + 1, x_4 \rangle$.

By definition of Gödel numbers, we have:

$$n = 2^{x_1+1} \cdot 3^{x_2+1} \cdot 5^{x_3+1} \cdot 7^{x_4+1}$$

while:

$$\begin{aligned}m &= 2^{x_1+3+1} \cdot 3^{x_2+1} \cdot 5^{x_3+1+1} \cdot 7^{x_4+1} \\&= (2^{x_1+1} \cdot 2^3) \cdot 3^{x_2+1} \cdot (5^{x_3+1} \cdot 5^1) \cdot 7^{x_4+1} \\&= 2^3 \cdot 5^1 \cdot (2^{x_1+1} \cdot 3^{x_2+1} \cdot 5^{x_3+1} \cdot 7^{x_4+1}) \\&= 40n\end{aligned}$$

Problem 25 Let n be the Gödel number of the sequence:

$$\langle x_1, x_2, x_3, x_4 \rangle$$

Express the Gödel number of the sequence:

$$\langle x_1, x_2 + 2, x_3, x_4 + 1 \rangle$$

in terms of n .

Answer: Let $m = \mathcal{G}(\langle x_1, x_2 + 2, x_3, x_4 + 1 \rangle)$.

By definition of Gödel numbers, we have:

$$n = 2^{x_1+1} \cdot 3^{x_2+1} \cdot 5^{x_3+1} \cdot 7^{x_4+1}$$

while:

$$\begin{aligned}m &= 2^{x_1+1} \cdot 3^{x_2+1+2} \cdot 5^{x_3+1} \cdot 7^{x_4+1+1} \\&= 2^{x_1+1} \cdot (3^{x_2+1} \cdot 3^2) \cdot 5^{x_3+1} \cdot (7^{x_4+1} \cdot 7^1) \\&= 3^2 \cdot 7^1 \cdot (2^{x_1+1} \cdot 3^{x_2+1} \cdot 5^{x_3+1} \cdot 7^{x_4+1}) \\&= 63n\end{aligned}$$

Problem 26 Let s be a sequence of six natural numbers:

$$s = \langle x_1, x_2, x_3, x_4, x_5, x_6 \rangle$$

and let $k = \mathcal{G}(\langle s \rangle)$ be the image of the sequence s under Gödel numbering.

(a) Is k divisible by 38? Prove your answer.

Answer: No. Observe that $38 = 2 \cdot 19$. By definition, k is divisible (only) by the first six prime numbers: 2 through 13, since the length of the sequence s is equal to 6. However, 19 is the eighth prime, so it does not divide k .

(b) Is k divisible by 39? Prove your answer.

Answer: Yes. Observe that $39 = 3 \cdot 13$, and k is divisible by 3 (the second prime) and by 13 (the sixth prime).

Problem 27 (a) Let m be the image of the sequence

$$s = \langle x_1, x_2, x_3, x_4, x_5, x_6 \rangle$$

under Gödel numbering. Represent the pre-image of the number $104m$ as a function of the (components of) sequence s . If such a representation does not exist, prove it.

Answer:

$$\langle x_1 + 3, x_2, x_3, x_4, x_5, x_6 + 1 \rangle$$

To prove this, observe that:

$$m = 2^{x_1+1} \cdot 3^{x_2+1} \cdot 5^{x_3+1} \cdot 7^{x_4+1} \cdot 11^{x_5+1} \cdot 13^{x_6+1}$$

while

$$104 = 8 \cdot 13 = 2^3 \cdot 13$$

hence

$$104 \cdot m = 2^{x_1+3+1} \cdot 3^{x_2+1} \cdot 5^{x_3+1} \cdot 7^{x_4+1} \cdot 11^{x_5+1} \cdot 13^{x_6+1+1}$$

(b) Let m be the image of the sequence

$$s = \langle x_1, x_2, x_3, x_4 \rangle$$

under Gödel numbering. Represent the pre-image of the number $143m$ as a function of the (components of) sequence s . If such a representation does not exist, prove it.

Answer:

$$\langle x_1, x_2, x_3, x_4, 0, 0 \rangle$$

To prove this, observe that:

$$m = 2^{x_1+1} \cdot 3^{x_2+1} \cdot 5^{x_3+1} \cdot 7^{x_4+1}$$

while

$$143 = 11 \cdot 13$$

hence

$$143 \cdot m = 2^{x_1+1} \cdot 3^{x_2+1} \cdot 5^{x_3+1} \cdot 7^{x_4+1} \cdot 11^{0+1} \cdot 13^{0+1}$$

(c) Let m be the image of the sequence

$$s = \langle x_1, x_2, x_3, x_4 \rangle$$

under Gödel numbering. Represent the pre-image of the number $34m$ as a function of the (components of)

sequence s . If such a representation does not exist, prove it.

Answer: Such a representation does not exist, since $34m$ is not a Gödel number. To prove this, observe that:

$$m = 2^{x_1+1} \cdot 3^{x_2+1} \cdot 5^{x_3+1} \cdot 7^{x_4+1}$$

while

$$34 = 2 \cdot 17$$

Thus, $34m$ is divisible (only) by 17 and by all primes that divide m , which are 2 through 7. Hence, $34m$ is not divisible by 13, which is the sixth prime, but is divisible by 17, which is the seventh prime—impossible for a Gödel number.

Specific instructions for Problems 28–32:

Fill the empty box at the end of each of the numbered claims with one of the following three signs:

✓ — ?

so that the signs have the following meaning.

- ✓ means that the preceding claim is always true;
- — means that the preceding claim is always false;
- ? means that the preceding claim may be true but may be false, depending on the values of the variable(s) appearing in the claim.

Problem 28 Problem variables:

L is a language over the alphabet $\Sigma = \{a, b, c\}$.

Claims:

1. If L is infinite, it is uncountable. _____ []
2. If L is infinite, its complement is finite. _____ [?]
3. Σ^* is countable. _____ [✓]
4. L^* is empty. _____ []
5. L^* is infinite. _____ [?]
6. Set of subsets of Σ^* is countable. _____ []

Problem 29 Problem variables:

L is a language over the alphabet $\Sigma = \{a, b, c\}$.

Claims:

1. L is empty. _____ [?]
2. L is finite. _____ [?]
3. L is infinite and countable. _____ [?]

4. L is infinite and uncountable. _____ []

5. L^* is empty. _____ []

6. L^* is infinite. _____ [?]

7. Set of languages over Σ^* is countable. _____ []

8. Set of languages over Σ^* is infinite and uncountable. _____ [✓]

Problem 30 Problem variables:

$s = \langle x_1, x_2, x_3, \dots, x_k \rangle$ is a sequence of natural numbers for some $k \geq 1$;

n is a natural number.

Claims:

1. Sequence s has a Gödel number. _____ [✓]
2. Sequence s has a Gödel number, and this Gödel number is divisible by 3. _____ [?]
3. n is a Gödel number. _____ [?]
4. n need not be a Gödel number, but there exists an algorithm to determine whether n is a Gödel number. _____ [✓]
5. Sequence s need not have a Gödel number, but there exists an algorithm to determine if s has a Gödel number. _____ []
6. Gödel numbering is surjective, which is proved based on the fact that every natural number can be expressed as a product of primes. _____ []
7. Gödel numbering is not surjective, but is injective. _____ [✓]

Problem 31 Problem variables:

$s = \langle x_1, x_2, x_3, \dots, x_k \rangle$ is a sequence of natural numbers for some $k \geq 1$;

n is a natural number.

Claims:

1. Sequence s need not have a Gödel number, but there exists an algorithm to determine if s has a Gödel number. _____ []
2. Sequence s has a Gödel number, and this Gödel number is divisible by 2. _____ [✓]
3. If $k \geq 3$ then the Gödel number of s is divisible by 15. _____ [✓]
4. n is a Gödel number. _____ [?]

5. n need not be a Gödel number, and there is no algorithm to determine whether n is a Gödel number.

6. Gödel numbering is injective, which is proved based on the fact that the prime factorization is unique for any natural number greater than 1. _____

7. Gödel numbering is not injective, but is surjective.

Problem 32 Problem variables:

s is an arbitrary non-empty finite sequence of natural numbers; n is an arbitrary natural number; N is the set of all natural numbers.

Claims:

1. s has a Gödel number. _____
2. There exists a non-empty finite sequence s' of natural numbers whose Gödel number is equal to n . ?
3. Gödel numbering is a bijection between the set of all non-empty finite sequences of natural numbers and the set N . _____
4. Gödel numbering may be employed to assign a unique integer to every JAVA program. _____
5. Gödel numbering may be employed to assign a unique integer to every total function from N to N .
6. Gödel numbering may be employed to assign a unique integer to every subset of the set N .

Problem 33 (a) Give an example of a finite language that is not regular. Give a precise definition of this language and explain your answer briefly. If such a language does not exist, explain why.

Answer: Impossible, every finite language is regular.

(b) Give an example of a regular language that is not finite. Give a precise definition of this language and explain your answer briefly. If such a language does not exist, explain why.

Answer: a^* is evidently regular because it has a regular expression, and also infinite because it contains a string of any length.

(c) Give an example of an infinite uncountable language. Give a precise definition of this language and explain your answer briefly. If such a language does not exist, explain why.

Answer: Impossible—every language is countable, since it is a set of finite sequences drawn from a countable set. Gödel numbering is an injection that maps every language into N .

Problem 34 Let L be the language defined by the regular expression:

$$(a \cup b \cup c)(a \cup b \cup c)$$

Let N be the set of natural numbers.

State the cardinality of each of the following sets. (For a finite set, state the exact number. For an infinite set, specify if it is countable or not.)

(Note: If you are to receive any credit for this problem, the number of your correct answers must significantly exceed the number of the incorrect ones. A missing answer is neither correct nor incorrect.)

Answer:

1. $|L| = 9$.
2. $|L \times L| = 9^2 = 81$.
3. $|L \cup L| = |L| = 9$.
4. $|LL| = 9^2 = 81$.
5. $|L^0| = |\{\lambda\}| = 1$.
6. Set of all subsets of L^0 has 2 elements (\emptyset and $\{\lambda\}$).
7. $|L^*| = \aleph_0$ — L^* is infinite and countable.
8. Set of all subsets of L has $2^{|L|} = 2^9 = 512$ elements.
9. Set of all subsets of L^* is infinite and uncountable.
10. Set of all finite subsets of L^* is infinite and countable.
11. $|L \cup L^*| = |L^*| = \aleph_0$ — $L \cup L^*$ is infinite and countable.
12. Set of all total functions from L to $\{0, 1\}$ has $|\{0, 1\}^{|L|}| = 2^9 = 512$ elements.
13. Set of all functions from L^* to $\{0, 1\}$ is infinite and uncountable.
14. Set of all functions from N to $\{0, 1\}$ is infinite and uncountable.
15. Set of all functions from N to L is infinite and uncountable.
16. Set of all total functions from L^* to $\{0, 1\}$ is infinite and uncountable.
17. Set of all total functions from N to $\{0, 1\}$ is infinite and uncountable.
18. Set of all total functions from N to L is infinite and uncountable.

Problem 35 Let:

$$\Sigma = \{a, b, c\}$$

and let L be the language defined by the regular expression:

$$(a \cup b)(b \cup c)$$

Let N be the set of natural numbers.

State the cardinality of each of the following sets. (For a finite set, state the exact number. For an infinite set, specify if it is countable or not.)

Answer:

1. $|L| = 4$
2. $|LL| = 16$
3. $|(LL)^*| = \aleph_0$
 $(LL)^*$ is infinite and countable.
4. $|\mathcal{P}(L)| = 2^4 = 16$
5. $|\mathcal{P}(L^*)| > \aleph_0$
 $\mathcal{P}(L^*)$ is infinite and uncountable.
6. $|\Sigma| = 3$
7. $|\Sigma^*| = \aleph_0$
 Σ^* is infinite and countable.
8. $|\Sigma \cap L| = 0$
9. $|(\Sigma \cap L)^*| = 1$.
10. set of regular expressions over Σ is infinite and countable.
11. set of context-free grammars over Σ is infinite and countable.
12. set of languages over Σ that do not have a context-free grammar is infinite and uncountable.

Problem 36 Let:

$$\Sigma = \{a, b, c, d\}$$

and let L be the language defined by the regular expression:

$$(a \cup b)abc(d \cup dd) \cup (a \cup b \cup c)$$

State the cardinality of each of the following sets. (For a finite set, state the exact number. For an infinite set, state that it is infinite and specify whether it is countable or not.)

Answer:

1. Set of all languages over Σ is infinite and uncountable.
2. Set of all strings over Σ is infinite and countable.
3. Set of all finite languages over Σ is infinite and countable.

4. Set of all context-free languages over Σ is infinite and countable.

5. Set of all infinite languages over Σ is infinite and uncountable.

$$6. |\Sigma \times \Sigma| = 16 = 4^2$$

$$7. |\mathcal{P}(\Sigma)| = 16 = 2^4$$

$$8. |L| = 7 = 2 \cdot 2 + 3$$

$$9. |L \cap \Sigma| = 3$$

$$10. |\mathcal{P}(L)| = 128 = 2^7$$

11. $L^* \cup \Sigma^*$ is infinite and countable.

12. \bar{L} (complement of L in Σ^*) is infinite and countable.

$$13. |\emptyset^*| = 1$$

Problem 37 Let:

$$\Sigma = \{0, 1\}$$

and let L be the language defined by the regular expression:

$$(0 \cup 1)01$$

Let N be the set of natural numbers.

State the cardinality of each of the following sets. (For a finite set, state the exact number. For an infinite set, specify if it is countable or not.)

Answer:

1. $|L| = 2$, since $L = \{001, 101\}$.
2. $|L^*| = \aleph_0$, i.e., L^* is infinite and countable.
3. $|L^0| = 1$, since $L^0 = \{\lambda\}$.
4. set of all strings of length 5 over Σ has 2^5 elements.
5. set of all strings of finite length over Σ is infinite and countable.
6. set of subsets of Σ has $2^2 = 4$ elements.
7. set of subsets of Σ^* is infinite and uncountable.
8. set of all languages over Σ is infinite and uncountable.
9. set of all functions from N to Σ is infinite and uncountable.
10. set of regular expressions over Σ is infinite and countable.

Problem 38 Let:

$$\Sigma = \{0, 1\}$$

and let L be the language defined by the regular expression:

$$(0 \cup 1)(0 \cup 1)$$

Let N be the set of natural numbers.

State the cardinality of each of the following sets. (For a finite set, state the exact number. For an infinite set, specify if it is countable or not.)

Answer:

1. $|L| = 4$
2. $|L^*| = \aleph_0$ — L^* is infinite and countable.
3. $|(LL)^*| = \aleph_0$ — $(LL)^*$ is infinite and countable.
4. $|\mathcal{P}(L)| = 2^4 = 16$
5. $|\mathcal{P}(L^*)| > \aleph_0$ — $\mathcal{P}(L^*)$ is infinite and uncountable.
6. $|\mathcal{P}(\Sigma)| = 2^2 = 4$
7. $|\mathcal{P}(\Sigma^*)| > \aleph_0$ — $\mathcal{P}(\Sigma^*)$ is infinite and uncountable.
8. $|\Sigma \cup L| = 6$
9. set of all functions from $N = \{0, 1, \dots\}$ to Σ is infinite and uncountable.
10. set of regular expressions over Σ is infinite and countable.
11. set of languages over Σ that do not have a regular expression is infinite and uncountable.
12. $|(S \cup L)^*| = \aleph_0$ — $(S \cup L)^*$ is infinite and countable.

Problem 39 Let L be the language defined by the regular expression:

$$a(a \cup b \cup c)c$$

Let N be the set of natural numbers.

State the cardinality of each of the following sets. (For a finite set, state the exact number. For an infinite set, specify if it is countable or not.)

Answer:

1. $|L| = 3$.
2. $|L \times L| = 3^2 = 9$.
3. $|LL| = 9$.
4. $|L^*| = \aleph_0$ — L^* is infinite and countable.
5. Set of all subsets of L has $2^{|L|} = 2^3 = 8$ elements.
6. Set of all subsets of L^* is infinite and uncountable.

7. Set of all subsets of $L \times L$ has $2^{|L \times L|} = 2^9 = 512$ elements.
8. $|L \cup LL| = 12$.
9. $|L \cup L^*| = \aleph_0$ — $L \cup L^*$ is infinite and countable.
10. $(LL)^* = \aleph_0$ — $(LL)^*$ is infinite and countable.
11. $(L \cup LL)^* = \aleph_0$ — $(L \cup LL)^*$ is infinite and countable.
12. Set of all functions from L to $\{0, 1\}$ has $|\{0, 1\}|^{|L|} = 2^3 = 8$ elements.
13. Set of all functions from LL to $\{0, 1\}$ has $|\{0, 1\}|^{|LL|} = 2^9 = 512$ elements.
14. Set of all functions from L^* to $\{0, 1\}$ is infinite and uncountable.
15. Set of all functions from N to $\{0, 1\}$ is infinite and uncountable.
16. Set of all functions from N to L is infinite and uncountable.
17. set of all functions from L to L has $|L|^{|L|} = 3^3 = 27$ elements.

Problem 40 Let: $\Sigma = \{a, b, c, d\}$

and let L be the language defined by the regular expression:

$$(a \cup b \cup c)d(c \cup d)$$

Let N be the set of natural numbers.

State the cardinality of each of the following sets. (For a finite set, state the exact number. For an infinite set, specify if it is countable or not.)

Answer:

1. L has 6 elements.
2. \bar{L} (the complement of L in Σ^*) is infinite and countable.
3. $\mathcal{P}(L)$ (the set of subsets of L) has 64 elements.
4. L^* is infinite and countable.
5. $(\bar{L})^*$ is infinite and countable.
6. $\mathcal{P}(L^*)$ is infinite and uncountable.
7. Σ^* is infinite and countable.
8. $\mathcal{P}(\Sigma^*)$ is infinite and uncountable.
9. set of finite languages over Σ is infinite and countable.
10. set of infinite languages over Σ is infinite and uncountable.
11. set of finite subsets of Σ has 16 elements.
12. set of infinite subsets of Σ has 0 elements.

Problem 41 Let: $\Sigma = \{a, b, c, d\}$ and let L be the language defined by the regular expression:

$$(a \cup b \cup c) d \cup d(c \cup d)$$

Let N be the set of natural numbers.

State the cardinality of each of the following sets. (For a finite set, state the exact number. For an infinite set, specify if it is countable or not.)

Answer:

1. L has 5 elements.
2. \bar{L} (the complement of L in Σ^*) is infinite and countable.
3. $\mathcal{P}(L)$ (the set of subsets of L) has 32 elements.
4. L^* is infinite and countable.
5. $(\bar{L})^*$ is infinite and countable.
6. $\mathcal{P}(L^*)$ is infinite and uncountable.
7. Σ^* is infinite and countable.
8. $\mathcal{P}(\Sigma^*)$ is infinite and uncountable.
9. set of finite languages over Σ is infinite and countable.
10. set of infinite languages over Σ is infinite and uncountable.
11. set of finite subsets of Σ has 16 elements.
12. set of infinite subsets of Σ has 0 elements.

Problem 42 Let:

$$\Sigma = \{a, b, c, d, e\}$$

and let L be the language defined by the regular expression:

$$(a \cup b \cup c \cup d)(c \cup e)d dd$$

Let N be the set of natural numbers.

State the cardinality of each of the following sets. (For a finite set, state the exact number. For an infinite set, specify if it is countable or not.)

Answer:

1. L has 8 elements.
2. LL has 64 elements.
3. $L \cup LL$ has 72 elements.
4. $L \cap LL$ has 0 elements.
5. $(L \cup LL)^*$ is infinite and countable.
6. $(L \cap LL)^*$ has 1 element.
7. $\mathcal{P}(L)$ (set of subsets of L) has 256 elements.

8. set of regular languages over Σ is infinite and countable.
9. set of context-free languages over Σ is infinite and countable.
10. set of those languages over Σ that are not regular is infinite and uncountable.
11. set of those languages over Σ that are not context-free is infinite and uncountable.
12. set of all languages over Σ is infinite and uncountable.

Problem 43 Let:

$$\Sigma = \{a, b, c, d, e\}$$

and let L be the language defined by the regular expression:

$$(a \cup b \cup c)(d \cup e) \cup (aa \cup bb \cup cc) dd$$

State the cardinality of each of the following sets. (For a finite set, state the exact number. For an infinite set, specify if it is countable or not.)

Answer:

1. Σ has 5 elements.
2. $\mathcal{P}(\Sigma)$ (set of subsets of Σ) has 32 elements.
3. set of all strings over of Σ is infinite and countable.
4. set of all languages over of Σ is infinite and uncountable.
5. set of all finite languages over of Σ is infinite and countable.
6. set of all regular languages over of Σ is infinite and countable.
7. set of all non-regular languages over of Σ is infinite and uncountable.
8. L has 9 elements.
9. $L \cup \Sigma$ has 14 elements.
10. $L \cap \Sigma$ has 0 elements.
11. $\mathcal{P}(L)$ has 512 elements.
12. Σ^* is infinite and countable.
13. L^* is infinite and countable.
14. \emptyset^* has 1 element.
15. \bar{L} (complement of L in Σ^*) is infinite and countable.
16. $\overline{\Sigma^*}$ has 0 elements.
17. $\mathcal{P}(L^*)$ is infinite and uncountable.

Problem 44 Let: $\Sigma = \{a, b, c, d\}$ and let L be the language defined by the regular expression:

$$(a \cup b \cup \lambda) c (d \cup dd \cup \lambda)$$

State the cardinality of each of the following sets. (For a finite set, state the exact number. For an infinite set, state that it is infinite and specify whether it is countable or not.)

Answer:

1. L has 9 elements.
2. L^* is infinite and countable.
3. $L \cup \Sigma$ has 12 elements.
4. $L \cap \Sigma$ has 1 element.
5. set of all subsets of L has 512 elements.
6. set of all finite subsets of L has 512 elements.
7. set of all infinite subsets of L has zero elements.
8. set of all subsets of L^* is infinite and uncountable.
9. set of all infinite subsets of L^* is infinite and uncountable.
10. LL has 81 elements.
11. $L \cup LL$ has 90 elements.
12. $L \cap LL$ has zero elements.
13. $LL \setminus L$ has 81 elements.
14. $(L \cup LL)^*$ is infinite and countable.
15. $(L \cap LL)^*$ has 1 element.
16. \overline{L} (complement of L in Σ^*) is infinite and countable.
17. $\overline{L^*}$ (complement of L^* in Σ^*) is infinite and countable.

Problem 45 Let: $\Sigma = \{a, b, c, d, e\}$ and let L be the language defined by the regular expression:

$$(a \cup b \cup c) (b \cup c \cup d) (\lambda \cup a \cup e)$$

State the cardinality of each of the following sets. (For a finite set, state the exact number. For an infinite set, state that it is infinite and specify whether it is countable or not.)

(Note: If you are to receive any credit for this problem, the number of your correct answers must exceed the number of incorrect ones. (In other words, every incorrect answer cancels out the credit earned by one correct answer; a missing answer is neither correct nor incorrect.))

Answer:

1. set of all subsets of Σ that have exactly 2 elements has 10 elements.
2. set of all strings over Σ with length equal to 2 has 25 elements.
3. set of all languages over Σ that contain exactly 2 words is infinite and countable.
4. set of all strings in L whose length is equal to 2 has 9 elements.
5. set of all languages over Σ that are not context-free is infinite and uncountable.
6. set of all context-free languages over Σ is infinite and countable.
7. set of all infinite subsets of Σ^* is infinite and uncountable.
8. set of all strings over Σ is infinite and countable.
9. L has 27 elements.
10. L^* is infinite and countable.
11. $\mathcal{P}(L)$ (set of subsets of L) has $2^{27} = 134,217,728$ elements.
12. $\mathcal{P}(L^*)$ (set of subsets of L^*) is infinite and uncountable.
13. \overline{L} (complement of L in Σ^*) is infinite and countable.
14. set whose regular expression is $\lambda \cup \emptyset$ has zero elements.
15. set whose regular expression is $\lambda^* \cup \emptyset^*$ has one element.

Problem 46 (a) Let L_1 be the set of all strings over alphabet $\{a, b, c\}$ in which all the a 's precede the b 's, which in turn precede the c 's. (It is possible that any or all of the three letters is missing, but those letters that do appear must appear in the given order.)

Write a regular expression that defines L_1 . If such a regular expression does not exist, explain why.

Answer:

$$a^* b^* c^*$$

(b) Let L_2 be the same set as L_1 , but without the empty string:

$$L_2 = L_1 \setminus \{\lambda\}$$

Write a regular expression that defines L_2 . If such a regular expression does not exist, explain why.

Answer:

$$aa^* b^* c^* \cup a^* bb^* c^* \cup a^* b^* cc^*$$

Problem 47 (a) Give an example of two countably infinite sets, S_1 and S_2 , such that $S_1 \cap S_2$ is finite. (Define S_1 and S_2 precisely.) If such sets do not exist, explain why.

Answer: There are infinitely many correct answers. A trivial one is to employ two disjoint countably infinite sets, say the set of positive natural numbers and the set of negative natural numbers. Their intersection is empty and thereby finite.

For a less trivial example, where S_1 and S_2 have a non-empty intersection, let:

$$\begin{array}{lll} S_1 : a^*b & S_2 : ab^* & S_1 \cap S_2 : ab \\ |S_1| = \aleph_0 & |S_2| = \aleph_0 & |S_1 \cap S_2| = 1 \end{array}$$

(b) Give an example of two countably infinite sets, S_1 and S_2 , such that $S_1 \setminus S_2$ is infinite. (Define S_1 and S_2 precisely.) If such sets do not exist, explain why.

Answer: There are infinitely many correct answers. A trivial one is to employ two disjoint countably infinite sets, say the set of positive natural numbers and the set of negative natural numbers. Then: $S_1 \setminus S_2 = S_1$, which is infinite.

For a less trivial example, where S_1 and S_2 have an infinite intersection, let:

$$\begin{array}{lll} S_1 : (a \cup b)^* & S_2 : a^* & S_1 \setminus S_2 : (a \cup b)^*b(a \cup b)^* \\ |S_1| = \aleph_0 & |S_2| = \aleph_0 & |S_1 \setminus S_2| = \aleph_0 \end{array}$$

Problem 48 Let:

$$\Sigma = \{a, b, c\}$$

and let L be the language defined by the regular expression:

$$11(0 \cup 1)^* 11$$

Let N be the set of natural numbers.

State the cardinality of each of the following sets. (For a finite set, state the exact number. For an infinite set, specify if it is countable or not.)

Answer:

1. $|L| = \aleph_0$ — L is infinite and countable.
2. $|\Sigma^*| = \aleph_0$ — Σ^* is infinite and countable.
3. $|\Sigma^0| = |\{\lambda\}| = 1$
4. $|L^*| = \aleph_0$ — L^* is infinite and countable.
5. $|\mathcal{P}(L)| > \aleph_0$ — $\mathcal{P}(L)$ (set of subsets of L) is infinite and uncountable.
6. $|\mathcal{P}(\Sigma)| = 2^3 = 8$
7. $|\Sigma \cup L| = \aleph_0$ — $\Sigma \cup L$ is infinite and countable.
8. $|\Sigma^* \cap L| = |\emptyset| = 0$
9. $|\mathcal{P}(\Sigma^*)| > \aleph_0$ —Set of subsets of Σ^* is infinite and uncountable.

10. Set of all languages over Σ is infinite and uncountable.
11. Set of all functions from N to Σ is infinite and uncountable.
12. Set of all context-free grammars over Σ is infinite and countable.

Problem 49 Let L be the set of strings over alphabet $\{a, b\}$ defined by the regular expression:

$$a^*b^*$$

Let $\{N = 0, 1, \dots\}$ be the set of natural numbers.

- (a) Construct an injective function from N to L . Justify your answer briefly. If such function does not exist, explain why.

Answer: There are infinitely many correct answers. A simple one:

$$g : m \mapsto a^m$$

To see that g is injective, observe that:

$$g(m) = g(n)$$

means:

$$a^m = a^n$$

which is possible only if $m = n$.

- (b) Construct a surjective function from L to N . Justify your answer briefly. If such function does not exist, explain why.

Answer: There are infinitely many correct answers. A simple one maps a string into its length:

$$h : a^m b^n \mapsto m + n$$

To see that h is surjective, observe that for any $x \in N$, there are exactly $x + 1$ distinct strings (which?) in L of length x that are mapped into x .

- (c) Construct an injective function from L to N . Justify your answer briefly. If such function does not exist, explain why.

Answer: There are infinitely many correct answers. A simple one:

$$f : a^m b^n \mapsto 2^m \cdot 3^n$$

To see that f is injective, we have to show that

$$f(m, n) = f(k, j)$$

is impossible unless $(m, n) = (k, j)$. Observe that 2 and 3 are prime. Hence:

$$2^m \cdot 3^n = 2^k \cdot 3^j$$

is possible only if:

$$m = k \text{ and } n = j$$

which means:

$$a^m b^n = a^k b^j$$

whence the claim.

Problem 50 Let $N = \{0, 1, \dots\}$ be the set of natural numbers and let L be the language defined by the regular expression:

$$(a \cup \lambda) (a \cup \lambda) b^*$$

Construct a bijection from L to N . If such bijection does not exist, prove it.

Answer: Observe that L is a union of 3 sets:

$$b^* \cup a b^* \cup aa b^*$$

which means that every string in L is of the form:

$$a^r b^q, q \geq 0, r \in \{0, 1, 2\}$$

Set $\{0, 1, 2\}$ is the set of all possible remainders in the division by 3. We map $a^r b^q$ to that number that gives remainder r and quotient q when divided by 3:

$$f : a^r b^q \rightarrow 3q + r$$

To see that f is surjective, note that every natural number has a quotient and a remainder in the division by 3, and thereby has an original in L under f . To see that f is injective, observe that the representation by the pair (quotient, remainder) is unique. Two distinct strings from L differ in the number of a 's or in the number of b 's. Hence, their images differ in their quotients or in their remainders in the division by 3, which suffices to guarantee that they are different.

Problem 51 Let L be the language defined by the regular expression:

$$((0 \cup 1)(0 \cup 1))^*$$

and let $N = \{0, 1, \dots\}$ be the set of non-negative integers.

(a) Construct an injective function $f : N \rightarrow L$, such that f is not surjective. Justify your answer briefly. If such a function does not exist, explain why.

Answer: Observe that L is the set of binary strings with even length. There are infinitely many injections from N to L . A simple one assigns to each number n a string of zeros of length $2n$:

$$f(n) = 0^{2n}$$

Evidently, two different numbers are mapped to two strings of different lengths, whence the injectiveness. To see that f is not surjective, observe that those strings in L that contain letter 1 do not have a pre-image under f .

(b) Write the strings that the function f which you constructed in part (a) assigns to numbers 0, 2, 3, 7.

Answer:

$$\begin{aligned} f(0) &= \lambda \\ f(2) &= 0000 \\ f(3) &= 000000 \\ f(7) &= 0000000000000000 \end{aligned}$$

(c) Construct a surjective function $g : L \rightarrow N$. Justify your answer briefly. If such a function does not exist, explain why.

Answer: There are infinitely many surjections from L to N . A simple one maps every string to the number represented in binary by that string:

$$g(\alpha_{n-1}\alpha_{n-2}\dots\alpha_1\alpha_0) = \sum_{k=0}^{n-1} \alpha_k \cdot 2^k$$

for $\alpha_k \in \{0, 1\}$.

To see that g is surjective, recall that every number x has a binary representation, say w_x , where $w_x \in \{0, 1\}^*$. If $|w_x|$ is even, then $w_x \in L$, and w_x is a pre-image of x under g . If $|w_x|$ is odd, then $0w_x \in L$, and $0w_x$ is a pre-image of x under g . In fact, every natural number has infinitely many pre-images under g . To see this, note that if string w is a binary representation of n , then so are strings of the form $(00)^k w$, for any $k \geq 0$, and all these strings belong to L if $w \in L$.

(d) Write the numbers that the function g which you constructed in part (c) assigns to strings $\lambda, 01, 10, 0110$.

Answer:

$$\begin{aligned} g(\lambda) &= 0 \\ g(01) &= 1 \cdot 2^0 + 0 \cdot 2^1 = 1 \\ g(10) &= 0 \cdot 2^0 + 1 \cdot 2^1 = 2 \\ g(0110) &= 0 \cdot 2^0 + 1 \cdot 2^1 + 1 \cdot 2^2 + 0 \cdot 2^3 = 6 \end{aligned}$$

Problem 52 Let L_1 be the language defined by the regular expression:

$$(ab)^* \cup c^*$$

and let L_2 be the language defined by the regular expression:

$$c^*$$

Construct a bijection from L_1 to L_2 . If such bijection does not exist, prove it.

Answer: Both L_1 and L_2 are infinite and countable, and there are infinitely many bijections between them. A simple one is a function $f : L_1 \rightarrow L_2$, defined as follows:

$$\begin{aligned} f(\lambda) &= \lambda \\ f((ab)^n) &= c^{2n} \text{ for } n \geq 1 \\ f(c^n) &= c^{2n-1} \text{ for } n \geq 1 \end{aligned}$$

Let $L' = (ab)^*$ and $L'' = c^*$. All strings in L' are of the form $(ab)^n$, while all strings in L'' are of the form c^n , $n \geq 0$. This means that within each of the two languages, any string is identified by its length only.

To see that f is injective, observe that every non-empty string from L' is mapped to an even-length string of c 's, whose length is equal to the number of ab -pairs in the original string. Next, every non-empty string from L'' is mapped to an odd-length string of c 's, whose length is one less than twice the length of the original string. Distinct strings from L' do not collide among themselves

because their images are distinguished by length; distinct strings from L'' do not collide for the same reason. A string from L' cannot collide with a string from L'' because of different parities of the lengths of their images. The only element of $L' \cap L''$ is λ , which is correctly mapped to otherwise un-assigned $\lambda = c^0 \in L_2$.

All strings in L_2 are of the form c^n , $n \geq 0$. To see that f is surjective, observe that the preimage of any $c^n \in L_2$ is computed as follows:

$$f^{-1}(c^n) = \begin{cases} (ab)^{n/2} & \text{if } n \text{ is even} \\ c^{(n+1)/2} & \text{if } n \text{ is odd} \end{cases}$$

2 Regular Expressions

Problem 53 Let L be the language defined by the regular expression:

$$(a \cup b)(a \cup b)(a \cup b)^*$$

(a) Write 8 distinct strings that belong to L . If such strings do not exist, explain why.

Answer:

$aa, ab, ba, bb, aab, abb, baa, bbb$

(b) Write 8 distinct strings over alphabet $\{a, b\}$ that do not belong to L . If such strings do not exist, explain why.

Answer: By inspection of the regular expression, we conclude that L consists of all strings over $\{a, b\}$ that have length greater than or equal to 2. Hence, there are only three strings over $\{a, b\}$, precisely those with length less than 2:

λ, a, b

that are not in L . Thus, it is impossible to list more than 3 of them.

Problem 54 Let L be the language defined by the regular expression:

$$(c \cup ab \cup b)^*bb(c \cup ab)$$

(a) Write 10 distinct strings that belong to L . If such strings do not exist, explain why.

(b) Write 10 distinct strings over alphabet $\{a, b, c\}$ that do not belong to L . If such strings do not exist, explain why.

Answer:

$\in L$	$\notin L$
bbc	λ
$bbab$	bb
$cbbc$	bba
$abbac$	$bbca$
$bbbc$	bbb
$ccabbbbc$	a
$bbcbab$	b
$bbabbbab$	ab
$cccbcbbab$	$aabb$
$cabcbbbab$	$abbab$

Problem 55 Let L be the language defined by the regular expression:

$$(a \cup b \cup c)((ab \cup ac \cup b)^*(a \cup b) \cup (aa)^*)$$

(a) Write 10 distinct strings that belong to L . If such strings do not exist, explain why.

(b) Write 10 distinct strings over alphabet $\{a, b, c\}$ that do not belong to L . If such strings do not exist, explain why.

Answer:

$\in L$	$\notin L$
a	λ
aa	cc
aaa	ccc
$aaaaaa$	$cccc$
abb	bc
$babacbab$	ac
b	$caaa$
$baaaaaaa$	$baaa$
c	$aaaa$
$caaaaa$	$aaaaaa$

Problem 56 Let L be the language defined by the regular expression

$$c^*(b \cup (ac^*)^*)$$

(a) Write 10 distinct strings that belong to L .

(b) Write 10 distinct strings over alphabet $\{a, b, c\}$ that do not belong to L .

Answer: Note that L is the set of strings over $\{a, b\}$ that do not contain bc as a substring.

$\in L$	$\notin L$
c	bc
cb	bcc
cac	$bcabc$
$cbac$	$cacac$
b	bbc
ac	abc
$aaaaccc$	$aaaabcc$
$bbbacac$	$bbbabcac$
$cccbcac$	$cccbcac$
$ccbaccbb$	$ccbccccbb$

Problem 57 Let L be the language defined by the regular expression

$$(b \cup ab \cup aab)^*(\lambda \cup a \cup aa)$$

(a) Write 10 distinct strings that belong to L .

(b) Write 10 distinct strings over alphabet $\{a, b\}$ that do not belong to L .

Answer: Note that L is the set of strings over $\{a, b\}$ that do not contain aaa as a substring.

$\in L$	$\notin L$
b	aaa
ba	aaaaa
baa	baaa
ab	aaab
aba	abaaa
abaa	aabaaa
aab	aaaab
aaba	aaaaba
aabaa	aabaaaa
bbbbbbabaabaa	bbbbbbbabaaabaa

Problem 58 Let L_1 be the language defined by the regular expression:

$$(aa)^*(bb)^*$$

Let L_2 be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSb \mid \lambda$$

1. List 5 distinct strings that belong to L_1 and L_2 (belong to $L_1 \cap L_2$). If this is impossible, state it and explain why.

Answer:

λ
aabb
aaaabbbb
aaaaaaabbbbb
aaaaaaaaabbbbbbb

2. List 5 distinct strings that belong to L_1 and do not belong to L_2 (belong to $L_1 \cap \overline{L_2}$). If this is impossible, state it and explain why.

Answer:

aa, aaaa, bb, aabb, aaaabb

3. List 5 distinct strings that belong to L_2 and do not belong to L_1 (belong to $\overline{L_1} \cap L_2$). If this is impossible, state it and explain why.

Answer:

ab
aaabbb
aaaaabbbbb
aaaaaaaabbbbbbb
aaaaaaaaabbbbbbb

4. List 5 distinct strings over the alphabet $\{a, b\}$ that do not belong to L_1 and do not belong to L_2 (belong to $\overline{L_1} \cap \overline{L_2}$). If this is impossible, state it and explain why.

Answer:

ba, baa, abb, aab, bbb

5. List 5 distinct strings that belong to L_1^* and do not belong to L_1 (belong to $L_1^* \cap \overline{L_1}$). If this is impossible, state it and explain why.

Answer:

aabbaa, aabbaabb, bbaa, bbaaaa, bbbbbaa

6. List 5 distinct strings that belong to L_1 and do not belong to L_1^* (belong to $L_1 \cap \overline{L_1^*}$). If this is impossible, state it and explain why.

Answer: Impossible---every language is a subset of its Kleene star. In particular, $L_1 \subset L_1^*$. Hence, $L_1 \cap \overline{L_1^*} = \emptyset$, and there are no strings to list.

Problem 59 (a) Let L_1 be the set of exactly those strings over the alphabet $\{a, b, c\}$ that begin with c .

Write a regular expression that represents the language L_1 . If such a regular expression does not exist, prove it.

Answer:

$$c(a \cup b \cup c)^*$$

(b) Let L_2 be the set of exactly those strings over the alphabet $\{a, b, c\}$ that do not begin with c .

Write a regular expression that represents the language L_2 . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b)(a \cup b \cup c)^* \cup \lambda$$

(c) Let L_3 be the set of exactly those strings over the alphabet $\{a, b, c\}$ that satisfy all of the following properties: the first symbol is b , the last symbol is b , and symbol c occurs exactly once.

Write a regular expression that represents the language L_3 . If such a regular expression does not exist, prove it.

Answer:

$$b(a \cup b)^* c(a \cup b)^* b$$

(d) Let L_4 be the set of exactly those strings over the alphabet $\{a, b, c\}$ that satisfy all of the following properties: the first symbol is different from b , the last symbol is different from b , and symbol b occurs exactly twice.

Write a regular expression that represents the language L_4 . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup c)(a \cup c)^* b(a \cup c)^* b(a \cup c)^* (a \cup c)$$

(e) Let L_5 be the set of exactly those strings over the alphabet $\{a, b, c\}$ where the symbol a occurs at least twice but at most four times.

Write a regular expression that represents the language L_5 . If such a regular expression does not exist, prove it.

Answer: Let $\beta = (b \cup c)^*$. Then the answer is:

$$\beta a \beta a \beta (a \cup \lambda) \beta (a \cup \lambda) \beta$$

Problem 60 Let L be the set of all strings over the alphabet $\{a, b, c\}$ that contain a substring ac .

Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)^* ac (a \cup b \cup c)^*$$

Problem 61 Let L be the set of all strings over the alphabet $\{a, b, c\}$ that contain at least one of the substrings: ab, ba .

Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)^* (ab \cup ba) (a \cup b \cup c)^*$$

Problem 62 Let L be the set of all strings over the alphabet $\{a, b, c\}$ that contain exactly one b .

Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup c)^* b (a \cup c)^*$$

Problem 63 (a) Write a regular expression that represents the set of strings over $\{a, b, c\}$ in which the total number of b 's is not greater than 1. If such a regular expression does not exist, prove it.

Answer:

$$(a \cup c)^* (b \cup \lambda) (a \cup c)^*$$

(b) Write a regular expression that represents the set of strings over $\{a, b, c\}$ in which the total number of b 's is greater than 1. If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)^* b (a \cup b \cup c)^* b (a \cup b \cup c)^*$$

Problem 64 Let L be the set of all strings over the alphabet $\{a, b\}$ that contain at least two a 's.

Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b)^* a (a \cup b)^* a (a \cup b)^*$$

Problem 65 (a) Let L_1 be the set of all strings over alphabet $\{0, 1\}$ that contain exactly two zeros.

Write a regular expression that defines L_1 . If such a regular expression does not exist, explain why.

Answer:

$$1^* 01^* 01^*$$

(b) Let L_2 be the set of all strings over alphabet $\{0, 1\}$ that contain at least two zeros.

Write a regular expression that defines L_2 . If such a regular expression does not exist, explain why.

Answer:

$$(0 \cup 1)^* 0 (0 \cup 1)^* 0 (0 \cup 1)^*$$

(c) Let L_3 be the set of all strings over alphabet $\{0, 1\}$ that contain at most two zeros.

Write a regular expression that defines L_3 . If such a regular expression does not exist, explain why.

Answer:

$$1^* (0 \cup \lambda) 1^* (0 \cup \lambda) 1^*$$

Problem 66 Let L be the set of all strings over the alphabet $\{a, b, c\}$ that contain at most one a .

Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(b \cup c)^* (a \cup \lambda) (b \cup c)^*$$

Problem 67 (a) Let L_1 be the set of all strings over alphabet $\{0, 1\}$ that do not contain the substring 11 .

Write a regular expression that defines L_1 . If such a regular expression does not exist, explain why.

Answer:

$$(0 \cup 10)^* (1 \cup \lambda)$$

(b) Let L_2 be the set of all strings over alphabet $\{0, 1\}$ that contain an odd number of 1's.

Write a regular expression that defines L_2 . If such a regular expression does not exist, explain why.

Answer:

$$0^* 10^* (0^* 10^* 10^*)^*$$

Problem 68 (a) Let L_1 be the set of all strings over alphabet $\{a, b\}$ whose length is not equal to 2.

Write a regular expression that defines L_1 . If such a regular expression does not exist, explain why.

Answer:

$$\lambda \cup a \cup b \cup (a \cup b)(a \cup b)(a \cup b)^*$$

(b) Let L_2 be the set of all strings over alphabet $\{a, b, c\}$ that contain exactly 3 occurrences of letter a .

Write a regular expression that defines L_2 . If such a regular expression does not exist, explain why.

Answer:

$$(b \cup c)^* a (b \cup c)^* a (b \cup c)^* a (b \cup c)^*$$

Problem 69 Let L_1 be the set of exactly those strings over the alphabet $\{a, b, c\}$ that have exactly two b 's.

Let L_2 be the set of exactly those strings over the alphabet $\{a, b, c\}$ where the number of b 's is even.

Let $L_3 = (L_2 \setminus L_1^*) \cup \{\lambda\}$.

(a) Write a regular expression that represents the language L_1 . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup c)^* b (a \cup c)^* b (a \cup c)^*$$

(b) Write a regular expression that represents the language L_1^* . If such a regular expression does not exist, prove it.

Answer:

$$((a \cup c)^* b (a \cup c)^* b (a \cup c)^*)^*$$

(c) Write a regular expression that represents the language L_2 . If such a regular expression does not exist, prove it.

Answer:

$$((a \cup c)^* b (a \cup c)^* b (a \cup c)^*)^* (a \cup c)^*$$

(d) Write a regular expression that represents the language L_3 . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup c)^*$$

(e) State the cardinality of the set L_1 . (If L_1 is finite set, state the exact number of elements of L_1 . Otherwise, state that L_1 is infinite and specify whether it is countable or not.)

Answer: Infinite and countable.

(f) State the cardinality of the set L_2 . (If L_2 is a finite set, state the exact number of elements of L_2 . Otherwise, state that L_2 is infinite and specify whether it is countable or not.)

Answer: Infinite and countable.

(g) State the cardinality of the set L_3 . (If L_3 is a finite set, state the exact number of its elements. Otherwise, state that L_3 is infinite and specify whether it is countable or not.)

Answer: Infinite and countable.

Problem 70 Let L be the set of exactly those strings over the alphabet $\Sigma = \{a, b, c\}$ that begin with b , end with a , and contain as substring at least one of the following two strings: bc , ca .

Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$\begin{aligned} & b(a \cup b \cup c)^* (bc \cup ca)(a \cup b \cup c)^* a \\ & \quad \cup \\ & \quad bc(a \cup b \cup c)^* a \\ & \quad \cup \\ & \quad b(a \cup b \cup c)^* ca \end{aligned}$$

Problem 71 Let L_1 be the language defined by the regular expression:

$$(a \cup b)((a \cup b)(a \cup b))^*$$

and let L_2 be the language defined by the regular expression:

$$(a \cup b)(a \cup b)(a \cup b)^*$$

In your answers to the following questions, state cardinalities of finite sets by giving the exact numbers. For infinite sets, specify if they are countable or not.

(a) Let $S_1 = L_1 \cup L_2$. Write a regular expression that defines S_1 . If such a regular expression does not exist, explain why.

Answer: Observe that L_1 is the set of all strings of odd length over $\{a, b\}$, while L_2 is the set of all strings of length greater than 1 over $\{a, b\}$. Their union S_1 is the set of all strings of length greater than zero:

$$(a \cup b)(a \cup b)^*$$

(b) Let $S_2 = L_1 \setminus L_2$. Write a regular expression that defines S_2 . If such a regular expression does not exist, explain why.

Answer: S_2 is the set of strings of length equal to 1:

$$(a \cup b)$$

(c) Let $S_3 = L_1 \cap L_2$. Write a regular expression that defines S_3 . If such a regular expression does not exist, explain why.

Answer: S_3 is the set of strings of odd length, greater than 1:

$$(a \cup b)(a \cup b)(a \cup b)((a \cup b)(a \cup b))^*$$

(d) What is the cardinality of S_1 ?

Answer: $|S_1| = \aleph_0$; S_1 is infinite and countable.

(e) What is the cardinality of S_2 ?

Answer: $|S_2| = 2$.

(f) What is the cardinality of S_3 ?

Answer: $|S_3| = \aleph_0$; S_3 is infinite and countable.

Problem 72 Let L be the set of all strings over alphabet $\{a, b, c\}$ that begin with b , contain exactly two c 's, and end with aa .

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$b(a \cup b)^*c(a \cup b)^*c(a \cup b)^*aa$$

(b) Let \mathcal{S} be a class of languages over alphabet $\{a, b, c\}$, defined as follows:

Language L is a member of \mathcal{S} if and only if L can be represented by a regular expression.

What is the cardinality of \mathcal{S} ? Explain your answer briefly.

Answer:

$$|\mathcal{S}| = \aleph_0$$

Class \mathcal{S} is infinite and countable. There are infinitely many regular expressions, yet every regular expression is a finite string, and the set of all finite strings over any alphabet is countable.

Problem 73 Let L be the set of strings over alphabet $\{a, b, c\}$ that do not contain the substring cc .

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup ca \cup cb)^*(\lambda \cup c)$$

(b) State the cardinalities of L and \overline{L} (the complement of L) and determine which one is greater. Explain your answer briefly. (For finite sets, state exact numbers. For infinite sets, specify if countable or not.)

Answer:

$$|L| = |\overline{L}| = \aleph_0$$

Both L and \overline{L} are infinite and countable. To see that both are infinite, observe that \overline{L} is:

$$(a \cup b \cup c)^*cc(a \cup b \cup c)^*$$

Hence, both sets contain a Kleene star of a non-empty set. They are both countable, as subsets of the countable set Σ^* .

Problem 74 Let L be the set of all strings over the alphabet $\{a, b\}$ such that the first letter is different from the last letter.

Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$a(a \cup b)^*b \cup b(a \cup b)^*a$$

Problem 75 (a) Let L be the set of strings over the alphabet $\{0, 1\}$ whose first symbol is different from the last symbol. Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$0(0 \cup 1)^*1 \cup 1(0 \cup 1)^*0$$

(b) Let \mathcal{R} be the class of languages that can be represented by a regular expression, and let \mathcal{N} be the class of languages that cannot be represented by a regular expression. State the cardinalities of \mathcal{R} and \mathcal{N} , and compare them.

Answer: Class \mathcal{R} is infinite and countable:

$$|\mathcal{R}| = \aleph_0$$

Class \mathcal{N} is infinite and uncountable; its cardinality is equal to the cardinality of the set of subsets of an infinite countable set:

$$|\mathcal{N}| = 2^{\aleph_0}$$

Hence:

$$|\mathcal{N}| > |\mathcal{R}|$$

Problem 76 (a) Let L be the set of strings over the alphabet $\{0, 1, 2\}$ that contain at least one occurrence of each alphabet letter. Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$\begin{aligned} & 0(0 \cup 1 \cup 2)^* 1(0 \cup 1 \cup 2)^* 2(0 \cup 1 \cup 2)^* \\ & \quad \cup \\ & 0(0 \cup 1 \cup 2)^* 2(0 \cup 1 \cup 2)^* 1(0 \cup 1 \cup 2)^* \\ & \quad \cup \\ & 1(0 \cup 1 \cup 2)^* 0(0 \cup 1 \cup 2)^* 2(0 \cup 1 \cup 2)^* \\ & \quad \cup \\ & 1(0 \cup 1 \cup 2)^* 2(0 \cup 1 \cup 2)^* 0(0 \cup 1 \cup 2)^* \\ & \quad \cup \\ & 2(0 \cup 1 \cup 2)^* 1(0 \cup 1 \cup 2)^* 0(0 \cup 1 \cup 2)^* \\ & \quad \cup \\ & 2(0 \cup 1 \cup 2)^* 0(0 \cup 1 \cup 2)^* 1(0 \cup 1 \cup 2)^* \end{aligned}$$

(b) Let \mathcal{B} be the class of all regular languages over the alphabet $\{0, 1\}$, and let \mathcal{T} be the class of all regular languages over the alphabet $\{0, 1, 2\}$. State the cardinalities of \mathcal{B} and \mathcal{T} , and compare them.

Answer: Classes \mathcal{B} and \mathcal{T} have equal cardinalities; both of them are infinite and countable:

$$|\mathcal{B}| = |\mathcal{T}| = \aleph_0$$

Problem 77 Let L be the set of all strings over alphabet $\{a, b, c\}$ whose first letter occurs at least once again in the string.

Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$\begin{aligned} & a(b \cup c)^*a(a \cup b \cup c)^* \\ & \quad \cup \\ & b(a \cup c)^*b(a \cup b \cup c)^* \\ & \quad \cup \\ & c(a \cup b)^*c(a \cup b \cup c)^* \end{aligned}$$

Problem 78 Let L be the set of all strings over the alphabet $\{a, b, c\}$ where both a and b occur at least once. Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)^* a (a \cup b \cup c)^* b (a \cup b \cup c)^*$$

$$\quad \cup$$

$$(a \cup b \cup c)^* b (a \cup b \cup c)^* a (a \cup b \cup c)^*$$

Problem 79 Let L be the set of strings over alphabet $\{a, b, c\}$ that contain as a substring at least one of the strings: ba , bc .

- (a) Write 5 distinct strings that belong to L .
- (b) Write 5 distinct strings over alphabet $\{a, b, c\}$ that do not belong to L .

Answer:

$\in L$	$\notin L$
ba	λ
bc	a
$baabc$	b
$bcbba$	c
$bababcbcbaaa$	ab

- (c) Write a regular expression that defines L .

Answer:

$$(a \cup b \cup c)^* ba (a \cup b \cup c)^* \cup (a \cup b \cup c)^* bc (a \cup b \cup c)^*$$

Problem 80 Let L be the set of strings over alphabet $\{a, b, c\}$ in which the substring cc occurs exactly once.

- (a) Write 5 distinct strings that belong to L . If such strings do not exist, explain why.
- (b) Write 5 distinct strings over alphabet $\{a, b, c\}$ that do not belong to L . If such strings do not exist, explain why.

Answer:

$\in L$	$\notin L$
cc	λ
acc	ccc
ccb	$ccacc$
$abcbcaccabcacb$	$accacc$
$abcbacahbcababcc$	$cccc$

- (c) Write a regular expression that defines L . If such expression does not exist, explain why.

Answer:

$$(a \cup b \cup ca \cup cb)^* cc (a \cup b \cup ac \cup bc)^*$$

Problem 81 Let L be the set of strings over alphabet $\{0, 1\}$ in which the number of 1s is even.

- (a) Write 5 distinct strings that belong to L . If such strings do not exist, explain why.
- (b) Write 5 distinct strings over alphabet $\{0, 1\}$ that do not belong to L . If such strings do not exist, explain why.

Answer:

$\in L$	$\notin L$
λ	1
000	01
011	10
0011	001
110010111	01101101

- (c) Write a regular expression that defines L . If such expression does not exist, explain why.

Answer:

$$(0^* 1 0^* 1 0^*)^* \cup 0^*$$

Problem 82 Let L be the set of all strings of odd length over $\{a, b, c\}$ that contain exactly one b . Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$((a \cup c)(a \cup c))^* b ((a \cup c)(a \cup c))^*$$

$$\quad \cup$$

$$((a \cup c)(a \cup c))^* (a \cup c) b ((a \cup c)(a \cup c))^* (a \cup c)$$

Problem 83 (a) Let L_1 be the set of all strings of length two or more over alphabet $\{a, b\}$ in which all the a 's follow all the b 's. Write 5 distinct strings that belong to L_1 . If such strings do not exist, explain why.

Answer:

$$aa, bb, ba, bbb, baa$$

- (b) Write a regular expression that defines L_1 . If such a regular expression does not exist, explain why.

Answer:

$$bbb^* a^* \cup baa^* \cup aaa^*$$

(c) Let L_2 be the set of all strings over alphabet $\{a, b, c\}$ that begin with c , end with b , and contain exactly two a 's. Write 5 distinct strings that belong to L_2 . If such strings do not exist, explain why.

Answer:

$$caab, ccaab, cabab, ccabab, cbababb$$

- (d) Write a regular expression that defines L_2 . If such a regular expression does not exist, explain why.

Answer:

$$c (b \cup c)^* a (b \cup c)^* a (b \cup c)^* b$$

Problem 84 Write a regular expression that represents the set of all strings over alphabet $\{a, b, c\}$ that contain the substring ac and the substring bc . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)^* ac (a \cup b \cup c)^* bc (a \cup b \cup c)^*$$

$$\quad \cup$$

$$(a \cup b \cup c)^* bc (a \cup b \cup c)^* ac (a \cup b \cup c)^*$$

Problem 85 Let L be the set of strings over alphabet $\{a, b, c\}$ that contain both aa and bc as substrings.

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer: The answer is obtained by substituting:

$$\sigma = (a \cup b \cup c)$$

in the following regular expression:

$$\sigma^* aa \sigma^* bc \sigma^* \cup \sigma^* bc \sigma^* aa \sigma^*$$

(b) Is it possible to write a computer program (algorithm) that operates as follows:

INPUT: An arbitrary string η consisting of letters $\{a, b, c\}$.

OUTPUT: yes if $\eta \in L$ and no if $\eta \notin L$.

Explain your answer briefly.

Answer: Yes—such an algorithm first converts the regular expression constructed in the answer to part (a) into a finite automaton that accepts L , and then simulates this automaton.

Problem 86 (a) Let L_1 be the set of all strings over alphabet $\{a, b\}$ that do not begin with the substring bb .

Write a regular expression that defines L_1 . If such a regular expression does not exist, explain why.

Answer:

$$\lambda \cup a(a \cup b)^* \cup b \cup ba(a \cup b)^*$$

(b) Let L_2 be the set of all strings over alphabet $\{a, b, c, d\}$ that contain the substrings ab , ac , and ad .

Write a regular expression that defines L_2 . If such a regular expression does not exist, explain why.

Answer: The answer is obtained by substituting:

$$\sigma = (a \cup b \cup c \cup d)$$

in the following regular expression:

$$\begin{aligned} & \sigma^* ab \sigma^* ac \sigma^* ad \sigma^* \cup \\ & \sigma^* ab \sigma^* ad \sigma^* ac \sigma^* \cup \\ & \sigma^* ac \sigma^* ab \sigma^* ad \sigma^* \cup \\ & \sigma^* ac \sigma^* ad \sigma^* ab \sigma^* \cup \\ & \sigma^* ad \sigma^* ab \sigma^* ac \sigma^* \cup \\ & \sigma^* ad \sigma^* ac \sigma^* ab \sigma^* \end{aligned}$$

Problem 87 (a) Let L_1 be the set of all strings over alphabet $\{a, b, c\}$ in which all the a 's precede the b 's, which in turn precede the c 's. (It is possible that any or all of the three letters is missing, but those letters that do appear must appear in the given order.)

Write a regular expression that defines L_1 . If such a regular expression does not exist, explain why.

Answer:

$$a^* b^* c^*$$

(b) Let L_2 be the same set as L_1 , but without the empty string:

$$L_2 = L_1 \setminus \{\lambda\}$$

Write a regular expression that defines L_2 . If such a regular expression does not exist, explain why.

Answer:

$$aa^* b^* c^* \cup a^* bb^* c^* \cup a^* b^* cc^*$$

Problem 88 Let L be a set of strings over alphabet $\{0, 1\}$ such that

$$L = \{w \mid w \neq 11 \wedge w \neq 111\}$$

(a) Write 5 distinct strings that belong to L . If such strings do not exist, explain why.

Answer:

$$\lambda, 0, 1, 00, 01, 10, 000, 001, 010, 011, 100, 101, 110$$

(b) Write 5 distinct strings over alphabet $\{0, 1\}$ that do not belong to L . If such strings do not exist, explain why.

Answer: It is impossible to find 5 such strings. Straightforwardly, by definition of L , there are only two strings over $\{0, 1\}$ that do not belong to L :

$$11, 111$$

(c) Write a regular expression that defines L . If such expression does not exist, explain why.

Answer:

$$\lambda \cup 1 \cup (0 \cup 10 \cup 110 \cup 111(0 \cup 1))(0 \cup 1)^*$$

Problem 89 Let L be the set of strings of odd length over alphabet $\{a, b\}$ that contain the substring aa . Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$\begin{aligned} & ((a \cup b)(a \cup b))^* (a \cup b) aa ((a \cup b)(a \cup b))^* \\ & \quad \cup \\ & ((a \cup b)(a \cup b))^* aa ((a \cup b)(a \cup b))^* (a \cup b) \end{aligned}$$

Problem 90 Write a regular expression that defines the set of strings over alphabet $\{a, b, c\}$ with length greater than 3. If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)(a \cup b \cup c)(a \cup b \cup c)(a \cup b \cup c)(a \cup b \cup c)^*$$

Problem 91 Write a regular expression that represents the set of strings over $\{a, b, c\}$ in which the total number of b 's and c 's is three. If such regular expression does not exist, prove it.

Answer:

$$a^*(b \cup c)a^*(b \cup c)a^*(b \cup c)a^*$$

Problem 92 Let L be the set of all strings over alphabet $\{a, b\}$ in which substring aa occurs exactly once. Write a regular expression that defines L . If such a regular expression does not exist, explain why.

Answer:

$$(ab \cup b)^* aa (ba \cup b)^*$$

Problem 93 (a) Let L_1 be the set of all strings over alphabet $\{a, b\}$ whose length gives remainder 2 if divided by 3.

Write a regular expression that defines L_1 . If such regular expression does not exist, explain why.

Answer:

$$((a \cup b)(a \cup b)(a \cup b))^* (a \cup b)(a \cup b)$$

(b) Let L_2 be the set of all strings over alphabet $\{a, b\}$ whose length gives a remainder different from 2 if divided by 3.

Write a regular expression that defines L_2 . If such regular expression does not exist, explain why.

Answer:

$$((a \cup b)(a \cup b)(a \cup b))^* (\lambda \cup a \cup b)$$

Problem 94 Let L be the set of all strings over alphabet $\{a, b\}$ in which the number of a 's is divisible by three. Write a regular expression that defines L . If such a regular expression does not exist, explain why.

Answer:

$$(b^* ab^* ab^* ab^*)^* \cup b^*$$

Problem 95 Let:

$$L = \{a^{i+3}b^{2j+1}c^{3k+2}a^{5\ell+4} \mid i, j, k, \ell \geq 0\}$$

Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$a^* aaa(bb)^* b(ccc)^* cc(aaaaa)^* aaaa$$

Problem 96 Let L be the set of strings over alphabet $\{a, b, c\}$ with at most three a 's.

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$(b \cup c)^* (\lambda \cup a)(b \cup c)^* (\lambda \cup a)(b \cup c)^* (\lambda \cup a)(b \cup c)^*$$

(b) Is \bar{L} (the complement of L) context-free? Explain your answer briefly.

Answer: Yes—language \bar{L} is regular as the complement of a regular language; every regular language is context-free.

Problem 97 Let L be the set of strings over alphabet $\{a, b, c\}$ that have odd length or odd number of b 's. Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$\begin{aligned} & (a \cup b \cup c) ((a \cup b \cup c)(a \cup b \cup c))^* \\ & \quad \cup \\ & (a \cup c)^* b (a \cup c)^* ((a \cup c)^* b (a \cup c)^* b (a \cup c)^*)^* \end{aligned}$$

Problem 98 Let L be the set of strings over alphabet $\{a, b, c\}$ that have even length and contain exactly one c .

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$\begin{aligned} & ((a \cup b)(a \cup b))^* c (a \cup b)((a \cup b)(a \cup b))^* \\ & \quad \cup \\ & ((a \cup b)(a \cup b))^* (a \cup b)c ((a \cup b)(a \cup b))^* \end{aligned}$$

(b) Write a regular expression that defines \bar{L} (the complement of L). If such regular expression does not exist, prove it.

Answer:

$$\begin{aligned} & ((a \cup b \cup c)(a \cup b \cup c))^* (a \cup b \cup c) \\ & \quad \cup \\ & (a \cup b)^* \cup (a \cup b)^* c (a \cup b)^* c (a \cup b \cup c)^* \end{aligned}$$

Problem 99 (a) Write a regular expression that represents the set of strings over $\{a, b, c\}$ that begin with a , end with c , and contain exactly two b 's.

Answer:

$$a(a \cup c)^* b(a \cup c)^* b(a \cup c)^* c$$

(b) Write a regular expression that represents the set of strings over $\{a, b, c\}$ that begin with a , end with c , and contain at most two b 's.

Answer:

$$a(a \cup c)^* (b \cup \lambda)(a \cup c)^* (b \cup \lambda)(a \cup c)^* c$$

Problem 100 Let L be the set of strings over the alphabet $\{a, b, c\}$ that have even length and begin and end with the same letter.

Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$\begin{aligned} & a ((a \cup b \cup c)(a \cup b \cup c))^* a \\ & \quad \cup \\ & b ((a \cup b \cup c)(a \cup b \cup c))^* b \\ & \quad \cup \\ & c ((a \cup b \cup c)(a \cup b \cup c))^* c \end{aligned}$$

Problem 101 (a) Let L_1 be the set of all strings over alphabet $\{0, 1\}$ that do not contain the substring 00.

Write a regular expression that defines L_1 . If such a regular expression does not exist, explain why.

Answer:

$$(1 \cup 01)^*(0 \cup \lambda)$$

(b) Let L_2 be the set of all strings over alphabet $\{0, 1\}$ that contain an odd number of 0's.

Write a regular expression that defines L_2 . If such a regular expression does not exist, explain why.

Answer:

$$1^*01^*(1^*01^*01^*)^*$$

Problem 102 Let L be the set of strings over alphabet $\{a, b, c\}$ with length less than four.

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c \cup \lambda)(a \cup b \cup c \cup \lambda)(a \cup b \cup c \cup \lambda)$$

(b) Is L context-free? Explain your answer briefly.

Answer: Yes—every regular language is context-free (and we know that L is regular, since it has a regular expression, given in the answer to part (a).)

3 Context-Free Grammars

Problem 103 Let L_1 be the language defined by the regular expression:

$$a^*b^*$$

Let L_2 be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSb \mid \lambda$$

(a) Write 5 distinct strings that belong to $L_1 \setminus L_2$. If such strings do not exist, explain why.

Answer:

$$a, b, aab, abbb, abbbbb$$

(b) Write 5 distinct strings that belong to $L_2 \setminus L_1$. If such strings do not exist, explain why.

Answer: It is impossible to list even one such string, since:

$$L_2 \setminus L_1 = \emptyset$$

Precisely:

$$L_2 \subset L_1$$

To see this, observe:

$$L_1 = \{a^m b^k \mid m, k \geq 0\}$$

$$L_2 = \{a^m b^k \mid m = k \text{ and } m, k \geq 0\}$$

(c) Write 5 distinct strings that belong to $L_1 \cap L_2$. If such strings do not exist, explain why.

Answer:

$$\lambda, ab, aabb, aaabbb, aaaabbbb$$

(Recall that $L_1 \cap L_2 = L_2$.)

(d) Write 5 distinct strings over alphabet $\{a, b\}$ that belong to $\overline{L_1 \cup L_2}$ (the complement of $L_1 \cup L_2$). If such strings do not exist, explain why.

Answer:

$$ba, aabba, bab, baabb, aba$$

(Recall that $L_1 \cup L_2 = L_1$. Hence, $\overline{L_1 \cup L_2} = \overline{L_1}$, which is exactly the set of strings that contain ba as a substring.)

Problem 104 Let L_1 be the language defined by the regular expression:

$$(ab)^* c$$

Let L_2 be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSc \mid b$$

(a) Write 5 distinct strings that belong to L_1 but do not belong to L_2 (belong to $L_1 \setminus L_2$). If such strings do not exist, state it and explain why.

Answer: $c, ababc, abababc, ababababc, abababababc$

(b) Write 5 distinct strings that belong to L_2 but do not belong to L_1 (belong to $L_2 \setminus L_1$). If such strings do not exist, state it and explain why.

Answer: $b, aabcc, aaabccc, aaaabcccc, aaaaabcccc$

(c) Write 5 distinct strings that belong to L_1 and L_2 (belong to $L_1 \cap L_2$). If such strings do not exist, state it and explain why.

Answer: Impossible—exactly one string, namely abc , belongs to $L_1 \cap L_2$, and we cannot list as many as five. To see this, observe that the general templates for the two languages are:

$$L_1 : (ab)^n c \quad \text{where } n \geq 0$$

$$L_2 : a^n b c^n \quad \text{where } n \geq 0$$

(d) Write 5 distinct strings over alphabet $\{a, b, c\}$ that do not belong to L_1 and do not belong to L_2 (belong to $\overline{L_1 \cap L_2}$). If such strings do not exist, state it and explain why.

Answer: ba, ca, cb, cc, aa

(e) Write 5 distinct strings that belong to L_2^* but do not belong to L_2 (belong to $L_2^* \setminus L_2$). If such strings do not exist, state it and explain why.

Answer: $bb, bbb, abc, abc, abcabc$

(f) Write 5 distinct strings that belong to $L_1 L_2$ but do not belong to $L_2 L_1$ (belong to $L_1 L_2 \setminus L_2 L_1$). If such strings do not exist, state it and explain why.

Answer: $cb, abc, cab, ababc, caabc$

Problem 105 Let L_1 be the language defined by the regular expression:

$$(aa \cup b)^*$$

Let L_2 be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSa \mid bSb \mid \lambda$$

(a) List 5 distinct strings that belong to L_1 and L_2 (belong to $L_1 \cap L_2$). If this is impossible, state it and explain why.

Answer:

$$\lambda, bb, bbbb, aaaa, aabbbbaa$$

(b) List 5 distinct strings that belong to L_1 but do not belong to L_2 (belong to $L_1 \cap \overline{L_2}$). If this is impossible, state it and explain why.

Answer:

$$b, baa, baabb, aab, aaaaab$$

(c) List 5 distinct strings that belong to L_2 but do not belong to L_1 (belong to $\overline{L_1} \cap L_2$). If this is impossible, state it and explain why.

Answer:

$$abba, abbbba, abaaba, aaabbaaa, abaaaaba$$

(d) List 5 distinct strings over the alphabet $\{a, b\}$ that belong to neither L_1 nor L_2 (belong to $\overline{L_1} \cap \overline{L_2}$). If this is impossible, state it and explain why.

Answer:

$$a, aaa, ab, ba, aba$$

(e) List 5 distinct strings that belong to L_1^* but do not belong to L_1 (belong to $L_1^* \cap \overline{L_1}$). If this is impossible, state it and explain why.

Answer: Impossible— L_1 itself is already a Kleene star of a regular language, and any positive number of applications of Kleene star is equivalent to the first application. Hence, $L_1^* = L_1$, meaning that $L_1^* \cap \overline{L_1} = \emptyset$, and there are no elements to list.

(f) List 5 distinct strings that belong to $L_2 L_2$ but do not belong to L_2 (belong to $L_2 L_2 \cap \overline{L_2}$). If this is impossible, state it and explain why.

Answer:

$$aabb, bbaa, abbabb, bbabba, aabbbb$$

Problem 106 Let L_1 be the language defined by the regular expression:

$$a^*b^*$$

Let L_2 be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aaSb \mid a$$

(a) List 5 distinct strings that belong to L_1 and L_2 (belong to $L_1 \cap L_2$). If this is impossible, state it and explain why.

Answer:

$$a, aaab, aaaaabb, aaaaaaaabbb, aaaaaaaaaabbbb$$

(b) List 5 distinct strings that belong to L_1 but do not belong to L_2 (belong to $L_1 \cap \overline{L_2}$). If this is impossible, state it and explain why.

Answer:

$$b, bb, bbb, ab, abbb$$

(c) List 5 distinct strings that belong to L_2 but do not belong to L_1 (belong to $\overline{L_1} \cap L_2$). If this is impossible, state it and explain why.

Answer: Impossible, since $L_2 \subseteq L_1$, which means that $\overline{L_1} \cap L_2 = \emptyset$, and there are no elements to list. To see this, observe that:

$$L_1 = \{a^n b^k \mid n, k \geq 0\}$$

$$L_2 = \{a^n b^k \mid n = 2k + 1 \wedge k \geq 0\}$$

Hence, L_2 contains exactly those strings of L_1 that satisfy an additional constraint, as stated.

(d) List 5 distinct strings over the alphabet $\{a, b\}$ that belong to neither L_1 nor L_2 (belong to $\overline{L_1} \cap \overline{L_2}$). If this is impossible, state it and explain why.

Answer:

$$ba, aba, aaba, bba, bab$$

(e) List 5 distinct strings that belong to L_2^* but do not belong to L_1^* (belong to $L_2^* \cap \overline{L_1^*}$). If this is impossible, state it and explain why.

Answer: Impossible— $L_2 \subseteq L_1$ implies that $L_2^* \subseteq L_1^*$, which means that $L_2^* \cap \overline{L_1^*} = \emptyset$ and there are no elements to list.

(f) List 5 distinct strings that belong to $L_2 L_2$ but do not belong to L_2 (belong to $L_2 L_2 \cap \overline{L_2}$). If this is impossible, state it and explain why.

Answer:

$$\begin{aligned} &aaabaaaab \\ &aaabaaaaabb \\ &aaaaaabbaaaab \\ &aaabaaaabaaab \\ &aaaabbaaaaabb \end{aligned}$$

Problem 107 Let L_1 be the language defined over alphabet $\Sigma = \{a, b\}$ by the regular expression:

$$b^*$$

Let L_2 be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aaSaa \mid b$$

1. Write a complete formal definition of a context-free grammar G_1 that generates language $L_2 \setminus L_1$. If such grammar does not exist, explain why.

Answer: Observe that:

$$L_2 = \{a^{2n}ba^{2n} \mid n \geq 0\}$$

The only string that belongs to both L_1 and L_2 is b , obtained by setting $n = 0$ in the template for L_2 . This means that:

$$L_2 \setminus L_1 = \{a^{2n}ba^{2n} \mid n \geq 1\}$$

whence the grammar: $G_1 = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aaSaa \mid aabaa$$

2. Write a regular expression that defines $L_1 \cap L_2$. If such regular expression does not exist, explain why.

Answer:

b

3. What is the cardinality of $L_1 \cup L_2$? (If possible, state the exact number. If the set is infinite, specify if it is countable or not.)

Answer: Every language is countable. Both L_1 and L_2 are infinite, and so is their union. In short:

$$|L_1 \cup L_2| = \aleph_0$$

4. What is the cardinality of $\mathcal{P}(L_1)$ —the set of subsets of L_1 ? (If possible, state the exact number. If the set is infinite, specify if it is countable or not.)

Answer: L_1 is infinite and countable, and the set of its subsets is uncountable. In short:

$$|\mathcal{P}(L_1)| > \aleph_0$$

5. Does every subset of L_1 have a finite description? Explain your answer.

Answer: No—there are only countably many finite descriptions, and uncountably many subsets of L_1 .

Problem 108 Let L_1 be the language defined by the regular expression:

$$a^* b^* (cd)^*$$

Let L_2 be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aScd \mid B \\ B &\rightarrow bB \mid \lambda \end{aligned}$$

- (a) Write 5 distinct strings that belong to L_1 but do not belong to L_2 (belong to $L_1 \setminus L_2$). If such strings do not exist, state it and explain why.

Answer:

$$a, cd, ab, bcd, aacd$$

- (b) Write 5 distinct strings that belong to L_2 but do not belong to L_1 (belong to $L_2 \setminus L_1$). If such strings do not exist, state it and explain why.

Answer: Impossible, since $L_2 \subset L_1$. To see this, observe that the general templates for the two languages are:

$$\begin{aligned} L_1 : \quad &a^n b^k (cd)^\ell && \text{where } n, k, \ell \geq 0 \\ L_2 : \quad &a^n b^k (cd)^\ell && \text{where } n, k, \ell \geq 0 \text{ and } n = \ell \end{aligned}$$

- (c) Write 5 distinct strings that belong to L_1 and L_2 (belong to $L_1 \cap L_2$). If such strings do not exist, state it and explain why.

Answer:

$$b, bb, acd, aacdcd, aabcdcd$$

- (d) Write 5 distinct strings over alphabet $\{a, b, c, d\}$ that do not belong to L_1 and do not belong to L_2 (belong to $\overline{L_1} \cap \overline{L_2}$). If such strings do not exist, state it and explain why.

Answer:

$$ba, dc, cb, db, ca$$

- (e) Write 5 distinct strings that belong to L_2^* but do not belong to L_2 (belong to $L_2^* \setminus L_2$). If such strings do not exist, state it and explain why.

Answer:

$$acdb, acdbb, acdacd, bacdb, babcd$$

- (f) Write 5 distinct strings that belong to $L_1 L_1$ but do not belong to L_1 (belong to $L_1 L_1 \setminus L_1$). If such strings do not exist, state it and explain why.

Answer:

$$acdb, abcdb, ba, cda, cdab$$

Problem 109 Let L_1 be the language defined over alphabet $\Sigma = \{0, 1\}$ by the regular expression:

$$(0 \cup 1)(0 \cup 1)(0 \cup 1)(0 \cup 1)(0 \cup 1)^*$$

Let L_2 be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{0, 1\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow 00S \mid 11S \mid \lambda$$

1. Write a complete formal definition of a context-free grammar G_1 that generates language $L_1 \cup L_2$. If such grammar does not exist, explain why.

Answer: $G_1 = (V, \Sigma, P, T)$, where:

$\Sigma = \{0, 1\}$, $V = \{S, T, W, A, B\}$, and P is:

$$\begin{aligned} T &\rightarrow S \mid W \\ S &\rightarrow 00S \mid 11S \mid \lambda \\ W &\rightarrow BBBBA \\ B &\rightarrow 0 \mid 1 \\ A &\rightarrow BA \mid \lambda \end{aligned}$$

2. Write a regular expression that defines $L_2 \setminus L_1$. If such regular expression does not exist, explain why.

Answer: Observe that L_1 contains all strings with length no less than 4. There are only 3 strings in L_2 with length less than 4, whence the answer:

$$\lambda \cup 00 \cup 11$$

3. List three distinct non-empty strings that belong to L_2 , and calculate the Gödel number of each of these three strings. Show your work. If this is impossible, prove it.

Answer:

$$00, 0000, 11$$

$$\begin{aligned} G(\langle 00 \rangle) &= 2^{0+1} \cdot 3^{0+1} = 6 \\ G(\langle 0000 \rangle) &= 2^{0+1} \cdot 3^{0+1} \cdot 5^{0+1} \cdot 7^{0+1} = 210 \\ G(\langle 11 \rangle) &= 2^{1+1} \cdot 3^{1+1} = 36 \end{aligned}$$

Problem 110 Let L_1 be the language defined by the regular expression:

$$(ab)^*$$

Let L_2 be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSa \mid bSb \mid a \mid b \mid \lambda$$

- (a) Write a regular expression that defines $L_1 \cap L_2$. If such a regular expression does not exist, explain why.

Answer:

$$\lambda$$

To verify the answer, observe first that L_2 is the language of all palindromes over $\{a, b\}$. In contrast, every non-empty string from L_1 begins with a and ends with b , and cannot be a palindrome. Hence, $L_1 \cap L_2$ contains no non-empty strings. However, the empty string is derivable in G by the last rule; it also belongs to L_1 , because L_1 is a Kleene star.

- (b) What is the cardinality of $L_1 \cap L_2$? (If possible, state the exact number. If the set is infinite, specify if it is countable or not.)

Answer: It follows from the answer to part (a) that:

$$|L_1 \cap L_2| = 1$$

- (c) What is the cardinality of L_1 ? (If possible, state the exact number. If the set is infinite, specify if it is countable or not.)

Answer:

$$|L_1| = \aleph_0$$

Every language is countable, and so is L_1 . To see that L_1 is infinite, observe that it is a Kleene star of a non-empty language.

- (d) What is the cardinality of L_2 ? (If possible, state the exact number. If the set is infinite, specify if it is countable or not.)

Answer:

$$|L_2| = \aleph_0$$

Every language is countable, and so is L_2 . To see that L_2 is infinite, observe that any of the first two rules can be applied an unbounded number of times, appending two symbols to the sentential form, before a non-empty terminal string is eventually produced by any of the next two rules.

- (e) Compare the cardinalities of L_1 and L_2 , and explain which one (if any) is greater.

Answer: It follows from the answers to parts (c) and (d) that the two cardinalities are equal:

$$|L_1| = |L_2| = \aleph_0$$

- (f) What is the cardinality of $L_1 \cup L_2$? (If possible, state the exact number. If the set is infinite, specify if it is countable or not.)

Answer: It follows from the answers to parts (c) and (d) that $L_1 \cup L_2$ is a union of two infinite and countable languages, and thereby itself infinite and countable:

$$|L_1 \cup L_2| = \aleph_0$$

- (g) Compare the cardinalities of $L_1 \cup L_2$ and $L_1 \cap L_2$, and explain which one (if any) is greater.

Answer: It follows from the answers to parts (b) and (f) that $L_1 \cap L_2$ is finite, while $L_1 \cup L_2$ is infinite. Hence:

$$|L_1 \cap L_2| < |L_1 \cup L_2|$$

Problem 111 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, E, D\}$, and P comprises:

$$\begin{aligned} S &\rightarrow ABE \mid D \\ A &\rightarrow cA \mid \lambda \\ B &\rightarrow bcb \\ E &\rightarrow aaE \mid \lambda \\ D &\rightarrow bDb \mid c \end{aligned}$$

- (a) Write 10 distinct strings that belong to L . If such strings do not exist, explain why.
- (b) Write 10 distinct strings over alphabet $\{a, b, c\}$ that do not belong to L . If such strings do not exist, explain why.

Answer:

$\in L$	$\notin L$
c	a
bcb	bcba
bbccb	aa
bbbcb	b
bbbbcb	cc
cbc	cbcba
bcba	bb
cbcba	ab
cccbcba	ac
cbcbaaaaaaa	bc

Answer: L is the set of strings of even length over Σ .

$\in L$	$\notin L$
λ	a
aa	b
bb	aab
ba	aba
ab	abb
abba	bba
abbbaa	bbb
aaaaaaaa	bab
aaaaaaaa	baa
aaaaaaaa	aaa

Problem 112 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{array}{l} S \rightarrow aSb \mid A \\ A \rightarrow cAb \mid B \\ B \rightarrow cb \end{array}$$

(a) Write 8 distinct strings that belong to L . If such strings do not exist, explain why.

(b) Write 8 distinct strings over alphabet $\{a, b, c\}$ that do not belong to L . If such strings do not exist, explain why.

Answer: Observe that:

$$L = \{a^m c^n b^{m+n} \mid m \geq 0, n \geq 1\}$$

whence:

$\in L$	$\notin L$
cb	b
ccbb	bc
cccbb	ba
acbb	bca
accbb	bcb
accbbb	aba
aaacbbb	ab
aaacbbb	abc

Problem 113 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, A\}$, and P comprises:

$$\begin{array}{l} S \rightarrow aA \mid bA \mid \lambda \\ A \rightarrow aS \mid bS \end{array}$$

(a) Write 10 distinct strings that belong to L . If such strings do not exist, explain why.

(b) Write 10 distinct strings over alphabet $\{a, b\}$ that do not belong to L . If such strings do not exist, explain why.

Problem 114 Let L_1 be the language defined by the regular expression:

$$(a \cup b)^* aa (a \cup b)^* bb (a \cup b)^*$$

Let L_2 be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{array}{l} S \rightarrow SS \mid A \mid \lambda \\ A \rightarrow BBB \\ B \rightarrow a \mid b \end{array}$$

(a) Write 5 distinct strings that belong to L_1 and do not belong to L_2 . If such strings do not exist, explain why.

Answer:

$$aabb, aaabb, baabb, aabba, baababb$$

(b) Write 5 distinct strings that belong to L_2 and do not belong to L_1 . If such strings do not exist, explain why.

Answer:

$$aaa, bbb, aba, bab, aab$$

(c) Write 5 distinct strings that belong to L_1 and L_2 . If such strings do not exist, explain why.

Answer:

$$aaabbb, baabbb, aaabba, baabba, aababb$$

(d) Write 5 distinct strings over alphabet $\{a, b\}$ that do not belong to L_1 and do not belong to L_2 . If such strings do not exist, explain why.

Answer:

$$a, b, ab, ba, aa$$

Problem 115 Let L_1 be the language defined by the regular expression:

$$(a \cup ba \cup ca)^* (b \cup c)$$

Let L_2 be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ABABA \\ A &\rightarrow AA \mid a \mid c \mid \lambda \\ B &\rightarrow b \end{aligned}$$

- (a) Write 5 distinct strings that belong to L_1 and do not belong to L_2 (belong to $L_1 \cap \overline{L_2}$). If such strings do not exist, explain why.

Answer:

$$ab, ac, aab, aac, cab$$

- (b) Write 5 distinct strings that belong to L_2 and do not belong to L_1 (belong to $\overline{L_1} \cap L_2$). If such strings do not exist, explain why.

Answer:

$$bb, abb, cbb, aabb, ccbb$$

- (c) Write 5 distinct strings that belong to L_1 and L_2 (belong to $L_1 \cap L_2$). If such strings do not exist, explain why.

Answer:

$$bab, babac, abab, ababac, cabab$$

- (d) Write 5 distinct strings over alphabet $\{a, b, c\}$ that do not belong to L_1 and do not belong to L_2 (belong to $\overline{L_1} \cap \overline{L_2}$). If such strings do not exist, explain why.

Answer:

$$a, aa, aaa, aaaa, aaaaa$$

Note that L_2 is the set of strings over $\{a, b, c\}$ that contain exactly 2 occurrences of letter b . It is given by the regular expression:

$$(a \cup c)^* b (a \cup c)^* b (a \cup c)^*$$

Problem 116 Let L_1 be the language defined by the regular expression:

$$(aa)^* \cup (bb)^*$$

Let L_2 be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSb \mid \lambda$$

- (a) Write 5 distinct strings that belong to L_1 and do not belong to L_2 . If such strings do not exist, explain why.

Answer:

$$aa, bb, aaaa, bbbb, aaaaaa$$

- (b) Write 5 distinct strings that belong to L_2 and do not belong to L_1 . If such strings do not exist, explain why.

Answer:

$$ab, aabb, aaabbb, aaaabbbb, aaaaabbbbb$$

- (c) Write 5 distinct strings that belong to L_1 and L_2 . If such strings do not exist, explain why.

Answer: Observe that:

$$L_1 = \{(aa)^m \mid m \geq 0\} \cup \{(bb)^m \mid m \geq 0\}$$

while:

$$L_2 = \{a^m b^m \mid m \geq 0\}$$

Every non-empty string in L_2 contains both a and b , whereas no non-empty string in L_1 contains both a and b . Hence, there exists exactly one string, λ , that belongs to $L_1 \cap L_2$.

- (d) Write 5 distinct strings over alphabet $\{a, b\}$ that do not belong to L_1 and do not belong to L_2 . If such strings do not exist, explain why.

Answer:

$$ba, bba, aab, bbaa, aba$$

Problem 117 Let L_1 be the language defined by the regular expression:

$$(a \cup b)^* c (a \cup b)^* c (a \cup b)^* c (a \cup b)^*$$

Let L_2 be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AAS \mid \lambda \\ A &\rightarrow a \mid b \mid c \end{aligned}$$

- (a) Write 5 distinct strings that belong to L_1 and do not belong to L_2 (belong to $L_1 \cap \overline{L_2}$). If such strings do not exist, explain why.

Answer: Observe that L_1 is the set of strings over alphabet $\{a, b, c\}$ that contain exactly 3 occurrences of letter c , while L_2 is the set of strings over alphabet $\{a, b, c\}$ that have even length.

$$ccc, cccaa, abccc, cacac, cbbcc$$

- (b) Write 5 distinct strings that belong to L_2 and do not belong to L_1 (belong to $\overline{L_1} \cap L_2$). If such strings do not exist, explain why.

Answer:

$$\lambda, aa, ab, ac, abc$$

- (c) Write 5 distinct strings that belong to L_1 and L_2 (belong to $L_1 \cap L_2$). If such strings do not exist, explain why.

Answer:

$$accc, cbccaa, abcacc, cacaca, cbbcca$$

- (d) Write 5 distinct strings over alphabet $\{a, b, c\}$ that do not belong to L_1 and do not belong to L_2 (belong to $\overline{L_1} \cap \overline{L_2}$). If such strings do not exist, explain why.

Answer:

$$a, b, c, aaa, aac$$

Problem 118 Let L_1 be the language defined by the regular expression:

$$((a \cup b)(a \cup b)(a \cup b))^*$$

Let L_2 be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow AA \mid \lambda \mid DDD \\ B &\rightarrow D \mid DD \\ D &\rightarrow a \mid b \end{aligned}$$

- (a) Write 5 distinct strings that belong to L_1 and do not belong to L_2 (belong to $L_1 \cap \overline{L_2}$). If such strings do not exist, explain why.

Answer: Observe that L_1 is the set of strings over alphabet $\{a, b\}$ whose length is divisible by 3, while L_2 is the set of strings over alphabet $\{a, b\}$ whose length is not divisible by 3.

$$\lambda, aaa, aab, aac, aba$$

- (b) Write 5 distinct strings that belong to L_2 and do not belong to L_1 (belong to $\overline{L_1} \cap L_2$). If such strings do not exist, explain why.

Answer:

$$a, b, c, aa, ab$$

- (c) Write 5 distinct strings that belong to L_1 and L_2 (belong to $L_1 \cap L_2$). If such strings do not exist, explain why.

Answer: The length of any string is either divisible by 3, or is not divisible by 3, but not both. Hence, every string in $\{a, b\}^*$ belongs to exactly one of the sets L_1, L_2 . This means that L_1 and L_2 are disjoint and jointly cover $\{a, b\}^*$:

$$\begin{aligned} L_1 \cap L_2 &= \emptyset \\ L_1 \cup L_2 &= \{a, b\}^* \end{aligned}$$

Consequently, not a single string over alphabet $\{a, b\}$ belongs to $L_1 \cap L_2$.

- (d) Write 5 distinct strings over alphabet $\{a, b\}$ that do not belong to L_1 and do not belong to L_2 (belong to $\overline{L_1} \cap \overline{L_2}$). If such strings do not exist, explain why.

Answer: Since every string in $\{a, b\}^*$ belongs to exactly one of the sets L_1, L_2 , not a single string over alphabet $\{a, b\}$ can be excluded from both L_1 and L_2 . Precisely, by the analysis given in the answer to part (c), and De-Morgan laws:

$$\overline{\emptyset} = \{a, b\}^* = L_1 \cup L_2 = \overline{L_1} \cap \overline{L_2}$$

whence:

$$\overline{L_1} \cap \overline{L_2} = \emptyset$$

Problem 119 Let L_1 be the language defined over alphabet $\Sigma = \{a, b, c\}$ by the regular expression:

$$(a \cup b \cup c)^* cab (a \cup b \cup c)^*$$

Let L_2 be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ZSZ \mid a \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

- (a) Write a complete formal definition of a context-free grammar G_1 that generates language L_1 . If such a grammar does not exist, explain why.

Answer: $G_1 = (V_1, \Sigma, P_1, S_1)$, where $\Sigma = \{a, b, c\}$, $V_1 = \{S_1, A\}$, and the production set P_1 is:

$$\begin{aligned} S_1 &\rightarrow AcabA \\ A &\rightarrow AA \mid \lambda \mid a \mid b \mid c \end{aligned}$$

- (b) Write a complete formal definition of a context-free grammar G_2 that generates language $L_1 \cup L_2$. If such a grammar does not exist, explain why.

Answer: $G_2 = (V_2, \Sigma, P_2, T)$, where $\Sigma = \{a, b, c\}$, $V_2 = \{S, Z, S_1, A, T\}$, and the production set P_2 is:

$$\begin{aligned} T &\rightarrow S \mid S_1 \\ S &\rightarrow ZSZ \mid a \\ Z &\rightarrow a \mid b \mid c \\ S_1 &\rightarrow AcabA \\ A &\rightarrow AA \mid \lambda \mid a \mid b \mid c \end{aligned}$$

- (c) List six different strings of length 3 that belong to $L_1 \cap L_2$. If this is impossible, explain why.

Answer: Impossible. L_1 is the set of all strings over Σ that contain cab as a substring. Hence, the only string of length 3 belonging to L_1 is cab itself. This means that we cannot find as many as six distinct strings of length 3 in L_1 (regardless of L_2 .)

- (d) List six different strings of length 3 that belong to $L_2 \setminus L_1$. If this is impossible, explain why.

Answer: L_2 contains odd-length strings whose middle symbol is a . There are nine such strings of length 3, one of which— cab —is disqualified as it belongs to L_1 . Any six of the remaining eight form a valid answer:

$$aaa, aab, aac, baa, bab, bac, caa, cac$$

- (e) List six different strings of length 3 that belong to $\overline{L_1 \cup L_2}$ (the complement of $L_1 \cup L_2$). If this is impossible, explain why.

Answer: Any six of the 18 strings of length 3 whose middle symbol is different from a are acceptable, for example:

$$aba, abb, abc, bba, bbb, bbc$$

Problem 120 Let L_1 be the language defined by the regular expression:

$$(ab)^* e^* (cd)^*$$

Let L_2 be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, e\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow abScd \mid e$$

- (a) Write 5 distinct strings that belong to L_1 but do not belong to L_2 (belong to $L_1 \setminus L_2$). If such strings do not exist, state it and explain why.

Answer:

$$ab, ee, cd, abe, ecd$$

- (b) Write 5 distinct strings that belong to L_2 but do not belong to L_1 (belong to $L_2 \setminus L_1$). If such strings do not exist, state it and explain why.

Answer: Impossible, since $L_2 \subset L_1$ and thereby $L_2 \setminus L_1 = \emptyset$. To see this, observe that the two templates are:

$$L_1 = \{(ab)^m e^k (cd)^n\}$$

$$L_2 = \{(ab)^i e (cd)^j\}$$

Evidently, L_2 is obtained from L_1 by setting $m = n$ and $k = 1$.

- (c) Write 5 distinct strings that belong to L_1 and L_2 (belong to $L_1 \cap L_2$). If such strings do not exist, state it and explain why.

Answer:

$$e, abecd, ababecdc, abababecdc, ababababecdc, abababababecdc$$

- (d) Write 5 distinct strings over alphabet $\{a, b, c\}$ that do not belong to L_1 and do not belong to L_2 (belong to $\overline{L_1} \cap \overline{L_2}$). If such strings do not exist, state it and explain why.

Answer:

$$a, b, c, d, ac$$

- (e) Write 5 distinct strings that belong to L_2^* but do not belong to L_2 (belong to $L_2^* \setminus L_2$). If such strings do not exist, state it and explain why.

Answer:

$$\lambda, ee, eee, abecde, eabecde$$

- (f) Write 5 distinct strings that belong to $L_1 L_1$ but do not belong to L_1 (belong to $L_1 L_1 \setminus L_1$). If such strings do not exist, state it and explain why.

Answer:

$$abcaab, abcdab, cdc, cdab, eab$$

Problem 121 Let L_1 be the language defined by the regular expression:

$$a^* b^* c^* d^*$$

Let L_2 be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AD \\ A &\rightarrow aAb \mid \lambda \\ D &\rightarrow cDd \mid \lambda \end{aligned}$$

- (a) Write 5 distinct strings that belong to L_1 but do not belong to L_2 (belong to $L_1 \setminus L_2$). If such strings do not exist, state it and explain why.

Answer: a, b, c, d, ac

- (b) Write 5 distinct strings that belong to L_2 but do not belong to L_1 (belong to $L_2 \setminus L_1$). If such strings do not exist, state it and explain why.

Answer: Impossible. To see this, note that the template for L_1 is:

$$L_1 = \{a^m b^n c^k d^\ell\}$$

while the template for L_2 is obtained from the template for L_1 by setting $m = n$ and $k = \ell$. Hence, $L_2 \subset L_1$ and $L_2 \setminus L_1$ is empty, and there are no elements to list.

- (c) Write 5 distinct strings that belong to L_1 and L_2 (belong to $L_1 \cap L_2$). If such strings do not exist, state it and explain why.

Answer: $\lambda, ab, cd, aabb, abcd$

- (d) Write 5 distinct strings over alphabet $\{a, b, c\}$ that do not belong to L_1 and do not belong to L_2 (belong to $\overline{L_1} \cap \overline{L_2}$). If such strings do not exist, state it and explain why.

Answer: ba, ca, da, cb, db

- (e) Write 5 distinct strings that belong to L_2 but do not belong to L_2^* (belong to $L_2 \setminus L_2^*$). If such strings do not exist, state it and explain why.

Answer: Impossible—for any language, say L_2 , $L_2 \subseteq L_2^*$. Hence, $L_2 \setminus L_2^*$ is empty, and there are no elements to list.

- (f) Write 5 distinct strings that belong to $L_1 L_2$ but do not belong to $L_2 L_1$ (belong to $L_1 L_2 \setminus L_2 L_1$). If such strings do not exist, state it and explain why.

Answer: $aab, bab, cab, dab, acab$

Problem 122 Let L_1 be the language accepted by the finite automaton given on Figure 1.

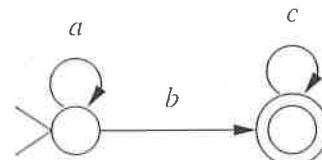


Figure 1:

Let L_2 be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSc \mid b \mid bb$$

- (a) Write 5 distinct strings that belong to L_1 but do not belong to L_2 (belong to $L_1 \setminus L_2$). If such strings do not exist, state it and explain why.

Answer:

$$ab, bc, aab, aabc, abcc$$

- (b) Write 5 distinct strings that belong to L_2 but do not belong to L_1 (belong to $L_2 \setminus L_1$). If such strings do not exist, state it and explain why.

Answer:

$$bb, abbc, aabbcc, aaabbccc, aaaabcccc$$

- (c) Write 5 distinct strings that belong to L_1 and L_2 (belong to $L_1 \cap L_2$). If such strings do not exist, state it and explain why.

Answer:

$$b, abc, aabcc, aaabccc, aaaabcccc$$

- (d) Write 5 distinct strings over alphabet $\{a, b, c\}$ that do not belong to L_1 and do not belong to L_2 (belong to $\overline{L_1} \cap \overline{L_2}$). If such strings do not exist, state it and explain why.

Answer:

$$\lambda, bbb, ba, cb, ca$$

- (e) Write 5 distinct strings that belong to L_2^* but do not belong to L_1 (belong to $L_2^* \setminus L_1$). If such strings do not exist, state it and explain why.

Answer:

$$bbb, bbbb, abcb, babc, bbabc$$

- (f) Write 5 distinct strings that belong to $L_1 L_2$ but do not belong to $L_2 L_1$ (belong to $L_1 L_2 \setminus L_2 L_1$). If such strings do not exist, state it and explain why.

Answer:

$$abb, cbb, acbb, aabb, ababc$$

Problem 123 Let L be the language defined by the regular expression:

$$((a \cup bc)^* \cup (b \cup ac)^*)aaa$$

- (a) Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow Daaa \\ D &\rightarrow A \mid B \\ A &\rightarrow AA \mid \lambda \mid a \mid bc \\ B &\rightarrow BB \mid \lambda \mid b \mid ac \end{aligned}$$

- (b) Does there exist an algorithm to convert an arbitrary regular expression into an equivalent context-free grammar? Explain your answer briefly.

Answer: Yes. The construction is performed by a recursive decomposition of the regular expression into two expressions connected by the outer-most regular operator. If such two expressions, R_1 and R_2 , are derived by two disjoint variable sets from start symbols S_1 and S_2 , then their composition is derived from a new start symbol S and additional rules which are:

$$\begin{aligned} S &\rightarrow S_1 \mid S_2 && \text{for union } R_1 \cup R_2 \\ S &\rightarrow S_1 S_2 && \text{for concatenation } R_1 \cdot R_2 \\ S &\rightarrow SS \mid \lambda \mid S_1 && \text{for For Kleene star } R_1^* \end{aligned}$$

In the base case, \emptyset is derived by an empty rule set, empty string by a rule of the form $S \rightarrow \lambda$, and individual letters by rules of the form $S \rightarrow a$.

Problem 124 (a) Let L_1 be the language defined by the regular expression:

$$(a \cup b)(ca \cup da)^*(gh)^*(a \cup d)$$

Write a complete formal definition of a context-free grammar G_1 that generates language L_1 . If such grammar does not exist, explain why.

Answer: $G_1 = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, g, h\}$, $V = \{S, A, B, D, E\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ABDE \\ A &\rightarrow a \mid b \\ E &\rightarrow a \mid d \\ B &\rightarrow BB \mid \lambda \mid ca \mid da \\ D &\rightarrow DD \mid \lambda \mid gh \end{aligned}$$

(b) Let L_2 be the language defined by the regular expression:

$$b^*(aa)b^* \cup ddc^*$$

Write a complete formal definition of a context-free grammar G_2 that generates language L_2 . If such grammar does not exist, explain why.

Answer: $G_2 = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow BaaB \mid ddD \\ B &\rightarrow BB \mid \lambda \mid b \\ D &\rightarrow DD \mid \lambda \mid c \end{aligned}$$

Problem 125 Let L be the language defined by the regular expression:

$$(a \cup b \cup dd)^*cc \cup b(dca)^*d$$

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$ is the set of terminals; $V = \{S, A, B\}$ is the set of variables; S is the start symbol, and the production set P is:

$$\begin{aligned} S &\rightarrow Acc \mid bBd \\ A &\rightarrow \lambda \mid AA \mid a \mid b \mid dd \\ B &\rightarrow dcaB \mid \lambda \end{aligned}$$

Problem 126 (a) Let L_1 be the language defined by the regular expression:

$$(ab \cup cb)^* b d^* a \cup (aa)^*$$

Write a complete formal definition of a context-free grammar G_1 that generates language L_1 . If such a grammar does not exist, state it and explain why.

Answer: $G_1 = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D, E\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow A \mid B \\ A &\rightarrow DbEa \\ D &\rightarrow \lambda \mid DD \mid ab \mid cb \\ E &\rightarrow dE \mid \lambda \\ B &\rightarrow aaB \mid \lambda \end{aligned}$$

(b) Let L_2 be the language defined by the regular expression:

$$(ab \cup c b^* b d^*)(a \cup (aa)^*)$$

Write a complete formal definition of a context-free grammar G_2 that generates language L_2 . If such a grammar does not exist, state it and explain why.

Answer: $G_2 = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D, E, F\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow ab \mid cEbF \\ E &\rightarrow bE \mid \lambda \\ F &\rightarrow dF \mid \lambda \\ B &\rightarrow a \mid D \\ D &\rightarrow aaD \mid \lambda \end{aligned}$$

Problem 127 (a) Let S_1 be the language defined by the regular expression:

$$((a \cup b)(bee)^* \cup c)^*(dd \cup d)$$

Write a complete formal definition of a context-free grammar G_1 that generates language S_1 . If such a grammar does not exist, explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, e\}$, $V = \{S, A, B, D, E, F\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow AA \mid \lambda \mid c \mid D \\ D &\rightarrow EF \\ E &\rightarrow a \mid b \\ F &\rightarrow FF \mid \lambda \mid bee \\ B &\rightarrow dd \mid d \end{aligned}$$

(b) Let S_2 be the language defined by the regular expression:

$$(b \cup bb \cup bbb)^*(a \cup c)^*$$

Write a complete formal definition of a context-free grammar G_2 that generates language S_2 . If such a grammar does not exist, explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow AA \mid \lambda \mid b \mid bb \mid bbb \\ B &\rightarrow BB \mid \lambda \mid a \mid c \end{aligned}$$

Problem 128 (a) Let L_1 be the language defined by the regular expression:

$$(dd(a \cup b \cup cc)^*) \cup (db^* \cup (ca)^*b)$$

Write a complete formal definition of a context-free grammar that generates L_1 . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, E, F, H\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow A \mid B \\ A &\rightarrow ddE \\ E &\rightarrow \lambda \mid EE \mid a \mid b \mid cc \\ B &\rightarrow dF \mid Hb \\ F &\rightarrow bF \mid \lambda \\ H &\rightarrow caH \mid \lambda \end{aligned}$$

(b) Let L_2 be the language defined by the regular expression:

$$(dd \cup (ab \cup cc)^*) (db^* \cup (ca)^*b)$$

Write a complete formal definition of a context-free grammar that generates L_2 . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, E, F, H\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow dd \mid E \\ E &\rightarrow \lambda \mid EE \mid ab \mid cc \\ B &\rightarrow dF \mid Hb \\ F &\rightarrow bF \mid \lambda \\ H &\rightarrow caH \mid \lambda \end{aligned}$$

Problem 129 Let L be the language defined by the regular expression:

$$(c \cup cc)(ab)^*(bb \cup cc)^*$$

(a) Write 5 distinct strings that belong to L .

(b) Write 5 distinct strings over alphabet $\{a, b, c\}$ that do not belong to L .

Answer:

$\in L$	$\notin L$
c	a
cc	b
cabab	ab
cbb	bb
ccabccccbb	bcb

(c) Write a complete formal definition of a context-free grammar G that generates L .

Answer: $G = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b, c\}$, $V = \{S, A, B, D\}$, and P comprises:

$$\begin{aligned} S &\rightarrow ABD \\ A &\rightarrow c \mid cc \\ B &\rightarrow abB \mid \lambda \\ D &\rightarrow bbD \mid ccD \mid \lambda \end{aligned}$$

Problem 130 Let L be the language defined by the regular expression:

$$(a \cup b)^*(c^* \cup d^*)$$

Write a complete formal definition of a context-free grammar G that generates L . If such a grammar does not exist, explain why.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c, d\}$, $V = \{S, L, R, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow LR \\ L &\rightarrow aL \mid bL \mid \lambda \\ R &\rightarrow B \mid D \\ B &\rightarrow cB \mid \lambda \\ D &\rightarrow dD \mid \lambda \end{aligned}$$

Problem 131 Write a complete formal definition of a context-free grammar G that generates the language defined by the regular expression:

$$d(ab \cup bc)^*(d \cup c)^*a$$

If such grammar G does not exist, explain why.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c, d\}$ is the set of terminals;

$V = \{S, A, B\}$ is the set of variables;

S is the start symbol;

and the set of productions P comprises:

$$\begin{aligned} S &\rightarrow dABA \\ A &\rightarrow abA \mid bcA \mid \epsilon \\ B &\rightarrow dB \mid cB \mid \epsilon \end{aligned}$$

Problem 132 (a) Write a complete formal definition of a context-free grammar G over alphabet $\{a, b, c, d\}$ that generates the language defined by the regular expression:

$$(a \cup b)^*(cc \cup dddd)^*$$

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$ is the set of terminals; $V = \{S, A, B\}$ is the set of variables; S is the start symbol; and the set of productions P comprises:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow AA \mid a \mid b \mid \lambda \\ B &\rightarrow BB \mid cc \mid dddd \mid \lambda \end{aligned}$$

(b) Is the grammar G which you constructed in your answer to part (a) regular? Justify briefly your answer.

Answer: No. The right-hand side of the production $B \rightarrow dddd$ has as many as 4 terminals, whereas the right-hand side of every production in a regular grammar contains at most one terminal, possibly followed by exactly one variable.

Problem 133 (a) Let S_1 be the language defined by the regular expression:

$$(ab \cup b)^* \cup (c^*(d \cup b)^*)$$

Write 5 distinct strings that belong to S_1 . If such strings do not exist, explain why.

Answer:

$$\lambda, ab, b, cd, cb$$

(b) Write a complete formal definition of a context-free grammar G_1 that generates language S_1 . If such a grammar does not exist, explain why.

Answer: $G_1 = (V, \Sigma, P_1, S)$, where

$\Sigma = \{a, b, c, d\}$, $V = \{S, A_1, A_2, D, E\}$, and P_1 is:

$$\begin{aligned} S &\rightarrow A_1 \mid A_2 \\ A_1 &\rightarrow A_1 A_1 \mid \lambda \mid ab \mid b \\ A_2 &\rightarrow DE \\ D &\rightarrow cD \mid \lambda \\ E &\rightarrow EE \mid \lambda \mid d \mid b \end{aligned}$$

(c) Let S_2 be the language defined by the regular expression:

$$(a \cup d)^*(abc \cup bcd \cup da)^*$$

Write 5 distinct strings that belong to S_2 . If such strings do not exist, explain why.

Answer:

$$\lambda, a, d, abc, addabcdabcddada$$

(d) Write a complete formal definition of a context-free grammar G_2 that generates language S_2 . If such a grammar does not exist, explain why.

Answer: $G_2 = (V, \Sigma, P_2, S)$, where

$\Sigma = \{a, b, c, d\}$, $V = \{S, A_1, A_2\}$, and P_2 is:

$$\begin{aligned} S &\rightarrow A_1 A_2 \\ A_1 &\rightarrow A_1 A_1 \mid \lambda \mid a \mid d \\ A_2 &\rightarrow A_2 A_2 \mid \lambda \mid abc \mid bcd \mid da \end{aligned}$$

Problem 134 Let L be the language defined by the regular expression:

$$((a \cup b^*)^* (c \cup cd)^* (c \cup (ac)^*)^*)^*$$

Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D, E, F\}$,

and the production set P is:

$$\begin{aligned} S &\rightarrow SS \mid \lambda \mid ABD \\ A &\rightarrow AA \mid \lambda \mid a \mid E \\ E &\rightarrow bE \mid \lambda \\ B &\rightarrow BB \mid \lambda \mid c \mid cd \\ D &\rightarrow DD \mid \lambda \mid c \mid F \\ F &\rightarrow FF \mid \lambda \mid ac \end{aligned}$$

Problem 135 Let L be the language defined by the regular expression:

$$(ab \cup dd)^* \cup (c \cup a^* \cup acc)^*$$

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow A \mid B \\ A &\rightarrow AA \mid \lambda \mid ab \mid dd \\ B &\rightarrow BB \mid \lambda \mid c \mid acc \mid D \\ D &\rightarrow aD \mid \lambda \end{aligned}$$

Problem 136 (a) Let L_1 be the language defined by the regular expression:

$$(a \cup b)(ca \cup da)^*(gh)^*(a \cup d)$$

Write a complete formal definition of a context-free grammar G_1 that generates language L_1 . If such a grammar does not exist, explain why.

Answer: $G_1 = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, g, h\}$, $V = \{S, A, B, D, E\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ABDE \\ A &\rightarrow a \mid b \\ E &\rightarrow a \mid d \\ B &\rightarrow BB \mid \lambda \mid ca \mid da \\ D &\rightarrow DD \mid \lambda \mid gh \end{aligned}$$

(b) Let L_2 be the language defined by the regular expression:

$$b^*(aa)b^* \cup ddc^*$$

Write a complete formal definition of a context-free grammar G_2 that generates language L_2 . If such a grammar does not exist, explain why.

Answer: $G_2 = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow BaaB \mid ddD \\ B &\rightarrow BB \mid \lambda \mid b \\ D &\rightarrow DD \mid \lambda \mid c \end{aligned}$$

Problem 137 (a) Let S_1 be the language defined by the regular expression:

$$ab(be \cup e)^* \cup d(ba)^*d(\lambda \cup a)(baa)^*$$

Write a complete formal definition of a context-free grammar G_1 that generates language S_1 . If such a grammar does not exist, explain why.

Answer: $G_1 = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, e, d\}$, $V = \{S, A, B, D, E, F, H\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow A \mid B \\ A &\rightarrow abD \\ D &\rightarrow DD \mid \lambda \mid be \mid e \\ B &\rightarrow dEdFH \\ E &\rightarrow EE \mid \lambda \mid ba \\ F &\rightarrow \lambda \mid a \\ H &\rightarrow HH \mid \lambda \mid baa \end{aligned}$$

(b) Let S_2 be the language defined by the regular expression:

$$(ba \cup baa)^*(eb \cup eeb)^* \cup d(a \cup b)^*dda^*$$

Write a complete formal definition of a context-free grammar G_2 that generates language S_2 . If such a grammar does not exist, explain why.

Answer: $G_2 = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, e, d\}$, $V = \{S, A, B, D, E\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \mid dDddE \\ A &\rightarrow AA \mid \lambda \mid ba \mid baa \\ B &\rightarrow BB \mid \lambda \mid eb \mid eeb \\ D &\rightarrow DD \mid \lambda \mid a \mid b \\ E &\rightarrow EE \mid \lambda \mid a \end{aligned}$$

Problem 138 Let L be the language defined by the regular expression:

$$g(ab^* \cup b^*d)^*g \cup hh(a^*b \cup ba^*)$$

Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, d, g, h\}$, $V = \{S, A, B, D, E, F, H\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow gAg \mid hhB \\ A &\rightarrow AA \mid \lambda \mid D \mid E \\ D &\rightarrow Db \mid a \\ E &\rightarrow bE \mid d \\ B &\rightarrow F \mid H \\ F &\rightarrow aF \mid b \\ H &\rightarrow Ha \mid b \end{aligned}$$

Problem 139 Let L be the language defined by the regular expression:

$$(a \cup cc)^*(b \cup bb)cd^*$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

(b) Is L regular? Explain your answer briefly.

Problem 140 (a) Let L_1 be the language defined by the regular expression:

$$(a \cup b)(eb \cup ec \cup ecb)^*(b \cup c)^*(a \cup d)$$

Write a complete formal definition of a context-free grammar G_1 that generates language L_1 . If such grammar does not exist, explain why.

Answer: $G_1 = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c, d, e\}$, $V = \{S, A, B, D, E\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ABDE \\ A &\rightarrow a \mid b \\ B &\rightarrow BB \mid \lambda \mid eb \mid ec \mid ecb \\ D &\rightarrow DD \mid \lambda \mid b \mid c \\ E &\rightarrow a \mid d \end{aligned}$$

(b) Let L_2 be the language defined by the regular expression:

$$(a^*(ba)b^* \cup eee^* \cup cac \cup aca)^*$$

Write a complete formal definition of a context-free grammar G_2 that generates language L_2 . If such grammar does not exist, explain why.

Answer: $G_2 = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c, e\}$, $V = \{S, A, B, E\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow SS \mid \lambda \mid AbaB \mid eeE \mid cac \mid aca \\ A &\rightarrow aA \mid \lambda \\ B &\rightarrow bB \mid \lambda \\ E &\rightarrow eE \mid \lambda \end{aligned}$$

Problem 141 (a) Let L be the language defined by the regular expression:

$$(a(ca \cup da)^* b) \cup (b(ca \cup da)^* a)$$

Write a complete formal definition of a context-free grammar G_1 that generates language L . If such grammar does not exist, explain why.

Answer: $G_1 = (V_1, \Sigma, P_1, S)$, where $\Sigma = \{a, b, c, d\}$, $V_1 = \{S, A, B, D\}$, and the production set P_1 is:

$$\begin{aligned} S &\rightarrow A \mid B \\ A &\rightarrow aDb \\ B &\rightarrow bDa \\ D &\rightarrow DD \mid \lambda \mid ca \mid da \end{aligned}$$

(b) Let L be the language defined in part (a).

Write a complete formal definition of a context-free grammar G_2 that generates language L^* . If such grammar does not exist, explain why.

Answer: $G_2 = (V_2, \Sigma, P_2, S_2)$, where $\Sigma = \{a, b, c, d\}$, $V_2 = \{S_2, S, A, B, D\}$, and the production set P_2 is:

$$\begin{aligned} S_2 &\rightarrow S_2S_2 \mid \lambda \mid S \\ S &\rightarrow A \mid B \\ A &\rightarrow aDb \\ B &\rightarrow bDa \\ D &\rightarrow DD \mid \lambda \mid ca \mid da \end{aligned}$$

Problem 142 Let L be the language defined by the regular expression:

$$(c \cup ba)^* bcb(b \cup ddd)^* \cup a(ab)^* d$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D, E, F, H\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow A \mid B \\ A &\rightarrow EFH \\ E &\rightarrow EE \mid \lambda \mid c \mid ba \\ F &\rightarrow bcb \\ H &\rightarrow HH \mid \lambda \mid b \mid ddd \\ B &\rightarrow aDd \\ D &\rightarrow abD \mid \lambda \end{aligned}$$

(b) Is the complement of L regular? Prove your answer.

Answer: Yes— L itself is regular, since it has a regular expression, and the complement of every regular language is regular.

Problem 143 Let L be the set of strings over alphabet $\{0, 1\}$ in which every 0 is followed immediately by 111.

1. Write a regular expression that defines L . If such regular expression does not exist, explain why.

Answer:

$$(1 \cup 0111)^*$$

2. Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{0, 1\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow SS \mid \lambda \mid 1 \mid 0111$$

3. What is the cardinality of L ? (If possible, state the exact number. Otherwise, specify if the set is countable or not.) Explain your answer briefly.

Answer:

$$|L| = \aleph_0$$

L is infinite and countable. To see that it is infinite, observe that it is the Kleene star of a non-empty set. It is countable, since it is a subset of the set of all finite strings over $\{0, 1\}$, which is countable.

4. What is the cardinality of $\mathcal{P}(L)$ —the set of subsets of L ? (If possible, state the exact number. Otherwise, specify if the set is countable or not.) Explain your answer briefly.

Answer: $\mathcal{P}(L)$ is uncountable—by Cantor's diagonal argument, the set of subsets of an infinite countable set is uncountable.

Problem 144 Let L be the set of strings over alphabet $\Sigma = \{a, b, c\}$ defined as follows:

$$L = \{a^m b^k c^\ell \mid m, k, \ell \geq 0 \wedge m = k + \ell\}$$

1. Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, explain why.

Answer: The general template for strings in L is:

$$a^\ell a^k b^k c^\ell \text{ for } k, \ell \geq 0$$

whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aSc \mid B \\ B &\rightarrow aBb \mid \lambda \end{aligned}$$

2. List six different strings that belong to \overline{L} (where $\overline{L} = \Sigma \setminus L$). If this is impossible, explain why.

Answer:

$$cba, ca, cb, ba, abab, abca$$

3. List six different strings that belong to

$$\overline{L} \cap a^* b^* c^*$$

(where $\overline{L} = \Sigma \setminus L$). If this is impossible, explain why.

Answer:

$$a, b, c, aaabc, abc, acc$$

4. State the cardinalities of L and \overline{L} , and determine which one is greater (if any.) If possible, give exact numbers, otherwise state if sets are countable or not. Explain your answer briefly.

Answer: L and \overline{L} are infinite and countable:

$$|L| = |\overline{L}| = \aleph_0$$

To see that L is infinite, observe that it contains, for example, a word $(aa)^n b^n c^n$ for every $n \geq 0$. To see that \overline{L} is infinite, note that it contains, for example, aa^* . L and \overline{L} are countable, since every language is countable.

Problem 145 (a) Let:

$$\Sigma = \{a, b, c\}$$

and let L_1 be defined as follows:

$$L_1 = \{a^m b^{m+1} c \mid m \geq 0\}$$

Write a regular expression that defines L_1 . If such expression does not exist, explain why.

Answer: Such regular expression does not exist, because L_1 is not regular.

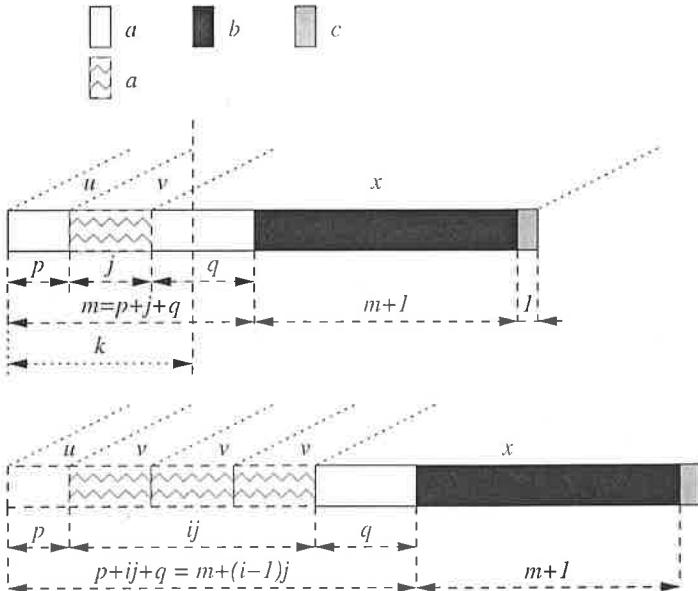


Figure 2:

Assume the opposite. Let k be the constant as in the Pumping Lemma for L_1 . Let $w = a^m b^{m+1} c$, such that $m > k$. Let $w = uvx$ be a decomposition of w , where v is the pumping part, as on Figure 2. By the Lemma, it must be that $|uv| < k < m$. Hence, $v = a^j$ for some $j > 0$. By the pumping, for every $i \geq 0$, we have: $uv^i x \in L_1$. However:

$$uv^i x = uvv^{i-1} x = uv a^{(i-1)j} x = a^{m+(i-1)j} b^{m+1} c$$

Since:

$$m + (i-1)j > m \text{ whenever } i \geq 2$$

we have:

$$a^{m+(i-1)j} b^{m+1} c \notin L_1$$

whence the contradiction.

(b) Let:

$$\Sigma = \{a, b, c\}$$

and let L_2 be defined as follows:

$$L_2 = \{a^m b^n c^k a^{n+1} c^{m+2} b^j \mid k \geq 0, m \geq 0, n \geq 0, j \geq 0\}$$

Write a complete formal definition of a context-free grammar G that generates L_2 . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c\}$, $V = \{S, J, M, N, K\}$, and P is:

$$\begin{aligned} S &\rightarrow MJ \\ J &\rightarrow bJ \mid \lambda \\ M &\rightarrow aMc \mid Ncc \\ N &\rightarrow bNa \mid Ka \\ K &\rightarrow cK \mid \lambda \end{aligned}$$

Problem 146 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \mid AD \mid DB \\ A &\rightarrow Aa \mid d \\ B &\rightarrow bB \mid d \\ D &\rightarrow cD \mid d \end{aligned}$$

(a) Write 5 distinct strings that belong to L . If such strings do not exist, state it and explain why.

Answer:

$$dd, dad, dabd, dc, cdd$$

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$da^*b^*d \cup da^*c^*d \cup c^*db^*d$$

Problem 147 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \mid AD \\ A &\rightarrow Aa \mid \lambda \\ B &\rightarrow BB \mid \lambda \mid b \\ D &\rightarrow cDc \mid d \end{aligned}$$

(a) Write 5 distinct strings that belong to L . If such strings do not exist, state it and explain why.

Advice for Answer: First, observe that all strings of L belong to one of the following templates:

$$a^*b^*$$

or

$$a^n c^j d c^j$$

for some $n, j \geq 0$.

Answer:

$$\lambda, a, b, d, acdc$$

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: This regular expression does not exist, since L is not regular. To prove this, first observe that the general template mandates that all those strings that contain c 's contain an even number of c 's in a consecutive run, except for a single d in the middle.

Assume the opposite—that L is regular. Let k be the constant as in the Pumping Lemma for L . Consider a word $w = c^p dc^p$, where $p > k$, obtained from the general template for L by setting $n = 0$.

Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < k < p$. Hence, the pumping part v is entirely within the c -segment, meaning that $v = c^m$ for some m such that $0 < m < k$. After pumping once, we obtain a word: $w_1 = c^{m+n} dc^p$. However, since $m > 0$ (as the length of the pumping substring), we conclude that $m + p > p$, meaning that there are more c 's to the left of the single d than to the right of it. This means that $w_1 \notin L$, whence the contradiction.

Problem 148 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, e\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \mid D \\ A &\rightarrow aA \mid b \\ B &\rightarrow Bcc \mid d \\ D &\rightarrow DD \mid \lambda \mid eee \end{aligned}$$

Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$a^* bd (cc)^* \cup (eee)^*$$

Problem 149 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow A \mid DBA \\ A &\rightarrow aA \mid cb \\ B &\rightarrow BB \mid \lambda \mid db \\ D &\rightarrow Db \mid c \end{aligned}$$

(a) List 5 distinct strings that belong to L . If such strings do not exist, state it and explain why.

Advice for Answer: See part (b).

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$a^*cb \cup cb^* (db)^* a^*cb$$

Problem 150 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow ABD \mid BA \\ A &\rightarrow Aa \mid bc \\ B &\rightarrow BB \mid \lambda \mid bd \\ D &\rightarrow ccD \mid a \end{aligned}$$

(a) List 5 distinct strings that belong to L . If such strings do not exist, state it and explain why.

Advice for Answer: See part (b).

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$bca^* (bd)^* (cc)^* a \cup (bd)^* bca^*$$

Problem 151 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, e\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow BA \mid DB \\ A &\rightarrow aaA \mid b \\ B &\rightarrow Bcc \mid d \\ D &\rightarrow DD \mid \lambda \mid eee \end{aligned}$$

Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$d(cc)^*(aa)^*b \cup (eee)^*d(cc)^*$$

Problem 152 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \mid D \\ A &\rightarrow AA \mid \lambda \mid abc \\ B &\rightarrow bB \mid \lambda \\ D &\rightarrow aaD \mid bbbD \mid cD \mid \lambda \end{aligned}$$

Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$(abc)^*b^* \cup (aa \cup bbb \cup c)^*$$

Problem 153 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D\}$, and P comprises:

$$\begin{aligned} S &\rightarrow SS \mid A \mid \lambda \\ A &\rightarrow BD \\ B &\rightarrow a \mid b \\ D &\rightarrow ccc \end{aligned}$$

- (a) Write 5 distinct strings that belong to L .
- (b) Write 5 distinct strings over alphabet $\{a, b, c\}$ that do not belong to L .

Answer:

$\in L$	$\notin L$
λ	a
acc	b
bcc	c
$acccaccc$	ac
$bcccaccc$	aa

- (c) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$((a \cup b) ccc)^*$$

Problem 154 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$ is the set of terminals; $V = \{S, A, B, T\}$ is the set of variables; S is the start symbol; and the set of productions P comprises:

$$\begin{aligned} S &\rightarrow TA \\ T &\rightarrow TT \mid Bd \mid \lambda \\ B &\rightarrow BB \mid bc \mid \lambda \\ A &\rightarrow aa \end{aligned}$$

Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$((bc)^*d)^*aa.$$

Problem 155 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$; $V = \{S, A\}$; and the set of productions P is:

$$\begin{aligned} S &\rightarrow aS \mid bS \mid cS \mid dA \\ A &\rightarrow AA \mid \lambda \mid a \mid b \mid c \end{aligned}$$

Write a regular expression that defines L . If such a regular expression does not exist, explain why.

Answer:

$$(a \cup b \cup c)^*d(a \cup b \cup c)^*$$

Problem 156 Let L_1 be the language defined over alphabet $\Sigma = \{a, b\}$ by the regular expression:

$$(a \cup bb)^*$$

Let L_2 be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSbb \mid \lambda$$

1. Write a complete formal definition of a context-free grammar G_1 that generates language L_1 . If such grammar does not exist, explain why.

Answer: $G_1 = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow SS \mid \lambda \mid a \mid bb$$

2. Write a complete formal definition of a context-free grammar G_2 that generates language L_2L_1 . If such grammar does not exist, explain why.

Answer: $G_2 = (V, \Sigma, P, T)$, where $\Sigma = \{a, b\}$, $V = \{S, T\}$, and the production set P is:

$$\begin{aligned} T &\rightarrow SS \\ S &\rightarrow aSbb \mid \lambda \end{aligned}$$

3. List six different strings that belong to $L_1 \setminus L_2$. If this is impossible, explain why.

Answer:

$$a, bb, bba, aa, bbbb, bbaabbbbaaababb$$

4. List six different strings that belong to $L_2 \setminus L_1$. If this is impossible, explain why.

Answer: Impossible, since $L_2 \setminus L_1 = \emptyset$. All strings in L_2 are of the form:

$$a^m(bb)^m \text{ for } m \geq 0$$

and each can be expressed as a concatenation of strings a and bb . Language L_1 is exactly the set of all such concatenations.

5. List six different strings that belong to $L_2L_2 \setminus L_2$. If this is impossible, explain why.

Answer:

$abbabb, abbaabbbb, aabbbbabb,$
 $aabbbbaabbbb, abbaaabbbbb, aaabbbbbbabb$

Problem 157 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \mid BA \\ A &\rightarrow ab \\ B &\rightarrow BB \mid \lambda \mid a \mid b \mid c \end{aligned}$$

- (a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$ab(a \cup b \cup c)^* \cup (a \cup b \cup c)^*ab$$

- (b) Is \overline{L} (the complement of L) context-free? Explain your answer.

Answer: Yes. The complement of a regular language is regular, so \overline{L} is regular. Every regular language is context-free, so \overline{L} is context-free.

Problem 158 (a) Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, B, D, E\}$, and P is:

$$\begin{aligned} S &\rightarrow SS \mid \lambda \mid B \\ B &\rightarrow cD \mid bbD \mid aE \mid b \\ D &\rightarrow aD \mid cD \mid \lambda \\ E &\rightarrow EE \mid \lambda \mid b \end{aligned}$$

Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(c(a \cup c)^* \cup bb(a \cup c)^* \cup ab^* \cup b)^*$$

- (b) Let \mathcal{R} be the class of languages that can be represented by a regular expression, and let \mathcal{C} be the class of languages that can be represented by a context-free grammar. State the cardinalities of \mathcal{R} and \mathcal{C} , and compare them.

Answer: Classes \mathcal{R} and \mathcal{C} have equal cardinalities; both of them are infinite and countable:

$$|\mathcal{R}| = |\mathcal{C}| = \aleph_0$$

Problem 159 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow ABDB \mid BABD \\ A &\rightarrow AA \mid \lambda \mid ba \\ B &\rightarrow ccB \mid \lambda \\ D &\rightarrow acAD \mid aacaaD \mid \lambda \end{aligned}$$

- (a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$\begin{aligned} (ba)^*(cc)^*(aca \cup aacaa)^*(cc)^* \\ \cup \\ (cc)^*(ba)^*(cc)^*(aca \cup aacaa)^* \end{aligned}$$

- (b) Is the grammar G regular? Prove your answer.

Answer: No. In a regular grammar, the right-hand side of any production is either empty or consists of a single terminal, possibly followed by a single non-terminal. All but three of the productions in the grammar G violate this rule.

Problem 160 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D, E\}$, and P is:

$$\begin{aligned} S &\rightarrow ABD \mid E \\ A &\rightarrow \lambda \mid AA \mid aa \mid cc \\ B &\rightarrow bB \mid b \\ D &\rightarrow acD \mid aaa \\ E &\rightarrow aE \mid \lambda \end{aligned}$$

Write a regular expression that represents L . If such a regular expression does not exist, prove it.

Answer:

$$(aa \cup cc)^*b^*b(ac)^*aaa \cup a^*$$

Problem 161 (a) Let L_1 be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, g\}$, $V = \{S, A, B, D, E\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \mid BD \mid E \\ A &\rightarrow aaA \mid \lambda \\ B &\rightarrow BB \mid \lambda \mid bb \mid bc \\ D &\rightarrow aD \mid cD \mid \lambda \\ E &\rightarrow gE \mid bbb \end{aligned}$$

Write a regular expression that represents L_1 . If such a regular expression does not exist, prove it.

Answer:

$$(aa)^*(bb \cup bc)^* \cup (bb \cup bc)^*(a \cup c)^* \cup g^*bbb$$

Problem 162 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, e\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \mid BD \\ A &\rightarrow aaA \mid b \\ B &\rightarrow Bcc \mid d \\ D &\rightarrow DD \mid \lambda \mid eee \end{aligned}$$

Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(aa)^* bd (cc)^* \cup d (cc)^* (eee)^*$$

Problem 163 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow A \mid B \\ A &\rightarrow aAa \mid b \\ B &\rightarrow cBc \mid d \end{aligned}$$

(a) List 5 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: The general template is:

$$a^nba^n \cup c^jdc^j, n, j \geq 0$$

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Advice for Answer: The language is not regular. Pumping is observed in a word:

$$w = a^nba^n$$

obtained from the general template by selecting a word from the first term in the union, where n is selected so as to exceed the pumping constant. Every word in L that does not contain any c 's or d 's must honor the template of w , for some n ; pumping, however, breaks the template.

Problem 164 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow A \mid B \\ A &\rightarrow Aaa \mid b \\ B &\rightarrow ccB \mid d \end{aligned}$$

(a) List 5 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: See part (b).

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$b(aa)^* \cup (cc)^*d$$

Problem 165 Let L_1 be a language over the alphabet $\{a, b, c, d\}$, defined as follows:

$$L_1 = \{a^m c^j b^{4m}, m, j \geq 0\}$$

Let L_2 be a language over the alphabet $\{a, b, c, d\}$, defined as follows:

$$L_2 = \{c^{k+1}d^{k+2}, k \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar that generates the language $L_1 L_2$. If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, S_1, S_2, K\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow S_1 S_2 \\ S_1 &\rightarrow aS_1 bbbb \mid K \\ K &\rightarrow cK \mid \lambda \\ S_2 &\rightarrow cS_2 d \mid cdd \end{aligned}$$

(b) Write a complete formal definition of a context-free grammar that generates the language $L_1 \cup L_2$. If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, S_1, S_2, K\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow S_1 \mid S_2 \\ S_1 &\rightarrow aS_1 bbbb \mid K \\ K &\rightarrow cK \mid \lambda \\ S_2 &\rightarrow cS_2 d \mid cdd \end{aligned}$$

(c) Write a complete formal definition of a context-free grammar that generates the language $(L_1)^*$. If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, S_1, K\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow \lambda \mid SS \mid S_1 \\ S_1 &\rightarrow aS_1 bbbb \mid K \\ K &\rightarrow cK \mid \lambda \end{aligned}$$

(d) Write a complete formal definition of a context-free grammar that generates the language $(L_2)^*$. If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, S_2\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow \lambda \mid SS \mid S_2 \\ S_2 &\rightarrow cS_2 d \mid cdd \end{aligned}$$

Problem 166 Let:

$$L = \{d^{2n}b^j a^{2p}c^{\ell+1} \mid n = p \wedge j = \ell \wedge n, j, p, \ell \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: This grammar does not exist, since L is not context-free.

To show that L is not context-free, we assume the opposite—that L is context-free and that it honors the Pumping Lemma, with the constant k , as given in the Lemma.

The general template for strings in L is:

$$L = \{d^{2n}b^j a^{2n}c^{\ell+1} \mid n, j \geq 0\}$$

Observe that in every word of L the number of a 's is equal to the number of d 's, while the number of b 's is by exactly one less than the number of c 's. We show that pumping violates these relationships.

Select a word w such that $w = d^{2n}b^ja^{2n}c^{j+1}$ where $n > k$ and $j > k$. The word w is a concatenation of four substrings, each consisting of repetitions of a single letter. In any possible pumping decomposition of w such that $w = xy_1ty_2z$, the Lemma guarantees that:

$$|y_1ty_2| < k < n < 2n \wedge |y_1ty_2| < k < j < j + 1$$

meaning that the entire pumping window y_1ty_2 either falls within one substring formed by a single letter or spans two such adjacent substrings—the pumping window is too short to span three (or more) of these substrings.

Once the pumping is admitted, the following possibilities exist:

- If any part of d^{2n} is pumped, then a^{2n} is not pumped, and the resulting word has fewer a 's than d 's.
- If any part of b^j is pumped, then c^{j+1} is not pumped, and the resulting word no longer has fewer b 's than c 's.
- If any part of a^{2n} is pumped, then d^{2n} is not pumped, and the resulting word has fewer d 's than a 's.
- If any part of c^{j+1} is pumped, then b^j is not pumped, and the number of c 's in the resulting word exceeds the number of b 's by more than one.

In each case, at least one of the mandatory relationships is violated, meaning that the resulting word no longer belongs to L , which is a contradiction, since the Lemma guarantees that words produced by pumping remain in L .

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: This regular expression does not exist, because L is not regular. As is proved in the answer to part (a), L is not context-free, and thereby not regular, since every regular language is context-free.

Problem 167 Let:

$$L = \{d^{2n+1}a^pb^{2j+1}c^{\ell+1} \mid n=p \wedge j=\ell \wedge n, j, p, \ell \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The general template for strings in L is:

$$L = \{d^{2n+1}a^n b^{2j+1} c^{j+1} \mid n, j \geq 0\}$$

yielding the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow ddAa \mid d \\ B &\rightarrow bbBc \mid bc \end{aligned}$$

- (b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: This regular expression does not exist, since L is not regular.

To show that L is not regular, we assume the opposite—that L is regular and that it honors the Pumping Lemma, with the constant k , as given in the Lemma.

Observe that the number of a 's and the number of d 's in every word of L are related so that if the number of a 's is equal to n , then the number of d 's is equal to $2n + 1$. We show that pumping violates this relationship.

Select a word w such that $w = d^{2n+1}a^nbc$, which is obtained from the general template by setting $j = 0$. Moreover, select n such that $n > k$. In any possible pumping decomposition of w such that $w = xyz$, the Lemma guarantees that:

$$|xy| < k < n < 2n + 1$$

meaning that the pumping substring y consists entirely of d 's. Say that $y = d^m$ for some $m > 0$. By pumping once, we obtain the word

$$w_1 = xy_1yz = d^{2n+1+m}a^nbc$$

where the number of a 's is still equal to n , but the number of d 's is strictly greater than $2n + 1$, since:

$$m > 0 \implies 2n + 1 + m > 2n + 1$$

Hence, the mandatory relationship between the number of a 's and the number of d 's does not hold for the word w_1 , meaning that $w_1 \notin L$, which is a contradiction, since the Lemma guarantees that words produced by pumping remain in L .

Problem 168 Let:

$$L = \{d^{p+1}b^j a^p c^{\ell+2} \mid n = j + p \wedge n, j, p, \ell \geq 0\}$$

- (a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The general template for strings in L is:

$$L = \{d^{p+1}d^j b^j a^p c^{\ell+2} \mid j, p, \ell \geq 0\}$$

yielding the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow dAD \\ A &\rightarrow dAa \mid B \\ B &\rightarrow dBb \mid \lambda \\ D &\rightarrow cD \mid cc \end{aligned}$$

- (b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: This automaton does not exist, since L is not regular.

To show that L is not regular, we assume the opposite—that L is regular and that it honors the Pumping Lemma, with the constant k , as given in the Lemma.

Observe that the number of d 's, number of b 's and the number of a 's in every word of L are related so that the number of d 's is by one greater than the total number of b 's and a 's. We show that pumping violates this relationship.

Select a word w such that $w = d^{p+1}a^pcc$, which is obtained from the general template by setting $j = \ell = 0$. Moreover, select p such that $p > k$. In any possible pumping decomposition of w such that $w = xyz$, the Lemma guarantees that:

$$|xy| < k < p < p + 1$$

meaning that the pumping substring y consists entirely of d 's. Say that $y = d^q$ for some $q > 0$. By pumping once, we obtain the word

$$w_1 = xyzz = d^{p+1+q}a^pcc$$

where the number of a 's is still equal to p , but the number of d 's is now longer equal to $p + 1$:

$$q > 0 \implies p + 1 + q > p + 1$$

Hence, the mandatory relationship between the number of d 's and the number of a 's (and b 's) does not hold for the word w_1 , meaning that $w_1 \notin L$, which is a contradiction, since the Lemma guarantees that words produced by pumping remain in L .

Problem 169 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \mid BA \mid D \\ A &\rightarrow aA \mid c \\ B &\rightarrow BB \mid \lambda \mid b \\ D &\rightarrow Dcc \mid a \end{aligned}$$

(a) Write 5 distinct strings that belong to L . If such strings do not exist, state it and explain why.

Answer:

$c, ac, acb, bbac, acc$

(b) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$a^*cb^* \cup b^*a^*c \cup a(cc)^*$$

(c) Is \overline{L} (the complement of L) regular? Explain your answer.

Answer: Yes. L is regular, as is claimed in the answer to part (b), and every regular language has a regular complement.

Problem 170 Let:

$$L = \{d^{n+1}b^ja^pc^{\ell+1} \mid n < \ell \wedge j > p \wedge n, j, p, \ell \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: To clarify the general template for strings of L , observe that the condition $n < \ell$ is equivalent to $n + 1 \leq \ell$, which is in turn equivalent to $\ell = n + 1 + m$, for some $m \geq 0$. Hence, the general template for strings in L is:

$$L = \{d^{n+1}b^{p+i+1}a^pc^{n+m+2} \mid n, p, i, m \geq 0\}$$

or, equivalently,

$$L = \{d^{n+1}b^pb^{i+1}a^pc^{n+1}c^{m+1} \mid n, p, i, m \geq 0\}$$

yielding the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D, E\}$, and P is:

$$\begin{aligned} S &\rightarrow AD \\ A &\rightarrow dAc \mid dBc \\ B &\rightarrow bBa \mid E \\ E &\rightarrow bE \mid b \\ D &\rightarrow cD \mid c \end{aligned}$$

(b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: This automaton does not exist, since L is not regular.

To show that L is not regular, we assume the opposite—that L is regular and that it honors the Pumping Lemma, with the constant k , as given in the Lemma.

Observe that the number of d 's and the number of c 's in every word of L are related so that if the number of d 's is equal to $n + 1$, then the number of c 's is greater than $n + 1$. We show that pumping violates this relationship.

Select a word w such that $w = d^{n+1}bc^{n+1}c$, which is obtained from the general template by setting $p = i = m = 0$. Moreover, select n such that $n > k$. In any possible pumping decomposition of w such that $w = xyz$, the Lemma guarantees that:

$$|xy| < k < n < n + 1$$

meaning that the pumping substring y consists entirely of d 's. Say that $y = d^q$ for some $q > 0$. By pumping once, we obtain the word

$$w_1 = xyzz = d^{n+1+q}bc^{n+1}c$$

where the number of c 's is equal to $n + 2$, but the number of d 's is now longer less than $n + 2$, since:

$$q > 0 \implies q \geq 1 \implies n + 1 + q \geq n + 2$$

Hence, the mandatory relationship between the number of d 's and the number of c 's does not hold for the word w_1 , meaning that $w_1 \notin L$, which is a contradiction, since the Lemma guarantees that words produced by pumping remain in L .

Problem 171 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \mid BA \mid D \\ A &\rightarrow Aa \mid c \\ B &\rightarrow BB \mid \lambda \mid b \\ D &\rightarrow cDc \mid d \end{aligned}$$

(a) Write 5 distinct strings that belong to L . If such strings do not exist, state it and explain why.

Answer:

$$ca, caa, caab, bca, cdc$$

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: This regular expression does not exist, since L is not regular.

To show that L is not regular, we assume the opposite—that L is regular and that it honors the Pumping Lemma, with the constant k , as given in the Lemma.

Let $L_1 \subset L$ be the set of those words in L that contain at least one d . By inspection of the grammar, we find that L_1 is precisely the set of strings derived from the variable D , which in turn is given by the template:

$$L_1 = c^n dc^n$$

This template is characterized by the property that the number of c 's preceding the single d is equal to the number of c 's following the single d . We show that pumping violates this property.

Select a word w such that $w = c^n dc^n$, where $n > k$. In any possible pumping decomposition of w such that $w = xyz$, the Lemma guarantees that:

$$|xy| < k < n$$

meaning that the pumping substring y consists entirely of c 's. Say that $y = c^m$ for some $m > 0$. By pumping once, we obtain the word

$$w_1 = xy'yz = c^{n+m} dc^n$$

where the number of c 's following the single d is still equal to n , but the number of c 's preceding the single d is now strictly greater than n , since:

$$m > 0 \implies n + m > n$$

Hence, the mandatory property (that the d is exactly in the middle of the run of c 's contained in the word) does not hold for the word w_1 , meaning that $w_1 \notin L$, which is

a contradiction, since the Lemma guarantees that words produced by pumping remain in L .

(c) Is the intersection of L with the language $c^*(a^*b^*)^*$ context-free? Explain your answer.

Answer: Yes. Language L is context-free (as witnessed by its grammar), while $c^*(a^*b^*)^*$ is a regular language (since it is given by a regular expression.) The intersection of any context-free language with any regular language is known to be context-free.

Problem 172 Let L_1 be the language defined by the regular expression:

$$a^* b a^*$$

Let L_2 be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSa \mid b \mid bb$$

(a) Write 5 distinct strings that belong to L_1 but do not belong to L_2 . If such strings do not exist, explain why.

Answer:

$$ba, ab, abaa, aaba, baaa$$

(b) Write 5 distinct strings that belong to L_2 but do not belong to L_1 . If such strings do not exist, explain why.

Answer:

$$bb, abba, aabbba, aaabbbaaa, aaaabbaaaa$$

(c) Write 5 distinct strings that belong to L_1 and L_2 . If such strings do not exist, explain why.

Answer:

$$b, aba, aabaa, aaabaaa, aaaabaaaa$$

(d) Write 5 distinct strings over alphabet $\{a, b\}$ that do not belong to L_1 and do not belong to L_2 . If such strings do not exist, explain why.

Answer:

$$bab, aa, bba, abb, bbb$$

(e) Write a complete formal definition of a context-free grammar G that generates $L_1 \cup L_2$. If such a grammar does not exist, explain why.

Answer: $G = (V, \Sigma, P, R)$, where $\Sigma = \{a, b\}$, $V = \{R, T, S, A\}$, and the production set P is:

$$\begin{aligned} R &\rightarrow T \mid S \\ T &\rightarrow AbA \\ A &\rightarrow aA \mid \lambda \\ S &\rightarrow aSa \mid b \mid bb \end{aligned}$$

Problem 173 Let L be the set of all non-empty strings of even length over the alphabet $\{a, b, c\}$ that begin and end with the same letter.

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aAa \mid bAb \mid cAc \\ A &\rightarrow \lambda \mid AA \mid ZZ \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 174 Let L be the set of all strings over the alphabet $\Sigma = \{a, b, c\}$ in which each one of the alphabet letters occurs at least once.

(a) Write a regular expression that defines L . If such a regular expression does not exist, state it and explain why.

Answer:

$$\begin{aligned} &(a \cup b \cup c)^* a (a \cup b \cup c)^* b (a \cup b \cup c)^* c (a \cup b \cup c)^* \\ &\quad \cup \\ &(a \cup b \cup c)^* a (a \cup b \cup c)^* c (a \cup b \cup c)^* b (a \cup b \cup c)^* \\ &\quad \cup \\ &(a \cup b \cup c)^* b (a \cup b \cup c)^* a (a \cup b \cup c)^* c (a \cup b \cup c)^* \\ &\quad \cup \\ &(a \cup b \cup c)^* c (a \cup b \cup c)^* a (a \cup b \cup c)^* b (a \cup b \cup c)^* \\ &\quad \cup \\ &(a \cup b \cup c)^* c (a \cup b \cup c)^* b (a \cup b \cup c)^* a (a \cup b \cup c)^* \end{aligned}$$

A more compact, equivalent expression is as follows.

$$\begin{aligned} &a a^* b (a \cup b)^* c (a \cup b \cup c)^* \\ &\quad \cup \\ &a a^* c (a \cup c)^* b (a \cup b \cup c)^* \\ &\quad \cup \\ &b b^* a (a \cup b)^* c (a \cup b \cup c)^* \\ &\quad \cup \\ &b b^* c (b \cup c)^* a (a \cup b \cup c)^* \\ &\quad \cup \\ &c c^* a (a \cup c)^* b (a \cup b \cup c)^* \\ &\quad \cup \\ &c c^* b (b \cup c)^* a (a \cup b \cup c)^* \end{aligned}$$

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V =$

$\{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AaAbAcA \\ S &\rightarrow AaAcAbA \\ S &\rightarrow AbAaAcA \\ S &\rightarrow AbAcAaA \\ S &\rightarrow AcAaAbA \\ S &\rightarrow AcAbAaA \\ A &\rightarrow \lambda \mid AA \mid a \mid b \mid c \end{aligned}$$

Problem 175

Let L be the set of those strings over the alphabet $\{a, b, c\}$ that satisfy all of the following properties:

1. No more than two different letters occur in the string.
2. The two letters always occur in the alphabetic order within the string.
3. The number of occurrences of the letter b is always even.
4. The letter c , if it occurs in the string at all, occurs at least three times.

(a) Write a regular expression that defines L . If such a regular expression does not exist, state it and explain why.

Answer:

$$a^* (bb)^* \cup a^* ccc c^* \cup (bb)^* ccc c^*$$

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \mid AD \mid BD \\ A &\rightarrow aA \mid \lambda \\ B &\rightarrow bbB \mid \lambda \\ D &\rightarrow cD \mid ccc \end{aligned}$$

Problem 176 Let L be the set of those strings over the alphabet $\{a, b, c\}$ that satisfy all of the following properties:

1. The letters always occur in the alphabetic order within the string.
2. The number of occurrences of the letter b is always odd.
3. The number of occurrences of the letter a is never a multiple of 3.
4. The letter c always occurs twice as often as the letter a .

(a) List 5 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer:

$abcc, aabccc, abbbcc, aabbccccc, abbbbbcc$

(b) List 5 distinct strings that belong to $a^*b^*c^*$ but do not belong to L . If this is impossible, state it and explain why.

Answer:

$\lambda, a, b, abbc, abc$

(c) List 5 distinct strings that belong to L but do not belong to $a^*b^*c^*$. If this is impossible, state it and explain why.

Answer: Impossible, since $L \subset a^*b^*c^*$. To verify this, observe that the first of the four conjuncts in the definition of the language L in fact defines $a^*b^*c^*$.

(d) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, state it and explain why.

Answer: Observe that the general template for the strings of L is:

$$a^{3k+r} b^{2n+1} c^{6k+2r} \mid k, n \geq 0, r \in \{1, 2\}$$

whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aaaScfffff \mid aBcc \mid aaBcccc \\ B &\rightarrow bbB \mid b \end{aligned}$$

Problem 177 (a) Let L_1 be a language over alphabet $\{a, b, c, d, e\}$, defined as follows:

$$L_1 = \{a^n b^m c^k d^{m+2} e^{n+3} \mid k, n, m \geq 0\}$$

Write a complete formal definition of a context-free grammar G_1 that generates language L_1 . If such grammar does not exist, explain why.

Answer: $G_1 = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, e\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aSe \mid Aceee \\ A &\rightarrow bAd \mid Bdd \\ B &\rightarrow cB \mid \lambda \end{aligned}$$

(b) Let L_2 be a language over alphabet $\{a, b, c, d, e\}$, defined as follows:

$$L_2 = \{a^{2n} b^{3n} c^{k+2} d^{3k+1} e^{m+3} a^{2m+5} \mid k, n, m \geq 0\}$$

Write a complete formal definition of a context-free grammar G_2 that generates language L_2 . If such grammar does not exist, explain why.

Answer: $G_2 = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, e\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ABD \\ A &\rightarrow aaAbbb \mid \lambda \\ B &\rightarrow cBddd \mid ccd \\ D &\rightarrow eDaa \mid eeeaaaaa \end{aligned}$$

Problem 178 (a) Let L_1 be the set of all strings over alphabet $\{0, 1\}$ that contain at least two 1's.

Write a complete formal definition of a context-free grammar G_1 that generates language L_1 . If such grammar does not exist, explain why.

Answer: $G_1 = (V, \Sigma, P, S)$, where $\Sigma = \{0, 1\}$, $V = \{S, Z\}$,

and the production set P is:

$$\begin{aligned} S &\rightarrow Z1Z1Z \\ Z &\rightarrow ZZ \mid \lambda \mid 0 \mid 1 \end{aligned}$$

(b) Let L_2 be the set of all strings over alphabet $\{0, 1\}$ that contain an even number of 1's.

Write a complete formal definition of a context-free grammar G_2 that generates language L_2 . If such grammar does not exist, explain why.

Answer: $G_2 = (V, \Sigma, P, S)$, where $\Sigma = \{0, 1\}$, $V = \{S, D\}$,

and the production set P is:

$$\begin{aligned} S &\rightarrow 0S \mid 1D \mid \lambda \\ D &\rightarrow 0D \mid 1S \end{aligned}$$

Problem 179 (a) Let:

$$L = \{a^i b^j c^k d^m \mid i = j + k \text{ and } m = 2\ell, i, j, k, m, \ell \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AD \\ A &\rightarrow aAc \mid B \\ B &\rightarrow aBb \mid \lambda \\ D &\rightarrow DD \mid \lambda \mid dd \end{aligned}$$

(b) What is the cardinality of the set of context-free grammars? Answer by giving the exact number (if this set is finite) or by specifying if it is countable or uncountable.

Answer: The set of context-free grammars is infinite and countable (cardinality \aleph_0).

Problem 180 (a) Let:

$$L = \{a^i b^j c^k d^m \mid i = j \text{ and } j = 2k, i, j, k, m \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: L does not have a context-free grammar.

Observe that:

$$L = \{a^{2k}b^{2k}c^kd^m \mid k, m \geq 0\}$$

To prove that language L is not context-free, assume the opposite—that the Pumping Lemma holds for L .

Let ν be the constant as in the Pumping Lemma for L_2 . Let $w = a^{2n}b^{2n}c^n$, such that $n > \nu$. (Note that $m = 0$ for this choice, and $d^0 = \lambda$.) Let $w = uvxyz$ be a decomposition of w , where vy is the pumping part.

Every substring of w that contains all the three letters (a, b, c) must contain the entire run of b 's of length $2n$, plus some a 's before this run of b 's and some c 's after it. Hence, every substring of w that contains all the three letters has length at least $2n+2$. By the Lemma, it must be that $|vxy| < \nu < n < 2n+2$. Thus, the pumping part vy is too short to contain all the three letters—it lacks at least one of them, say letter ξ , for some $\xi \in \{a, b, c\}$.

After the pumping is admitted, any word of the form $uv^i xy^i z$, for $i > 1$, is claimed to be in L . However, since the other two letters are pumped, while ξ is not pumped, such a word has a surplus of some of the other two letters, relative to ξ , thereby violating the pattern $a^{2k}b^{2k}c^k$ —a contradiction.

(b) Is every countable language context-free? Explain your answer briefly.

Answer: No—every language is countable, but (infinitely and uncountably) many of them are not context-free.

Problem 181 (a) Let L_1 be a language over alphabet $\{a, b, c, d, e\}$, defined as follows:

$$L_1 = \{d^j a^{3n} b^{2m} c^{k+2} d^{2m+1} e^{n+1} d^\ell \mid j, k, \ell, m, n \geq 0\}$$

Write a complete formal definition of a context-free grammar G_1 that generates language L_1 . If such grammar does not exist, explain why.

Answer: $G_1 = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c, d, e\}$, $V = \{S, A, B, D, E\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow DAD \\ D &\rightarrow dD \mid \lambda \\ A &\rightarrow aaaAc \mid Be \\ B &\rightarrow bbBdd \mid Ed \\ E &\rightarrow cE \mid cc \end{aligned}$$

(b) Let L_2 be the set of strings over alphabet $\{a, b, c, d, e\}$ in which the total number of d 's and e 's is 3.

Write a complete formal definition of a context-free grammar G_2 that generates language L_2 . If such grammar does not exist, explain why.

Answer: $G_2 = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c, d, e\}$, $V = \{S, A, Z\}$,

and the production set P is:

$$\begin{aligned} S &\rightarrow AZAZAZA \\ A &\rightarrow AA \mid \lambda \mid a \mid b \mid c \\ Z &\rightarrow d \mid e \end{aligned}$$

Problem 182 (a) Let L_1 be a language over alphabet $\{a, b, c, d, e\}$, defined as follows:

$$L_1 = \{a^{2n} d^\ell b^m c^k d^{2m+1} e^{n+3} \mid k, n, m, \ell \geq 0\}$$

Write a complete formal definition of a context-free grammar G_1 that generates language L_1 . If such grammar does not exist, explain why.

Answer: $G_1 = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c, d, e\}$, $V = \{S, A, B, D, E\}$, and P is:

$$\begin{aligned} S &\rightarrow aaSe \mid Aeee \\ A &\rightarrow DB \\ D &\rightarrow dD \mid \lambda \\ B &\rightarrow bBdd \mid Ed \\ E &\rightarrow cE \mid \lambda \end{aligned}$$

(b) Let L_2 be a language over alphabet $\{a, b, c, d, e\}$, consisting of those strings that have an even number of e 's.

Write a complete formal definition of a context-free grammar G_2 that generates language L_2 . If such grammar does not exist, explain why.

Answer: $G_2 = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c, d, e\}$, $V = \{S, Z, E\}$, and P is:

$$\begin{aligned} S &\rightarrow Z \mid E \\ Z &\rightarrow ZZ \mid \lambda \mid a \mid b \mid c \mid d \\ E &\rightarrow EE \mid \lambda \mid ZeZeZ \end{aligned}$$

Problem 183 Write a complete formal definition of a context-free grammar that generates the set of strings over $\{a, b, c\}$ in which the total number of b 's and c 's is three. If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow ABABABA \\ A &\rightarrow aA \mid \lambda \\ B &\rightarrow b \mid c \end{aligned}$$

Problem 184 (a) Let L_1 be the set of all strings over alphabet $\{a, b\}$ that contain an equal number of a 's and b 's.

Write a complete formal definition of a context-free grammar G_1 that generates language L_1 . If such a grammar does not exist, explain why.

Answer: $G_1 = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow SS \mid aSb \mid bSa \mid \lambda$$

(b) Let L_2 be the set of all strings over alphabet $\{a, b\}$ in which all a 's follow all b 's and there is an equal number of a 's and b 's.

Write a complete formal definition of a context-free grammar G_2 that generates language L_2 . If such a grammar does not exist, explain why.

Answer: $G_2 = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow bSa \mid \lambda$$

Problem 185 Let L be the language over alphabet $\{a, b, c\}$ defined as follows:

$$L = \{c^p b^n a^m \mid m, n, p \geq 0 \wedge (m = p \vee n = p)\}$$

Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c\}$, $V = \{S, S_1, S_2, A, B, D\}$, and P comprises:

$$\begin{aligned} S &\rightarrow S_1 \mid S_2 \\ S_1 &\rightarrow cS_1a \mid A \\ A &\rightarrow bA \mid \lambda \\ S_2 &\rightarrow BD \\ B &\rightarrow cBb \mid \lambda \\ D &\rightarrow aD \mid \lambda \end{aligned}$$

Problem 186 Let L be the set of strings over alphabet $\{p, q, r\}$ whose length is divisible by 3.

(a) Write 5 distinct strings that belong to L .

(b) Write 5 distinct strings over alphabet $\{p, q, r\}$ that do not belong to L .

Answer:

$\in L$	$\notin L$
λ	p
ppp	pp
$qqqqqq$	$qqqq$
$rrrrrrrrrr$	$rrrrrr$
$pqrprqqrqrp$	$pqrpqrqrp$

(c) Write a complete formal definition of a context-free grammar G that generates L .

Answer: $G = (V, \Sigma, P, S)$, where

$\Sigma = \{p, q, r\}$, $V = \{S, A\}$, and P comprises:

$$\begin{aligned} S &\rightarrow SS \mid AAA \mid \lambda \\ A &\rightarrow p \mid q \mid r \end{aligned}$$

Observe that L is regular, and defined by the regular expression:

$$((p \cup q \cup r)(p \cup q \cup r)(p \cup q \cup r))^*$$

A regular grammar that generates L :

$G' = (V', \Sigma, P', S)$, where

$\Sigma = \{p, q, r\}$, $V' = \{S, A, B\}$, and P' comprises:

$$\begin{aligned} S &\rightarrow pA \mid qA \mid rA \mid \lambda \\ A &\rightarrow pB \mid qB \mid rB \\ B &\rightarrow pS \mid qS \mid rS \end{aligned}$$

Problem 187 (a) Write a complete formal definition of a context-free grammar G that generates the language of all strings over alphabet $\{a, b, c, d\}$ which contain at least one a and at least one b (in any order.)

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$ is the set of terminals; $V = \{S, T\}$ is the set of variables; S is the start symbol; and the set of productions P comprises:

$$\begin{aligned} S &\rightarrow TaTbT \mid TbTaT \\ T &\rightarrow TT \mid a \mid b \mid c \mid d \mid \lambda \end{aligned}$$

(b) Is the grammar G which you constructed in your answer to part (a) regular? Justify briefly your answer.

Answer: No. The right-hand side of the first production has as many as 3 variables, whereas the right-hand side of every production in a regular grammar contains at most one variable along with exactly one terminal that precedes it.

Problem 188 (a) Construct a context-free grammar G over alphabet $\{a, b, c\}$ that generates the language

$$L(G) = \{a^m b^n c^i \mid m + n < i\}$$

where m, n, i are non-negative integers.

Answer: Let $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$ is the set of terminals; $V = \{S, A, B\}$ is the set of variables; S is the start symbol. The set of productions P comprises:

$$\begin{aligned} S &\rightarrow Sc \mid Ac \\ A &\rightarrow aAc \mid B \\ B &\rightarrow bBc \mid \lambda \end{aligned}$$

(b) Is the grammar G that you constructed in your answer to part (a) regular? Justify briefly your answer.

Answer: No. The right-hand side of all productions in a regular grammar is of the form λ , or a , or aA , where $a \in \Sigma$, $A \in V$. For example, the production $A \rightarrow aAc$ in part (a) is not of this form.

Problem 189 (a) Let L be the set of all strings of the form $a^k b^m$ such that $k = 2m$, where $k, m \geq 0$.

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S\}$, and P is:

$$S \rightarrow aaSb \mid \lambda$$

(b) Let L be the set of all strings of the form $a^k b^m$ such that $k \neq 2m$, where $k, m \geq 0$.

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Advice for Answer: $k \neq 2m \iff k < 2m \vee k > 2m$.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow aaSb \mid A \mid B \\ A &\rightarrow aA \mid \lambda \\ B &\rightarrow bB \mid \lambda \end{aligned}$$

Problem 190 Let:

$$L = \{a^k b^n c^j d^\ell \mid k \leq \ell, n \geq j, k = 2m + 1\}$$

where $j, k, \ell, m, n \geq 0$.

Advice for Answer: The template is:

$$a^{2m+1} b^{j+t} c^j d^{2m+1+p}$$

where $j, m, t, p \geq 0$.

Answer: $G = (V, \Sigma, P, S)$,

where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D, T\}$, and P is:

$$\begin{aligned} S &\rightarrow aTd \\ T &\rightarrow aaTdd \mid AD \\ D &\rightarrow dD \mid \lambda \\ A &\rightarrow bAc \mid B \\ B &\rightarrow bB \mid \lambda \end{aligned}$$

Problem 191 Let L be the set of strings over the alphabet $\{a, b, c\}$ such that the length of the string is equal to 6.

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ZZZZZZ \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 192 Let L_1 be a language over the alphabet $\{a, b, c, d\}$, defined as follows:

$$L_1 = \{a^{3m+3} c^{2j+2} b^{m+1} \text{ where } m, j \geq 0\}$$

Let L_2 be a language over the alphabet $\{a, b, c, d\}$, defined as follows:

$$L_2 = \{(da)^k bb (ad)^{2k} \text{ where } k \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar that generates L_1 . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aaaSb \mid aaaAb \\ A &\rightarrow ccA \mid cc \end{aligned}$$

(b) Write a complete formal definition of a context-free grammar that generates L_2 . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, T)$, where $\Sigma = \{a, b, d\}$, $V = \{T\}$, and the production set P is:

$$T \rightarrow daTadad \mid bb$$

(c) Write a complete formal definition of a context-free grammar that generates $L_1 L_2$. If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, Q)$, where $\Sigma = \{a, b, c, d\}$, $V = \{Q, S, A, T\}$, and the production set P is:

$$\begin{aligned} Q &\rightarrow ST \\ S &\rightarrow aaaSb \mid aaaAb \\ A &\rightarrow ccA \mid cc \\ T &\rightarrow daTadad \mid bb \end{aligned}$$

(d) Write a complete formal definition of a context-free grammar that generates $(L_1 \cup L_2)^*$. If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, Q)$, where $\Sigma = \{a, b, c, d\}$, $V = \{Q, S, A, T\}$, and the production set P is:

$$\begin{aligned} Q &\rightarrow QQ \mid \lambda \mid S \mid T \\ S &\rightarrow aaaSb \mid aaaAb \\ A &\rightarrow ccA \mid cc \\ T &\rightarrow daTadad \mid bb \end{aligned}$$

Problem 193 Let L be the set of all strings over the alphabet $\{a, b\}$ such that the length of the string is at least 3.

(a) Write 5 distinct strings that belong to L . If such strings do not exist, state it and explain why.

Answer:

aaa, aab, aba, abb, baa

(b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 3.

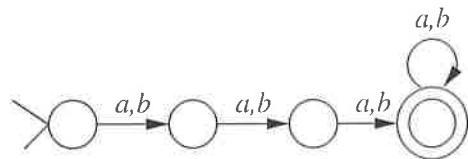


Figure 3:

(c) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, Z\}$, and P is:

$$\begin{aligned} S &\rightarrow ZS \mid ZZZ \\ Z &\rightarrow a \mid b \end{aligned}$$

Problem 194 Let L be the set of all strings over the alphabet $\{a, b\}$ such that all a 's come before any b 's, and the number of a 's is at least 2, but the number of b 's is at least 3.

(a) Write 5 distinct strings that belong to L . If such strings do not exist, state it and explain why.

Answer:

aabbb, aaabbb, aaabbbb, aaaabbbb, aaaabbbbb

(b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 4.

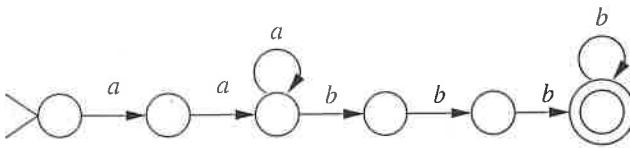


Figure 4:

(c) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aA \mid aa \\ B &\rightarrow bB \mid bbb \end{aligned}$$

Problem 195 Let L be the set of strings over the alphabet $\{a, b, c\}$ such that the length of the string is equal to 5 or 6.

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ZZZZZ \mid ZZZZZZ \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 196 Let L be the set of all strings over the alphabet $\{a, b\}$ whose length gives an odd remainder after a division by 3.

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, A, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ZA \\ A &\rightarrow ZZZA \mid \lambda \\ Z &\rightarrow a \mid b \end{aligned}$$

Problem 197 Let L be the set of all strings over the alphabet $\{a, b\}$ whose length gives a remainder of 1 after a division by 4.

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ZZZZS \mid Z \\ Z &\rightarrow a \mid b \end{aligned}$$

Problem 198 Let:

$$L = \{a^i c^j a^k b^\ell g^m b^n a^x f^y\}$$

where $i, j, k, \ell, m, n, x, y \geq 0$
are natural numbers such that:

$$y = i + j, \ell = n + 1, m = 0, k = x$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The template for L is:

$$L = \{a^i c^j a^k b^{2n+1} a^k f^j f^i\}$$

whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, g, f\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aSf \mid A \\ A &\rightarrow cAf \mid B \\ B &\rightarrow aBa \mid D \\ D &\rightarrow bbD \mid b \end{aligned}$$

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: This regular expression does not exist, since L is not regular. To prove this, we show that Pumping Lemma does not hold for L .

Observe that all words of L satisfy the following characteristic property: number of f 's is equal to the number of c 's plus the number of a 's to the left of c 's.

Assume the opposite, that L is regular. Let ξ be the constant as in the Pumping Lemma for L . Let $j > \xi$; then the word $w = c^j b f^j$ belongs to L , as it is obtained from the template by setting $i = k = n = 0$.

In any “pumping” decomposition such that $c^j b f^j = uvx$, we have: $|uv| \leq \xi < j$. Hence, the “pumping” substring v consists entirely of c 's, say $v = c^\ell$. Recall that $\ell > 0$, since the “pumping” substring cannot be empty. Pump up once, obtaining the word:

$$w_1 = c^{j+\ell} b f^j$$

Since $j + \ell > j$, word w_1 violates the stated characteristic property and thus $w_1 \notin L$, in violation of the Pumping Lemma.

Problem 199 Let:

$$L = \{a^n b^k a^p d^\ell c^j d^m\}$$

where $n + 3 = m, \ell = j + 2, k = p = 0, n, k, \ell, j, m, p \geq 0$.

- (a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: Observe that the template for L is:

$$a^n d^{j+2} c^j d^{n+3}$$

where $n, j \geq 0$, whence the grammar:

$G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aSd \mid Dddd \\ D &\rightarrow dDc \mid dd \end{aligned}$$

- (b) Draw a state transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: Such an automaton does not exist since L is not regular. To prove that L is not regular, first observe that every string in the language L satisfies the following property:

the number of d 's exceeds the number of a 's by at least 5.

Assume that L is regular, and let η be the constant as in the Pumping Lemma for L . Select a word:

$$w = a^n d^{n+5}, \text{ for some } n > \eta$$

obtained by setting $j = 0$ in the template of L . The word w is long enough and must pump. In any pumping decomposition: $w = xyz$, the length of the substring $|xy|$ is not greater than η , which in turn is less than n . Hence, the pumping window y is located entirely within the segment a^n , and pumping up once produces a word which has $n + |y| > n$ a 's but only $n + 5$ d 's, thereby violating the considered property. Since the Pumping Lemma does not hold for L , L is not regular.

Problem 200 Let:

$$L = \{a^n b^k a^p d^\ell c^j d^m\}$$

where $n = p, k = j = 0, \ell = m + 2, n, k, \ell, j, m, p \geq 0$.

- (a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: Observe that the template for L is:

$$a^{2n} d^{2m+2}$$

where $n, m \geq 0$, whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aaA \mid \lambda \\ B &\rightarrow ddB \mid dd \end{aligned}$$

- (b) Draw a state transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 5.

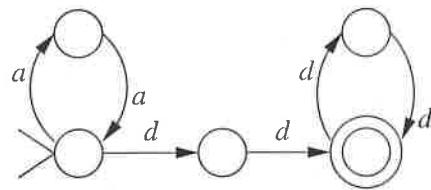


Figure 5:

Problem 201 Let:

$$L = \{a^n b^k a^p d^\ell c^j d^m\}$$

where $n = \ell + 3, k = j + 2, m = p = 0, n, k, \ell, j, m, p \geq 0$.

- (a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: Such a grammar does not exist, since L is not context-free. To prove that L is not context-free, first observe that the template for L is:

$$a^{\ell+3} b^{j+2} d^\ell c^j$$

where $\ell, j \geq 0$.

Assume that L is context-free, and let η be the constant as in the Pumping Lemma for L . Select a word:

$$w = a^{\ell+3} b^{j+2} d^\ell c^j \text{ where } \ell, j > \eta$$

The word w is long enough and must pump. In any pumping decomposition: $w = xyz$, the length of the pumping window y_1ty_2 is not greater than η . Hence, the pumping window is located either entirely within one of the four substrings $a^{\ell+3}, b^{j+2}, d^\ell, c^j$, or it is shared by an adjacent pair of them, since it is too short to span more than two of these substrings. If the pumping window is within any one of the four named substrings, the pumping produces an excess of the letter pumped; if the window is shared by two substrings, the pumping produces a lack of a another letter, thereby in all cases leading to a word that violates the considered template. Hence, L does not honor the Pumping Lemma and thereby cannot be context-free.

- (b) Draw a state transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: This automaton does not exist, since language L is not regular. To prove this, recall that L is not context-free, as is proved in the answer to part (a). However, if L was regular, it would be context-free.

Problem 202 Let L be the language over the alphabet $\Sigma = \{a, b, c, d\}$ that contains exactly those strings whose form is:

$$d^{m+1} c^{m+k+2} a^k d^j c^{j+1}$$

where $j, k, m \geq 0$ are natural numbers.

If L is context free, then use part (a) of the answer space below to write a complete formal definition of a context free grammar that generates L , and do not write anything in part (b).

If L is not context free, then do not write anything in part (a) of the answer space, but complete the missing parts of the text given in part (b) so as to obtain a proof that L is not context free.

(a) Context free grammar for L :

Answer: Observe that the template is:

$$d^{m+1}c^{m+1}c^{k+1}a^kd^jc^{j+1}$$

whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow ABD \\ A &\rightarrow dAc \mid dc \\ B &\rightarrow cBa \mid c \\ D &\rightarrow dDc \mid c \end{aligned}$$

(b) Proof that L is not context free (no answer allowed): Observe that all words of L satisfy the following characteristic property:

Assume the opposite, that L is context free. Let π be the constant as in the Pumping Lemma for L . Let $w_0 \in L$ be a string defined as follows:

$$w_0 =$$

w_0 belongs to L because

w_0 must pump because

In any “pumping” decomposition of w_0 , the pumping window satisfies the following property:

because

By pumping times, we obtain a string:

which violates the stated characteristic property because and thus does not belong to L . Since L violates the Pumping Lemma, L is not context free.

Problem 203 Let L be the language over the alphabet $\Sigma = \{a, b, c, d\}$ that contains exactly those strings whose form is:

$$c^k a^m d^{2k} b^m c^{3\ell}$$

where $k, \ell, m \geq 0$ are natural numbers.

If L is context free, then use part (a) of the answer space below to write a complete formal definition of a context free grammar that generates L , and do not write anything in part (b).

If L is not context free, then do not write anything in part (a) of the answer space, but complete the missing

parts of the text given in part (b) so as to obtain a proof that L is not context free.

(a) Context free grammar for L :

Answer:

(b) Proof that L is not context free (answer underlined):

Observe that all words of L satisfy the following characteristic property:

conjunction:

1. number of a 's = number of b 's

2. number of d 's = $2 \times$ number of c 's to the left of d 's

Assume the opposite, that L is context free. Let j be the constant as in the Pumping Lemma for L . Let $w_0 \in L$ be a string defined as follows:

$$w_0 = c^k a^m d^{2k} b^m, \quad k, m > j$$

w_0 belongs to L because

it is obtained from the template by setting $\ell = 0$.

w_0 must pump because

$$|w_0| = 3k + 2m > 3j + 2j = 5j > j$$

In any “pumping” decomposition of w_0 , the pumping window satisfies the following property:

it is either within a single segment
or within two adjacent segments

because

it is too short to extend through 3 segments.

By pumping 1 times, we obtain a string which violates the stated characteristic property because no more than 2 letters are pumped, which means that the number of occurrences of at least one other letter is not correct.

and thus does not belong to L . Since L violates the Pumping Lemma, L is not context free.

Problem 204 Let L be the language over the alphabet $\Sigma = \{a, b, c, d\}$ that contains exactly those strings whose form is:

$$c^k a^{m+1} d^{3j+2} b^{\ell+3} c^{2j} d^{2m+1} a^{3k}$$

where $j, k, \ell, m \geq 0$ are natural numbers.

If L is context free, then use part (a) of the answer space below to write a complete formal definition of a context free grammar that generates L , and do not write anything in part (b).

If L is not context free, then do not write anything in part (a) of the answer space, but complete the missing parts of the text given in part (b) so as to obtain a proof that L is not context free.

(a) Context free grammar for L :

Answer: $G = (V, \Sigma, P, S)$,where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow cSa^a \mid A \\ A &\rightarrow aAdd \mid aBd \\ B &\rightarrow dddBcc \mid ddD \\ D &\rightarrow bD \mid bbb \end{aligned}$$

(b) Proof that L is not context free (no answer allowed):Observe that all words of L satisfy the following characteristic property:Assume the opposite, that L is context free. Let τ be the constant as in the Pumping Lemma for L . Let $w_0 \in L$ be a string defined as follows: $w_0 =$ w_0 belongs to L because w_0 must pump becauseIn any “pumping” decomposition of w_0 , the pumping window satisfies the following property:

because

By pumping times, we obtain a string:

which violates the stated characteristic property because and thus does not belong to L . Since L violates the Pumping Lemma, L is not context free.**Problem 205** Let L be the language over the alphabet $\Sigma = \{a, b, c, d\}$ that contains exactly those strings whose form is:

$$a^k c^m d^k b^k a^{3\ell}$$

where $k, \ell, m \geq 0$ are natural numbers.If L is context free, then use part (a) of the answer space below to write a complete formal definition of a context free grammar that generates L , and do not write anything in part (b).If L is not context free, then do not write anything in part (a) of the answer space, but complete the missing parts of the text given in part (b) so as to obtain a proof that L is not context free.(a) Context free grammar for L :**Answer:**(b) Proof that L is not context free:Observe that all words of L satisfy the following characteristic property:number of a 's on the left of d 's is equal to the number of d 's, which is equal to the number of b 's.Assume the opposite, that L is context free.Let ρ be the constant as in the Pumping Lemma for L . Let $w_0 \in L$ be a string defined as follows: $w_0 = a^k d^k b^k$, where $k > \rho$ w_0 belongs to L becauseit is obtained from the template by setting $m = 0$. w_0 must pump because

$|w_0| = 3k > k > \rho$

In any “pumping” decomposition of w_0 , the pumping window satisfies the following property:pumping window y is entirely either within one of the three segments (of a 's, d 'c or b 's) or it is within some two adjacent ones (a 's and d 'c or d 'c and b 's.)

because

pumping window is shorter than the constant ρ , so $|y| < \rho < k$, hence y is too short to extend through all 3 segments.By pumping 1 times, we obtain a string in which one or two letters occur more than k times, but at least one letter occurs exactly k times, which violates the stated characteristic property because the number of occurrences of at least one letter is different from the number of occurrences of at least one other letterand thus does not belong to L . Since L violates the Pumping Lemma, L is not context free.**Problem 206** Let L be the language over the alphabet $\Sigma = \{a, b, c, d\}$ that contains exactly those strings whose form is:

$$c^i d^j a^k d^\ell b^m a^n b^p c^q$$

where $i, j, k, \ell, m, n, p, q \geq 0$ are natural numbers such that: $i = \ell$, $k = 3\ell$, $n = 0$, $p = 0$, $q = 0$, $j = 0$.If L is context free, then use part (a) of the answer space below to write a complete formal definition of a context free grammar that generates L , and do not write anything in part (b). If L is not context free, then do not write anything in part (a) of the answer space, but complete the missing parts of the text given in part (b) so as to obtain a proof that L is not context free.(a) Grammar that generates L :**Answer:**(b) Proof that L is not context free (answer underlined): Observe that all words of L satisfy the following characteristic property:template $c^\ell a^{3\ell} d^\ell b^m$, hencenumber of c 's is equal to number of d 's and number of a 's is 3 times as much.Assume the opposite, that L is context free. Let π be the constant as in the Pumping Lemma for L . Let $w_0 \in L$ be a string defined as follows: $w_0 = c^\ell a^{3\ell} d^\ell$ where $\ell > \pi$ w_0 belongs to L becauseit is obtained from the template with $m = 0$ w_0 must pump because $|w_0| = 5\ell > \pi$

In any “pumping” decomposition of w_0 , the pumping window satisfies the following property:

it is within a single-symbol segment
or within at most two adjacent segments

because it is too short to intersect with all three segments
since it is shorter than π and $\ell > \pi$

By pumping 1 times, we obtain a string w_1
 which violates the stated characteristic property because

w_1 differs from w_0 in that it has
more occurrences of some symbol(s)
but equal number of some other(s)

hence $w_1 \notin L$.

Since L violates the Pumping Lemma, it is not context free.

Problem 207 Let L be the language over the alphabet $\Sigma = \{a, b, c\}$ that contains exactly those strings which satisfy all of the following properties:

1. the three symbols a, b, c occur in the string in the alphabetic order;
2. the length of the substring which consists of a 's is positive and even;
3. the length of the substring which consists of b 's is equal to twice the length of the substring which consists of a 's;
4. the length of the substring which consists of c 's is equal to twice the length of the substring which consists of b 's;

If L is context free, then use part (a) of the answer space below to write a complete formal definition of a context free grammar that generates L , and do not write anything in part (b).

If L is not context free, then do not write anything in part (a) of the answer space, but complete the missing parts of the text given in part (b) so as to obtain a proof that L is not context free.

(a) Context free grammar for L :

Answer:

(b) Proof that L is not context free:

Observe that all words of L satisfy the following characteristic property:

template:

$$a^{2n+2}b^{4n+4}c^{8n+8}$$

Assume the opposite, that L is context free. Let π be the constant as in the Pumping Lemma for L . Let $w_0 \in L$ be a string defined as follows:

$$w_0 = a^{2n+2}b^{4n+4}c^{8n+8} \text{ where } n > \pi$$

w_0 belongs to L because

it satisfies the definition (which is equivalent to the template.)

w_0 must pump because

$$|w_0| = 14n + 14 > n > \pi$$

In any “pumping” decomposition of w_0 , the pumping window satisfies the following property:

spans no more than two adjacent segments

because

it is too short to extend through all three segments, since it is shorter than π by the Lemma, but each segment is longer than π by construction.

By pumping 1 times, we obtain a string which violates the stated characteristic property because

the number of occurrences of at least one of the letters has increased, but the number of occurrences of at least one other letter has remained unchanged,

and thus does not belong to L . Since L violates the Pumping Lemma, L is not context free.

Problem 208 Let L be the language over the alphabet $\Sigma = \{a, b, c\}$ that contains exactly those strings which satisfy all of the following properties:

1. the three symbols a, b, c occur in the string in the alphabetic order;
2. the length of the substring which consists of a 's is odd;
3. the length of the substring which consists of b 's is positive and even;
4. the length of the substring which consists of c 's is equal to the sum of lengths of the two substrings which consist of a 's and b 's.

If L is context free, then use part (a) of the answer space below to write a complete formal definition of a context free grammar that generates L , and do not write anything in part (b).

If L is not context free, then do not write anything in part (a) of the answer space, but complete the missing parts of the text given in part (b) so as to obtain a proof that L is not context free.

(a) Context free grammar for L :

Advice for Answer: Template:

$$a^{2n+1}b^{2k+2}c^{2k+2}c^{2n+1}$$

Answer: $G = (V, \Sigma, P, S)$, where
 $\Sigma = \{a, b, c\}$, $V = \{S, A\}$, and P is:

$$\begin{aligned} S &\rightarrow aaSc | aAc \\ A &\rightarrow bbAcc | bbcc \end{aligned}$$

(b) Proof that L is not context free:

Observe that all words of L satisfy the following characteristic property:

Assume the opposite, that L is context free. Let π be the constant as in the Pumping Lemma for L . Let $w_0 \in L$ be a string defined as follows:

$$w_0 =$$

w_0 belongs to L because

w_0 must pump because

In any “pumping” decomposition of w_0 , the pumping window satisfies the following property:

because

By pumping times, we obtain a string:
which violates the stated characteristic property because

and thus does not belong to L . Since L violates the Pumping Lemma, L is not context free.

Problem 209 Let L be the language over the alphabet $\Sigma = \{a, b, c, g\}$ that contains exactly those strings whose form is:

$$g^m c^{2n} b^{3m} a^{4n}$$

where $m, n \geq 0$ are natural numbers.

If L is context free, then use part (a) of the answer space below to write a complete formal definition of a context free grammar that generates L , and do not write anything in part (b).

If L is not context free, then do not write anything in part (a) of the answer space, but complete the missing parts of the text given in part (b) so as to obtain a proof that L is not context free.

(a) Context free grammar for L :

Answer:

(b) Proof that L is not context free:

Observe that all words of L satisfy the following characteristic property:

In each of the two pairs of non-adjacent segments, the number of symbols is related: number of b 's is equal to 3 times the number of g 's, and number of a 's is equal to 4 times the number of c 's.

Assume the opposite, that L is context free.

Let β be the constant as in the Pumping Lemma for L . Let $w_0 \in L$ be a string defined as follows:

$$w_0 = g^m c^{2n} b^{3m} a^{4n}$$

where $m, n > \beta$.

w_0 belongs to L because it satisfies the template.

w_0 must pump because

$$\|w_0\| = 4m + 6n > 4\beta + 6\beta > \beta$$

In any “pumping” decomposition of w_0 , the pumping window satisfies the following property:

it is either within a single segment or within at most two adjacent segments

because

the pumping window is shorter than β while every segment is longer than β .

By pumping 1 times, we obtain a string

in which at least one segment is pumped while its related non-adjacent segment is not pumped,

which violates the stated characteristic property because the lengths of these two non-adjacent segments fail the characteristic property, since one of them has been increased by pumping while the other has not,

and thus does not belong to L . Since L violates the Pumping Lemma, L is not context free.

Problem 210 Let L be the language over the alphabet $\Sigma = \{a, b, c, g\}$ that contains exactly those strings whose form is:

$$g^m c^n b^{m+n+2} a^{3k+1}$$

where $m, n, k \geq 0$ are natural numbers.

If L is context free, then use part (a) of the answer space below to write a complete formal definition of a context free grammar that generates L , and do not write anything in part (b).

If L is not context free, then do not write anything in part (a) of the answer space, but complete the missing parts of the text given in part (b) so as to obtain a proof that L is not context free.

(a) Context free grammar for L :

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, g\}$,
 $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow gAb | D \\ D &\rightarrow cDb | bb \\ B &\rightarrow aaaB | a \end{aligned}$$

(b) Proof that L is not context free:

Observe that all words of L satisfy the following characteristic property:

Assume the opposite, that L is context free.

Let γ be the constant as in the Pumping Lemma for L . Let $w_0 \in L$ be a string defined as follows:

$$w_0 =$$

w_0 belongs to L because

w_0 must pump because

In any “pumping” decomposition of w_0 , the pumping window satisfies the following property:

because

By pumping times, we obtain a string

which violates the stated characteristic property because

and thus does not belong to L . Since L violates the Pumping Lemma, L is not context free.

Problem 211 Let L be the set of all palindromes (strings equal to their reversal) over the alphabet $\{a, b, c\}$.

(a) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: Such automaton does not exist, since L is not regular.

To prove this, assume the opposite, that L is regular. Let k be the constant as in the Pumping Lemma for L . Let $m > k$; then $a^m b a^m \in L$. In any “pumping” decomposition such that $a^m b a^m = uvx$, we have: $|uv| \leq k < m$. Hence, the “pumping” substring v consists entirely of a 's, say $v = a^\ell$. Recall that $\ell > 0$, since the “pumping” substring cannot be empty. After pumping once, we obtain a word:

$$w_1 = a^{m+\ell} b a^m$$

while its reversal is:

$$w_1^R = a^m b a^{m+\ell}$$

Since $m + \ell > m$, word w_1 has more a 's before the single b than w_1^R , and it must be that:

$$w_1 \neq w_1^R$$

Since w_1 is not equal to its reversal w_1^R , it follows that w_1 is not a palindrome and $w_1 \notin L$, which is a contradiction.

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S\}$, and P is:

$$S \rightarrow aSa \mid bSb \mid cSc \mid a \mid b \mid c \mid \lambda$$

Problem 212 Let:

$$L = \{a^m b^k c^p d^\ell \mid m = k = p = \ell, m, k, p, \ell \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: This grammar does not exist, since L is not context-free.

To prove this, observe that the general template for all strings that belong to L is:

$$a^m b^m c^m d^m \text{ where } m \geq 0$$

and every such string has the property that the number of occurrences of any of the four letters is equal to the number of occurrences of any other.

Now, assume that L is context-free, and let η be the constant as in the Pumping Lemma for L . Select a word:

$$w = a^m b^m c^m d^m \text{ where } m > \eta$$

In any “pumping” decomposition: $w = uvxyz$, the length of the “pumping window” vxy is not greater than η : $|vxy| \leq \eta < m$. The “pumping” window vxy either falls within one of the four substrings containing a single letter: a^m , b^m , c^m , d^m , or it spans two such adjacent substrings—it is too short to extend through as many as three (or four) of them. If the pumping is only within one of the designated substrings, it will make this substring longer than is appropriate with respect to the length of another substring. In the other case, if two adjacent letters are pumped, then there will be two other substrings with fewer letters than required (in addition to possible appearance of letters out of order.) This means that the word obtained by pumping does not honor the required template and cannot belong to L , which is a contradiction.

(b) Write a complete formal definition of a *regular* context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: Such a grammar does not exist—if it existed, L would be regular and thereby context-free, since every regular language is context-free. However, the answer to the part (a) shows that L is not context-free, meaning that L cannot be regular.

Problem 213 Let:

$$L = \{a^m b^k c^p d^\ell \mid m = k + p + \ell, m, k, p, \ell \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The general template for all strings of L is:

$$a^\ell a^p a^k b^k c^p d^\ell, \quad m, k, p, \ell \geq 0$$

and L is generated by the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow aSd \mid A \\ A &\rightarrow aAc \mid B \\ B &\rightarrow aBb \mid \lambda \end{aligned}$$

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: This regular expression does not exist, since L is not regular.

To prove this, recall that the general template for all strings that belong to L is:

$$a^\ell a^p a^k b^k c^p d^\ell$$

and every such string has the property that the number of occurrences of letter a is equal to the number of occurrences of all other letters together.

Now, assume that L is regular, and let η be the constant as in the Pumping Lemma for L . Select a word:

$$w = a^k b^k k > \eta$$

obtained by setting $p = \ell = 0$ in the general template. In any “pumping” decomposition: $w = uvx$, the length of the “pumping window” v is not greater than η : $|vxy| \leq \eta < k$. Hence, the “pumping” substring v consists entirely of a 's, say $v = a^\ell$. Recall that $\ell > 0$, since the “pumping” substring cannot be empty. After pumping once, we obtain a word:

$$w_1 = a^{k+\ell} b^k \text{ and } k + \ell > k$$

However, the number of a 's in w_1 is not equal to the number of b 's, but it would have to be if w_1 belonged to L . Hence, $w_1 \notin L$, which is a contradiction.

Problem 214 Write a complete formal definition of a context-free grammar $G = (V, \Sigma, P, S)$ over alphabet $\{a, b, c\}$ such that G generates the language of all strings whose length is even or gives remainder 1 if divided by 3. If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S_1, S_2, D\}$, and P is:

$$\begin{aligned} S &\rightarrow S_1 \mid S_2 \\ S_1 &\rightarrow DDS_1 \mid \lambda \\ S_2 &\rightarrow DDDSS_2 \mid D \\ D &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 215 Let $\Sigma = \{a, b, c, g\}$; let L_1 be defined as follows:

$$L_1 = \{w \in \Sigma^* \mid w = a^k b^m c^n, \quad m > k + n, \quad k, m, n \geq 0\}$$

and let L_2 be defined as follows:

$$L_2 = \{w \in \Sigma^* \mid w = g^m b^k e^k a^n g^n a^m, \quad k, m, n \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar G_1 that generates language L_1 . If such a grammar does not exist, explain why.

Answer: Observe that: $m = k + \ell + n$, for some $\ell > 0$, whence:

$$a^k b^m c^n = a^k b^{k+\ell+n} c^n = a^k b^k b^\ell b^n c^n$$

These strings are generated by the grammar:

$G_1 = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, e, g\}$, $V = \{S, A, B, E\}$,

and the production set P is:

$$\begin{aligned} S &\rightarrow ABE \\ A &\rightarrow aAb \mid \lambda \\ E &\rightarrow bEe \mid \lambda \\ B &\rightarrow bB \mid b \end{aligned}$$

(b) Write a complete formal definition of a context-free grammar G_2 that generates language L_2 . If such a grammar does not exist, explain why.

Answer: $G_2 = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, e, g\}$, $V = \{S, A, B\}$,

and the production set P is:

$$\begin{aligned} S &\rightarrow gSa \mid BA \\ B &\rightarrow bBe \mid \lambda \\ A &\rightarrow aAg \mid \lambda \end{aligned}$$

Problem 216 Let:

$$L = \{a^{2i} b^j d^{3k} a^{\ell+2} \mid i = k, \text{ and } i, j, k, \ell \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, d\}$, $V = \{S, A, B, D\}$,

and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aaAddd \mid D \\ D &\rightarrow bD \mid \lambda \\ B &\rightarrow aB \mid aa \end{aligned}$$

Problem 217 Let:

$$L = \{a^{2i+3} d^{k+1} b^{3j+2} a^{\ell+1} \mid i = k, \quad j = \ell, \quad i, j, k, \ell \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, d\}$, $V = \{S, A, B\}$,

and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aaAd \mid aaad \\ B &\rightarrow bbbBa \mid bba \end{aligned}$$

Problem 218 Let:

$$L_1 = \{a^i b^j c^j \mid i, j \geq 0\}$$

and

$$L_2 = \{a^i b^i c^j \mid i, j \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates $L_1 \cup L_2$. If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:

$$\Sigma = \{a, b, c\}, V = \{S, S_1, S_2, A, B, D, E\},$$

and the production set P is:

$$\begin{aligned} S &\rightarrow S_1 \mid S_2 \\ S_1 &\rightarrow AB \\ S_2 &\rightarrow DE \\ A &\rightarrow aA \mid \lambda \\ B &\rightarrow bBc \mid \lambda \\ D &\rightarrow aDb \mid \lambda \\ E &\rightarrow cE \mid \lambda \end{aligned}$$

Problem 219 Let:

$$L_1 = \{a^i b^j c^j \mid i, j \geq 0\}$$

and

$$L_2 = \{a^i b^i c^j \mid i, j \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates $L_1 \cap L_2$. If such grammar does not exist, prove it.

Answer: Observe that:

$$L_1 \cap L_2 = \{a^j b^j c^j \mid j \geq 0\}$$

which is a canonical non-context-free language. Hence, there is no context-free grammar that generates it.

To prove that $L_1 \cap L_2$ is not context-free, assume the opposite—that the Pumping Lemma holds for the language.

Let k be the constant as in the Pumping Lemma for $L_1 \cap L_2$. Let $w = a^m b^m c^m$, such that $m > k$. Let $w = uvxyz$ be a decomposition of w , where vy is the pumping part.

Observe that every substring of w that contains all the three letters (a, b, c) must contain the entire run of b 's of length m , plus some a 's before this run of b 's and some c 's after it. Hence, every substring of w that contains all the three letters has length at least $m + 2$. By the Lemma, it must be that $|vxy| < k < m < m + 2$. Thus, the pumping part vy is too short to contain all the three letters—it lacks at least one of them, say letter ξ , for some $\xi \in \{a, b, c\}$.

After the pumping is admitted, any word of the form $uv^i xy^i z$, for $i > 1$, is claimed to be in $L_1 \cap L_2$. However, since the other two letters are pumped, while ξ is not pumped, such a word has a surplus of the other two letters, relative to ξ , thereby violating the pattern $a^j b^j c^j$ —a contradiction.

Problem 220 Let:

$$L = \{a^{2i+1} b^j d^{3k+2} a^{\ell+1} \mid i = k, \text{ and } i, j, k, \ell \geq 0\}$$

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer: Such automaton does not exist, since:

$$L = \{a^{2i+1} b^j d^{3i+2} a^{\ell+1} \mid i, j, \ell \geq 0\}$$

and this language is not regular.

To prove this, assume the opposite, that L is regular. Let η be the constant as in the Pumping Lemma for L . Let $m > \eta$; then $a^{2m+1} b^{3m+2} a \in L$, since $a^{2m+1} b^{3m+2} a = a^{2m+1} b^0 d^{3m+2} a^{0+1}$. In any “pumping” decomposition such that: $a^{2m+1} b^{3m+2} a = uvx$, we have: $|uv| \leq \eta < m < 2m + 1$, hence uv is a prefix of a^{2m+1} and the “pumping” substring v consists entirely of a 's, say $v = a^p$. Recall that $p > 0$, since the “pumping” substring cannot be empty. By the pumping, every word of the form $uv^i x$, $i \geq 0$, belongs to L . However, such a word has exactly $3m + 2$ occurrences of d and $2m + 1 + (i - 1)p$ occurrences of a . Whenever $i > 1$, it is true that $2m + 1 + (i - 1)p > 2m + 1$. Hence, in this case, such a word has more a 's than is appropriate for its number of d 's, and cannot belong to L , which is a contradiction.

(b) Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where

$$\Sigma = \{a, b, d\}, V = \{S, A, B, D\}, \text{ and } P \text{ is:}$$

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aaAddd \mid aDdd \\ D &\rightarrow DD \mid \lambda \mid b \\ B &\rightarrow aB \mid a \end{aligned}$$

(c) Is the grammar (if any) which you constructed in part (b) regular? Explain your answer.

Answer: No—the first production, for example, violates the prescribed form. More importantly, a regular grammar for L cannot exist, lest L would be regular, which is not the case, by the proof given in the answer to part (a).

Problem 221 Let:

$$L = \{a^n c^k d^\ell b^{j+3} d^m\}$$

where $k = n + 1$, $m = 0$, $\ell = n + 2$, $n, k, \ell, j, m \geq 0$.

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: Such a grammar does not exist, since L is not context-free. To prove that L is not context-free, first observe that

$$L = \{a^n c^{n+1} d^{n+2} b^{j+3} \mid n, j \geq 0\}$$

Next, consider the following two languages:

$$L_1 = \{a^n c^{n+1} d^{n+2} bbb \mid n, j \geq 0\}$$

$$R = a^* c^* d^* bbb$$

Observe that R is regular, but also:

$$L_1 = L \cap R$$

Now, if L was context free then its intersection with any regular language would be context-free. In particular, L_1 would be context-free, since it is the intersection of L with the regular language R . To prove that L is not context-free, it is then sufficient to prove that L_1 is not context-free.

Furthermore, observe that every string in the language L_1 satisfies the following three properties:

1. number of c 's is by one greater than the number of a 's;
2. number of d 's is by two greater than the number of a 's;
3. number of b 's is equal to three.

Assume that L_1 is context-free, and let η be the constant as in the Pumping Lemma for L . Select a word:

$$w = a^n c^{n+1} d^{n+2} bbb \text{ where } n > \eta$$

The word w is long enough and must pump. In any pumping decomposition: $w = xyz$, the length of the pumping window y_1ty_2 is not greater than η . Additionally, the substring bbb must remain outside any pumping window (or else pumping would produce more than three b 's, violating the third property.) Hence, the pumping window is located either entirely within one of the three substrings a^n, c^{n+1}, d^{n+2} , or it is shared by a^n and c^{n+1} , or it is shared by c^{n+1} and d^{n+2} , since it is too short to span more than two of these substrings. If the pumping window is within any one of the three named substrings, the pumping produces an excess of the letter pumped; if the window is shared by two substrings, the pumping produces a lack of the third letter, thereby in all cases leading to a word that violates at least one of the first two properties. Hence, L_1 does not honor the Pumping Lemma and thereby cannot be context-free, meaning that L cannot be context-free either.

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: This regular expression does not exist, since language L is not regular. To prove this, recall that L is not context-free, as is proved in the answer to part (a). However, if L was regular, it would be context-free.

Problem 222 Let:

$$L = \{a^n c^k d^\ell b^j d^{m+3}\}$$

where $n = k + 1, m = 0, \ell = j + 2, n, k, \ell, j, m \geq 0$.

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ABddd \\ A &\rightarrow aAc \mid a \\ B &\rightarrow dBb \mid dd \end{aligned}$$

(b) Draw a state transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: Such a finite automaton does not exist since L is not regular. To prove this, first observe that:

$$L = \{a^{k+1} c^k d^{\ell+2} b^j d^{m+3} \mid k, j \geq 0\}$$

Observe that all strings of L satisfy the following property:

1. number of a 's is equal to the number of c 's plus one.

Assume that L is regular, and let η be the constant as in the Pumping Lemma for L . Select a word:

$$w = a^{n+1} c^n ddd$$

for some $n > \eta$, obtained from the general template by setting $j = 0$. The word w is long enough and must pump. In any pumping decomposition: $w = xyz$, the length of the pumping window xy is not greater than η , and is thereby less than n . Hence, the pumping window y is located entirely within the segment a^{n+1} . Pumping up once produces an excess of a 's over c 's, violating the property. Hence, L does not honor the Pumping Lemma and thereby cannot be regular.

Problem 223 Let:

$$L = \{a^n c^k d^{\ell+3} b^j d^m\}$$

where $m = \ell + 1, j = 0, k = j + 1, n, k, \ell, j, m \geq 0$.

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: Observe that:

$$L = \{a^n c d^{2\ell+4} \mid n, \ell \geq 0\}$$

whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AcBdd \\ A &\rightarrow aA \mid \lambda \\ B &\rightarrow ddB \mid \lambda \end{aligned}$$

(b) Draw a state transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 6.

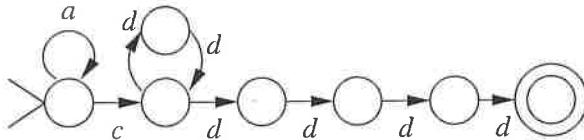


Figure 6:

Problem 224 Let:

$$L = \{a^n c^k d^\ell b^j d^m \mid k = n + p, \ell = j + p, m > 0\}$$

where $n, k, \ell, j, p, m \geq 0$.(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.**Answer:** The general template for the strings of L is:

$$L = \{a^n c^n c^p d^p d^j b^j d^m \mid n, j, p, m \geq 0\}$$

but $m > 0$, whence the grammar:
 $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$,
 $V = \{S, A, K, D, E\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AKDE \\ A &\rightarrow aAc \mid \lambda \\ K &\rightarrow cKd \mid \lambda \\ D &\rightarrow dDb \mid \lambda \\ E &\rightarrow dE \mid d \end{aligned}$$

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.**Answer:** Such a regular expression does not exist since L is not regular. To prove this, assume the opposite and observe that all strings of L satisfy the following property:

1. number of a 's is not greater than the number of c 's.

Let η be the constant as in the Pumping Lemma for L . Select a word:

$$w = a^n c^n d$$

for some $n > \eta$, obtained from the general template by setting $p = j = 0$ and $m = 1$. The word w is long enough and must pump. In any “pumping” decomposition: $w = xyz$, the length of the “pumping window” xy is not greater than η , and is thereby less than n . Hence, the “pumping window” y is located entirely within the segment a^n . Pumping up once produces an excess of a 's over c 's, violating property (1). Hence, L does not honor the Pumping Lemma and thereby cannot be regular.

(See also Problem 526.)

Problem 225 Let:

$$L = \{a^n c^k d^\ell b^j d^m \mid j > 0, \ell = m, n = 2\ell\}$$

where $n, k, \ell, j, p, m \geq 0$.(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.**Answer:** Such a grammar does not exist, since L is not context-free. To prove that L is not context-free, consider first the following two languages:

$$L_1 = \{a^{2m} d^m b d^m \mid m \geq 0\}$$

$$R = a^* d^* b d^*$$

Observe that R is regular, but also:

$$L_1 = L \cap R$$

Now, if L was context free then its intersection with any regular language would be context-free. In particular, L_1 would be context-free, since it is the intersection of L with the regular language R . To prove that L is not context-free, it is then sufficient to prove that L_1 is not context-free.Next, observe that every string in the language L_1 satisfies the following two properties:

1. number of a 's is equal to the number of d 's;
2. number of d 's to the left of the single b is equal to the number of d 's to the right of the single b .

Now, assume that L_1 is context-free, and let η be the constant as in the Pumping Lemma for L . Select a word:

$$w = a^{2m} d^m b d^m, m > \eta$$

The word w is long enough and must pump. In any “pumping” decomposition: $w = xyz$, the length of the “pumping window” y is not greater than η . Additionally, the single b must remain outside any pumping window (or else pumping would produce more than one b .) Hence, the “pumping window” is located either entirely within one of the three substrings a^{2m}, d^m, d^m , or it is shared by a^{2m} and d^m —it is too short to span more than two of these substrings, and it must not include the single b . If the “pumping window” is within a^{2m} then pumping up once produces an excess of a 's, violating property (1); if it is within any one of the two segments d^m , then pumping up once produces an excess of d 's, again violating property (1); if it is shared by a^{2m} and d^m , then pumping up once violates the property (2). Hence, L_1 does not honor the Pumping Lemma and thereby cannot be context-free, meaning that L cannot be context-free either.(b) Draw a state transition graph of a finite automaton that accept L . If such an automaton does not exist, prove it.**Answer:** This automaton does not exist, since language L is not regular. To prove this, recall that L is not context-free, as is proved in the answer to part (a). However, if L was regular, it would be context-free.

Problem 226 Let:

$$L = \{a^n c^k a^\ell b^j d^m \mid k = j, n > 2, \ell = 0, n, k, \ell, j, m \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: Observe that the template for L is:

$$a^{n+3} c^k b^k d^m$$

where $n, k, m \geq 0$, whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aaaABD \\ A &\rightarrow aA \mid \lambda \\ B &\rightarrow cBb \mid \lambda \\ D &\rightarrow dD \mid \lambda \end{aligned}$$

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: Such a regular expression does not exist since L is not regular. To prove that L is not regular, consider first the following two languages:

$$L_1 = \{aaa c^k b^k d^m \mid k, m \geq 0\}$$

$$R = aaa c^* b^* d^*$$

Observe that R is regular, but also:

$$L_1 = L \cap R$$

Now, if L was regular then its intersection with any regular language would be regular. In particular, L_1 would be regular, since it is the intersection of L with the regular language R . To prove that L is not regular, it is then sufficient to prove that L_1 is not regular.

Next, observe that every string in the language L_1 satisfies the following property:

1. number of c 's is equal to the number of b 's;
2. number of a 's is equal to 3.

Assume that L_1 is regular, and let η be the constant as in the Pumping Lemma for L . Select a word:

$$w = aaa c^k b^k, k > \eta$$

The word w is long enough and must pump. In any pumping decomposition: $w = xyz$, the length of the substring xy is not greater than η , which in turn is less than k . Hence, the pumping window y is located entirely within the segment $aaa c^k$. However, no part of the pumping window may reside in the segment aaa , since pumping any a 's would violate property (2). Then, the pumping window is entirely within the segment c^k , and pumping up once produces an excess of c 's, violating property (1). Thus L_1 violates the Pumping Lemma and thereby cannot be regular, meaning that L cannot be regular either.

Problem 227 Let:

$$L = \{a^i b^j c^k a^\ell \mid i = 2k, j = 2\ell, i, j, k, \ell \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: Such grammar does not exist, since L is not a context-free language. To prove this, assume the opposite, and let k be the constant as in the Pumping Lemma for L . Select a word $a^{2m} b^{2n} c^m a^n \in L$ such that $m > k$ and $n > k$. In any “pumping” decomposition: $a^{2m} b^{2n} c^m a^n = uvxyz$, the length of the “pumping window” vxy is not greater than k : $|vxy| \leq k < m$ and also $|vxy| \leq k < n$. Consider a non-empty, effectively “pumping” substring of the “pumping window”, which is at least one of v, y . There are two cases: this “pumping” substring either falls within one of the four substrings containing a single letter: a^{2m}, b^{2n}, c^m, a^n , or it spans two such substrings—it is too short to extend through as many as three of them. In the first case, the “pumping” produces a surplus of occurrences of one letter, over against what should be the matching number of occurrences of another letter. In the second case, the “pumping” produces two kinds of letters that are out of sequence. In both cases, the word produced by “pumping” violates the given form of the strings in L , whence the contradiction.

Problem 228 Let:

$$L = \{a^i b^k c^j a^\ell \mid i = 2k + 3, j = 2\ell + 1, i, j, k, \ell \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aaAb \mid aaa \\ B &\rightarrow ccBa \mid c \end{aligned}$$

Problem 229 Let:

$$L = \{a^i b^m c^{2i+1} d^{k+2} h^{2k} \mid i, m, k \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b, c, d, h\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aAcc \mid Dc \\ D &\rightarrow bD \mid \lambda \\ B &\rightarrow dBhh \mid dd \end{aligned}$$

(b) Is L recursively enumerable? Explain your answer briefly.

Answer: Yes—the answer to part (a) shows that L is context-free. Every context-free language is recursively enumerable.

Problem 230 Let:

$$L = \{a^i b^k c^{2i+1} d^{k+2} h^{2i} \mid i, k \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: L does not have a context-free grammar. To prove that L is not context-free, assume the opposite—that the Pumping Lemma holds for the language L .

Let ν be the constant as in the Pumping Lemma for L . Let $w = a^i b^k c^{2i+1} d^{k+2} h^{2i}$, such that $i > \nu$ and $k > \nu$. Let $w = uvxyz$ be a decomposition of w , where vy is the pumping part.

Consider a non-empty, effectively “pumping” substring of the “pumping window”, which is at least one of v , y . There are two cases. In the first case this “pumping” substring falls within one of the five segments containing a single letter: a^i , b^k , c^{2i+1} , d^{k+2} , h^{2i} . In this case, the “pumping” produces a surplus of occurrences of one letter, over against what should be the matching number of occurrences of another letter. To see this, observe that the number of occurrences of a, c, h is governed by i , while the number of occurrences of b, d is governed by k . In the second case; the “pumping” substring spans two of the five segments—it is too short to extend through as many as three of them. In this case, the “pumping” produces two kinds of letters that are out of sequence.

(b) Is L uncountable? Explain your answer briefly.

Answer: No— L is a subset of $\{a, b, c, d, h\}^*$, which is (infinite and) countable.

Problem 231 Let:

$$L_1 = \{a^i b^j c^k \mid i = 2j \text{ or } j = 2k, i, j, k \geq 0\}$$

and

$$L_2 = \{a^i b^j c^k \mid i = 2j \text{ and } j = 2k, i, j, k \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar that generates L_1 . If such grammar does not exist, prove it.

Answer: Observe that $L_1 = L' \cup L''$ where:

$$\begin{aligned} L' &= \{a^{2j} b^j c^k \mid j, k \geq 0\} \\ L'' &= \{a^i b^{2k} c^k \mid i, k \geq 0\} \end{aligned}$$

$G = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b, c\}$, $V = \{S, S', S'', A, B, D, E\}$, and P is:

$$\begin{aligned} S &\rightarrow S' \mid S'' \\ S' &\rightarrow AB \\ A &\rightarrow aaAb \mid \lambda \\ B &\rightarrow cB \mid \lambda \\ S'' &\rightarrow DE \\ D &\rightarrow aD \mid \lambda \\ E &\rightarrow bbEc \mid \lambda \end{aligned}$$

(b) Write a complete formal definition of a context-free grammar that generates L_2 . If such grammar does not exist, prove it.

Answer: L_2 does not have a context-free grammar. Observe that:

$$L_2 = \{a^{4k} b^{2k} c^k \mid k \geq 0\}$$

To prove that L_2 is not context-free, assume the opposite—that the Pumping Lemma holds for the language.

Let ν be the constant as in the Pumping Lemma for L_2 . Let $w = a^{4m} b^{2m} c^m$, such that $m > \nu$. Let $w = uvxyz$ be a decomposition of w , where vy is the pumping part.

Observe that every substring of w that contains all the three letters (a, b, c) must contain the entire run of b 's of length $2m$, plus some a 's before this run of b 's and some c 's after it. Hence, every substring of w that contains all the three letters has length at least $2m + 2$. By the Lemma, it must be that $|vxy| < \nu < m < 2m + 2$. Thus, the pumping part vy is too short to contain all the three letters—it lacks at least one of them, say letter ξ , for some $\xi \in \{a, b, c\}$.

After the pumping is admitted, any word of the form $uv^i xy^i z$, for $i > 1$, is claimed to be in L_2 . However, since the other two letters are pumped, while ξ is not pumped, such a word has a surplus of the other two letters, relative to ξ , thereby violating the pattern $a^{4k} b^{2k} c^k$ —a contradiction.

Problem 232 Let:

$$L = \{a^\ell b^j c^k d^m \mid m = 2k \text{ and } k = 2\ell, \ell, j, k, m \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: Such grammar does not exist, because L is not context-free. Observe that:

$$L = \{a^\ell b^j c^{2\ell} d^{4\ell} \mid \ell, j \geq 0\}$$

To prove that L is not context-free, assume the opposite—that the Pumping Lemma holds for the language L .

Let ν be the constant as in the Pumping Lemma for L . Let:

$$w = a^m b^0 c^{2m} d^{4m} = a^m c^{2m} d^{4m}$$

where $m > \nu$. Let $w = uvxyz$ be a decomposition of w , where vy is the pumping part.

Observe that every substring of w that contains all the three letters (a, c, d) must contain the entire run of c 's of length $2m$, plus some a 's before this run of c 's and some d 's after it. Hence, every substring of w that contains all the three letters has length at least $2m + 2$. By the Lemma, it must be that $|vxy| < \nu < m < 2m + 2$. Thus, the pumping part vy is too short to contain all the three letters—it lacks at least one of them, say letter ξ , for some $\xi \in \{a, b, c\}$.

After the pumping is admitted, any word of the form $uv^i xy^i z$, for $i > 1$, is claimed to be in L . However, since the other two letters are pumped, while ξ is not pumped, such a word has a surplus of the other two letters, relative to ξ , thereby violating the pattern $a^m c^{2m} d^{4m}$ —a contradiction.

(b) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer: Such regular expression does not exist— L cannot be regular because it is not context-free, as is proved in the answer to part (a).

Problem 233 Let:

$$L = \{a^\ell b^{2j} c^k d^{2m} \mid \ell, j, k, m \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D, E\}$, and P is:

$$\begin{aligned} S &\rightarrow ABED \\ A &\rightarrow aA \mid \lambda \\ B &\rightarrow bbB \mid \lambda \\ E &\rightarrow cE \mid \lambda \\ D &\rightarrow ddD \mid \lambda \end{aligned}$$

(b) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$a^*(bb)^*c^*(dd)^*$$

Problem 234 Let:

$$L = \{a^i b^j c^k d^m \mid i = 2m \text{ and } j = 2k, i, j, k, m \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Problem 235 Let:

$$L = \{a^i b^j c^k d^m \mid i = 2j \text{ and } j = 2k, i, j, k, m \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Problem 236 (a) Let:

$$L = \{d^{m+1} a^i b^j c^{2k} d^m \mid j = 2(i + k) + 3, i, j, k, m \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: The general template for strings in L is:

$$d d^m a^i b^{2i} bbb b^{2k} c^{2k} d^m \text{ for } i, k, m \geq 0$$

whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow dA \\ A &\rightarrow dAd \mid BbbbD \\ B &\rightarrow aBbb \mid \lambda \\ D &\rightarrow bbDDcc \mid \lambda \end{aligned}$$

(b) What is the cardinality of the class of languages that are not context-free? Answer by giving the exact number (if this set is finite) or by specifying if it is countable or uncountable. Explain your answer briefly.

Answer: The class of languages that are not context-free is infinite and uncountable. To see this, recall that the class of all languages is infinite and uncountable. Only countably many of them are context-free—every context-free grammar is a finite string, and the set of finite strings over any (non-empty) alphabet is (infinite and) countable.

Problem 237 (a) Let:

$$L = \{d^{m+1} a^i b^j c^{2k} d^m \mid j = i + m, i, j, k, m \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: The general template for strings in L is:

$$d^{m+1} a^i b^i b^m c^{2k} d^m \text{ for } i, k, m \geq 0$$

The required grammar does not exist, since L is not context-free. To prove this, assume the opposite, and let η be the constant as in the Pumping Lemma for L . Select a word $w = d^{m+1} b^m d^m$, which belongs to L because it is obtained by setting $i = k = 0$ in the general template, and select m such that $m > \eta$. In any “pumping” decomposition: $w = uvxyz$, the length of the “pumping window” vxy is not greater than η : $|vxy| \leq \eta < m$. Consider a non-empty, effectively “pumping” substring of the “pumping window”, which is at least one of v , y . There are two cases: this “pumping” substring either falls within one of the three substrings containing a single letter: d^{m+1} , b^m , d^m , or it spans two such substrings—it is too short to extend through as many as three of them. In the first case, the “pumping” produces a surplus of occurrences of one letter, over against what should be the matching number of occurrences of another letter. In the second case, the “pumping” produces two kinds of letters that are out of sequence. In both cases, the word produced by “pumping” violates the given form of the strings in L , whence the contradiction.

(b) Let \mathcal{L} be the class of all languages over alphabet $\{0, 1\}$. Let \mathcal{C} be the class of all compilers. State the cardinalities of \mathcal{L} and \mathcal{C} and determine which one is greater. Explain your answer briefly. (For finite sets, state exact numbers. For infinite sets, specify if countable or not.)

Answer: Class \mathcal{L} is infinite and uncountable, since it is the set of subsets of an infinite and countable set $\{0, 1\}^*$. Class \mathcal{C} is infinite and countable, since every compiler is a finite string of symbols in some programming language, and the set of all finite strings over any alphabet is countable. Thus:

$$|\mathcal{L}| > |\mathcal{C}|$$

Problem 238 (a) Let:

$$L_1 = \{a^{\ell+1}b^{j+2}c^{k+3}d^{m+4} \mid k = 2\ell, \ell, j, k, m \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L_1 . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AD \\ A &\rightarrow aAcc \mid aBccc \\ B &\rightarrow bB \mid bb \\ D &\rightarrow dD \mid dddd \end{aligned}$$

(b) Let:

$$L_2 = \{a^{\ell+1}b^{j+2}c^{k+3}d^{m+4} \mid k = 2n, \ell, j, k, m, n \geq 0\}$$

Write a regular expression that defines L_2 . If such regular expression does not exist, prove it.

Answer:

$$aa^*bbb^*ccc(cc)^*ddddd^*$$

Problem 239 (a) Let L_1 be a language over alphabet $\Sigma = \{a, b, c, d, e\}$, defined as follows:

$$L_1 = \{a^m b^n c^k d^\ell e^j \mid k = m + n, \ell = 2j, m, n, k, \ell, j \geq 0\}$$

Write a complete formal definition of a context-free grammar G_1 that generates language L_1 . If such grammar does not exist, prove it.

Answer: Observe that the general form of a string that belongs to L_1 is:

$$a^m b^n c^n c^m (dd)^j e^j$$

whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, e\}$ is the set of terminals; $V = \{S, A, B, D\}$ is the set of variables; S is the start symbol, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aAc \mid D \\ D &\rightarrow bDc \mid \lambda \\ B &\rightarrow ddBe \mid \lambda \end{aligned}$$

(b) Let L_2 be a language over alphabet $\Sigma = \{a, b, c, d, e\}$, defined as follows:

$$L_2 = \{a^m b^n c^k d^\ell e^j \mid k = m + n, \ell = 2k, m, n, k, \ell, j \geq 0\}$$

Write a complete formal definition of a context-free grammar G_2 that generates language L_2 . If such grammar does not exist, prove it.

Answer: Observe that the general form of a string that belongs to L_2 is:

$$a^m b^n c^{m+n} d^{2(m+n)} e^j$$

from which we see that L_2 is not context-free, and cannot have a context-free grammar.

To prove that L_2 is not context-free, assume the opposite—that the Pumping Lemma holds for the language L_2 .

Let ν be the constant as in the Pumping Lemma for L_2 . Consider a word $w = a^m b^n c^{m+n} d^{2(m+n)}$, such that $m + n > \nu$, obtained by setting $j = 0$ in the template for L_2 . Let $w = uvxyz$ be a decomposition of w , where vy is the pumping part.

Consider the “pumping” window vzy . There are two cases. In the first case the entire “pumping” falls within one of the three segments: $a^m b^n$, c^{m+n} , $d^{2(m+n)}$, where the length of each segment is governed by $m + n$. In this case, the “pumping” produces a surplus of occurrences of one letter type, over against what should be the matching number of occurrences of another letter type, where the letter types of the three segments are: $\{a, b\}$, $\{c\}$, and $\{d\}$. In the second case, the “pumping” substring spans two of the three segments—it is too short to extend through as many as three of them. In this case, letters of two types are “pumped”, producing a deficit of letters of the third type.

Problem 240 Let:

$$L = \{(bab)^n (aba)^k b^\ell (cdc)^j b^m \mid n = 0, \ell = p+1, j = 2\ell\}$$

where: $j, k, \ell, m, n, p \geq 0$

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The template for the strings of L is:

$$L = \{(aba)^k b^{p+1} (cdc)^{2p+2} b^m \text{ where } k, p, m \geq 0\}$$

whence the grammar: $G = (V, \Sigma, P, S)$, $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ADB \\ A &\rightarrow abaA \mid \lambda \\ D &\rightarrow bDcdecdc \mid bcdeccdc \\ B &\rightarrow bB \mid \lambda \end{aligned}$$

(b) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: This automaton does not exist since L is not regular. To prove that L is not regular, first observe that every string in the language L satisfies the following property:

the number of d 's is equal to twice the number of b 's that appear to the left of the first c .

Assume that L is regular, and let η be the constant as in the Pumping Lemma for L . Select a word:

$$w = b^{p+1} (cdc)^{2p+2}, \text{ for some } p > \eta$$

obtained by setting $k = m = 0$ in the template of L . The word w is long enough and must pump. In any pumping decomposition: $w = xyz$, the length of the substring $|xy|$ is not greater than η , which in turn is less than p . Hence, the pumping window y is located entirely within the segment b^{p+1} , and pumping up once produces a word which has $p + 1 + |y| > p + 1$ b 's but still only $2p + 2$ d 's, thereby violating the considered property. Since the Pumping Lemma does not hold for L , L is not regular.

Problem 241 Let:

$$L = \{(dbd)^k a^\ell (bcm)^j a^m (aba)^n \mid j = 0, \ell > 3, n = 2p\}$$

where: $j, k, \ell, m, n, p \geq 0$

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The template for the strings of L is:

$$L = \{(dbd)^k aaaa a^i (aba)^{2p} \text{ where } k, i, p \geq 0\}$$

whence the grammar: $G = (V, \Sigma, P, S)$, $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow DaAAAB \\ D &\rightarrow dbdD \mid \lambda \\ A &\rightarrow aA \mid \lambda \\ B &\rightarrow abaabaB \mid \lambda \end{aligned}$$

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(dbd)^* aaaa a^* (abaaba)^*$$

Problem 242 Let:

$$L = \{c^{n+2} a^m b^j a^p d^{\ell+1} \mid n = j + p \wedge p = \ell\}$$

where $m, n, j, p, \ell \in N$.

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The general template for the strings of L is:

$$cc c^j c^m a^m b^j a^p d^p d$$

whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ccABd \\ A &\rightarrow cAb \mid D \\ D &\rightarrow cDa \mid \lambda \\ B &\rightarrow aBd \mid \lambda \end{aligned}$$

(b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: This automaton does not exist, since L is not regular.

To prove this, assume the opposite—that L is regular. Observe that every word of L satisfies the following property:

the number of c 's is greater by 2 than the number of b 's plus the number of a 's between c 's and b 's.

Let k be the constant as in the Pumping Lemma for L . Consider a word $w = c^{2+j+m} a^m b^j a^p d^{p+1}$, where $j > k$.

Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| \leq k < j$. Hence, the pumping part v is entirely within the c -segment, meaning that $v = c^\ell$ for some ℓ such that $0 < \ell \leq k$.

After pumping once (up) we obtain a word: $w_1 = c^{2+j+m+\ell} a^m b^j a^p d^{p+1}$, where there are $2 + j + m + \ell$ c 's, but still only $j + m$ b 's and relevant a 's. This means that w_1 violates the stated property of L and $w_1 \notin L$, whence the contradiction.

(c) Is L decidable? Prove your answer.

Answer: Yes. L is context free (as is demonstrated in the answer to part (a)), and every context language is decidable.

(d) Is L recursively enumerable? Prove your answer.

Answer: Yes. L is context free, and every context language is recursively enumerable.

Problem 243 Let:

$$L = \{c^{n+2} a^m b^j a^p d^{\ell+1} \mid n = j + p \wedge m = 0\}$$

where $m, n, j, p, \ell \in N$:

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The general template for the strings of L is:

$$c^p c^j cc b^j a^p d^{\ell+1}$$

whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AD \\ A &\rightarrow cAa \mid B \\ B &\rightarrow cBb \mid cc \\ D &\rightarrow dD \mid d \end{aligned}$$

(b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: This automaton does not exist, since L is not regular.

To prove this, assume the opposite—that L is regular. Observe that every word of L satisfies the following property:

the number of c 's is greater by 2 than the number of b 's plus the number of a 's.

Let k be the constant as in the Pumping Lemma for L . Consider a word $w = c^{2+p+j} b^j a^p d^{\ell+1}$, where $j > k$.

Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| \leq k < j$. Hence, the pumping part v is entirely within the c -segment, meaning that $v = c^m$ for some m such that $0 < m \leq k$.

After pumping once (up) we obtain a word: $w_1 = c^{2+p+j+m} b^j a^p d^{\ell+1}$, where there are $2 + p + j + m$ c 's, but still only $p + j$ b 's and a 's. This means that w_1 violates the stated property of L and $w_1 \notin L$, whence the contradiction.

(c) Is L recursively enumerable? Prove your answer.

Answer: Yes. L is context free (as is demonstrated in the answer to part (a)), and every context language is recursively enumerable.

(d) Is L decidable? Prove your answer.

Answer: Yes. L is context free, and every context language is decidable.

Problem 244 Let:

$$L = \{c^{n+2} a^m b^j a^p d^{\ell+1} \mid j = n \wedge m = p\}$$

where $m, n, j, p, \ell \in N$.

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: Such a grammar does not exist, since L is not context-free. To prove that L is not context-free, recall that

$$L = \{c^{n+2} a^m b^n a^m d^{\ell+1} \mid m, n, \ell \geq 0\}$$

and consider the following two languages:

$$L_1 = \{c^{n+2} a^m b^n a^m d \mid m, n \geq 0\}$$

$$R = c^* a^* b^* a^* d$$

Observe that R is regular, but also:

$$L_1 = L \cap R$$

If L was context free then its intersection with any regular language would be context-free. L_1 would be context-free, since it is the intersection of L with the regular language R . To prove that L is not context-free, it is sufficient to prove that L_1 is not context-free. Observe that every string in the language L_1 satisfies all the following properties:

1. number of c 's is by two greater than the number of b 's;

2. number of a 's in the first a -segment is equal to the number a 's in the second a -segment;
3. number of d 's is equal to one.

Assume that L_1 is context-free, and let η be the constant as in the Pumping Lemma for L . Select a word:

$$w = c^{n+2} a^m b^n a^m d \text{ where } n, m > \eta$$

The word w is long enough and must pump. In any pumping decomposition: $w = xy_1ty_2z$, the length of the pumping window y_1ty_2 is not greater than η . Additionally, the substring d must remain outside any pumping window (or else pumping would produce more than one d 's, violating the third property.) Hence, the pumping window is located either entirely within one of the four substrings c^{n+2}, a^m, b^n, a^m , or it is shared by two adjacent ones, since it is too short to span more than two of these substrings. If the pumping window is within any one of the four named substrings, the pumping produces an excess of the letter pumped; if the window is shared by two substrings, the pumping produces a lack of a third letter, thereby in all cases leading to a word that violates at least one of the first two properties. Hence, L_1 violates the Pumping Lemma and thereby cannot be context-free, meaning that L cannot be context-free either.

(b) Write a complete formal definition of a context-free grammar that generates $L \cap c^* a^* d^*$. If such a grammar does not exist, prove it.

Answer: The template for elements of $L \cap c^* a^* d^*$ is:

$$L = \{c^2 a^{2m} d^{\ell+1} \mid m, \ell \geq 0\}$$

whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, c, d\}$, $V = \{S, A, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ccAD \\ A &\rightarrow aaA \mid \lambda \\ D &\rightarrow dD \mid d \end{aligned}$$

Problem 245 Let L be the set of those strings over the alphabet $\{a, b, c\}$ such that all c 's come before any b 's, all b 's come before any a 's, the number of a 's is equal to $n + 1$, the number of b 's is equal to $n + 2$, and the number of c 's is equal to $n + 3$ for some natural number n .

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: Such a grammar does not exist, since L is not context-free. To prove this, we show that Pumping Lemma does not hold for L .

Observe that the general template for strings of L is:

$$c^{n+3} b^{n+2} a^{n+1}$$

Assume the opposite, that L is context-free. Let k be the constant as in the Pumping Lemma for L . Select $n > k$ and consider a word $w = c^{n+3} b^{n+2} a^{n+1} \in L$,

which must pump. In any pumping decomposition, the pumping window is entirely within one of the following three segments: $c^{n+3}, b^{n+2}, a^{n+1}$ or it spans two adjacent segments, since the pumping window is too short to extend into more than two of them, because its length is less than k , and thereby less than n . Thus, if we pump up once, there will be at least one of the segments $c^{n+3}, b^{n+2}, a^{n+1}$ where pumping occurs and at least one where it does not occur. The number of occurrences of at least one of the letters c, b, a will have increased, while the number of occurrences of at least one other of these letters will not have changed. The new word will violate the template and thus will not belong to L . Since L violates the Pumping Lemma, L is not context free.

(b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: This automaton does not exist, since L is not regular. If L was regular, then L would be content free, because every regular language is content free. However, we have proved that L is not context free, hence L cannot be regular.

(c) State the cardinality of L . If L is finite, state the exact number; if L is infinite, specify whether it is countable or not countable.

Answer: L is infinite and countable.

(d) State the cardinality of L^* . If L^* is finite, state the exact number; if L^* is infinite, specify whether it is countable or not countable.

Answer: L^* is infinite and countable.

Problem 246 Let L be a language over the alphabet $\{a, b, c, d\}$, defined as follows:

$$L = \{a^{2\ell}b^n c^n d^\ell \mid n, \ell \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$ is the set of terminals; $V = \{S, A\}$ is the set of variables; S is the start symbol, and the production set P is:

$$\begin{aligned} S &\rightarrow aaSd \mid A \\ A &\rightarrow bAc \mid \lambda \end{aligned}$$

Problem 247 Let L be a language over the alphabet $\{a, b, c, d\}$, defined as follows:

$$L = \{a^\ell b^k c^j d^k b^\ell \mid j, k, \ell \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aSb \mid A \\ A &\rightarrow bAd \mid B \\ B &\rightarrow cB \mid \lambda \end{aligned}$$

Problem 248 Let L be a language over alphabet $\Sigma = \{a, b, c, d, e\}$, defined as follows:

$$L = \{a^{m+1}b^{2n}c^{k+2}d^{3\ell}e^{j+3} \mid n = 2m, \ell = k + j\}$$

where $m, n, k, \ell, j \geq 0$.

(a) Write a complete formal definition of a context-free grammar G that generates language L . If such grammar does not exist, prove it.

Answer: The template for strings of L is:

$$a^{m+1}b^{4m}c^{k+2}d^{3k}e^{3j} \mid m, k, j \geq 0$$

whence the grammar: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b, c, d, e\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow ABD \\ A &\rightarrow aAbbbb \mid a \\ B &\rightarrow cBddd \mid cc \\ D &\rightarrow dddDe \mid eee \end{aligned}$$

(b) Let \mathcal{S} be a class of languages over alphabet $\{a, b, c, d, e\}$, defined as follows:

Language L is a member of \mathcal{S} if and only if L is generated by some context-free grammar, but there does not exist a pushdown automaton that accepts L .

What is the cardinality of \mathcal{S} ? Explain your answer briefly.

Answer:

$$|\mathcal{S}| = 0, \text{ since } \mathcal{S} = \emptyset$$

Class \mathcal{S} is empty, since every language generated by some context-free grammar is also accepted by some pushdown automaton. In fact, this pushdown automaton is obtained by an algorithmic conversion of the original context-free grammar.

Problem 249 Let:

$$L = \{(ab)^n c^{k+1} d^{\ell+3} (ge)^{n+1} d^k \mid n, k, \ell \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: Such a grammar does not exist, since L is not context-free. To prove that L is not context-free, consider the language L_1 , defined as the intersection of L and the regular language:

$$(ab)^* c^* ddd (ge)^* d^*$$

Then:

$$L_1 = \{(ab)^n c^{k+1} ddd (ge)^{n+1} d^k \mid n, k \geq 0\}$$

In other words, strings of L_1 contain exactly three d 's to the left of the first g , while strings of L contain at least three such d 's. If L was context-free, then L_1 would also be context-free, since the intersection of a context-free language with any regular language is context-free. However, we show that L_1 is not context-free.

Observe that every word of L_1 satisfies the following property:

- the number of c 's is by one greater than the number of final d 's;
- the number of (ab) 's is by one less than the number of (ge) 's;

To prove that L_1 is not context-free, assume the opposite—that L_1 is context-free. Let η be the constant as in the Pumping Lemma for L . Select $n > \eta, k > \eta$ and consider a word $w = (ab)^n c^{k+1} ddd (ge)^{n+1} d^k$. The word w must pump. By the Lemma, in any pumping decomposition, the pumping window cannot intersect with the 3-symbol segment ddd , since correct strings contain exactly three d 's, which thus cannot be pumped. Then it must be that the pumping window either is entirely within one of the following four segments: $(ab)^n, c^{k+1}, (ge)^{n+1}, dk$ or it spans two adjacent segments, since the pumping window is too short to extend into more than two of them. Hence, by pumping up we obtain a word where the number of occurrences of at least one letter has increased, while the number of occurrences of at least one other letter essential for maintaining the stated property of L_1 has not changed. This means that the new word violates the stated property, whence the contradiction.

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: Such a regular expression does not exist, since L is not regular. To see this, recall that L is not context-free, as is proved in the answer to part (a). If L was regular, it would be context-free (since every regular language is context-free.)

Problem 250 Let:

$$L = \{c^n (ab)^n d^{n+k} (ge)^{k+n} \mid n, k \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: Such a grammar does not exist, since L is not context-free.

Observe that every word of L satisfies all of the following properties.

1. the number of c 's is equal to the number of (ab) 's;

2. the number of d 's is equal to the number of (ge) 's;
3. the number of d 's is not less than the number of c 's;

To prove that L is not context-free, assume the opposite—that L is context-free. Let η be the constant as in the Pumping Lemma for L . Select $n > \eta, k = 0$ and consider a word $w = c^n (ab)^n d^n (ge)^n$. The word w must pump. By the Lemma, in any pumping decomposition, the pumping window cannot extend through more than two of the four segments—it is too short to intersect with three (or more) of them. Then all the cases of acceptable positions of the pumping window and the corresponding pumping instances which violate the stated properties are as follows.

1. the pumping window is entirely within the segment c^n : pump once up \Rightarrow property (1) is lost;
2. the pumping window is entirely within the segment $(ab)^n$: pump once up \Rightarrow property (1) is lost;
3. the pumping window is entirely within the segment d^n : pump once up \Rightarrow property (2) is lost;
4. the pumping window is entirely within the segment $(ge)^n$: pump once up \Rightarrow property (2) is lost;
5. the pumping window contains parts of segments $c^n, (ab)^n$: pump once up \Rightarrow property (3) is lost;
6. the pumping window contains parts of segments $(ab)^n, d^n$: pump once up \Rightarrow property (2) is lost;
7. the pumping window contains parts of segments $d^n, (ge)^n$: pump once down \Rightarrow property (3) is lost;

Hence, the Pumping Lemma does not hold for L and L is not context-free.

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: Such a regular expression does not exist, since L is not regular. To see this, recall that L is not context-free, as is proved in the answer to part (a). If L was regular, it would be context-free (since every regular language is context-free.)

Problem 251 Let:

$$L = \{(ab)^n c^{k+1} (ge)^{n+2} \mid n, k \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, g, e\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow abSge \mid Agege \\ A &\rightarrow cA \mid c \end{aligned}$$

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: Such a regular expression does not exist, since L is not regular. To prove this, assume the opposite—that L is regular.

Observe that every word of L satisfies the following property:

The number of a 's is equal to the number of b 's, which is by 2 less than the number of g 's, which is in turn equal to the number of e 's.

Let η be the constant as in the Pumping Lemma for L . Consider a word $w = (ab)^n c (ge)^{n+2}$, where $n > \eta$, obtained from the general template by setting $k = 0$.

Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < \eta < n$. Hence, the pumping part v is entirely within the (ab) -segment, and after pumping once (up) we obtain a word where the number of a 's and/or b 's is greater than n , while the number of g 's and e 's has remained unchanged. This means that the new word violates the stated property of L , whence the contradiction.

Problem 252 Let:

$$L = \{a^n b^n c^{k+1} g^{n+2} e^{n+2} \mid n, k \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: Such a grammar does not exist, since L is not context-free. To prove that L is not context-free, consider the language L_1 , defined as the intersection of L and the regular language:

$$a^* b^* c g^* e^*$$

Then:

$$L_1 = \{a^n b^n c g^{n+2} e^{n+2} \mid n, k \geq 0\}$$

In other words, strings of L_1 contain exactly one c while strings of L contain at least one c . If L was context-free, then L_1 would also be context-free, since the intersection of a context-free language with any regular language is context-free. However, we show that L_1 is not context-free.

To prove that L_1 is not context-free, assume the opposite—that L_1 is context-free. Observe that every word of L_1 satisfies the following property:

The number of a 's is equal to the number of b 's, which is by 2 less than the number of g 's, which is in turn equal to the number of c 's, and there is exactly one c .

Let η be the constant as in the Pumping Lemma for L . Consider a word $w = a^n b^n c g^{n+2} e^{n+2}$, where $n > \eta$.

By the Lemma, in any pumping decomposition, the pumping window cannot contain c , since correct strings contain exactly one c , which thus cannot be pumped. Then it must be that the pumping window either is entirely within one of the following four segments: $a^n, b^n, g^{n+2}, e^{n+2}$ or it spans two adjacent segments, since the pumping window is too short to extend into more than two of them. Hence, by pumping up we obtain a word where the number of occurrences of at least one letter has increased, while the number of occurrences of at least one other letter has not changed. This means that the new word violates the stated property of L_1 , whence the contradiction.

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: Such a regular expression does not exist, since L is not regular. To see this, recall that L is not context-free, as is proved in the answer to part (a). If L was regular, it would be context-free (since every regular language is context-free.)

Problem 253 Let L be a language over alphabet $\Sigma = \{a, b, c, d, e\}$, defined as follows:

$$L = \{a^{m+1} b^n c^{k+2} d^\ell e^{j+3} \mid n = 2k, \ell = m + j\}$$

where $m, n, k, \ell, j \geq 0$.

(a) Write a complete formal definition of a context-free grammar G that generates L . If such a grammar does not exist, prove it.

Answer: The general template of the strings in L is:

$$L = \{a^{m+1} b^{2k} c^{k+2} d^m d^j e^{j+3} \mid m, k, j \geq 0\}$$

This language is generated by the grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, e\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aAd \mid aD \\ D &\rightarrow bbDc \mid cc \\ B &\rightarrow dBe \mid eee \end{aligned}$$

(b) Let \mathcal{S} be a class of languages over alphabet $\{a, b, c, d, e\}$, defined as follows:

Language T is a member of \mathcal{S} if and only if T is accepted by some finite automaton, but there does not exist a context-free grammar that generates T .

What is the cardinality of \mathcal{S} ? Explain your answer.

Answer:

$$|\mathcal{S}| = 0, \text{ since } \mathcal{S} = \emptyset$$

Class \mathcal{S} is empty, since for every finite automaton there exists an equivalent context-free grammar—the conversion is algorithmic.

Problem 254 Let L be a language over alphabet $\Sigma = \{a, b, c\}$, defined as follows:

$$L = \{a^m b^n c^k \mid n = 2j + 3, k = 3\ell + 2p + 5\}$$

where $m, n, k, \ell, j, p \geq 0$.

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: The general form of the strings that belong to the language L is:

$$a^m b^{2j+3} c^{3\ell+2p+5} \text{ where } m, j, \ell, p \geq 0$$

This language is represented by the regular expression:

$$a^* bbb (bb)^* ccccc (ccc)^* (cc)^*$$

(b) Let \mathcal{S} be a class of languages over alphabet $\{a, b, c\}$, defined as follows:

Language T is a member of \mathcal{S} if and only if T is accepted by some non-deterministic finite automaton, but there does not exist a regular context-free grammar that generates T .

What is the cardinality of \mathcal{S} ? Explain your answer.

Answer:

$$|\mathcal{S}| = 0 \text{ since } \mathcal{S} = \emptyset$$

Class \mathcal{S} is empty, since for every finite automaton there exists an equivalent regular context-free grammar—the conversion is, in fact, algorithmic.

Problem 255 (a) Let L_1 be a language over alphabet $\{a, b, c, d, e\}$, defined as follows:

$$L_1 = \{a^n b^m c^k d^{m+2} e^{n+3} \mid k, n, m \geq 0\}$$

Write a complete formal definition of a context-free grammar G_1 that generates language L_1 . If such a grammar does not exist, explain why.

Answer: $G_1 = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b, c, d, e\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aSe \mid Acee \\ A &\rightarrow bAd \mid Bdd \\ B &\rightarrow cB \mid \lambda \end{aligned}$$

(b) Let L_2 be a language over alphabet $\{a, b, c, d, e\}$, defined as follows:

$$L_2 = \{a^{2n} b^{3n} c^{k+2} d^{3k+1} e^{m+3} a^{2m+5} \mid k, n, m \geq 0\}$$

Write a complete formal definition of a context-free grammar G_2 that generates language L_2 . If such a grammar does not exist, explain why.

Answer: $G_2 = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b, c, d, e\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ABD \\ A &\rightarrow aaAbbb \mid \lambda \\ B &\rightarrow cBddd \mid ccd \\ D &\rightarrow cDaa \mid eeeaaaaa \end{aligned}$$

Problem 256 (a) Let L_1 be a language over the alphabet $\{a, b, d\}$, defined as follows:

$$L_1 = \{a^m b^{4n} d^k b^{2\ell} a^j \mid k=2m \wedge j=m \wedge j, k, \ell, m, n \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L_1 . If such a grammar does not exist, prove it.

Answer: The required grammar does not exist, since L_1 is not contest-free.

Observe that the general template for the strings that belong to L_1 is:

$$a^m b^{4n} d^{2m} b^{2\ell} a^m \text{ where } m, n \geq 0$$

To prove that L_1 is not context-free, assume the opposite, and let η be the constant as in the Pumping Lemma for L_1 . Select a word:

$$w = a^m d^{2m} a^m \text{ where } m > \eta$$

obtained by setting $n = \ell = 0$ in the general template. In any “pumping” decomposition: $w = uvxyz$, the length of the “pumping window” vxy is not greater than η : $|vxy| \leq \eta < m < 2m$. The “pumping” window vxy either falls within one of the three substrings containing a single letter: a^m, d^{2m}, a^m , or it spans two such adjacent substrings—it is too short to extend through as many as three of them. If the pumping is only within one of the designated substrings, it will make this substring longer than is appropriate with respect to the length of another substring. In the other case, if two adjacent letters are pumped, then there will still exist a third substring with fewer letters than required (in addition to possible appearance of letters out of order.) Hence, pumping produces strings that are not in L_1 , which is a contradiction that proves that L_1 cannot be context-free since it violates the Pumping Lemma.

(b) Let L_2 be a language over the alphabet $\{a, b, d\}$, defined as follows:

$$L_2 = \{a^m b^{4n} d^k b^{2\ell} a^j \mid k=2m \wedge j=\ell \wedge j, k, \ell, m, n \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L_2 . If such a grammar does not exist, prove it.

Answer: Observe that the general template for the strings of L_2 is:

$$a^m b^{4n} d^{2m} b^{2\ell} a^\ell \text{ where } m, n, \ell \geq 0$$

Hence, L_2 is generated by the grammar $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b, d\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aAdd \mid D \\ D &\rightarrow bbbbD \mid \lambda \\ B &\rightarrow bbBa \mid \lambda \end{aligned}$$

Problem 257 Let L be a language over the alphabet $\{a, b, c\}$, defined as follows:

$$L = \{a^\ell b^n c^j b^m a^p \mid \ell, n, j, m, p \geq 0 \wedge j = 2n \wedge \ell = 2m \wedge m = 2p\}$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The general template for strings in L is:

$$L = a^{4p} b^n c^{2n} b^{2p} a^p, \text{ where } n, p \geq 0$$

The required grammar does not exist, since L is not context-free. To prove this, assume the opposite, and let k be the constant as in the Pumping Lemma for L . Observe that the template for all words of L that do not contain any c 's is: $w = a^{4p} b^{2p} a^p$, $p \geq 0$. and is obtained by setting $n = 0$ in the general template for L . Select a word $w = a^{4p} b^{2p} a^p$, $p > k$. In any “pumping” decomposition: $w = uvxyz$, the length of the “pumping window” vxy is not greater than k : $|vxy| \leq k < k$. The “pumping” window vxy either falls within one of the three substrings containing a single letter: a^{4p} , b^{2p} , a^p , or it spans two such substrings—it is too short to extend through as many as three of them. In all cases, it omits at least one of the letters, so that pumping results in the deficit of that letter with respect to others.

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: This language is not context-free, as is proved in the answer to part (a). Since it is not context-free, it is not regular, and cannot have a regular expression.

Problem 258 (a) Let L_1 be a language over the alphabet $\{a, b, c\}$, defined as follows:

$$L_1 = \{a^\ell b^n c^j b^m a^p \mid \ell, n, j, m, p \geq 0 \wedge \ell = j = p\}$$

Write a complete formal definition of a context-free grammar that generates L_1 . If such a grammar does not exist, prove it.

Answer: The general template for strings in L_1 is:

$$L_1 = a^p b^n c^p b^m a^p, \text{ where } m, n, p \geq 0$$

The required grammar does not exist, since L_1 is not context-free. To prove this, assume the opposite, and let k be the constant as in the Pumping Lemma for L_1 . Observe that the template for all words of L_1 that do not contain any b 's is: $w = a^p c^p a^p$, $p \geq 0$, and is obtained by setting $m = n = 0$ in the general template for L_1 . Select a word $w = a^p c^p a^p$, $p > k$. In any “pumping” decomposition: $w = uvxyz$, the length of the “pumping window” vxy is not greater than k : $|vxy| \leq k < p$. The “pumping” window vxy either falls within one of the three substrings containing a single letter: a^p , c^p , a^p , or it spans two such substrings—it is too short to extend through as many as three of them. In all cases, it omits at least one of the letters, so that pumping results in the deficit of that letter with respect to others.

at least one of the letters, so that pumping results in the deficit of that letter with respect to others.

(b) Let L_2 be a language over the alphabet $\{a, b, c\}$, defined as follows:

$$L_2 = \{a^\ell b^n c^j b^m a^p \mid \ell, n, j, m, p \geq 0 \wedge \ell = j + p\}$$

Write a complete formal definition of a context-free grammar that generates L_2 . If such a grammar does not exist, prove it.

Answer: Observe that the general template for strings in L_2 is:

$$L_2 = a^p a^j b^n c^j b^m a^p, \text{ where } j, m, n, p \geq 0$$

whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aSa \mid DB \\ B &\rightarrow bB \mid \lambda \\ D &\rightarrow aDc \mid B \end{aligned}$$

Problem 259 Let L be a language over the alphabet $\{a, b, c, d\}$, defined as follows:

$$L = \{a^\ell b^n c^j d^m b^p \mid j, \ell, m, n, p \geq 0 \wedge j = \ell \wedge m = p\}$$

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow AD \\ A &\rightarrow aAc \mid B \\ B &\rightarrow bB \mid \lambda \\ D &\rightarrow dDb \mid \lambda \end{aligned}$$

Problem 260 Let L be a language over the alphabet $\{a, b, c, d\}$, defined as follows:

$$L = \{a^\ell b^n c^j d^m b^p \mid j, \ell, m, n, p \geq 0 \wedge j = \ell + n \wedge m = 2p+1\}$$

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow AD \\ A &\rightarrow aAc \mid B \\ B &\rightarrow bBc \mid \lambda \\ D &\rightarrow ddDb \mid d \end{aligned}$$

Problem 261 Let L be a language over the alphabet $\{a, b, c\}$, defined as follows:

$$L = \{a^\ell b^j c^m a^p \mid \ell, j, m, p \geq 0 \wedge j = 2m \wedge m = 2p\}$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The required grammar does not exist, since L is not context-free. To prove this, assume the opposite, and let k be the constant as in the Pumping Lemma for L . Observe that the template for all words of L is: $a^\ell b^{4p} c^{2p} a^p$, $\ell, p \geq 0$.

Observe that all words on L have to satisfy the following two claims.

number of c 's is equal to twice the number of trailing a 's;

number of b 's is equal to twice the number of c 's;

Select a word $w = b^{4p} c^{2p} a^p$, $p > k$, obtained from the general template by setting $\ell = 0$. (Observe that the only a 's are trailing a 's, and no pumping can produce any leading a 's.)

In any “pumping” decomposition: $w = uvxyz$, the length of the “pumping window” vxy is not greater than k : $|vxy| \leq k < p$. The “pumping” window vxy either falls within one of the three substrings containing a single letter: b^{4p} , c^{2p} , a^p , or it spans two such substrings—it is too short to extend through as many as three of them. In all cases, it omits at least one of the letters, so that pumping results in the deficit of that letter with respect to others.

(b) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: The required automaton does not exist, since L is not regular—by the answer to the part (a), L is not even context-free, but it would have to be context-free if it was regular, since every regular language is context-free.

Problem 262 Let L be a language over the alphabet $\{a, b, c\}$, defined as follows:

$$L = \{a^\ell b^j c^m a^p \mid \ell, j, m, p \geq 0 \wedge j = 2\ell \wedge m = 2p\}$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The general template for strings in L is: $a^\ell b^{2\ell} c^{2p} a^p$, which is generated by the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aAbb \mid \lambda \\ B &\rightarrow ccBa \mid \lambda \end{aligned}$$

(b) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: This finite automaton does not exist, since L is not regular.

To prove that L is not regular, assume the opposite, that L is regular. Let k be the constant as in the Pumping Lemma for L . Consider a word $w \in L$, such that $w = a^m b^{2m}$, for some $m > k$; w is obtained from the general template by setting $p = 0$.

Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < k < m$. Hence, the pumping part v is entirely within the a -segment, meaning that $v = a^j$ for some j such that $0 < j \leq k$. After pumping once, we obtain a word: $w_p = a^{m+j} b^{2m}$. However, $m + j > m$, meaning that w_p has more a 's than bb -pairs, which in turn implies that $w_p \notin L$.

Problem 263 (a) Let L_1 be a language over the alphabet $\{a, b, c\}$, defined as follows:

$$L_1 = \{a^\ell b^j c^m \mid \ell, j, m \geq 0 \wedge \ell = j = m\}$$

Write a complete formal definition of a context-free grammar that generates L_1 . If such a grammar does not exist, prove it.

Answer: The required grammar does not exist, since L_1 is not context-free. To prove this, assume the opposite, and let k be the constant as in the Pumping Lemma for L_1 . The template for all words of L_1 is: $a^\ell b^\ell c^\ell$, $\ell \geq 0$. Select a word $w = a^\ell b^\ell c^\ell$, $\ell > k$. In any “pumping” decomposition: $w = uvxyz$, the length of the “pumping window” vxy is not greater than k : $|vxy| \leq k < \ell$. The “pumping” window vxy either falls within one of the three substrings containing a single letter: a^ℓ , b^ℓ , c^ℓ , or it spans two such substrings—it is too short to extend through as many as all three of them. In all cases, it omits at least one of the letters, so that pumping results in the deficit of that letter with respect to others.

(b) Let L_2 be a language over the alphabet $\{a, b, c\}$, defined as follows:

$$L_2 = \{a^\ell b^j c^m \mid \ell, j, m \geq 0 \wedge \ell = j + m\}$$

Write a complete formal definition of a context-free grammar that generates L_2 . If such a grammar does not exist, prove it.

Answer: The general template for strings in L is: $a^m a^{j\ell} b^j c^m$, which is generated by the grammar:

$G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A\}$, and P is:

$$\begin{aligned} S &\rightarrow aSc \mid A \\ A &\rightarrow aAb \mid \lambda \end{aligned}$$

Problem 264 Let:

$$L = \{a^k b^n c^j d^\ell \mid k \leq \ell, n \leq j, k = 2m, j, k, \ell, m, n \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, B, D, K\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aaSdd \mid BD \\ D &\rightarrow dD \mid \lambda \\ B &\rightarrow bBc \mid K \\ K &\rightarrow cK \mid \lambda \end{aligned}$$

Problem 265 (a) Let:

$$L = \{a^{m+1}a^i b^j c^k d^m \mid j = 2(i+k), i, j, k, m \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c, d\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow dSd \mid dAB \\ A &\rightarrow aAbb \mid \lambda \\ B &\rightarrow bbBc \mid \lambda \end{aligned}$$

(b) What is the cardinality of the class of languages that are not context-free? Answer by giving the exact number (if this set is finite) or by specifying if it is countable or uncountable.

Answer: The class of languages that are not context-free is infinite and uncountable. This is because the set of all languages is uncountable, and there are only countably many context-free grammars (and languages), since every such grammar is a finite string, and the set of all finite strings is countable.

Problem 266 (a) Construct a context-free grammar G over alphabet $\{a, b, c, d\}$ that generates the language

$$L(G) = \{a^n b^{3m} c^m d^{2n} \mid n \geq 0 \wedge m > 0\}$$

where m, n are integers.

Answer: Let $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$ is the set of terminals; $V = \{S, A\}$ is the set of variables; S is the start symbol. The set of productions P comprises:

$$\begin{aligned} S &\rightarrow aSdd \mid A \\ A &\rightarrow bbbAc \mid bbac \end{aligned}$$

(b) Is the grammar G that you constructed in your answer to part (a) regular? Justify briefly your answer.

Answer: No. The right-hand side of all productions in a regular grammar is of the form λ , or a , or aA , where $a \in \Sigma$, $A \in V$. None of the productions in part (a) are of this form.

Problem 267 Write a complete formal definition of a context-free grammar $G = (V, \Sigma, P, S)$ over alphabet $\{a, b, c\}$ such that G generates the language of all strings that do not contain cba as a substring. If such G does not exist, explain why.

Problem 268 Write a complete formal definition of a context-free grammar $G = (V, \Sigma, P, S)$ over alphabet $\{a, b, c\}$ such that G generates the language of all strings that contain exactly 2 occurrences of b . If such G does not exist, explain why.

Problem 269 Let L be a set of strings over alphabet $\{0, 1\}$ defined as follows:

$$L = \{w \mid w = 1^m 0^{2m} 1^n, m \geq 1, n \geq 0\}$$

(a) Write 5 distinct strings that belong to L . If such strings do not exist, explain why.

(b) Write 5 distinct strings over alphabet $\{0, 1\}$ that do not belong to L . If such strings do not exist, explain why.

Answer:

$\in L$	$\notin L$
100	λ
1001	1
10011	10
110000	01
110000111	101

(c) Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, explain why.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{0, 1\}$, $V = \{S, A, B\}$, and P comprises:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow 1A00 \mid 100 \\ B &\rightarrow 1B \mid \lambda \end{aligned}$$

Problem 270 Let L be a language over alphabet $\{a, b, c\}$ with the following property:

For every word $w \in L$:

if w contains c then w is a palindrome of even length;

if w does not contain c then the length of w is divisible by 3.

(a) Write 5 distinct strings that belong to L .

(b) Write 5 distinct strings over alphabet $\{a, b, c\}$ that do not belong to L .

Answer:

$\in L$	$\notin L$
acca	c
caccac	cca
abbcabaaabacbab	a
aba	ab
abbaaaabb	abab

(c) Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, explain why.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c\}$, $V = \{S, S_1, S_2, A, B, D\}$, and P comprises:

$$\begin{aligned} S &\rightarrow S_1 \mid S_2 \\ S_1 &\rightarrow aS_1a \mid bS_1b \mid B \\ B &\rightarrow cDc \\ D &\rightarrow aDa \mid bDb \mid cDc \mid \lambda \\ S_2 &\rightarrow S_2S_2 \mid AAA \mid \lambda \\ A &\rightarrow a \mid b \end{aligned}$$

Problem 271 Let L be the set of strings over alphabet $\Sigma = \{a, b, c\}$ defined as follows:

String w belongs to the set L if w is an odd-length palindrome or an even-length non-palindrome.

(a) Write a complete formal definition of a context-free grammar G that generates L . If such a grammar does not exist, explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, D, E, Z, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow D \mid E \\ D &\rightarrow aDa \mid bDb \mid cDc \mid a \mid b \mid c \\ E &\rightarrow aEa \mid bEb \mid cEc \mid A \\ A &\rightarrow aBb \mid aBc \mid bBa \mid bBc \mid cBa \mid cBb \\ B &\rightarrow ZZB \mid \lambda \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

(b) Let \mathcal{C} be a class of languages over alphabet $\{a, b, c\}$, defined as follows:

Language X is a member of \mathcal{C} if and only if X is generated by some context-free grammar.

What is the cardinality of the class \mathcal{C} ? (If possible, state the exact number. Otherwise, specify if it is countable or not.) Explain your answer.

Answer:

$$|\mathcal{C}| = \aleph_0$$

Class \mathcal{C} is infinite and countable. There are infinitely many context-free grammars, yet every such grammar is a finite string, and the set of all finite strings over any alphabet is countable.

(c) Let \mathcal{N} be a class of languages over alphabet $\{a, b, c\}$, defined as follows:

Language X is a member of \mathcal{N} if and only if X is not generated by any context-free grammar.

What is the cardinality of the class \mathcal{N} ? (If possible, state the exact number. Otherwise, specify if it is countable or not.) Explain your answer.

Answer:

$$|\mathcal{N}| > \aleph_0$$

Class \mathcal{N} is infinite and uncountable. The class of all languages over alphabet $\{a, b, c\}$ is uncountable, while only countably many of them are generated by some context-free grammar.

(d) Does there exist a (finite) description in plain English words of every language that belongs to the class \mathcal{N} defined in the previous part? Explain your answer.

Answer: No—every such description is a finite string over the English alphabet, and there are only countably many such strings. In contrast, as explained in the answer to the previous part, there are uncountably many languages in the class \mathcal{N} . Hence, there are not enough finite descriptions to provide for all languages in \mathcal{N} .

Problem 272 Write a complete formal definition of a context-free grammar G that generates language L , defined as follows:

$$L = \{a^m b^n ac \mid m \geq 0, n > m\}$$

If such a grammar does not exist, explain why.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow Aac \\ A &\rightarrow aAb \mid B \\ B &\rightarrow bB \mid b \end{aligned}$$

Problem 273 Write a complete formal definition of a context-free grammar that generates the language:

$$\{a^m b^i a^n \mid i = m + n\}$$

where $i, m, n \geq 0$. If such grammar does not exist, prove it.

Answer: Observe that:

$$a^m b^{m+n} a^n = a^m b^m b^n a^n$$

Hence, this language is generated by the context-free grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$ is the set of terminals; $V = \{S, L, R\}$ is the set of variables; S is the start symbol, and the production set P is:

$$\begin{aligned} S &\rightarrow LR \\ L &\rightarrow aLb \mid \lambda \\ R &\rightarrow bRa \mid \lambda \end{aligned}$$

Problem 274 Let:

$$L = \{a^i (bb)^{k+1} e^{\ell+2} (ddd)^{m+3} g^{n+4} h^j\}$$

where

$$k = 2s \text{ and } j = 3t \text{ and } m = 2p + 1$$

for some non-negative integers $i, k, \ell, m, n, j, s, t, p$.

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The general template of the words in the language L is:

$$a^i (bb)^{2s+1} e^{\ell+2} (ddd)^{2p+4} g^{n+4} h^{3t}$$

L is generated by the context-free grammar:
 $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, d, e, g, h\}$,
 $V = \{S, A, B, E, D, K, H, J, T\}$, and the production set
 P is:

$$\begin{aligned} S &\rightarrow ABEDKH \\ A &\rightarrow aA \mid \lambda \\ B &\rightarrow JJB \mid J \\ J &\rightarrow bb \\ E &\rightarrow eE \mid ee \\ D &\rightarrow TTD \mid TTTT \\ T &\rightarrow ddd \\ K &\rightarrow gK \mid gggg \\ H &\rightarrow hhH \mid \lambda \end{aligned}$$

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$a^* (bbbb)^* bb e^* ee (dddd)^* dddddd ddd g^* gggg (hhh)^*$$

Problem 275 Let:

$$L = \{a^i(bb)^{k+1}e^{\ell+2}(ddd)^{m+3}g^{n+4}h^j\}$$

where

$$k = 3s \text{ and } j = t \text{ and } m = s + t$$

for some non-negative integers $i, k, \ell, m, n, j, s, t$.

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The general template of the words in the language L is:

$$a^i(bb)^{3s+1}e^{\ell+2}(ddd)^{s+3}(ddd)^t g^{n+4}h^t$$

L is generated by the context-free grammar:
 $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, d, e, g, h\}$,
 $V = \{S, A, B, E, D, H, J, K, L\}$, and the production set
 P is:

$$\begin{aligned} S &\rightarrow ABD \\ A &\rightarrow aA \mid \lambda \\ B &\rightarrow JJJBK \mid JEKKK \\ J &\rightarrow bb \\ K &\rightarrow ddd \\ E &\rightarrow cE \mid ee \\ D &\rightarrow LDh \mid H \\ L &\rightarrow ddd \\ H &\rightarrow gH \mid gggg \end{aligned}$$

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: This language is not regular, and there does not exist a regular expression that represents it. To prove this, we show that the Pumping Lemma does not hold for the language L .

First, note that if β is the number of b 's in some word contained in L and if δ is the number of d 's in the same word, then the general template requires:

$$\delta > \frac{\beta}{3}$$

since:

$$\delta = s + 3 + t \geq s + 3 > s + \frac{1}{3} = \frac{1}{3}(3s + 1) = \frac{\beta}{3}$$

Now, to prove that L is not regular, assume the opposite—that L is regular. Let η be the constant as in the Pumping Lemma for L . Let $s > \eta + 2$; then $(bbb)^{3s+1}ee(ddd)^{s+3}gggg \in L$, as it is obtained from the general template by setting $i = \ell = t = n = 0$.

In any “pumping” decomposition such that:

$$(bbb)^{3s+1}ee(ddd)^{s+3}gggg = uvx$$

we have: $|uv| \leq \eta < s$. Hence, the “pumping” substring v consists entirely of b 's, say $v = b^p$. By the pumping, every word of the form $uv^q x$, $q \geq 0$, belongs to L . However, such a word is of the form:

$$w_1 = (bbb)^{3s+1+p(q-1)}ee(ddd)^{s+3}gggg$$

By selecting, say: $q = 6s + 1$, we obtain a word whose number of b 's is equal to:

$$\beta = 3s + 1 + p(6s + 1 - 1) = 3s + 1 + 6sp > 3s(2p + 1) \geq 9s$$

where the last inequality holds because $p \geq 1$, as the “pumping” substring cannot be empty. However, the number of d 's in this word is still:

$$\delta = s + 3 < s + 2s = 3s = \frac{\beta}{3}$$

(recall that $s > 2$, by the choice of s .) Since this word violates the required relationship between β and δ , we conclude that it does not belong to the language L , whence the contradiction.

Problem 276 Let:

$$L = \{a^i b^{k+1} e^{\ell+2} d^{m+2} g^n h^j\}$$

where

$$k = 2s \text{ and } j = 3t \text{ and } m = s + 1 \text{ and } \ell = t$$

for some non-negative integers $i, k, \ell, m, n, j, s, t$.

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The general template for strings in L is:

$$a^i b^{2s+1} e^{\ell+2} d^{s+3} g^n h^{3t} \mid i, s, t, n \geq 0$$

The required grammar does not exist, since L is not context-free. To prove this, assume the opposite, and

let η be the constant as in the Pumping Lemma for L .
Select a word:

$$w = b^{2s+1}e^{t+2}d^{s+3}h^{3t} \text{ where } s, t > \eta$$

obtained by setting $i = n = 0$ in the general template. In any “pumping” decomposition: $w = uvxyz$, the length of the “pumping window” vxy is not greater than η : $|vxy| \leq \eta < t$ and $|vxy| \leq \eta < s$. The “pumping” window vxy either falls within one of the four substrings containing a single letter: b^{2s+1} , e^{t+2} , d^{s+3} , h^{3t} , or it spans two such adjacent substrings—it is too short to extend through as many as three (or four) of them. If only letter b is pumped, then there will be only $s + 3$ occurrences of letter d and more than $2s + 1$ occurrences of the letter b , thus invalidating the word. Analogously, pumping any of the other three letters will produce a deficit of another letter, and thereby a word that cannot be in the language. In the other case, if two adjacent letters are pumped, then both of their corresponding letters will be in deficit simultaneously, thus confirming the contradiction.

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: By the answer to the part (a), language L is not context-free, and thereby not regular either. Hence, there does not exist a regular expression that defines it.

Problem 277 Let L be the set of all strings over alphabet $\{a, b\}$ with length divisible by 2 or 3. Write a complete formal definition of a context-free grammar G that generates L . If such a grammar does not exist, explain why.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b\}$, $V = \{S, S_2, S_3, A\}$, and P is:

$$\begin{aligned} S &\rightarrow S_2 \mid S_3 \\ S_2 &\rightarrow S_2 S_2 \mid \lambda \mid AA \\ S_3 &\rightarrow S_3 S_3 \mid \lambda \mid AAA \\ A &\rightarrow a \mid b \end{aligned}$$

Problem 278 Write a complete formal definition of a context-free grammar that generates the language consisting of those strings over $\{a, b\}$ that have even length or an even number of b 's. If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b\}$ is the set of terminals;

$V = \{S, L, A, B\}$ is the set of variables;

S is the start symbol,

and the production set P is:

$$\begin{aligned} S &\rightarrow L \mid B \\ L &\rightarrow AAL \mid \lambda \\ A &\rightarrow a \mid b \\ B &\rightarrow aB \mid bO \mid \lambda \\ O &\rightarrow aO \mid bB \end{aligned}$$

Problem 279 (a) Let:

$$\Sigma = \{a, b, c\}$$

and let L_1 be defined as follows:

$$L_1 = \{a^m b^{2n} c^{3p} \mid m \geq 0, n \geq 0, p \geq 0\}$$

Write a regular expression that defines L_1 . If such expression does not exist, explain why.

Answer:

$$a^*(bb)^*(ccc)^*$$

(b) Let:

$$\Sigma = \{a, b, c\}$$

and let L_2 be defined as follows:

$$L_2 = \{a^m b^{2m} c^{3m} \mid m \geq 0\}$$

Write a complete formal definition of a context-free grammar G that generates L_2 . If such grammar does not exist, prove it.

Answer: Such grammar does not exist, because L_2 is not context-free.

Assume the opposite. Let k be the constant as in the Pumping Lemma for L_2 . Let $w = a^m b^{2m} c^{3m}$, such that $m > k$. Let $w = uvxyz$ be a decomposition of w , where vy is the pumping part.

Observe that every substring of w that contains all the three letters (a, b, c) must contain the entire run of b 's of length $2m$, plus some a 's before this run of b 's and some c 's after it. Hence, every substring of w that contains all the three letters has length at least $2m + 2$. By the Lemma, it must be that $|vxy| < k < m < 2m + 2$. Thus, the pumping part vy cannot contain all the three letters, but lacks at least one of them, say letter ξ , for some $\xi \in \{a, b, c\}$.

After the pumping is admitted, any word of the form $uv^i xy^i z$, for $i > 1$, is claimed to be in L_2 . However, since the other two letters are pumped, while ξ is not pumped, such a word has a surplus of the other two letters, relative to ξ , thereby violating the pattern $a^m b^{2m} c^{3m}$ —a contradiction.

Problem 280 (a) Let L_1 be the set of all strings over alphabet $\{a, b\}$ with an even number of a 's. Write 5 distinct strings that belong to L_1 . If such strings do not exist, explain why.

Answer:

$$\lambda, b, aa, aba, aab, baba$$

(b) Write a complete formal definition of a context-free grammar G_1 that generates language L_1 . If such a grammar does not exist, explain why.

Answer: $G = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b\}$, $V = \{S, A\}$, and P is:

$$\begin{aligned} S &\rightarrow bS \mid aA \mid \lambda \\ A &\rightarrow bA \mid aS \end{aligned}$$

(c) Let L_2 be the set of all strings over alphabet $\{a, b, c\}$ that have even length. Write 5 distinct strings that belong to L_2 . If such strings do not exist, explain why.

Answer:

$$\lambda, ab, ac, bc, ba, ca$$

(d) Write a complete formal definition of a context-free grammar G_2 that generates language L_2 . If such a grammar does not exist, explain why.

Answer: $G = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b, c\}$, $V = \{S, A\}$, and P is:

$$\begin{aligned} S &\rightarrow aA \mid bA \mid cA \mid \lambda \\ A &\rightarrow aS \mid bS \mid cS \end{aligned}$$

Problem 281 Let:

$$L = \{a^i b^j c^k a^\ell b^m c^n \mid i = j+k, m = \ell+n, i, j, k, \ell, m, n \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c\}$, $V = \{S, S_1, S_2, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow S_1 S_2 \\ S_1 &\rightarrow aS_1 c \mid A \\ A &\rightarrow aAb \mid \lambda \\ S_2 &\rightarrow BD \\ B &\rightarrow aBb \mid \lambda \\ D &\rightarrow bDc \mid \lambda \end{aligned}$$

Problem 282 Let:

$$L = \{a^i b^j c^k a^\ell \mid i = k, j = \ell, i, j, k, \ell \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: Such a grammar does not exist, since L is not a context-free language. To prove this, assume the opposite, and let k be the constant as in the Pumping Lemma. Select a word $a^i b^j c^i a^j \in L$ such that $i > k$ and $j > k$. In the “pumping” decomposition: $a^i b^j c^i a^j = uvxyz$, the length of the “pumping window” vxy is not greater than k : $|vxy| \leq k < i$ and also $|vxy| \leq k < j$. Consider a non-empty, effectively “pumping” substring of the “pumping window”, which is at least one of v , y . There are two cases: this “pumping” substring either falls within one of the four substrings containing a single letter: a^i , b^j , c^i , a^j , or it spans two such substrings—it is too short to extend through as many as three (or four) of them. In the first case, the “pumping” produces a surplus of occurrences of one letter, over against what should be the matching number of occurrences of another letter. In the second case, the “pumping” produces two kinds of letters that are out of sequence.

Problem 283 (a) Let L_1 be a language over the alphabet $\{a, b, c\}$, defined as follows:

$$L_1 = \{a^{m+2} b^{n+1} c^k b^\ell a^j \mid k = m \wedge j = \ell \wedge j, k, \ell, m, n \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L_1 . If such a grammar does not exist, prove it.

Answer: Observe that the general template for the strings that belong to L_1 is:

$$a^{k+2} b^{n+1} c^k b^j a^j \text{ where } j, k, n \geq 0$$

whence the grammar: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aAc \mid aaD \\ D &\rightarrow bD \mid b \\ B &\rightarrow bBa \mid \lambda \end{aligned}$$

(b) Let L_2 be a language over the alphabet $\{a, b, c\}$, defined as follows:

$$L_2 = \{a^{m+2} b^{n+1} c^k b^\ell a^j \mid k = m \wedge j = n \wedge j, k, \ell, m, n \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L_2 . If such a grammar does not exist, prove it.

Answer: The required grammar does not exists, since L_2 is not contest-free.

Observe that the general template for the strings that belong to L_2 is:

$$a^{k+2} b^{j+1} c^k b^\ell a^j \text{ where } j, k, \ell \geq 0$$

To prove that L_2 is not context-free, assume the opposite, and let η be the constant as in the Pumping Lemma for L_2 . Select a word:

$$w = a^{k+2} b^{j+1} c^k a^j \text{ where } k, j > \eta$$

obtained by setting $\ell = 0$ in the general template. In any “pumping” decomposition: $w = uvxyz$, the length of the “pumping window” vxy is not greater than η : $|vxy| \leq \eta < k$, and $|vxy| \leq \eta < j$. The “pumping” window vxy either falls within one of the four substrings containing a single letter: a^{k+2} , b^{j+1} , c^k , a^j , or it spans two such adjacent substrings—it is too short to extend through as many as three (or four) of them. If the pumping is only within one of the designated substrings, it will make this substring longer than is appropriate with respect to the length of another substring. In the other case, if two adjacent letters are pumped, then there will be two other substrings with fewer letters than required (in addition to possible appearance of letters out of order.)

Problem 284 Let:

$$L = \{(ab)^n c^k (ab)^\ell b^j\}$$

for some $k = 0, j = 2k + 1, n = 2\ell + 1, n, k, \ell, j, p \geq 0\}$

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The template is:

$$L = \{(ab)^{3\ell+1}b\}$$

whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow Ab \\ A &\rightarrow ababAb \mid ab \end{aligned}$$

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$ab(ababab)^*b$$

Problem 285 Let L be a language over the alphabet $\{a, b, c, d\}$, defined as follows:

$$L = \{a^\ell b^m c^j d^q b^p \mid j, \ell, m, q, p \geq 0 \wedge j = \ell \wedge m = p\}$$

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Advice for Answer: The general template is:

$$a^\ell b^m c^\ell d^q b^m$$

and the language is not context-free. Pumping is observed in a word:

$$w = a^\ell b^m c^\ell d^q b^m$$

obtained from the general template by setting $q = 0$, where m and ℓ are selected so as to exceed the pumping constant. Every word in L that does not contain any d 's must honor the template of w , for some m and ℓ ; pumping, however, breaks the template.

Problem 286 Let L be a language over the alphabet $\{a, b, c, d\}$, defined as follows:

$$L = \{a^\ell b^m c^j d^q b^p \mid j, \ell, m, q, p \geq 0 \wedge j = \ell + 1 \wedge q = p + 1\}$$

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The general template is:

$$a^\ell b^m c^{\ell+1} d^{p+1} b^p$$

whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aAc \mid Dc \\ D &\rightarrow bD \mid \lambda \\ B &\rightarrow dBb \mid d \end{aligned}$$

Problem 287 Let L be a language over the alphabet $\{a, b, c, d\}$, defined as follows:

$$L = \{a^\ell b^m c^j d^q b^p \mid j, \ell, m, q, p \geq 0 \wedge \ell = j + 1 \wedge m = q + 1\}$$

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Advice for Answer: The general template is:

$$a^{j+1} b^{q+1} c^j d^q b^p$$

and the language is not context-free. Pumping is observed in a word:

$$w = a^{j+1} b^{q+1} c^j d^q$$

obtained from the general template by setting $p = 0$, where j and q are selected so as to exceed the pumping constant. Every word in L where all b 's occur contiguously must honor the template of w , for some j and q ; pumping, however, breaks the template.

Problem 288 Let L be a language over the alphabet $\{a, b, c, d\}$, defined as follows:

$$L = \{a^\ell b^m c^j d^q b^p \mid j, \ell, m, q, p \geq 0 \wedge \ell = q + 1 \wedge m = j + 1\}$$

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The general template is:

$$a^{q+1} b^{j+1} c^j d^q b^p$$

whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, E\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aAd \mid aE \\ E &\rightarrow bEc \mid b \\ B &\rightarrow bB \mid \lambda \end{aligned}$$

Problem 289 Let L be the set of all strings over the alphabet $\{a, b\}$, except for strings abb and bba . In short:

$$L = \{w \in \{a, b\}^* \mid w \neq abb \wedge w \neq bba\}$$

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: Observe that all strings of length no less than 4 are good, and so are all strings of length less than or equal to 2. Among the eight strings whose length is equal to 3, six are good (and two must be excluded.) Hence, the required grammar is:

$G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, Z, T, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ZZZZT \mid ZZ \mid Z \mid \lambda \mid A \\ Z &\rightarrow a \mid b \\ T &\rightarrow aT \mid bT \mid \lambda \\ A &\rightarrow aaa \mid aab \mid aba \mid baa \mid bab \mid bbb \end{aligned}$$

Problem 290 (a) Let L_1 be the set of all strings over alphabet $\{a, b, d\}$ that do not contain the substring dba .

Write a complete formal definition of a context-free grammar G_1 that generates language L_1 . If such a grammar does not exist, explain why.

Answer: $G_1 = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, d\}$, $V = \{S, A, B\}$,

and the production set P is:

$$\begin{aligned} S &\rightarrow aS \mid bS \mid dA \mid \lambda \\ A &\rightarrow aS \mid bB \mid dA \mid \lambda \\ B &\rightarrow bS \mid dA \mid \lambda \end{aligned}$$

(b) Let L_2 be the set of all strings over alphabet $\{a, b\}$ that have even length or contain an even number of a 's.

Write a complete formal definition of a context-free grammar G_2 that generates language L_2 . If such a grammar does not exist, explain why.

Answer: $G_2 = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b\}$, $V = \{S, S_1, S_2, A, Z\}$,

and the production set P is:

$$\begin{aligned} S &\rightarrow S_1 \mid S_2 \\ S_1 &\rightarrow S_1S_1 \mid \lambda \mid ZZ \\ Z &\rightarrow a \mid b \\ S_2 &\rightarrow aA \mid bS_2 \mid \lambda \\ A &\rightarrow bA \mid aS_2 \end{aligned}$$

Problem 291 Write a complete formal definition of a context-free grammar that generates the set of all strings over alphabet $\{a, b\}$ whose number of b 's is equal to 2 or 3. If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, A\}$, and P is:

$$\begin{aligned} S &\rightarrow AbAbA \mid AbAbAbA \\ A &\rightarrow aA \mid \lambda \end{aligned}$$

Problem 292 Let L be the set of strings over alphabet $\{a, b, c\}$ in which at least one of the following strings occurs at least once as a substring:

$abbc, cbca, acba$

Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c\}$, $V = \{S, A\}$,

and the production set P is:

$$\begin{aligned} S &\rightarrow AabbcA \mid AcbcaA \mid AacbaA \\ A &\rightarrow AA \mid \lambda \mid a \mid b \mid c \end{aligned}$$

Problem 293 (a) Let L be the set of strings over alphabet $\{a, b\}$ in which all a 's come before all b 's, and the numbers of a 's and b 's have equal parities (i.e., both are either odd or even.)

Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, A, B, D, E\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow D \mid E \\ D &\rightarrow aAbB \\ E &\rightarrow AB \\ A &\rightarrow AA \mid \lambda \mid aa \\ B &\rightarrow BB \mid \lambda \mid bb \end{aligned}$$

(b) Does there exist an algorithm that solves the following problem:

INPUT: An arbitrary context-free grammar G and a string x .

QUESTION: Is x derivable in G ?

Explain your answer briefly.

Answer: Yes—there exists an algorithm to convert the grammar G into its equivalent (non-deterministic) push-down automaton M . To decide if an arbitrary string x is derivable in G , simulate M on input x , and answer as M answers.

Problem 294 Let L be the set of strings over the alphabet $\{a, b\}$ in which all a 's come before all b 's, and the numbers of a 's and b 's have opposite parities (i.e., when one is odd, the other is even.)

Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aAB \mid AbB \\ A &\rightarrow aaA \mid \lambda \\ B &\rightarrow bbB \mid \lambda \end{aligned}$$

Problem 295 Let L be the set of all nonempty strings over alphabet $\{a, b, d\}$ whose first symbol is different from a and the last symbol is different from b .

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(b \cup d)(a \cup b \cup d)^*(a \cup d) \cup d$$

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, d\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ADB \mid d \\ A &\rightarrow b \mid d \\ B &\rightarrow a \mid d \\ D &\rightarrow DD \mid \lambda \mid a \mid b \mid d \end{aligned}$$

Problem 296 (a) Let L be the set of all strings over alphabet $\{a, b\}$ that have the same symbol in the first and last positions.

Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b\}$, $V = \{S, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aDa \mid bDb \mid a \mid b \mid \lambda \\ D &\rightarrow DD \mid \lambda \mid a \mid b \end{aligned}$$

(b) Let L_1 be the set of all strings of odd length over alphabet $\{a, b\}$ that have the same symbol in the first, last, and middle positions.

Write a complete formal definition of a context-free grammar that generates L_1 . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b\}$, $V = \{S, A, B, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aAa \mid bBb \mid a \mid b \\ A &\rightarrow ZAZ \mid a \\ B &\rightarrow ZBZ \mid b \\ Z &\rightarrow a \mid b \end{aligned}$$

(c) Let L_2 be the set of all strings of odd length over alphabet $\{a, b\}$ that have the same symbol in the first and middle positions.

Write a complete formal definition of a context-free grammar that generates L_2 . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b\}$, $V = \{S, A, B, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aAa \mid aAb \mid bBb \mid bBa \mid a \mid b \\ A &\rightarrow ZAZ \mid a \\ B &\rightarrow ZBZ \mid b \\ Z &\rightarrow a \mid b \end{aligned}$$

Problem 297 Let L be a language over alphabet $\{a, b\}$ with the following property:

For every word $w \in L$:

if $|w|$ is odd, then the middle symbol of w is a ;
if $|w|$ is even, then w ends with bb .

Write a complete formal definition of a context-free grammar G that generates L . If such grammar G does not exist, explain why.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b\}$ is the set of terminals;

$V = \{S, E, O\}$ is the set of variables;

S is the start symbol;

and the set of productions P comprises:

$$\begin{aligned} S &\rightarrow O \mid E \\ O &\rightarrow aOa \mid aOb \mid bOa \mid bOb \mid a \\ E &\rightarrow aaE \mid abE \mid baE \mid bbE \mid bb \end{aligned}$$

Problem 298 Consider all those strings over the alphabet a, b, c whose length is odd and greater than 2. Let L be the set of all such strings whose middle symbol is equal to the last symbol and to the next-to-last symbol.

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, d\}$, $V = \{S, A, B, D, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ZZ Aaa \mid ZZ Bbb \mid ZZ Dcc \mid Zaa \mid Zbb \mid Zcc \\ Z &\rightarrow a \mid b \mid c \\ A &\rightarrow ZAZ \mid a \\ B &\rightarrow ZBZ \mid b \\ D &\rightarrow ZDZ \mid c \end{aligned}$$

(b) Write a complete formal definition of a regular context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The language L does not have a regular context-free grammar, since it is not regular. To prove that L is not regular, assume the opposite: that L is regular.

Let k be the constant as in the Pumping Lemma for L . Consider a word $w \in L$, such that $w = a^{m+2} b a^m bb$, where $m > k$. Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < k < m$. Hence, $v = a^j$ for some $j > 0$. By the pumping, for every $i \geq 0$, we have: $uv^i x \in L$. However:

$$uv^i x = uvv^{i-1} x = uva^{(i-1)j} x = a^{m+2+(i-1)j} b a^m bb$$

Since:

$$\beta = (i-1)j > 0 \text{ whenever } i \geq 2$$

we conclude that the word is in the form:

$$a^{m+2} a^\beta b a^m bb \text{ for some } \beta > 0$$

Observe that the $(m+3)$ rd symbol from the left is a , while the $(m+3)$ rd symbol from the right is b . Hence, this b cannot be the middle symbol. However, the middle symbol has to be b , since the word ends with bb . We conclude that this word is not in the language, whence the contradiction.

Problem 299 (a) Write a regular expression that represents the set of string over $\{a, b, c\}$ that have odd length and b as the last symbol. If such a regular expression does not exist, prove it.

Answer:

$$((a \cup b \cup c)(a \cup b \cup c))^* b$$

(b) Write a complete formal definition of a context-free grammar that generates the set of string over $\{a, b, c\}$ that have odd length and b as the middle symbol. If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:
 $\Sigma = \{a, b, c\}$, $V = \{S, Z\}$, and P is:

$$\begin{aligned} S &\rightarrow ZSZ \mid b \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 300 Let L be the set of strings over the alphabet $\{a, b, c\}$ that have odd length and the middle symbol different from a .

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, Z\}$, and P is:

$$\begin{aligned} S &\rightarrow ZSZ \mid b \mid c \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 301 Let L be the set of all strings over the alphabet $\{a, b, c\}$ whose length is odd, and the middle symbol is different from a and equal to either the first symbol or to the last symbol, but not to both.

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$ is the set of terminals; $V = \{S, B, D, Z\}$ is the set of variables; S is the start symbol, and the production set P is:

$$\begin{aligned} S &\rightarrow bBa \mid bBc \mid aBb \mid cBb \\ S &\rightarrow cDa \mid cDb \mid aDc \mid bDc \\ B &\rightarrow ZBZ \mid b \\ D &\rightarrow ZDZ \mid c \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 302 Let:

$$L = \{a^{2i+1}b^jd^{3k+2}a^{\ell+1} \mid i = k, j = \ell \text{ and } i, j, k, \ell \geq 0\}$$

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer: The general template for strings in L is:

$$a^{2k+1}b^jd^{3k+2}a^{j+1} \mid j, k \geq 0$$

This language is not context-free, as is proved in the answer to part (b). Since it is not context-free, it is not regular, and cannot have a regular expression.

(b) Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: The required grammar does not exist, since L is not context-free. To prove this, assume the opposite, and let η be the constant as in the Pumping Lemma for L . Select a word $w = a^{2k+1}b^kd^{3k+2}a^{k+1}$, $k > \eta$. In any “pumping” decomposition: $w = uvxyz$, the length of the “pumping window” vxy is not greater than η : $|vxy| \leq \eta < k$. The “pumping” window vxy either falls

within one of the four substrings containing a single letter: a^{2k+1} , b^k , d^{3k+2} , a^{k+1} , or it spans two such substrings—it is too short to extend through as many as three (or four) of them. In both cases, it omits at least one of the letters, so that pumping results in the deficit of that letter with respect to others.

Problem 303 Let L be a language over alphabet $\{a, b, c, d, e, f\}$, defined as follows:

$$L = \{a^n b^{k+1} c^{m+2} d^{m+1} e^{2k} f^{n+3} \mid k, n, m \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where
 $\Sigma = \{a, b, c, d, e, f\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow aSf \mid Afff \\ A &\rightarrow bAee \mid bB \\ B &\rightarrow cBd \mid ccd \end{aligned}$$

Problem 304 Let α and β be two functions defined as follows:

$$\begin{aligned} \alpha(x, y) &= x + y \\ \beta(x, y) &= x^y \end{aligned}$$

(where all variables assume values from the set of natural numbers.)

Let L be a language over alphabet $\{a, b, c\}$, defined as follows:

$$L = \{a^n b^{\alpha(1,2)} c^{\beta(2,2)} \mid n \geq 0\}$$

Write a regular expression that represents L . If such a regular expression does not exist, prove it.

Answer: Observe that:

$$\begin{aligned} \alpha(1, 2) &= 1 + 2 = 3 \\ \beta(2, 2) &= 2^2 = 4 \end{aligned}$$

Hence, the template for strings of L is:

$$a^n b^3 c^4, \text{ where } n \geq 0$$

whence the required regular expression:

$$a^* bbb \ cccc$$

Problem 305 Let α and β be two functions defined as follows:

$$\begin{aligned} \alpha(x, y) &= x + y \\ \beta(x, y) &= x^y \end{aligned}$$

(where all variables assume values from the set of natural numbers.)

Let L be a language over alphabet $\{a, b, c\}$, defined as follows:

$$L = \{a^n b^{\alpha(s,t)} c^{\beta(v,w)} \mid n \geq 0\}$$

Supply appropriate expressions in the place of the arguments s, t, v, w so that language L is context-free but not regular. If this is impossible, prove it.

Answer: Select the following assignment:

$$\begin{aligned}s &= n \\t &= 0 \\v &= 1 \\w &= 1\end{aligned}$$

This yields:

$$\begin{aligned}\alpha(s, t) &= \alpha(n, 0) = n + 0 = n \\ \beta(v, w) &= \beta(1, 1) = 1^1 = 1\end{aligned}$$

Hence, the template for strings of L is:

$$a^n b^n c, \text{ where } n \geq 0$$

which indeed defines a context-free but not a regular set.

Problem 306 Let α and β be two functions defined as follows:

$$\begin{aligned}\alpha(x, y) &= x + y \\ \beta(x, y) &= x^y\end{aligned}$$

(where all variables assume values from the set of natural numbers.)

Let L be a language over alphabet $\{a, b, c\}$, defined as follows:

$$L = \{a^n b^{\alpha(s, t)} c^{\beta(v, w)} \mid n \geq 0\}$$

Supply appropriate expressions in the place of the arguments s, t, v, w so that language L is not context-free. If this is impossible, prove it.

Answer: Select the following assignment:

$$\begin{aligned}s &= n \\t &= 0 \\v &= n \\w &= 1\end{aligned}$$

This yields:

$$\begin{aligned}\alpha(s, t) &= \alpha(n, 0) = n + 0 = n \\ \beta(v, w) &= \beta(n, 1) = n^1 = n\end{aligned}$$

Hence, the template for strings of L is:

$$a^n b^n c^n, \text{ where } n \geq 0$$

which indeed defines a set that is not context-free.

Problem 307 Let L_1 be the set of all strings over alphabet $\{0, 1\}$ that contain at least two 1's.

Write a complete formal definition of a context-free grammar G_1 that generates language L_1 . If such a grammar does not exist, explain why.

Answer: $G_1 = (V, \Sigma, P, S)$, where $\Sigma = \{0, 1\}$, $V = \{S, Z\}$, and the production set P is:

$$\begin{aligned}S &\rightarrow Z1Z1Z \\ Z &\rightarrow ZZ \mid \lambda \mid 0 \mid 1\end{aligned}$$

(b) Let L_2 be the set of all strings over alphabet $\{0, 1\}$ that contain an even number of 1's.

Write a complete formal definition of a context-free grammar G_2 that generates language L_2 . If such a grammar does not exist, explain why.

Answer: $G_2 = (V, \Sigma, P, S)$, where $\Sigma = \{0, 1\}$, $V = \{S, D\}$, and the production set P is:

$$\begin{aligned}S &\rightarrow 0S \mid 1D \mid \lambda \\ D &\rightarrow 0D \mid 1S\end{aligned}$$

Problem 308 Let L_1 be the set of all strings over the alphabet $\{a, b, c\}$ that contain exactly one b .

Let L_2 be the set of all strings over the alphabet $\{a, b, c\}$ that contain an even number of a 's.

Write a complete formal definition of a context-free grammar that generates the language $L_1 \cup L_2$. If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$ is the set of terminals; $V = \{S, A, B, D, E, F\}$ is the set of variables; S is the start symbol, and the production set P is:

$$\begin{aligned}S &\rightarrow A \mid B \\ A &\rightarrow DbD \\ D &\rightarrow \lambda \mid DD \mid a \mid c \\ B &\rightarrow \lambda \mid BB \mid E \mid F \\ E &\rightarrow bE \mid cE \mid \lambda \\ F &\rightarrow \lambda \mid FF \mid EaEaE\end{aligned}$$

Problem 309 Let L be the set of all odd-length strings over the alphabet $\Sigma = \{a, b, c\}$ whose first, last, and middle symbol are all equal to b .

(a) Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, B, Z\}$, and the production set P is:

$$\begin{aligned}S &\rightarrow bBb \mid b \\ B &\rightarrow ZBZ \mid b \\ Z &\rightarrow a \mid b \mid c\end{aligned}$$

(b) Write a complete formal definition of a context-free grammar that generates the language LL . If such a grammar does not exist, explain why.

Answer: $G = (V, \Sigma, P, T)$, where $\Sigma = \{a, b, c\}$, $V = \{T, S, B, Z\}$, and the production set P is:

$$\begin{aligned}T &\rightarrow SS \\ S &\rightarrow bBb \mid b \\ B &\rightarrow ZBZ \mid b \\ Z &\rightarrow a \mid b \mid c\end{aligned}$$

(c) Write a complete formal definition of a context-free grammar that generates the language L^* . If such a grammar does not exist, explain why.

Answer: $G = (V, \Sigma, P, T)$, where $\Sigma = \{a, b, c\}$, $V = \{T, S, B, Z\}$, and the production set P is:

$$\begin{aligned} T &\rightarrow \lambda \mid TT \mid S \\ S &\rightarrow bBb \mid b \\ B &\rightarrow ZBZ \mid b \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

(d) Write a complete formal definition of a context-free grammar that generates the language $L \cup LL$. If such a grammar does not exist, explain why.

Answer: $G = (V, \Sigma, P, T)$, where $\Sigma = \{a, b, c\}$, $V = \{T, S, B, Z\}$, and the production set P is:

$$\begin{aligned} T &\rightarrow S \mid SS \\ S &\rightarrow bBb \mid b \\ B &\rightarrow ZBZ \mid b \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 310 Let L be the set of all even-length strings over the alphabet $\Sigma = \{a, b, c\}$ whose two middle symbols are different from each other.

(a) Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, Z\}$, and P is:

$$\begin{aligned} S &\rightarrow ZSZ \mid ab \mid ac \mid ba \mid bc \mid ca \mid cb \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

(b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: This automaton does not exist, since L is not regular. To prove this, assume the opposite—that L is regular. Let k be the constant as in the Pumping Lemma for L . Consider a word $w = a^m b c a^m$, where $m > k$. Indeed, $w \in L$, since its length is equal to $2(m + 1)$ (even) and the middle two-symbol substring bc consists of two different symbols. We show that the pumping will push the characteristic substring bc into the right-hand side of the word, leaving an illegal substring aa in the middle.

Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < k < m$. Hence, the pumping part v is entirely within the a -segment, meaning that $v = a^j$ for some j such that $0 < j < k$. After pumping four times, we obtain a word: $w_1 = a^{m+4j} b c a^m$, whose length is equal to $m + 4j + 2 + m = 2(m + 2j) + 2$. Hence, there are exactly $m + 2j$ symbols at each side of the two-symbol substring in the middle. However, the leftmost $m + 2j$ a 's are followed by another a -segment of length equal to $2j$, which is no less than 2, since $j > 0$. This means that the two-symbol substring in the middle is covered by an a -segment, which means that it no longer consists of two different symbols. Hence, $w_1 \notin L$, whence the contradiction.

Problem 311 Let L be the set of exactly those strings over the alphabet $\{a, b, c, g\}$ that satisfy all of the following properties:

1. the first symbol is different from the last symbol;
2. the length of the string is odd;
3. if the first symbol was equal to the last symbol, the string would be a palindrome.

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, g\}$, $V = \{S, T, A, B, D, H\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aTA \mid bTB \mid cTD \mid gTH \\ A &\rightarrow b \mid c \mid g \\ B &\rightarrow a \mid c \mid g \\ D &\rightarrow a \mid b \mid g \\ H &\rightarrow a \mid b \mid c \\ T &\rightarrow aTa \mid bTb \mid cTc \mid gTg \mid a \mid b \mid c \mid g \end{aligned}$$

Problem 312 Let L be the set of exactly those strings over the alphabet $\{a, b, c, d\}$ which satisfy all of the following properties.

1. the string is a concatenation of four non-empty palindromes;
2. three of the four palindromes have an odd length;
3. one of the four palindromes has an even length;
4. the four palindromes may appear in any order;
5. the middle symbol of each of the three odd-length palindromes is different from b ;
6. the middle two symbols of the even-length palindrome are different from c .

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b, c, d\}$, $V = \{S, D, E\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow EDDD \mid DEDD \mid DDED \mid DDDE \\ D &\rightarrow aDa \mid bDb \mid cDc \mid dDd \mid a \mid c \mid d \\ E &\rightarrow aEa \mid bEb \mid cEc \mid dEd \mid aa \mid bb \mid dd \end{aligned}$$

Problem 313 Let L be the set of exactly those strings over the alphabet $\{a, b, c, d\}$ which satisfy all of the following properties.

1. the string is a concatenation of four non-empty palindromes;
2. three of the four palindromes have an even length;

3. one of the four palindromes has an odd length;
4. the four palindromes may appear in any order;
5. the middle symbol of the odd-length palindrome is different from c ;
6. the middle two symbols of each of the three even-length palindromes are different from b .

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b, c, d\}$, $V = \{S, D, E\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow DEEE \mid EDEE \mid EEDE \mid EEEE \\ D &\rightarrow aDa \mid bDb \mid cDc \mid dDd \mid a \mid b \mid d \\ E &\rightarrow aEa \mid bEb \mid cEc \mid dEd \mid aa \mid cc \mid dd \end{aligned}$$

Problem 314 Let L be the set of exactly those strings over the alphabet $\{a, b, c\}$ that are concatenations of an odd number of non-empty palindromes of even length.

In more words, L consists of exactly those strings, say s , over the alphabet $\{a, b, c\}$ that satisfy all of the following properties:

1. s is a concatenation: $s = w_1w_2\dots w_{2n+1}$ of an odd number of component strings;
2. each component string w_i is a palindrome whose length is positive and even.

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b, c\}$, $V = \{S, E\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow EES \mid E \\ E &\rightarrow aEa \mid bEb \mid cEc \mid aa \mid bb \mid cc \end{aligned}$$

Problem 315 Let L be the set of exactly those strings w over the alphabet $\{a, b, c, g\}$ that satisfy all of the following properties:

1. w is a concatenation: $w = w_1w_2w_3w_4$ of four component substrings;
2. substring w_1 is a palindrome whose length is positive and even;
3. the length of substring w_2 is odd and its middle symbol is b ;
4. the number of c 's in the substring w_3 is not greater than 3;
5. the length of substring w_4 is odd and its first symbol is equal to its middle symbol;

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, g\}$, $V = \{S, W_1, W_2, W_3, W_4, Z, E, K, A, B, D, H\}$, and P is:

$$S \rightarrow W_1W_2W_3W_4$$

$$W_1 \rightarrow aW_1a \mid bW_1b \mid cW_1c \mid gW_1g \mid aa \mid bb \mid cc \mid gg$$

$$W_2 \rightarrow ZW_2Z \mid b$$

$$Z \rightarrow a \mid b \mid c \mid g$$

$$W_3 \rightarrow EKEKEKE$$

$$E \rightarrow \lambda \mid EE \mid a \mid b \mid g$$

$$K \rightarrow c \mid \lambda$$

$$W_4 \rightarrow aAZ \mid bBZ \mid cDZ \mid gHZ \mid a \mid b \mid c \mid g$$

$$A \rightarrow ZAZ \mid a$$

$$B \rightarrow ZBZ \mid b$$

$$D \rightarrow ZDZ \mid c$$

$$H \rightarrow ZHZ \mid g$$

Problem 316 Let L be the set of those strings over the alphabet $\{a, b, c\}$ which satisfy all of the following properties.

1. the length of the string is odd;
2. the middle symbol is c ;
3. the number of a 's is positive and even.

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b, c, d\}$, $V = \{S, D, E, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ZSZ \mid aEa \mid aDZ \mid ZDa \\ D &\rightarrow ZDZ \mid aDa \mid aEZ \mid ZEa \\ E &\rightarrow ZEZ \mid aEa \mid aDZ \mid ZDa \mid c \\ Z &\rightarrow b \mid c \end{aligned}$$

Problem 317 Let L be the set of exactly those strings over the alphabet $\{a, b, c\}$ which satisfy all of the following properties.

1. the three letters occur in the string either in the alphabetic order or in the reversed alphabetic order;
2. each of the three letters occurs at least once;
3. if the letters occur in the alphabetic order, then there are more b 's than c 's;
4. if the letters occur in the reversed alphabetic order, then there are more a 's than c 's;

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b, c\}$, $V = \{S, L, R, A, B, D, E\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow L \mid R \\ L &\rightarrow ABD \\ A &\rightarrow aA \mid a \\ B &\rightarrow bB \mid b \\ D &\rightarrow bDc \mid bc \\ R &\rightarrow EA \\ E &\rightarrow cEa \mid cBa \end{aligned}$$

Problem 318 Let L be the set of exactly those strings over the alphabet $\{a, b, c, d\}$ that satisfy all of the following properties.

1. the length of the string is equal to $3n + 1$, for some natural number $n \geq 0$;
2. all of the first (leftmost) n symbols are b 's;
3. all of the last (rightmost) $2n$ symbols are elements of $\{a, d\}$;
4. the $(n + 1)$ st symbol (from the left) is c .

(a) List four distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: The general template for strings of this language is:

$$b^n c (a|d)^{2n}$$

(b) Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and prove your answer.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow bSAA \mid c \\ A &\rightarrow a \mid d \end{aligned}$$

(c) Write a complete formal definition of a *regular* context-free grammar that generates the language L . If such a grammar does not exist, state it and prove your answer.

Answer: This grammar does not exist. If L had a regular grammar, then L would be a regular language. However, L is not regular. To prove this, we show that Pumping Lemma does not hold for L .

Observe that all words of L satisfy the following characteristic property: number of b 's is equal to one half of the total number of a 's and d 's.

Assume the opposite, that L is regular. Let k be the constant as in the Pumping Lemma for L . Let $m > k$; then the word $w = b^m c a^{2m}$ belongs to L .

In any “pumping” decomposition such that

$b^m c a^{2m} = uvx$, we have: $|uv| \leq k < m$. Hence, the “pumping” substring v consists entirely of b 's, say $v = b^\ell$. Recall that $\ell > 0$, since the “pumping” substring cannot be empty. Pump up once, obtaining the word:

$$w_1 = b^{m+\ell} c a^{2m}$$

Since $2m \neq 2(m + \ell)$, word w_1 violates the stated characteristic property and thus $w_1 \notin L$, in violation of the Pumping Lemma.

Problem 319 Let L be the set of exactly those strings over the alphabet $\{a, b, c, d\}$ that satisfy all of the following properties.

1. the length of the string is equal to $3n + 1$, for some natural number $n \geq 0$;
2. all of the first (leftmost) n symbols are b 's;
3. all of the next n symbols (at positions $(n + 1)$ through $2n$ from the left) are c 's;
4. all of the last (rightmost) n symbols are d 's;
5. the symbol at position $2n + 1$ (from the left) is a .

(a) List four distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: The general template for strings of this language is:

$$b^n c^n a d^n$$

(b) Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and prove your answer.

Answer: Such a grammar does not exist, since L is not context-free. To prove this, we show that Pumping Lemma does not hold for L .

Observe that every word of L satisfies the following characteristic property: the number of b 's is equal to the number of c 's and equal to the number of d 's, and there is exactly one a .

Assume the opposite, that L is context-free. Let k be the constant as in the Pumping Lemma for L . Select $n > k$ and consider a word $w = b^n c^n a d^n \in L$, which must pump. If the pumping window is positioned so that a 's are pumped, the new word will have more than one a , violating the characteristic property. Otherwise, in any pumping decomposition, the pumping window is entirely within one of the following three segments: b^n, c^n, d^n or it spans two adjacent segments, since the pumping window is too short to extend into more than two of them, because its length is less than k , and thereby less than n . Thus, if we pump up once, there will be at least one of the segments b^n, c^n, d^n where pumping occurs and at least one where it does not occur. The number of occurrences of at least one of the letters b, c, d will have

increased, while the number of occurrences of at least one other of these letters will not have changed. The new word will violate the characteristic property and thus will not belong to L . Since L violates the Pumping Lemma, L is not context free.

(c) Draw a state-transition graph of a finite automaton M that accepts the language L . If such an automaton does not exist, state it and prove your answer.

Answer: This automaton does not exist, since L is not regular. If L was regular, then L would be content free, because every regular language is content free. However, we prove in part (b) that L is not context free, hence L cannot be regular.

Problem 320 Let L be the set of exactly those strings over the alphabet $\{a, b, c, d\}$ that satisfy all of the following properties.

1. the length of the string is equal to $3n + 1$, for some natural number $n \geq 0$;
2. all of the first (leftmost) $3n$ symbols are equal to each other;
3. none of the first (leftmost) $3n$ symbols is d ;
4. the last symbol (at position $3n + 1$ from the left) is d .

(a) List four distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: See the answer to part (c).

(b) Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and prove your answer.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, K\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow Ad \mid Bd \mid Kd \\ A &\rightarrow \lambda \mid AA \mid aaa \\ B &\rightarrow \lambda \mid BB \mid bbb \\ K &\rightarrow \lambda \mid KK \mid ccc \end{aligned}$$

(c) Write a regular expression that defines L . If such a regular expression does not exist, state it and prove your answer.

Answer:

$$((aaa)^* \cup (bbb)^* \cup (ccc)^*) \ d$$

Problem 321 (a) Let L_1 be the set of all strings over the alphabet $\{a, b, c\}$ that have an odd length and b as the middle symbol.

(In other words, the length of the string is equal to $2n + 1$, for some $n \geq 0$, and there are exactly n symbols to the left of b .)

Write a complete formal definition of a context-free grammar that generates L_1 . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ZSZ \mid b \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

(b) Let L_2 be the set of all strings over the alphabet $\{a, b, c\}$ whose length is of the form $3n + 1$ for $n \geq 0$, and the $(n + 1)$ st symbol from the left is b . (In other words, the length of the string is equal to $3n + 1$, for some $n \geq 0$, and there are exactly n symbols to the left of b .)

Write a complete formal definition of a context-free grammar that generates L_2 . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ZSZZ \mid b \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 322 Let L be the set of all strings over the alphabet $\{a, b, c\}$ which satisfy all of the following properties.

1. The length of the string is odd.
2. If the string contains any c 's, then the middle symbol is c .
3. If the string does not contain any c 's, then the length of the string is greater than or equal to 3, and the three middle symbols comprise a substring where letters appear in alphabetic order.

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, Z, X\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow A \mid B \\ A &\rightarrow ZAZ \mid c \\ Z &\rightarrow a \mid b \mid c \\ B &\rightarrow XBX \mid aaa \mid aab \mid abb \mid bbb \\ X &\rightarrow a \mid b \end{aligned}$$

Problem 323 Let L be the set of exactly those strings over the alphabet $\{a, b, c\}$ that are concatenations of an even number of palindromes of odd length.

In more words, L consists of exactly those strings, say s , over the alphabet $\{a, b, c\}$ that satisfy all of the following properties:

1. s is a concatenation: $s = w_1w_2 \dots w_{2n}$ of an even number of component substrings;
2. each component substring w_i is a palindrome whose length is odd.

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, W\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow WWS \mid \lambda \\ W &\rightarrow aWa \mid bWb \mid cWc \mid a \mid b \mid c \end{aligned}$$

Problem 324 Let L be the language over the alphabet $\Sigma = \{a, b, c, g\}$ that contains exactly those strings whose form is:

$$g^m c^{2n} b^{3m} a^{4n}$$

where $m, n \geq 0$ are natural numbers.

If L is context free, then use part (a) of the answer space below to write a complete formal definition of a context free grammar that generates L , and do not write anything in part (b).

If L is not context free, then do not write anything in part (a) of the answer space, but complete the missing parts of the text given in part (b) so as to obtain a proof that L is not context free.

(a) Context free grammar for L :

Answer:

(b) Proof that L is not context free:

Observe that all words of L satisfy the following characteristic property:

In each of the two pairs of non-adjacent segments, the number of symbols is related: number of b 's is equal to 3 times the number of g 's, and number of a 's is equal to 4 times the number of c 's.

Assume the opposite, that L is context free.

Let β be the constant as in the Pumping Lemma for L .

Let $w_0 \in L$ be a string defined as follows:

$$w_0 = g^m c^{2n} b^{3m} a^{4n}$$

where $m, n > \beta$.

w_0 belongs to L because it satisfies the template.

w_0 must pump because

$$\|w_0\| = 4m + 6n > 4\beta + 6\beta > \beta$$

In any “pumping” decomposition of w_0 , the pumping window satisfies the following property:

it is either within a single segment or within at most two adjacent segments

because

the pumping window is shorter than β while every segment is longer than β .

By pumping 1 times, we obtain a string

in which at least one segment is pumped while its related non-adjacent segment is not pumped,

which violates the stated characteristic property because the lengths of these two non-adjacent segments fail the characteristic property, since one of them has been increased by pumping while the other has not, and thus does not belong to L . Since L violates the Pumping Lemma, L is not context free.

Problem 325 Let L be the set of all strings over the alphabet $\{a, b, c\}$ which satisfy all of the following properties.

1. if the string ends with a , then it is an even-length palindrome;
2. if the string ends with b , then its length is odd, and the middle symbol is different from b ;
3. if the string does not end with either a or b , then it contains at least two c 's.

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aAa \mid ZBb \mid DcDc \\ A &\rightarrow aAa \mid bAb \mid cAc \mid \lambda \\ B &\rightarrow ZBZ \mid a \mid c \\ D &\rightarrow \lambda \mid DD \mid Z \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 326 Let L be the set of all non-empty strings over the alphabet $\{a, b, c\}$ which satisfy all of the following properties.

1. the length of the string is equal to $4n + 1$ for some natural number $n \geq 0$;
2. the first (leftmost) n symbols are all equal (to each other);
3. the last (rightmost) $3n$ symbols are all equal (to each other) but different from (any of) the first (leftmost) n symbols;
4. the $(n + 1)$ st symbol (from the left) is different from the n symbols to its left and different from the $3n$ symbols to its right.

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A_1, A_2, B_1, B_2, K_1, K_2\}$, and P is:

$$\begin{aligned} S &\rightarrow A_1 \mid A_2 \mid B_1 \mid B_2 \mid K_1 \mid K_2 \\ A_1 &\rightarrow bA_1ccc \mid a \\ A_2 &\rightarrow cA_2bbb \mid a \\ B_1 &\rightarrow aB_1ccc \mid b \\ B_2 &\rightarrow cB_2aaa \mid b \\ K_1 &\rightarrow aK_1bbb \mid c \\ K_2 &\rightarrow bK_2aaa \mid c \end{aligned}$$

Problem 327 Let L be the set of all strings over the alphabet $\{a, b, c\}$ which satisfy all of the following properties.

1. if the string has an odd length, then it is a palindrome.
2. if the string has an even length, then it is a concatenation of two palindromes.

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, D, E\}$, and P is:

$$\begin{aligned} S &\rightarrow D \mid EE \mid DD \\ D &\rightarrow aDa \mid bDb \mid cDc \mid a \mid b \mid c \\ E &\rightarrow aEa \mid bEb \mid cEc \mid \lambda \end{aligned}$$

Problem 328 Let L be the set of all strings over the alphabet $\{a, b, c\}$ which satisfy all of the following properties.

1. if the string has an even length, then it is a palindrome.
2. if the string has an odd length, then it is a concatenation of two palindromes.

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, L, N, A, B, Z\}$, and P is:

$$\begin{aligned} S &\rightarrow E \mid ED \mid DE \\ D &\rightarrow aDa \mid bDb \mid cDc \mid a \mid b \mid c \\ E &\rightarrow aEa \mid bEb \mid cEc \mid \lambda \end{aligned}$$

Note: Compare with Problem 327.

Problem 329 Let L be the set of all strings over the alphabet $\{a, b, c, d\}$ which satisfy all of the following properties.

1. if the string contains at least one a then its length is odd and its middle symbol is b ;
2. if the string does not contain any a 's, then it is a palindrome which contains exactly one c .

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, B, K, D, Z\}$, and P is:

$$\begin{aligned} S &\rightarrow B \mid K \\ B &\rightarrow ZBZ \mid aDZ \mid ZDa \\ D &\rightarrow ZDZ \mid b \\ Z &\rightarrow a \mid b \mid c \mid d \\ K &\rightarrow bKb \mid dKd \mid c \end{aligned}$$

Problem 330 Let L be the language over the alphabet $\Sigma = \{a, b, c\}$ that contains exactly those strings which satisfy all of the following properties:

1. if the string does not contain any a 's, then the string is a concatenation of exactly four non-empty palindromes, and the length of each of these palindromes is even;
2. if the string contains at least one a , then the length of the string is odd and the middle symbol is different from a .

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, L, N, A, B, Z\}$, and P is:

$$\begin{aligned} S &\rightarrow N \mid A \\ N &\rightarrow LLLL \\ L &\rightarrow bLb \mid cLc \mid bb \mid cc \\ A &\rightarrow ZAZ \mid aBZ \mid ZBa \\ B &\rightarrow ZBZ \mid b \mid c \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 331 Let L be the set of all strings over the alphabet $\{a, b, c\}$ which satisfy all of the following properties.

1. if the string begins with a , then both of the following conditions hold:
 - (a) the string does not end with a ;
 - (b) if the last symbol of the string was altered so that it becomes a , then the resulting string would be a palindrome;
2. if the string begins with b , then both of the following conditions hold:
 - (a) the string has an odd length;
 - (b) the middle symbol is equal to the last symbol;
3. if the string begins with c , then its length is divisible by 3.

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, K, P, E, F, H, Z, K, L\}$, and

the production set P is:

$$\begin{aligned}
 S &\rightarrow A \mid B \mid K \\
 A &\rightarrow aPb \mid aPc \\
 P &\rightarrow aPa \mid bPb \mid cPc \mid Z \mid \lambda \\
 B &\rightarrow bEa \mid bFb \mid bHc \\
 E &\rightarrow ZEZ \mid a \\
 F &\rightarrow ZFZ \mid b \\
 H &\rightarrow ZHZ \mid c \\
 K &\rightarrow cZZL \\
 L &\rightarrow ZZZL \mid \lambda \\
 Z &\rightarrow a \mid b \mid c
 \end{aligned}$$

Problem 332 Let L be the set of all strings over the alphabet $\{a, b\}$ which satisfy all of the following properties.

1. if the string belongs to a^*b^* then the number of b 's is equal to twice the number of a 's;
2. if the string does not belong to a^*b^* then it contains exactly two b 's.

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$,

where $\Sigma = \{a, b\}$, $V = \{S, E, B, A\}$, and P is:

$$\begin{aligned}
 S &\rightarrow E \mid B \\
 E &\rightarrow aEbb \mid \lambda \\
 B &\rightarrow AbaAbA \mid AbAbaA \\
 A &\rightarrow \lambda \mid AA \mid a
 \end{aligned}$$

Problem 333 Let L be the set of all strings over the alphabet $\{a, b\}$ which satisfy all of the following properties.

1. the length of the string is odd;
2. the middle symbol is b ;
3. the string contains exactly three b 's.

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Advice for Answer: There are three cases, according to the location of the remaining two b 's: these two b 's can be to the left of the middle b , or to the right of the middle b , or one to the left and one to the right.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, S_L, S_R, S_{LR}, L, R, M\}$, and the production set P is:

$$\begin{aligned}
 S &\rightarrow S_L \mid S_R \mid S_{LR} \\
 S_L &\rightarrow aS_La \mid bLa \\
 S_R &\rightarrow aS_Ra \mid aRb \\
 S_{LR} &\rightarrow aS_{LR}a \mid aLb \mid bRa \\
 L &\rightarrow aLa \mid bMa \\
 R &\rightarrow aRa \mid aMb \\
 M &\rightarrow aMa \mid b
 \end{aligned}$$

Problem 334 Let L be the set of all strings over the alphabet $\{a, b, c\}$ which satisfy all of the following properties.

1. If the length of the string is divisible by 4, then this length is not greater than 12.
2. If the length of the string gives a remainder 1 after division by 4, then the middle symbol is not an a .
3. If the length of the string gives a remainder 2 after division by 4, then the two middle symbols are different (from each other.)
4. If the length of the string gives a remainder 3 after division by 4, then the three middle symbols are equal (to each other.)

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, S_1, S_2, S_3, S_4, Z, F\}$, and the production set P is:

$$\begin{aligned}
 S &\rightarrow S_1 \mid S_2 \mid S_3 \mid S_4 \\
 S_1 &\rightarrow \lambda \mid F \mid FF \mid FFF \\
 F &\rightarrow ZZZZ \\
 Z &\rightarrow a \mid b \mid c \\
 S_2 &\rightarrow ZZS_2ZZ \mid b \mid c \\
 S_3 &\rightarrow ZZS_3ZZ \mid ab \mid ac \mid ba \mid bc \mid ca \mid cb \\
 S_4 &\rightarrow ZZS_4ZZ \mid aaa \mid bbb \mid ccc
 \end{aligned}$$

Problem 335 Let L be the set of all strings over the alphabet $\{a, b, c\}$ which satisfy all of the following properties.

1. if the length of the string is even, then both of the following conditions hold:
 - (a) the two middle symbols are different from each other;
 - (b) if the two middle symbols were removed, the resulting string would be a palindrome;
2. if the length of the string is odd, then the middle symbol does not occur anywhere else in the string.

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, E, D, A, B, K, X, Y, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow E \mid D \\ E &\rightarrow aEa \mid bEb \mid cEc \mid ab \mid ac \mid ba \mid bc \mid ca \mid cb \\ D &\rightarrow A \mid B \mid K \\ A &\rightarrow XAX \mid a \\ X &\rightarrow b \mid c \\ B &\rightarrow YBY \mid b \\ Y &\rightarrow a \mid c \\ K &\rightarrow ZKZ \mid c \\ Z &\rightarrow a \mid b \end{aligned}$$

Problem 336 Let D_{bc}^3 be the set of all palindromes which consist only of letters $\{b, c\}$ and have an odd length which is greater than or equal to 3.

Let D_{bc}^5 be the set of all palindromes which consist only of letters $\{b, c\}$ and have an odd length which is greater than or equal to 5.

Let L be the set of all strings over the alphabet $\{a, b, c\}$ which satisfy all of the following properties.

1. if the string contains a , then it also contains as a substring an element of D_{bc}^3 ;
2. if the string does not contain a , then it is itself an element of D_{bc}^5 .

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, D_3, D_5, A, E\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow A \mid D_5 \\ D_5 &\rightarrow bD_3b \mid cD_3c \\ D_3 &\rightarrow bD_3b \mid cD_3c \mid bbb \mid bcb \mid cbc \mid ccc \\ A &\rightarrow EaED_3E \mid ED_3EaE \\ E &\rightarrow \lambda \mid EE \mid a \mid b \mid c \end{aligned}$$

Problem 337 Let L be the set of all strings over the alphabet $\{a, b, c\}$ which satisfy all of the following properties.

1. if the symbol a does not occur in the string, then the string is a concatenation of three palindromes;
2. if the symbol a occurs in the string exactly once, then the length of the string is odd and the middle symbol is different from a ;
3. if the symbol a occurs in the string at least twice, then the length of the string is even.

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b, c\}$, $V = \{S, S_0, S_1, S_2, P, A, B, E, D, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow S_0 \mid S_1 \mid S_2 \\ S_0 &\rightarrow PPP \\ P &\rightarrow bPb \mid cPc \mid b \mid c \mid \lambda \\ S_1 &\rightarrow AS_1A \mid aBA \mid ABa \\ B &\rightarrow ABA \mid A \\ A &\rightarrow b \mid c \\ S_2 &\rightarrow EaEaE \mid EaDaD \mid DaEaD \mid DaDaE \\ E &\rightarrow ZZE \mid \lambda \\ D &\rightarrow ZE \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 338 Let L be the language over the alphabet $\Sigma = \{a, b, c, g\}$ that contains exactly those strings which satisfy all of the following properties:

1. the length of the string is equal to $8n + 1$ for some natural number n ;
2. the middle symbol is different from a ;
3. the second (from the left) symbol is g ;
4. the third (from the left) symbol is a ;
5. the next-to-last (i.e., second from the right) symbol is b ;
6. the last (i.e., first from the right) symbol is c .

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, g\}$, $V = \{S, A, Z\}$, and P is:

$$\begin{aligned} S &\rightarrow ZgaZAZZbc \\ A &\rightarrow ZZZZAZZZZ \mid b \mid c \mid g \\ Z &\rightarrow a \mid b \mid c \mid g \end{aligned}$$

Problem 339 Let L be the set of all strings over the alphabet $\{a, b, c\}$ whose length is odd, and the middle symbol is different from the second symbol (counting from the left.)

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D, E, F, H, Z\}$, and the production set

P is:

$$\begin{aligned} S &\rightarrow ZAZ \mid ZBZ \mid ZDZ \\ A &\rightarrow bEZ \mid cEZ \\ B &\rightarrow aFZ \mid cFZ \\ D &\rightarrow aHZ \mid bHZ \\ E &\rightarrow ZEZ \mid a \\ F &\rightarrow ZFZ \mid b \\ H &\rightarrow ZHZ \mid c \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 340 Let L be the set of those strings over the alphabet $\{a, b, c\}$ that satisfy all of the following properties:

1. The length of the string is odd.
2. The first symbol (counting from the left) is equal to the middle symbol.
3. The last symbol (counting from the left) is different from the first symbol.
4. The next-to-last (penultimate) symbol (counting from the left) is different from the last symbol and different from the first symbol.

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aZAbc \mid aZAc \\ S &\rightarrow bZBac \mid bZBca \\ S &\rightarrow cZDab \mid cZDba \\ A &\rightarrow ZAZ \mid a \\ B &\rightarrow ZBZ \mid b \\ D &\rightarrow ZDZ \mid c \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 341 Let L be the set of those strings over the alphabet $\{a, b, c\}$ that satisfy all of the following properties:

1. The length of the string is odd.
2. The last symbol (counting from the left) is equal to the middle symbol.
3. The first symbol (counting from the left) is different from the last symbol.
4. The next-to-last (penultimate) symbol (counting from the left) is different from the last symbol and different from the first symbol.

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow bZAca \mid cZAb \\ S &\rightarrow aZBcb \mid cZBab \\ S &\rightarrow aZDbc \mid bZDac \\ A &\rightarrow ZAZ \mid a \\ B &\rightarrow ZBZ \mid b \\ D &\rightarrow ZDZ \mid c \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 342 Let L be the set of strings over the alphabet $\{a, b, c\}$ with length greater than 1 and the following additional properties:

- If the length of the string is odd, then the middle three letters are equal.
- If the length of the string is even, then the string is a palindrome.

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, D, E, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow D \mid E \\ D &\rightarrow ZDZ \mid aaa \mid bbb \mid ccc \\ E &\rightarrow aEa \mid bEb \mid cEc \mid aa \mid bb \mid cc \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 343 Let L be the set of all strings with odd length (greater than 1) over the alphabet $\{a, b, c\}$ where the first symbol is different from the middle symbol but is equal to the last symbol.

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aAa \mid bBb \mid cDc \\ A &\rightarrow ZAZ \mid b \mid c \\ B &\rightarrow ZBZ \mid a \mid c \\ D &\rightarrow ZDZ \mid a \mid b \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 344 Let L be the set of all strings over the alphabet $\{a, b, c\}$ that satisfy all of the following properties.

1. The string is not empty and has an even length.
2. The middle two symbols are not equal to each other.
3. If the middle two symbols were equal, the string would be a palindrome.

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b, c\}$, $V = \{S, M\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aSa \mid bSb \mid cSc \mid M \\ M &\rightarrow ab \mid ac \mid ba \mid bc \mid ca \mid cb \end{aligned}$$

Problem 345 Let L be the set of all non-empty strings over the alphabet $\{a, b, c\}$ which satisfy all of the following properties.

1. if the string has an even length, then it also has an even number of b 's;
2. if the length of the string is odd, then only the middle symbol is b .

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, E, F, H, K, D, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow E \mid D \\ E &\rightarrow aH \mid cH \mid bK \mid \lambda \\ F &\rightarrow aK \mid cK \mid bH \\ H &\rightarrow aE \mid cE \mid bF \\ K &\rightarrow aF \mid cF \mid bE \\ D &\rightarrow ZDZ \mid b \\ Z &\rightarrow a \mid c \end{aligned}$$

Problem 346 Let L be the set of all strings over the alphabet $\{a, b, c\}$ which satisfy all of the following properties.

1. if the string has an odd length, then it is a palindrome whose middle symbol is different from the first symbol;
2. if the string has an even length, then it is not empty, and its second symbol is equal to the last symbol.

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, L, E, A, B, K, R, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow L \mid E \\ E &\rightarrow aAa \mid bBb \mid cKc \\ A &\rightarrow aAa \mid bAb \mid cAc \mid b \mid c \\ B &\rightarrow aBa \mid bBb \mid cBc \mid a \mid c \\ K &\rightarrow aKa \mid bKb \mid cKc \mid a \mid b \\ L &\rightarrow ZZ \mid ZaRZa \mid ZbRZb \mid ZcRZc \\ R &\rightarrow ZZR \mid \lambda \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 347 Let L be the set of all non-empty strings over the alphabet $\{a, b, c\}$ which satisfy all of the following properties.

1. the length of the string is not divisible by 2;
2. if the length of the string gives a remainder of 1 after division by 4, then there exists a letter b in the string such that there are exactly three times as many letters to the left of this b as there are to its right;
3. if the length of the string gives a remainder of 3 after division by 4, then its middle symbol is not b .

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, N, T, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow N \mid T \\ N &\rightarrow ZZZNZ \mid b \\ T &\rightarrow ZZTZZ \mid ZaZ \mid ZcZ \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 348 Let L be the set of all non-empty strings over the alphabet $\{a, b, c\}$ which satisfy all of the following properties.

1. if the length of the string is odd, then the first, last, and the middle symbols are equal;
2. if the length of the string is even, then it is divisible by four;
3. if the length of the string is odd but greater than four, then the second symbol is not c ;
4. if the length of the string is even and positive, then three of the four middle symbols are c .

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, K, Z, X, E, D, F\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow E \mid D \\ E &\rightarrow \lambda \mid F \\ F &\rightarrow ZZFZZ \mid Xccc \mid cXcc \mid ccXc \mid cccX \\ X &\rightarrow a \mid b \\ Z &\rightarrow a \mid b \mid c \\ D &\rightarrow aXAZa \mid bXBZb \mid cXKZc \mid aaa \mid bbb \mid ccc \mid a \mid b \mid c \\ A &\rightarrow ZAZ \mid a \\ B &\rightarrow ZBZ \mid b \\ K &\rightarrow ZKZ \mid c \end{aligned}$$

Problem 349 Let L be the set of all strings over the alphabet $\{a, b, c\}$ which satisfy all of the following properties.

1. if the string begins with a , then it is an odd-length palindrome;
2. if the string begins with b , then its length is divisible by 4, and the middle two symbols are equal;
3. if the string does not begin with either a or b , then it contains at most four c 's.

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, F, P, Z, Q, R, K, E, X\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow A \mid B \mid F \\ S &\rightarrow aPa \mid a \\ P &\rightarrow aPa \mid bPb \mid cPc \mid Z \\ B &\rightarrow bQZ \\ Q &\rightarrow ZZQZZ \mid aa \mid bb \mid cc \\ F &\rightarrow cR \mid \lambda \\ R &\rightarrow EKEKEKE \\ K &\rightarrow c \mid \lambda \\ E &\rightarrow XE \mid \lambda \\ Z &\rightarrow a \mid b \mid c \\ X &\rightarrow a \mid b \end{aligned}$$

Problem 350 Let L be the set of those strings over the alphabet $\{a, b, c\}$ that satisfy all of the following properties.

1. The string has an odd length.
2. The first symbol is equal to the middle symbol and to the last symbol.

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow Z \mid aAa \mid bBb \mid cDc \\ A &\rightarrow ZAZ \mid a \\ B &\rightarrow ZBZ \mid b \\ D &\rightarrow ZDZ \mid c \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 351 Let L be the set of those strings over the alphabet $\{a, b\}$ that satisfy all of the following properties.

1. The string has an odd length.
2. The second symbol is equal to the last symbol.

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, A, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ZaAa \mid ZbAb \\ A &\rightarrow ZZA \mid \lambda \\ Z &\rightarrow a \mid b \end{aligned}$$

Problem 352 Let L be the set of all strings over the alphabet $\{a, b, c\}$ that satisfy all of the following properties.

1. The length of the string is even, say equal to $2n$, for any natural number n .
2. The first (leftmost) symbol is equal to $(n+1)$ st symbol.

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aA \mid bB \mid cD \\ A &\rightarrow ZAZ \mid a \\ B &\rightarrow ZBZ \mid b \\ D &\rightarrow ZDZ \mid c \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 353 Let L be the set of all strings over the alphabet $\{a, b, c\}$ that satisfy all of the following properties.

1. The length of the string is odd.
2. If the middle symbol is a , then the string does not contain any c 's.
3. If the middle symbol is b , the string is a palindrome.
4. If the middle symbol is c , then the first symbol is a , but the last symbol is different from a .

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D, E, F, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow A \mid B \mid D \\ A &\rightarrow EAE \mid a \\ E &\rightarrow a \mid b \\ B &\rightarrow aBa \mid bBb \mid cBc \mid b \\ D &\rightarrow aFb \mid aFc \\ F &\rightarrow ZFZ \mid c \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 354 Let L be the set of those strings over the alphabet $\{a, b, c\}$ that satisfy all of the following properties.

1. The string has an odd length.
2. The first symbol is different from a .
3. The middle symbol is different from a .

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, X, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow XAZ \mid X \\ A &\rightarrow ZAZ \mid X \\ X &\rightarrow b \mid c \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 355 Let L be the set of those strings over the alphabet $\{a, b, c\}$ that satisfy all of the following properties.

1. The string has an even length greater than 2.
2. The first symbol is a .
3. Neither one of the two middle symbols is a .

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, X, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aAZ \\ A &\rightarrow ZAZ \mid XX \\ X &\rightarrow b \mid c \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 356 Let L be the set of all non-empty strings with even length over the alphabet $\{a, b, c\}$ where the two middle symbols are different (from each other) but at least one of them is equal to the first (leftmost) symbol of the string.

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, K, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aAZ \mid bBZ \mid cKZ \mid A \mid B \mid K \\ A &\rightarrow ZAZ \mid ab \mid ac \mid ba \mid ca \\ B &\rightarrow ZBZ \mid ba \mid bc \mid ab \mid cb \\ K &\rightarrow ZKZ \mid ca \mid cb \mid ac \mid bc \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 357 Let L be the set of all non-empty strings over the alphabet $\{a, b, c\}$ which satisfy all of the following properties.

1. if the string begins with a , then it is a palindrome;

2. if the string begins with b , then its length is odd, its middle letter is b , and its last letter is b ;
3. if the string begins with c , then its length is even.

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, K, P, Z, D, E\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow A \mid B \mid K \mid \lambda \\ A &\rightarrow aPa \mid a \\ P &\rightarrow aPa \mid bPb \mid cPc \mid Z \mid \lambda \\ Z &\rightarrow a \mid b \mid c \\ B &\rightarrow bDb \mid b \\ D &\rightarrow ZDZ \mid b \\ K &\rightarrow cZE \\ E &\rightarrow ZZE \mid \lambda \end{aligned}$$

Problem 358 Let L be the set of all non-empty strings over the alphabet $\{a, b, c\}$ which satisfy all of the following properties.

1. if the length of the string is divisible by 4, then its middle two symbols are equal to the last symbol;
2. if the length of the string is divisible by 2 but not divisible by 4, then the string does not contain any c 's;
3. if the length of the string is odd, then its middle symbol is a , and the first, middle and last symbols are all different from one another.

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, F, T, D, X, Z, A, B, K, T, Y\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow F \mid T \mid D \\ F &\rightarrow ZAa \mid ZBb \mid ZKc \\ A &\rightarrow ZZAZZ \mid aa \\ B &\rightarrow ZBZ \mid bb \\ K &\rightarrow ZZKZZ \mid cc \\ Z &\rightarrow a \mid b \mid c \\ T &\rightarrow YYYYT \mid YY \\ Y &\rightarrow a \mid b \\ D &\rightarrow bXc \mid cXb \\ X &\rightarrow ZXZ \mid a \end{aligned}$$

Problem 359 Let L be the language over the alphabet $\Sigma = \{a, b, c\}$ that contains exactly those strings which satisfy all of the following properties:

1. if the string does not contain any a 's, then the string is a concatenation of at least three palindromes whose length is odd and greater than 1;
2. if the string contains at least one a , then the string is a palindrome whose length is odd and the middle symbol is a .

Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, S_1, S_2, A, B, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow S_1 \mid S_2 \\ S_1 &\rightarrow AS_1 \mid AAA \\ A &\rightarrow bAb \mid cAc \mid B \\ B &\rightarrow bZb \mid cZc \\ Z &\rightarrow b \mid c \\ S_2 &\rightarrow aS_2a \mid bS_2b \mid cS_2c \mid a \end{aligned}$$

Problem 360 Let L_1 be the set of strings over alphabet $\{a, b, c\}$ that have odd length and c as the middle symbol. Let L_2 be the set of strings over alphabet $\{a, b, c\}$ that do not contain b .

(a) Write a complete formal definition of a context-free grammar that generates $L_1^* \cup L_1 L_2$. If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c\}$, $V = \{S, S_1, S_2, A, Z, Y\}$, and P is:

$$\begin{aligned} S &\rightarrow A \mid S_1 S_2 \\ A &\rightarrow AA \mid \lambda \mid S_1 \\ S_1 &\rightarrow ZS_1Z \mid c \\ Z &\rightarrow a \mid b \mid c \\ S_2 &\rightarrow YS_2 \mid \lambda \\ Y &\rightarrow a \mid c \end{aligned}$$

(b) Does there exist an algorithm that solves the following problem:

INPUT: An arbitrary context-free grammar G , which generates language $L(G)$.

OUTPUT: A context-free grammar G_1 that generates $L(G)^*$.

Explain your answer briefly.

Answer: Yes.

Let $G = (V, \Sigma, P, S)$; let $G_1 = (V_1, \Sigma, P_1, S_1)$.

To construct G_1 from G , select S_1 such that:

$$S_1 \notin V; V_1 = V \cup \{S_1\}$$

and form the new rule set P_1 by adding new rules to P as follows:

$$P_1 = P \cup \{S_1 \rightarrow S_1 S_1 \mid \lambda \mid S\}$$

Problem 361 (a) Let:

$$L = \{d^{i+1}a^ib^jc^kd^m \mid j = 2m + 1, i, j, k, m \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: The general template for the strings that belong to L is:

$$d^{i+1}a^ib^{2m+1}c^kd^m \mid i, k, m \geq 0$$

whence the grammar: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow dAa \mid d \\ B &\rightarrow bbBd \mid bD \\ D &\rightarrow cD \mid \lambda \end{aligned}$$

(b) Let L be the language defined in part (a). Is the language

$$L \cap d^*$$

context-free? Explain your answer briefly.

Answer: Yes—the intersection of a context free language with a regular language is always context free.

Problem 362 (a) Let L be the set of strings over alphabet $\{a, b\}$ in which all a 's come before all b 's, and the length of the string is divisible by 3.

Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: Observe that:

$$3 = 1 + 2 = 2 + 1$$

whence the grammar: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \mid AaBbb \mid AaaBb \\ A &\rightarrow aaaA \mid \lambda \\ B &\rightarrow bbbB \mid \lambda \end{aligned}$$

Problem 363 Let L_1 be the set of strings over alphabet $\{a, b, c\}$ that contain at least one b . Let L_2 be the set of strings over alphabet $\{a, b, c\}$ whose length gives remainder 2 when divided by 3.

(a) Write a complete formal definition of a context-free grammar that accepts L_1 . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, N)$, where $\Sigma = \{a, b, c\}$, $V = \{N, B\}$, and P is:

$$\begin{aligned} N &\rightarrow aN \mid cN \mid bB \\ B &\rightarrow aB \mid bB \mid cB \mid \lambda \end{aligned}$$

(b) Write a complete formal definition of a context-free grammar that accepts L_2 . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S_0)$, where $\Sigma = \{a, b, c\}$, $V = \{S_0, S_1, S_2, Z\}$, and P is:

$$\begin{aligned} S_0 &\rightarrow ZS_1 \\ S_1 &\rightarrow ZS_2 \\ S_2 &\rightarrow ZS_0 \mid \lambda \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

(c) Write a complete formal definition of a context-free grammar that accepts $L_1 \cup L_2$. If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, N, B, S_0, S_1, S_2, Z\}$, and P is:

$$\begin{aligned} S &\rightarrow N \mid S_0 \\ N &\rightarrow aN \mid cN \mid bB \\ B &\rightarrow aB \mid bB \mid cB \mid \lambda \\ S_0 &\rightarrow ZS_1 \\ S_1 &\rightarrow ZS_2 \\ S_2 &\rightarrow ZS_0 \mid \lambda \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

(d) Write a complete formal definition of a context-free grammar that accepts $L_1 \cap L_2$. If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, N_0)$, where $\Sigma = \{a, b, c\}$, $V = \{N_0, N_1, N_2, B_0, B_1, B_2, Z\}$, and P is:

$$\begin{aligned} N_0 &\rightarrow aN_1 \mid cN_1 \mid bB_1 \\ N_1 &\rightarrow aN_2 \mid cN_2 \mid bB_2 \\ N_2 &\rightarrow aN_0 \mid cN_0 \mid bB_0 \\ B_0 &\rightarrow ZB_1 \\ B_1 &\rightarrow ZB_2 \\ B_2 &\rightarrow ZB_0 \mid \lambda \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Note: All the four grammars are obtained straightforwardly, by algorithmic conversion of the automata constructed in Problem 579.

Problem 364 Let L be the set of strings over alphabet a, b, c that begin with a and end with c .

(a) Write a complete formal definition of a context-free grammar that generates \overline{L} (the complement of L). If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow bD \mid Db \mid aDa \mid cDa \mid cDc \mid \lambda \\ D &\rightarrow DD \mid \lambda \mid a \mid b \mid c \end{aligned}$$

(b) Does there exist an algorithm that solves the following problem:

INPUT: An arbitrary context-free grammar G .

OUTPUT: A context-free grammar G_1 that generates the complement of the language generated by G .

Explain your answer briefly.

Answer: No—some context-free languages do not have a context-free complement. In fact, it is undecidable for an arbitrary context-free grammar whether the complement of the language derived by it is context-free.

Problem 365 Let:

$$L = \{a^m b^m \mid m \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates the complement of L in $\{a, b\}^*$. If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b\}$, $V = \{S_{ba}, S_<, S_>, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow S_{ba} \mid S_< \mid S_> \\ S_{ba} &\rightarrow DbaD \\ D &\rightarrow aD \mid bD \mid \lambda \\ S_< &\rightarrow aS_< b \mid B \\ B &\rightarrow bB \mid b \\ S_> &\rightarrow aS_> b \mid A \\ A &\rightarrow aA \mid a \end{aligned}$$

Problem 366 Let L be a language over alphabet $\{a, b\}$, defined as follows:

$$L = \{a^{2n} b^n \mid n \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aaSb \mid \lambda$$

(b) Write a complete formal definition of a context-free grammar G_1 that generates \overline{L} (the complement of L). If such grammar does not exist, prove it.

Answer: $G_1 = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b\}$, $V = \{S_{ba}, S_1, S_<, S_>, A, B, D, E\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow S_{ba} \mid S_1 \mid S_< \mid S_> \\ S_{ba} &\rightarrow DbaD \\ D &\rightarrow aD \mid bD \mid \lambda \\ S_1 &\rightarrow aaS_1 \mid aE \\ E &\rightarrow bE \mid \lambda \\ S_< &\rightarrow aaS_< b \mid B \\ B &\rightarrow bB \mid b \\ S_> &\rightarrow aaS_> b \mid A \\ A &\rightarrow aA \mid a \end{aligned}$$

Problem 367 Let L be the set of strings over $\{a, b, c\}$ whose length is equal to 3.

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c\}$, $V = \{S, Z\}$, and P is:

$$\begin{aligned} S &\rightarrow ZZZ \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

(b) Write a complete formal definition of a context-free grammar that generates L^* . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, T)$, where:

$\Sigma = \{a, b, c\}$, $V = \{T, S, Z\}$, and P is:

$$\begin{aligned} T &\rightarrow TT \mid \lambda \mid S \\ S &\rightarrow ZZZ \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 368 Let $\Sigma = \{a, b, c\}$; let L_1 be defined as follows:

$$L_1 = \{w \in \Sigma^* \mid w = a^n b^{k+1} c^{2n}, n \geq 0, k \geq 0\}$$

and let L_2 be defined as follows:

$$L_2 = \{w \in \Sigma^* \mid w = b^k c^n, n \geq 0, k \geq 0\}$$

Let:

$$\begin{aligned} S_1 &= L_1 \cup L_2 \\ S_2 &= L_1 L_2 \end{aligned}$$

(a) Write 5 distinct strings that belong to S_1 . If such strings do not exist, explain why.

Answer:

$$b, c, bbb, aabbcccc, bccc$$

(b) Write a complete formal definition of a context-free grammar G_1 that generates language S_1 . If such a grammar does not exist, explain why.

Answer: $G_1 = (V, \Sigma, P_1, S)$, where

$\Sigma = \{a, b, c\}$, $V = \{S, A_1, A_2, B, D, E\}$, and P_1 is:

$$\begin{aligned} S &\rightarrow A_1 \mid A_2 \\ A_1 &\rightarrow aA_1cc \mid B \\ B &\rightarrow bB \mid b \\ A_2 &\rightarrow DE \\ D &\rightarrow bD \mid \lambda \\ E &\rightarrow cE \mid \lambda \end{aligned}$$

(c) Write 5 distinct strings that belong to S_2 . If such strings do not exist, explain why.

Answer:

$$b, abccbbc, bc, bbb, aabccccbbccc$$

(d) Write a complete formal definition of a context-free grammar G_2 that generates language S_2 . If such a grammar does not exist, explain why.

Answer: $G_2 = (V, \Sigma, P_2, S)$, where

$\Sigma = \{a, b, c\}$, $V = \{S, A_1, A_2, B, D, E\}$, and P_2 is:

$$\begin{aligned} S &\rightarrow A_1 A_2 \\ A_1 &\rightarrow aA_1cc \mid B \\ B &\rightarrow bB \mid b \\ A_2 &\rightarrow DE \\ D &\rightarrow bD \mid \lambda \\ E &\rightarrow cE \mid \lambda \end{aligned}$$

Problem 369 Let L be the set of all strings with length no less than two over the alphabet $\{a, b, c\}$ where the first symbol is equal to the second symbol.

Write a complete formal definition of a context-free grammar that generates \bar{L} (the complement of L). If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$,

$V = \{S, A, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow \lambda \mid Z \mid abA \mid acA \mid baA \mid bcA \mid caA \mid cbA \\ A &\rightarrow ZA \mid \lambda \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 370 Let L be the set of all strings over alphabet $\{a, b, c\}$ that have odd length and do not contain letter c .

(a) Write a complete formal definition of a context-free grammar G that generates \bar{L} (the complement of L). If such grammar does not exist, prove it.

Answer: \bar{L} consists of those strings that have even length or contain letter c , whence the grammar:

$$\begin{aligned} G &= (V, \Sigma, P, S), \text{ where} \\ \Sigma &= \{a, b, c\}, V = \{S, E, D, Z\}, \text{ and the production set} \\ P &\text{ is:} \\ S &\rightarrow E \mid DcD \\ E &\rightarrow EE \mid \lambda \mid ZZ \\ D &\rightarrow DD \mid \lambda \mid Z \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

(b) Let \mathcal{S} be a class of languages over alphabet $\{a, b, c\}$, defined as follows:

Language L is a member of \mathcal{S} if and only if L is generated by some context-free grammar.

What is the cardinality of \mathcal{S} ? Explain your answer briefly.

Answer:

$$|\mathcal{S}| = \aleph_0$$

Class \mathcal{S} is infinite and countable. There are infinitely many context-free grammars, yet every such grammar is a finite string, and the set of all finite strings over any alphabet is countable.

Problem 371 Let L be the set of all strings over alphabet $\{a, b\}$ that have even length and end with letter b .

(a) Write a complete formal definition of a context-free grammar G that generates \bar{L} (the complement of L). If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b\}$, $V = \{S, E, D, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ZE \mid Da \\ Z &\rightarrow a \mid b \\ E &\rightarrow EE \mid \lambda \mid ZZ \\ D &\rightarrow aD \mid bD \mid \lambda \end{aligned}$$

- (b) Let \mathcal{S} be a class of languages over alphabet $\{a, b\}$, defined as follows:

Language L is a member of \mathcal{S} if and only if L is not generated by any context-free grammar.

What is the cardinality of \mathcal{S} ? Explain your answer briefly.

Answer:

$$|\mathcal{S}| > \aleph_0$$

Class \mathcal{S} is infinite and uncountable. The class of all languages over alphabet $\{a, b\}$ is uncountable, while only countably many of them are generated by some context-free grammar.

Problem 372 Let L_1 be the set of all strings over the alphabet $\{a, b, c\}$ that contain at least one b . Let L_2 be the set of all strings over the alphabet $\{a, b, c\}$ that contain at least one c .

Write a complete formal definition of a context-free grammar that generates $L_1 \cap L_2$. If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AbAcA \mid AcAbA \\ A &\rightarrow \lambda \mid AA \mid a \mid b \mid c \end{aligned}$$

Problem 373 Let L_1 be the set of all strings over the alphabet $\{a, b, c\}$ that have odd length.

Let L_2 be the set of all strings over the alphabet $\{a, b, c\}$ that contain at most one c .

Write a complete formal definition of a context-free grammar that generates $L_1 \cap L_2$. If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, D, E, F, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow D \mid E \\ D &\rightarrow ZZD \mid Z \\ Z &\rightarrow a \mid b \\ E &\rightarrow FcF \mid DcD \\ F &\rightarrow ZZF \mid \lambda \end{aligned}$$

Problem 374 Let L_1 be the set of strings over alphabet $\{a, b\}$ that contain at least 3 b 's. Let L_2 be the set of strings over alphabet $\{a, b\}$ that contain bb as a substring.

(a) Write a complete formal definition of a context-free grammar G_1 that generates L_1 . If such grammar does not exist, prove it.

Answer: $G_1 = (V_1, \Sigma, P_1, S_1)$, where $\Sigma = \{a, b\}$, $V_1 = \{S_1, A\}$, and P_1 is:

$$\begin{aligned} S_1 &\rightarrow AbAbAbA \\ A &\rightarrow aA \mid bA \mid \lambda \end{aligned}$$

- (b) Write a complete formal definition of a context-free grammar G_2 that generates L_2 . If such grammar does not exist, prove it.

Answer: $G_2 = (V_2, \Sigma, P_2, S_2)$, where $\Sigma = \{a, b\}$, $V_2 = \{S_2, B\}$, and P_2 is:

$$\begin{aligned} S_2 &\rightarrow BbbB \\ B &\rightarrow aB \mid bB \mid \lambda \end{aligned}$$

- (c) Write a complete formal definition of a context-free grammar G_3 that generates $L_1 \cup L_2$. If such grammar does not exist, prove it.

Answer: $G_3 = (V_3, \Sigma, P_3, S_3)$, where $\Sigma = \{a, b\}$, $V_3 = \{S_3, S_1, A, S_2, B\}$, and P_3 is:

$$\begin{aligned} S_3 &\rightarrow S_1 \mid S_2 \\ S_1 &\rightarrow AbAbAbA \\ A &\rightarrow aA \mid bA \mid \lambda \\ S_2 &\rightarrow BbbB \\ B &\rightarrow aB \mid bB \mid \lambda \end{aligned}$$

- (d) Write a complete formal definition of a context-free grammar G_4 that generates $L_1 \cap L_2$. If such grammar does not exist, prove it.

Answer: $G_4 = (V_4, \Sigma, P_4, S_4)$, where $\Sigma = \{a, b\}$, $V_4 = \{S_4, L, R, A\}$, and P_4 is:

$$\begin{aligned} S_4 &\rightarrow L \mid R \\ L &\rightarrow AbbAbA \\ R &\rightarrow AbAbbA \\ A &\rightarrow aA \mid bA \mid \lambda \end{aligned}$$

Problem 375 Let L_1 be the set of strings over alphabet $\{a, b, c\}$ that contain at least two b 's. Let L_2 be the set of strings over alphabet $\{a, b, c\}$ that contain at least two c 's.

(a) Write a complete formal definition of a context-free grammar G_1 that generates $L_1 \cup L_2$. If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b, c\}$, $V = \{S, S_1, S_2, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow S_1 \mid S_2 \\ S_1 &\rightarrow AbAbAbA \\ S_2 &\rightarrow AcAcAcA \\ A &\rightarrow AA \mid \lambda \mid a \mid b \mid c \end{aligned}$$

- (b) Write a complete formal definition of a context-free grammar G_2 that generates $L_2 \setminus L_1$. If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b, c\}$, $V = \{S, S_1, S_2, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow S_2 S_1 \\ S_1 &\rightarrow AbAbAbA \\ S_2 &\rightarrow AcAcAcA \\ A &\rightarrow AA \mid \lambda \mid a \mid b \mid c \end{aligned}$$

Problem 376 (a) Let L be the set of strings over the alphabet $\{a, b, c\}$ whose length gives an odd remainder if divided by 7.

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ZA \mid ZZZA \mid ZZZZZA \\ A &\rightarrow AA \mid \lambda \mid ZZZZZZ \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

(b) Write a complete formal definition of a context-free grammar that generates \bar{L} , the complement of the language L described in part (a). If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow A \mid ZZA \mid ZZZA \mid ZZZZZA \\ A &\rightarrow AA \mid \lambda \mid ZZZZZZ \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 377 Let L be the set of all strings over alphabet $\{a, b, c\}$ that have odd length and b as the middle symbol.

(a) Write a complete formal definition of a context-free grammar G that generates L . If such a grammar does not exist, explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ZSZ \mid b \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

(b) Write a complete formal definition of a context-free grammar G that generates \bar{L} (the complement of L). If such a grammar does not exist, explain why.

Answer: We are looking for strings that either have even length or have odd length and the middle symbol different from b .

$G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, E, D, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow E \mid D \\ E &\rightarrow ZZE \mid \lambda \\ D &\rightarrow ZDZ \mid a \mid c \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 378 Let L be the set of all strings over alphabet $\{a, b, c\}$ that contain at most one b .

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow ABA \\ A &\rightarrow AA \mid \lambda \mid a \mid c \\ B &\rightarrow b \mid \lambda \end{aligned}$$

(b) Write a complete formal definition of a context-free grammar that generates \bar{L} (the complement of L). If such a grammar does not exist, prove it.

Answer: The language \bar{L} comprises strings with two or more b 's, which is generated by the grammar:

$G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow ABABA \\ A &\rightarrow AA \mid \lambda \mid a \mid b \mid c \\ B &\rightarrow b \end{aligned}$$

Problem 379 Let L be the set of strings over alphabet $\{a, b, c\}$ in which no two adjacent symbols are equal.

(a) Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aA \mid bB \mid cD \mid \lambda \\ A &\rightarrow bB \mid cD \mid \lambda \\ B &\rightarrow aA \mid cD \mid \lambda \\ D &\rightarrow aA \mid bB \mid \lambda \end{aligned}$$

(b) Write a complete formal definition of a context-free grammar that generates \bar{L} (the complement of L). If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, F\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AFA \\ A &\rightarrow AA \mid \lambda \mid a \mid b \mid c \\ F &\rightarrow aa \mid bb \mid cc \end{aligned}$$

Problem 380 Let L be the set of all non-empty strings over the alphabet $\{a, b, c\}$ where the first symbol is equal to the last symbol.

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aAa \mid bAb \mid cAc \mid a \mid b \mid c \\ A &\rightarrow aA \mid bA \mid cA \mid \lambda \end{aligned}$$

Problem 381 Let L be the set of all non-empty strings over the alphabet $\{a, b, c\}$ where the first symbol is equal to the last symbol.

Write a complete formal definition of a context-free grammar that generates \overline{L} (the complement of L .) If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aAb \mid aAc \mid bAa \mid bAc \mid cAa \mid cAb \mid \lambda \\ A &\rightarrow aA \mid bA \mid cA \mid \lambda \end{aligned}$$

Problem 382 Let L be the set of all strings over the alphabet $\{a, b, c\}$ that have even length.

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ZZS \mid \lambda \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

(b) Write a complete formal definition of a context-free grammar that generates \overline{L} (the complement of L .) If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ZZS \mid Z \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

Problem 383 Let L be the set of all strings over the alphabet $\{a, b, c\}$ that contain at least two b 's.

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AbAbA \\ A &\rightarrow \lambda \mid AA \mid a \mid b \mid c \end{aligned}$$

(b) Write a complete formal definition of a context-free grammar that generates \overline{L} (the complement of L .) If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow A \mid AbA \\ A &\rightarrow \lambda \mid AA \mid a \mid b \mid c \end{aligned}$$

Problem 384 Let L be the set of all strings over the alphabet $\{a, b, c\}$ that are palindromes. (A palindrome is a string that is equal to its reversal.)

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSa \mid bSb \mid cSc \mid \lambda \mid a \mid b \mid c$$

(b) Write a complete formal definition of a context-free grammar that generates \overline{L} (the complement of L .) If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aSa \mid bSb \mid cSc \mid A \\ A &\rightarrow aBb \mid aBc \mid bBa \mid bBc \mid cBa \mid cBb \\ B &\rightarrow \lambda \mid BB \mid a \mid b \mid c \end{aligned}$$

Problem 385 (a) Does there exist a pair of languages L_1 and L_2 such that all of the following three conditions hold?

- L_1 is regular;
- $L_2 \subseteq L_1$;
- L_2 is not regular, but is context-free.

If your answer to this part is “no” go to part (d), else complete parts (b)–(c).

Answer: Yes. In fact, for any alphabet Σ , the language Σ^* is regular, while every (non-regular) language over Σ is its subset. See parts (b)–(c) for a more interesting example where L_1 is infinite, with an infinite complement.

(b) Write a regular expression that defines L_1 (as in part (a)).

Answer:

$$a^*b^*$$

(c) Write a complete formal definition of a context-free grammar that generates L_2 (as in part (a)). Describe L_2 briefly, using words and set-notation.

Answer: Let L_2 be the language of strings over $\{a, b\}$ where all a 's precede all b 's and the number of a 's in the string equals the number of b 's:

$$L_2 = \{a^n b^n \mid n \geq 0\}$$

L_2 is generated by the context-free grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$ is the set of terminals; $V = \{S\}$ is the set of variables; S is the start symbol, and the production set P is:

$$S \rightarrow aSb \mid \lambda$$

The argument that shows that L_2 is not regular is almost identical to the one given in Problem 479.

(d) Explain why such a pair of languages L_1 and L_2 (as in part (a)) does not exist.

Problem 386 (a) Does there exist a pair of languages L_1 and L_2 such that all of the following three conditions hold?

- L_1 is not regular, but is context-free;
- $L_2 \subseteq L_1$;
- L_2 is regular.

If your answer to this part is “no” go to part (d), else complete parts (b)–(c).

Answer: Yes. In fact, for any alphabet Σ , the empty set \emptyset is regular, while every (non-regular) language over Σ contains it as a subset. See parts (b)–(c) for a more interesting example where L_2 is infinite.

(b) Write a regular expression that defines L_2 (as in part (a)).

Answer: Select L_2 to be any one of (infinitely many infinite and mutually disjoint) languages defined by regular expressions of the form:

$$\begin{aligned} &bb^* \\ &abbb^* \\ &aabb^* \\ &aaabbb^* \end{aligned}$$

etc. Each of these languages contains strings that begin with a certain *fixed* number of a 's, say k of them, which are followed by any number j of b 's, such that $j > k$.

(c) Write a complete formal definition of a context-free grammar that generates L_1 (as in part (a)). Describe L_1 briefly, using words and set-notation.

Answer: Let L_1 be the language of all strings over $\{a, b\}$ that contain more b 's than a 's, and all a 's precede all b 's. L_1 is generated by the context-free grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$ is the set of terminals; $V = \{S, B\}$ is the set of variables; S is the start symbol, and the production set P is:

$$\begin{aligned} S &\rightarrow aSb \mid B \\ B &\rightarrow bB \mid b \end{aligned}$$

The argument that shows that L_1 is not regular is almost identical to the one given in Problem 479.

(d) Explain why such a pair of languages L_1 and L_2 (as in part (a)) does not exist.

Problem 387 (a) Give an example of two context-free languages L_1 and L_2 whose union is not context-free. Give a precise definition of both languages and explain your answer. If such languages do not exist, explain why.

Answer: Such languages do not exist, since the class of context-free languages is closed under union.

(b) Give an example of two context-free languages L_1 and L_2 whose intersection is not context-free. Give a precise definition of both languages and explain your answer. If such languages do not exist, explain why.

Answer:

$$\begin{aligned} L_1 &= \{a^n b^n c^m \mid n, m \geq 0\} \\ L_2 &= \{a^n b^m c^n \mid n, m \geq 0\} \\ L_1 \cap L_2 &= \{a^n b^n c^n \mid n \geq 0\} \end{aligned}$$

Whereas L_1 and L_2 are context-free, $L_1 \cap L_2$ is a canonical non-context-free language, easily proved to be so by the Pumping Lemma.

(c) Give an example of two regular languages L_1 and L_2 whose intersection is not regular. Give a precise definition of both languages and explain your answer. If such languages do not exist, explain why.

Answer: Such languages do not exist, since the class of regular languages is closed under intersection.

(d) Give an example of a regular language L_1 that has a subset L_2 which is not context-free. Give a precise definition of both languages and explain your answer. If such a language does not exist, explain why.

Answer: Let:

$$\begin{aligned} L_1 &= a^* b^* c^* \\ L_2 &= \{a^n b^n c^n \mid n \geq 0\} \end{aligned}$$

L_1 is certainly regular, since it has a regular expression. L_2 is not context-free, as stated in part (b). Evidently, $L_2 \subset L_1$.

(e) Give an example of an infinite context-free language L_1 that is not regular but has a subset L_2 which is regular. Give a precise definition of both languages and explain your answer. If such a language does not exist, explain why.

Answer: Let:

$$\begin{aligned} L_1 &= \{a^n b^n \mid n \geq 0\} \\ L_2 &= \{ab\} \end{aligned}$$

L_1 is a canonical context-free language, easily shown not to be regular by the Pumping Lemma. L_2 is finite, and thereby certainly regular. Evidently, $L_2 \subset L_1$.

(f) Give an example of a regular language L_1 that has a subset L_2 which is not regular. Give a precise definition of both languages and explain your answer. If such a language does not exist, explain why.

Answer: Let:

$$\begin{aligned} L_1 &= a^* b^* \\ L_2 &= \{a^n b^n \mid n \geq 0\} \end{aligned}$$

L_1 is certainly regular, since it has a regular expression. L_2 is not regular, as stated in part (c). Evidently, $L_2 \subset L_1$.

Problem 388 (a) Give an example of a language that is not regular and does not have any regular subsets. Provide a precise definition of this language and explain your answer briefly. If such a language does not exist, explain why.

Answer: Impossible— \emptyset is straightforwardly regular, but is a subset of any other language.

(b) Give an example of a language that is not context-free and does not have any context-free supersets. Provide a precise definition of this language and explain your answer briefly. If such a language does not exist, explain why.

Answer: Impossible— Σ^* is straightforwardly context-free, but is a superset of any other language.

(c) Give an example of a language that is not countable. Provide a precise definition of this language and explain your answer briefly. If such a language does not exist, explain why.

Answer: Impossible—every language is a subset of the set of all finite strings, and this set is countable.

Problem 389 (a) Construct a regular expression that defines a language L_1 such that

$$L_1 \subseteq \{a, b, c\}^* \text{ and } |L_1| = 5$$

and list all the elements of L_1 . If such regular expression does not exist, explain why.

Answer: There are infinitely many correct constructions. For example, the regular expression:

$$\lambda \cup a \cup b \cup c \cup ab$$

defines the language

$$L_1 = \{\lambda, a, b, c, ab\}$$

(b) Construct a context-free grammar G that generates a language L_2 such that:

$$L_2 \subseteq \{a, b\}^* \text{ and } |L_2| > \aleph_0$$

If such grammar does not exist, explain why.

Answer: Impossible. The set $\{a, b\}^*$ of all strings over alphabet $\{a, b\}$ is countable, and it cannot have an uncountable subset.

(c) Construct a finite automaton M that accepts a language L_3 such that

$$L_3 \subset \{a\}^* \text{ and } |L_3| = \aleph_0 \text{ and } |\{a\}^* \setminus L_3| = \aleph_0$$

If such automaton does not exist, explain why.

Answer: We are looking for an infinite subset of $\{a\}^*$ whose complement in $\{a\}^*$ is also infinite. There are infinitely many possible constructions. For example, let L_3 be the set of all strings over alphabet $\{a\}$ that have even length. Then $\{a\}^* \setminus L_3$ is the set of all strings over the same alphabet that have odd length. Evidently, both L_3 and $\{a\}^* \setminus L_3$ are infinite.

The state transition graph of the finite automaton that accepts L_3 is given on Figure 7.

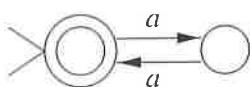


Figure 7:

Problem 390 Let L be the set of all strings over $\{a, b, c\}$ that contain ab or ac as a substring. Write

a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AabA \mid AacA \\ A &\rightarrow AA \mid \lambda \mid a \mid b \mid c \end{aligned}$$

Problem 391 For each of the following claims, circle the word “yes” that follows the claim if the claim is correct, and circle the word “no” that follows the claim if the claim is not correct.

1. Every regular language has a context-free grammar. yes no
2. The union of any two regular languages is context-free. yes no
3. The concatenation of any two context-free languages is context-free. yes no
4. Every regular language is finite. yes no
5. Every language over a finite alphabet is regular. no yes
6. Every language over a finite alphabet is context-free. no yes
7. Some infinite languages are not regular. yes no
8. Some infinite languages are not context-free. yes no
9. Some finite languages are not context-free. no yes
10. Every set of strings is countable. yes no
11. Every language is countable. yes no
12. Some infinite languages are not countable. no yes
13. There are infinitely many regular expressions. yes no
14. There are countably many context-free grammars. yes no
15. There are countably many regular languages. yes no
16. There are countably many languages over $\{0, 1\}$. no yes
17. Some subsets of $\{0, 1\}^*$ are uncountable. no yes

18. The set of all subsets of $\{0, 1\}^*$ is countable.

no

19. The set of all finite subsets of $\{0, 1\}^*$ is uncountable.

no

4 Finite Automata

Problem 392 Let M be the finite automaton represented by the state diagram on Figure 8, and let L be the language accepted by M .

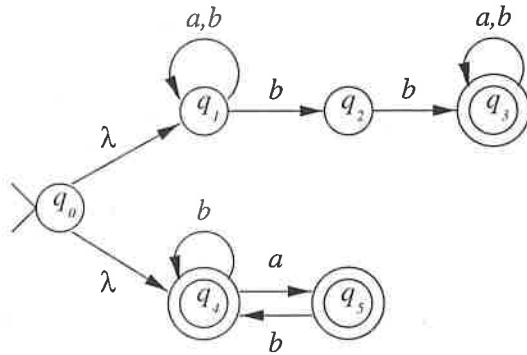


Figure 8:

- (a) Write 10 distinct strings that belong to L .
(b) Write 10 distinct strings over alphabet $\{a, b\}$ that do not belong to L .

Answer: Note that L is the set of strings over $\{a, b\}$ that do not contain aa as a substring or contain bb as a substring. \bar{L} is, therefore, the set of strings over $\{a, b\}$ that do not contain bb as substring, but do contain aa as a substring.

$\in L$	$\notin L$
a	aa
b	aab
$bbaa$	$aaab$
ba	$aabababab$
bb	$baab$
$bbaaa$	$babaa$
$babab$	$baabab$
$abbb$	$abaa$
$bbab$	$aabaa$
$aaaaaaaaabb$	$abaab$

Problem 393 Let M be the finite automaton represented by the state diagram on Figure 9, and let L be the language accepted by M .

- (a) Write 10 distinct strings that belong to L .
(b) Write 10 distinct strings over alphabet $\{a, b, c\}$ that do not belong to L .

Answer: Note that L is represented by a regular expression:

$$(abc)^*a^* \cup ((a \cup b \cup c)^*c(a \cup b \cup c))$$

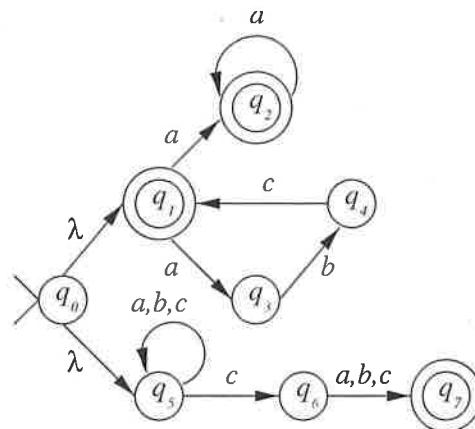


Figure 9:

$\in L$	$\notin L$
a	b
aa	bb
abc	c
$abca$	$aabc$
$abcabc$	bcb
ca	$abccb$
cb	cba
$abcabcaa$	bc
$abcabbbbcc$	ac
$bbbbca$	abb

Problem 394 Let L_1 be the language defined by the regular expression

$$(a \cup bb \cup bc)^*cc(a \cup bb \cup bc)^*$$

- (a) Write 10 distinct strings that belong to L_1 . If such strings do not exist, explain why.
(b) Write 10 distinct strings over alphabet $\{a, b, c\}$ that do not belong to L_1 . If such strings do not exist, explain why.

Answer:

$\in L_1$	$\notin L_1$
cc	ϵ
acc	ac
$bbacc$	bb
$bccc$	bcc
$accbca$	aa
$accbb$	abb
$bbccbc$	b
$bbcca$	$bacca$
$ccbb$	$ccba$
cca	ccb

Let M be the finite automaton represented by the state diagram on Figure 10, and let L_2 be the language accepted by M .

- (c) Write 10 distinct strings that belong to L_2 . If such strings do not exist, explain why.
(d) Write 10 distinct strings over alphabet $\{0, 1\}$ that do not belong to L_2 . If such strings do not exist, explain why.

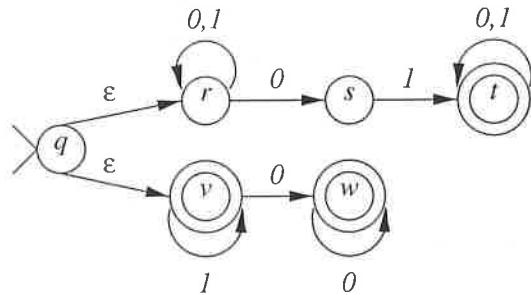
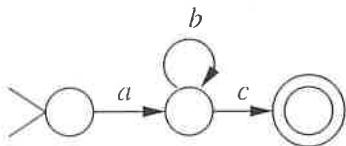


Figure 10:

Answer: $L_2 = \{0, 1\}^*$. To see this, note that M is taken to its accepting state t by strings that contain 01 as a substring, while it ends at its accepting states v and w after processing those strings that do not contain 01 as a substring. Thus, there are no strings over alphabet $\{0, 1\}$ that do not belong to L_2 .

$\in L_2$
ϵ
0
1
00
01
10
11
000
001
010

Problem 395 Let L_1 be the language accepted by the finite automaton given on Figure 11.

Figure 11: Language L_1

Let L_2 be the language generated by the context-free grammar $G_2 = (V, \Sigma, P, S_2)$, where $\Sigma = \{a, b, c\}$, $V = \{S_2\}$, and the production set P is:

$$S_2 \rightarrow aS_2c \mid b$$

(a) Write 5 distinct strings that belong to L_1 but do not belong to L_2 (belong to $L_1 \setminus L_2$). If such strings do not exist, state it and explain why.

Answer:

$$ac, abbc, abbac, abbbbc, abbbbcc$$

(b) Write 5 distinct strings that belong to L_2 but do not belong to L_1 (belong to $L_2 \setminus L_1$). If such strings do not exist, state it and explain why.

Answer:

$$b, aabcc, aaabccc, aaaabcccc, aaaaabcccc$$

(c) Write 5 distinct strings that belong to L_1 and L_2 (belong to $L_1 \cap L_2$). If such strings do not exist, state it and explain why.

Answer: Only one such string exists: abc . This happens because the template for L_1 is $ab^n c$, while the template for L_2 is $a^k b c^k$, and the only instance when the two templates yield the same string occurs when $n = 1$ and $k = 1$.

(d) Write 5 distinct strings over alphabet $\{a, b, c\}$ that do not belong to L_1 and do not belong to L_2 (belong to $\overline{L_1} \cap \overline{L_2}$). If such strings do not exist, state it and explain why.

Answer:

$$\lambda, a, c, ba, ca$$

(e) Write 5 distinct strings that belong to $L_2 L_2$ but do not belong to L_2 (belong to $L_2 L_2 \setminus L_2$). If such strings do not exist, state it and explain why.

Answer:

$$bb, abcb, babc, aabccb, abcabc$$

(f) Write all distinct strings that belong to $L_1 L_2$ and $L_2 L_1$ (belong to $L_1 L_2 \cap L_2 L_1$). If this is impossible, state it and explain why.

Answer:

$$abcabc$$

Problem 396 Let L be the language defined by the regular expression:

$$(a \cup c)^* (b \cup c)^* \cup bbb$$

(a) Construct a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 12.

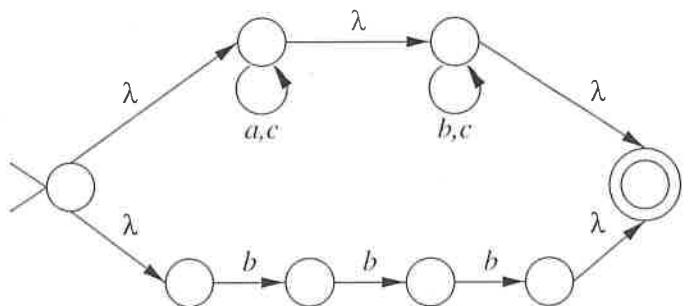


Figure 12:

(b) Does there exist an algorithm to convert an arbitrary regular expression into an equivalent finite automaton? Explain your answer briefly.

Answer: Yes. The construction is performed by a recursive decomposition of the regular expression into two

expressions connected by the outer-most regular operator. If such two expressions, R_1 and R_2 , are converted into two (canonical) finite automata, M_1 and M_2 , with initial states q_1, q_2 respectively, and final states f_1, f_2 respectively, then their composition is converted to an automaton M , whose new initial and final states are q, f respectively, so that the composition is effected by new transitions, which are:

1. for union $R_1 \cup R_2$:

$$\begin{aligned} [q, \lambda, q_1]; & [q, \lambda, q_2]; \\ [f_1, \lambda, f]; & [f_2, \lambda, f]; \end{aligned}$$

2. for concatenation $R_1 \cdot R_2$:

$$\begin{aligned} [q, \lambda, q_1]; \\ [f_1, \lambda, q_2]; \\ [f_2, \lambda, f]; \end{aligned}$$

3. for Kleene star R_1^* :

$$\begin{aligned} [q, \lambda, q_1]; \\ [q_1, \lambda, f_1]; \\ [f_1, \lambda, q_1]; \\ [f_1, \lambda, f]; \end{aligned}$$

In the base case, \emptyset is accepted by an automaton with the empty transition set, empty string is accepted by an automaton whose only transition is of the form:

$$[q, \lambda, f]$$

and an individual letter is accepted by an automaton whose only transition is of the form:

$$[q, a, f]$$

Problem 397 Let L be the language defined by the regular expression:

$$(a \cup bc)^* (dca)^* (b \cup d)$$

Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 13.

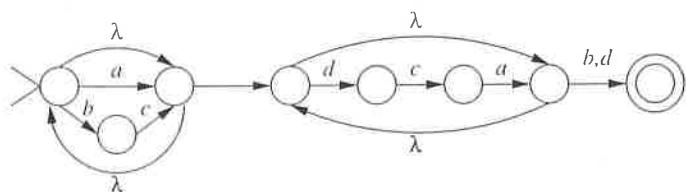


Figure 13:

Problem 398 Let L be the set of all strings over the alphabet $\{a, b, c, d\}$ that begin with d .

(a) Write a regular expression that represents the language L . If such a regular expression does not exist, explain why.

Answer:

$$d(a \cup b \cup c \cup d)^*$$

(b) Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S \rightarrow dA \\ A \rightarrow \lambda \mid AA \mid a \mid b \mid c \mid d \end{aligned}$$

(c) Draw a state-transition graph of a finite automaton that accepts the language L . If such an automaton does not exist, explain why.

Answer: See Figure 14.

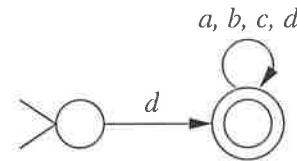


Figure 14:

Problem 399 Let L be the language defined by the regular expression:

$$cc(ba \cup d)^* ad \cup (acc)^*$$

(a) Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S \rightarrow ccBad \mid A \\ B \rightarrow \lambda \mid BB \mid ba \mid d \\ A \rightarrow accA \mid \lambda \end{aligned}$$

(b) Draw a state-transition graph of a finite automaton that accepts the language L . If such an automaton does not exist, explain why.

Answer: See Figure 15.

Problem 400 Let L be the language defined by the regular expression:

$$(ac)^* b^* \cup c(b \cup c)^*(bb \cup ac)$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

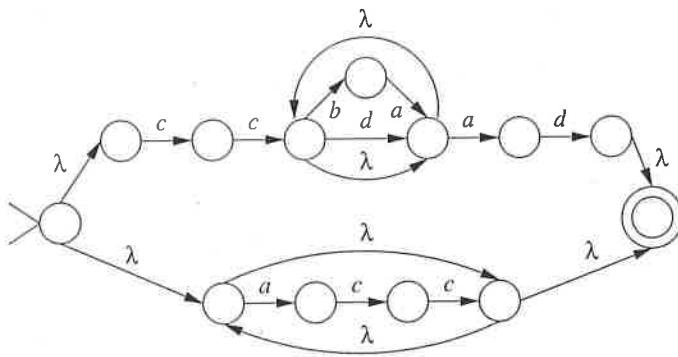


Figure 15:

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$,
 $V = \{S, A, B, D, E\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \mid cDE \\ A &\rightarrow \lambda \mid AA \mid ac \\ B &\rightarrow bB \mid \lambda \\ D &\rightarrow \lambda \mid DD \mid b \mid c \\ E &\rightarrow bb \mid ac \end{aligned}$$

(b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 16.

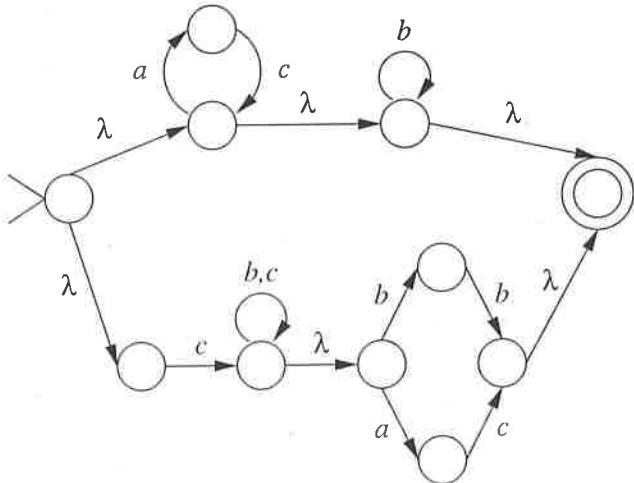


Figure 16:

Problem 401 Let L be the language defined by the regular expression:

$$(a \cup c)^* b^* c \cup a ((bc)^* \cup a^*)$$

(a) Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$,

$V = \{S, A, B, D, E, F\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ABC \mid aD \\ A &\rightarrow \lambda \mid AA \mid a \mid c \\ B &\rightarrow bB \mid \lambda \\ D &\rightarrow E \mid F \\ E &\rightarrow bcE \mid \lambda \\ F &\rightarrow aF \mid \lambda \end{aligned}$$

(b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 17.

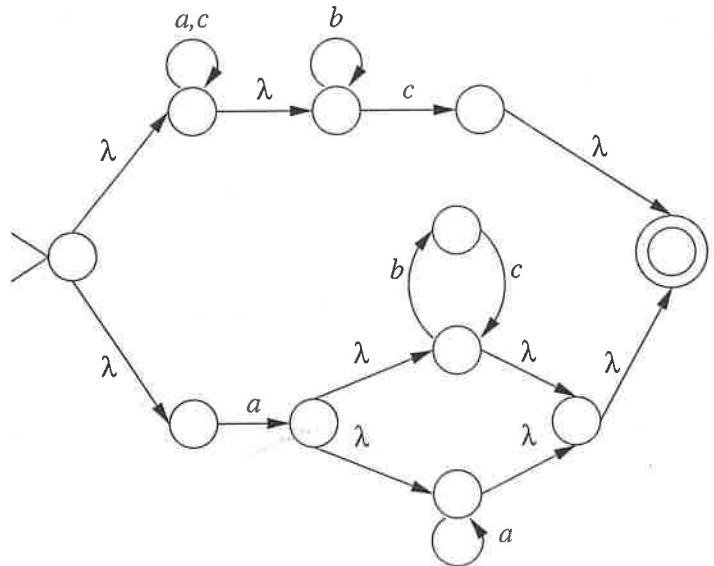


Figure 17:

Problem 402 Let L be the language defined by the regular expression:

$$cab^* (ba \cup d)^* \cup (bac)^*$$

(a) Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$,
 $V = \{S, A, B, D, E\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow A \mid B \\ A &\rightarrow caDE \\ D &\rightarrow bD \mid \lambda \\ E &\rightarrow \lambda \mid EE \mid ba \mid d \\ B &\rightarrow bacB \mid \lambda \end{aligned}$$

(b) Draw a state-transition graph of a finite automaton that accepts the language L . If such an automaton does not exist, state it and explain why.

Answer: See Figure 18.

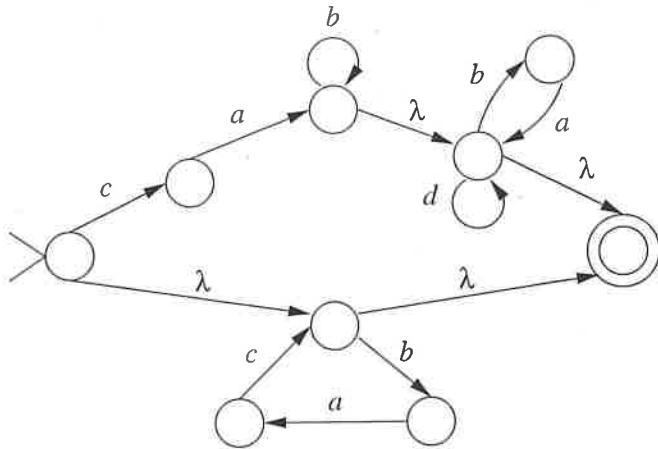


Figure 18:

Problem 403 Let L be the language defined by the regular expression:

$$(ca \cup b^*) ((bad)^* \cup (ba \cup c)^*)$$

(a) Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D, E, F\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow ca \mid D \\ D &\rightarrow bD \mid \lambda \\ B &\rightarrow E \mid F \\ E &\rightarrow \lambda \mid EE \mid bad \\ F &\rightarrow \lambda \mid FF \mid ba \mid c \end{aligned}$$

(b) Draw a state-transition graph of a finite automaton that accepts the language L . If such an automaton does not exist, state it and explain why.

Answer: See Figure 19.

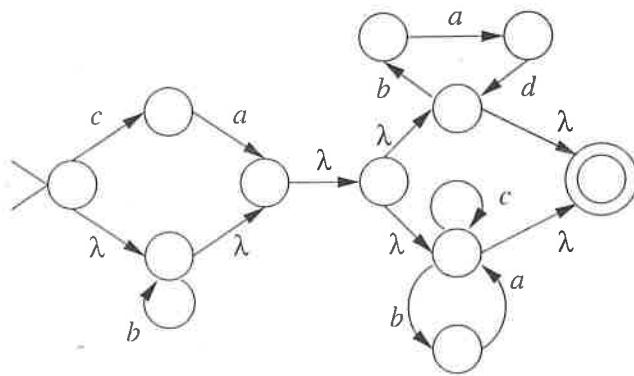


Figure 19:

Problem 404 Let L be the language defined by the regular expression:

$$a(b \cup e)^*a(b \cup d) \cup (ab)^*$$

(a) Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, d, e\}$, $V = \{S, A, B, D, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow A \mid B \\ A &\rightarrow aDaZ \\ Z &\rightarrow b \mid d \\ D &\rightarrow DD \mid \lambda \mid b \mid e \\ B &\rightarrow abB \mid \lambda \end{aligned}$$

(b) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 20.

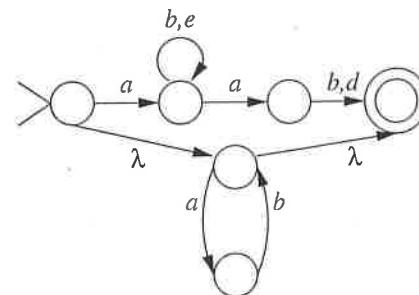


Figure 20:

Problem 405 Let L be the language defined by the regular expression:

$$(abc)^* \cup (ab)^*(ba)^*$$

(a) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 21.

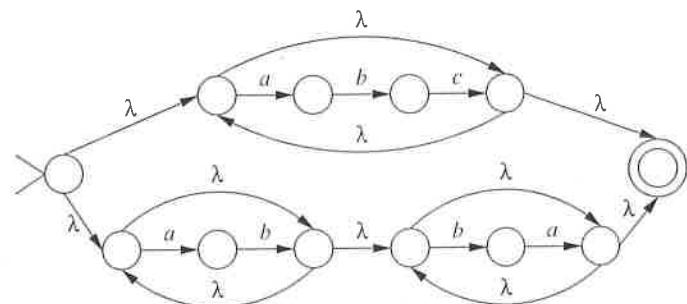


Figure 21:

(b) Is the complement \bar{L} of the language L regular? Explain your answer briefly.

Answer: Yes— \bar{L} is regular because it is defined by a regular expression; the complement of any regular language is regular.

Problem 406 Let L be the language defined by the regular expression:

$$(a \cup bc)^*(dd \cup g)^* \cup ab^*$$

Construct a state-transition graph of a finite automaton M that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 22.

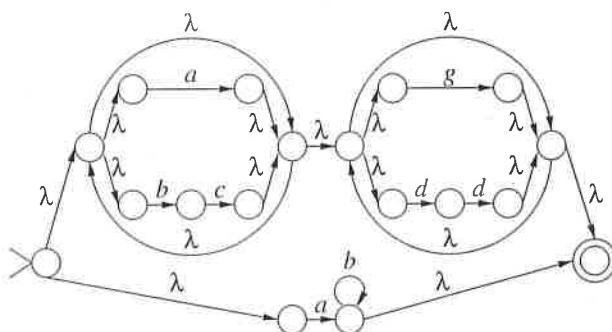


Figure 22:

Problem 407 Let L be the language defined by the regular expression:

$$(a \cup ba)^*(ba \cup bb \cup bc)^*$$

(a) Write a complete formal definition or a state-transition graph of a finite automaton M that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 23.

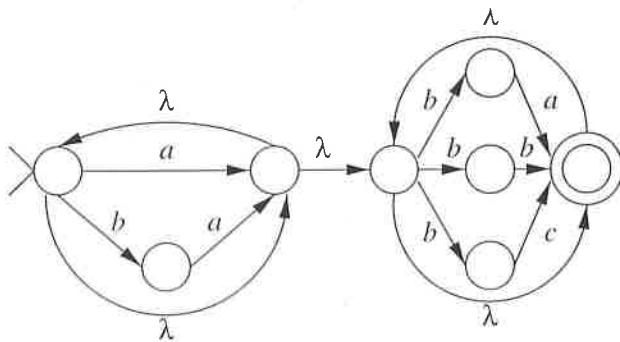


Figure 23:

(b) Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:
 $\Sigma = \{a, b, c\}$, $V = \{S, A, B\}$, and P comprises:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow AA \mid \lambda \mid a \mid ba \\ B &\rightarrow BB \mid \lambda \mid ba \mid bb \mid bc \end{aligned}$$

Problem 408 Let L be the set of all strings over the alphabet $\{a, b, c\}$ where the number of a 's is not divisible by 3.

(a) Write a regular expression that represents the language L . If such a regular expression does not exist, state it and explain why.

Answer:

$$(\beta a \beta a \beta a \beta)^* \beta a \beta (a \cup \lambda) \beta$$

where $\beta = (b \cup c)^*$

(b) Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, D, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow DaDaDaDS \mid A \\ A &\rightarrow DaD \mid DaDaD \\ D &\rightarrow bD \mid cD \mid \lambda \end{aligned}$$

(c) Draw a state-transition graph of a finite automaton that accepts the language L . If such an automaton does not exist, state it and explain why.

Answer: See Figure 24.

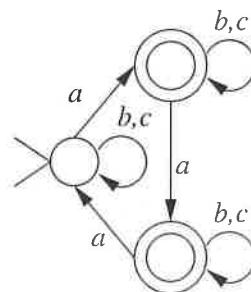


Figure 24:

Problem 409 Let L be the language defined by the regular expression

$$b(a \cup b^*((c^* \cup (cb)^*)ac)^*)b$$

(a) Construct a finite automaton M that accepts L . If such an automaton M does not exist, explain why.

Answer: The state diagram of M is given on Figure 25.

(b) If you constructed an automaton M in your answer to part (a), is M deterministic? Justify briefly your answer.

Answer: No—for example, M has ϵ transitions, etc.

Problem 410 Let L be the language defined by the regular expression

$$((abc \cup d)^* \cup (ab)^*)^*$$

(a) Write a complete formal definition or a state-transition graph of a finite automaton M that accepts L .

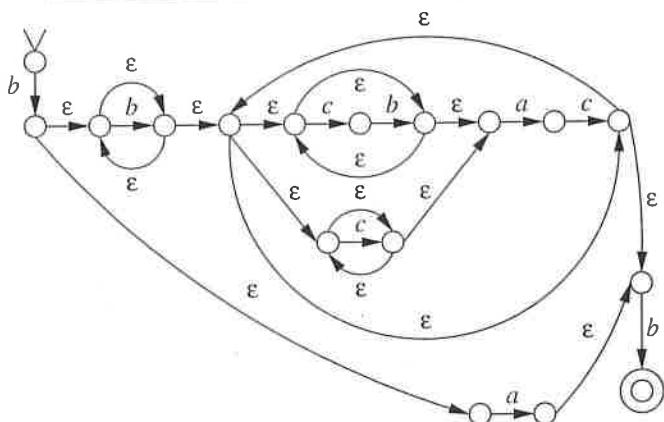


Figure 25:

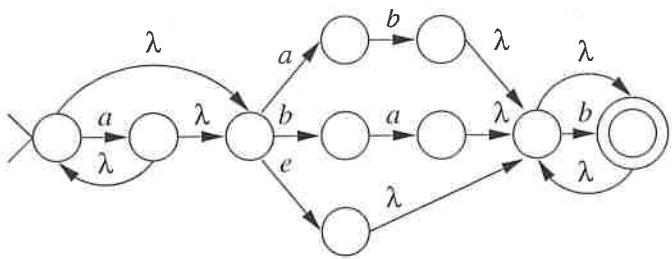


Figure 27:

Problem 412 Let L be the language defined by the regular expression

$$((ab)^* \cup (bc)^*)ab$$

- (a) Construct a finite automaton M that accepts L .

Answer: See Figure 28.

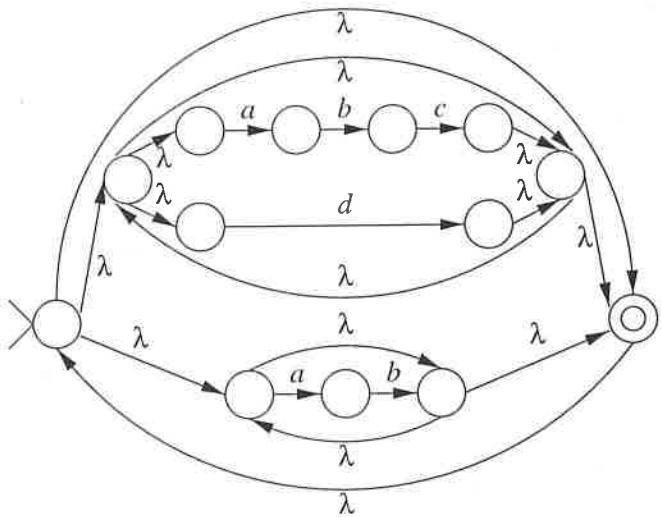


Figure 26:

Answer: See figure Figure 26.

- (b) Is the finite automaton M which you constructed in your answer to part (a) deterministic? Justify briefly your answer.

Answer: No. Its transition graph contains arcs labeled by λ , which correspond to non-deterministic transitions on null input string.

Problem 411 Let L be the language defined by the regular expression

$$a^*(ab \cup ba \cup e)b^*$$

- (a) Construct a finite automaton M that accepts L .

Answer: See Figure 27.

- (b) Is the finite automaton M that you constructed in your answer to part (a) deterministic? Justify briefly your answer.

Answer: No. For example, it does not have a b -transition out of the initial state; it has λ -transitions.

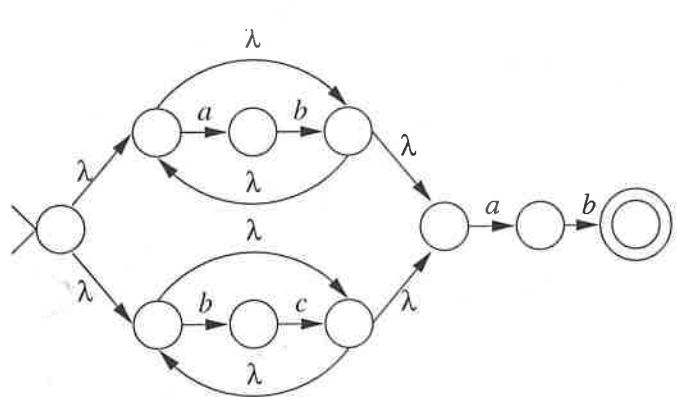


Figure 28:

- (b) Is the finite automaton M that you constructed in your answer to part (a) deterministic? Justify briefly your answer.

Answer: No. For example, it does not have an a -transition out of the initial state; it has λ -transitions.

Problem 413 Let L be the language defined by the regular expression

$$(a \cup b)^*(c \cup a) \cup (ccc(ab \cup cb \cup bb))^*$$

- (a) Write 5 distinct strings that belong to L . If such strings do not exist, explain why.

- (b) Write 5 distinct strings over alphabet $\{a, b, c\}$ that do not belong to L . If such strings do not exist, explain why.

Answer:

$\in L$	$\notin L$
λ	cc
c	ccc
a	$ccca$
ac	$cccb$
$cccab$	$aaab$

- (c) Write a complete formal definition or a state-transition graph of a finite automaton M that accepts L . If such automaton does not exist, explain why.

Answer: The state transition graph of the finite automaton that accepts L is given on Figure 29.

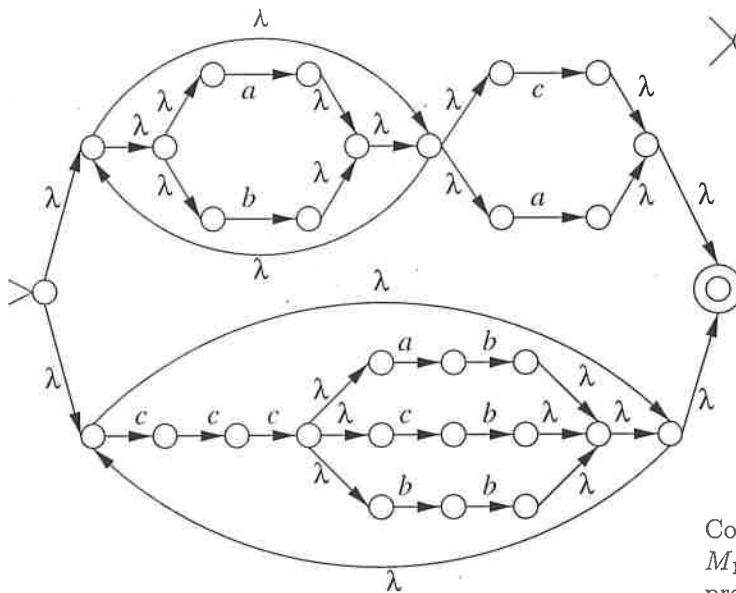


Figure 29:

Problem 414 Let L be the language defined by the regular expression:

$$(a \cup b \cup c)(a^* \cup ba^* \cup ca^*) \cup b^* \cup (b \cup c)^*$$

- (a) Write a complete formal definition or a state-transition graph of a finite automaton M that accepts L . If such automaton does not exist, explain why.

Answer: The state transition graph of the finite automaton that accepts L is given on Figure 30.

- (b) Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, explain why.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c\}$, $V = \{S, S_1, S_2, S_3, A, B, D\}$, and P comprises:

$$\begin{aligned} S &\rightarrow S_1 \mid S_2 \mid S_3 \\ S_1 &\rightarrow AB \\ A &\rightarrow a \mid b \mid c \\ B &\rightarrow D \mid bD \mid cD \\ D &\rightarrow DD \mid a \mid \lambda \\ S_2 &\rightarrow S_2 S_2 \mid b \mid \lambda \\ S_3 &\rightarrow S_3 S_3 \mid b \mid c \mid \lambda \end{aligned}$$

Problem 415 (a) Let:

$$\Sigma = \{a, b, c\}$$

and let L_1 be the set of all strings over Σ defined by the following regular expression:

$$(ab)^*(c \cup ba)(ba)^*$$

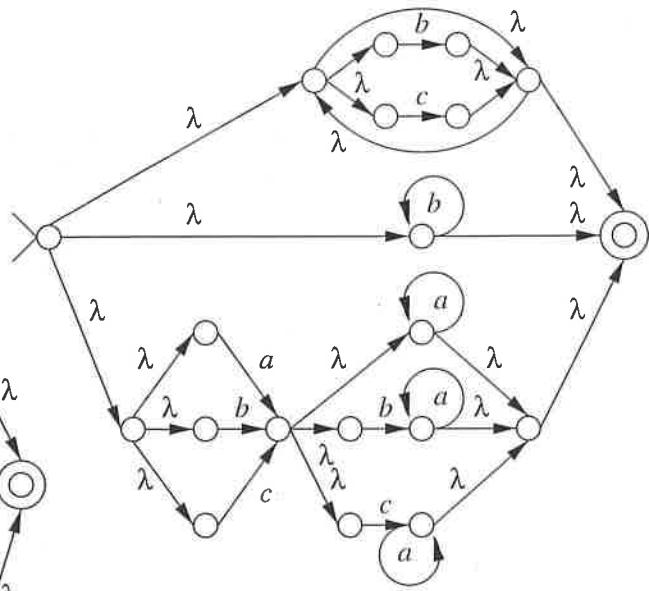


Figure 30:

Construct a state-transition graph of a finite automaton M_1 that accepts L_1 . If such automaton does not exist, prove it.

Answer: See Figure 31.

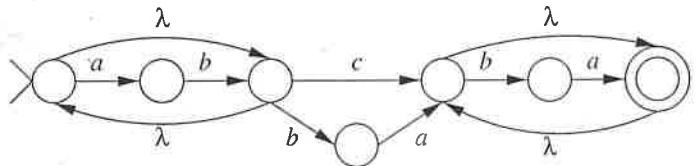


Figure 31:

- (b) Let:

$$\Sigma = \{a, b, c\}$$

and let L_2 be defined as follows:

$$L_2 = \{a^m b^n c \mid 0 \leq n \leq m \leq 3n\}$$

Write a complete formal definition of a context-free grammar G that generates L_2 . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c\}$, $V = \{S, A\}$, and P is:

$$\begin{aligned} S &\rightarrow Ac \\ A &\rightarrow aAb \mid aaAb \mid aaaAb \mid \lambda \end{aligned}$$

Problem 416 (a) Let L_1 be the language defined by the regular expression:

$$((a \cup bca)^* bab^*)^* \cup abb$$

Construct a state-transition graph of a finite automaton M_1 that accepts L_1 . If such automaton does not exist, prove it.

Advice for Answer: Conversion of a regular expression to a finite automaton is algorithmic.

(b) Let L_2 be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, B, D, E, F, G\}$, and P is:

$$\begin{aligned} S &\rightarrow aB \mid bD \mid cE \mid a \\ B &\rightarrow bD \mid cE \mid aF \mid b \\ D &\rightarrow cE \mid aF \mid bG \mid c \\ E &\rightarrow aF \mid bG \mid cS \mid \lambda \\ F &\rightarrow a \mid b \mid c \mid \lambda \\ G &\rightarrow a \mid b \mid c \end{aligned}$$

Construct a state-transition graph of a finite automaton M_2 that accepts L_2 . If such automaton does not exist, prove it.

Advice for Answer: Conversion of a regular grammar to a finite automaton is algorithmic.

Problem 417 Let L be the language defined by the regular expression:

$$((a \cup b)^* (bc \cup ca)^*)^* \cup (bac)^*$$

Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 32.

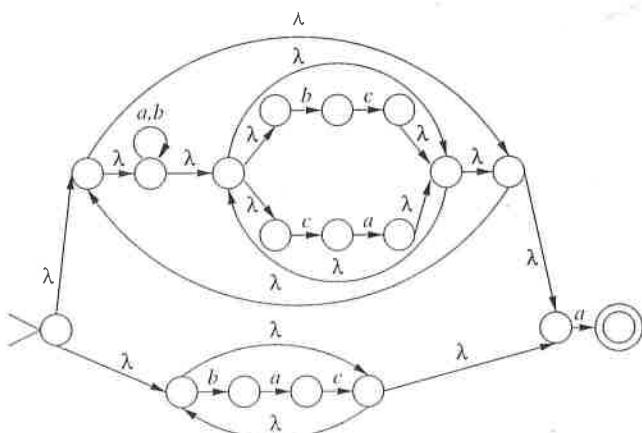


Figure 32:

Problem 418 Let L be the language defined by the regular expression:

$$((a \cup b) (c \cup d) b)^* \cup a^* b^* \cup (a \cup b)^*$$

Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 33.

Problem 419 Let L be the language defined by the regular expression:

$$(ab \cup bc)^* b^* ab \cup bca$$

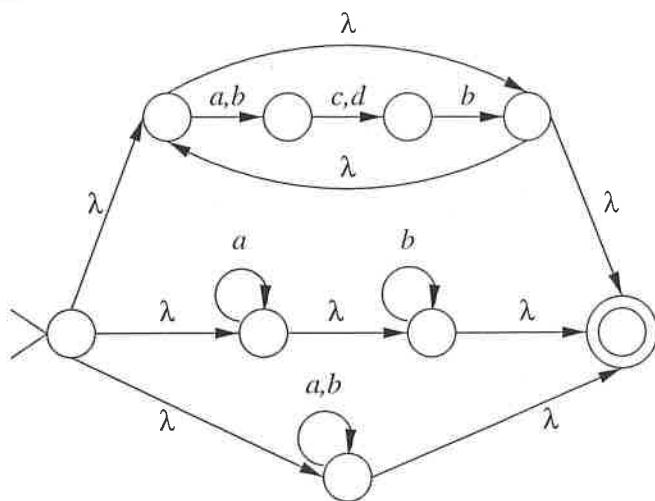


Figure 33:

(a) Construct a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Advice for Answer: Conversion of a regular expression to a finite automaton is algorithmic.

(b) Is L recursive (decidable?) Explain your answer briefly.

Advice for Answer: Every regular language is decidable.

Problem 420 Let L be the language defined by the regular expression:

$$(ac)^* (bc)^* \cup (bb)^* \cup (aa)^*$$

(a) Construct a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 34.

(b) Is \bar{L} (the complement of L) context-free? Explain your answer briefly.

Answer: Yes— \bar{L} is regular because L is regular and the class of regular languages is closed under complement; every regular language is context-free.

Problem 421 Let L be the language defined by the regular expression:

$$(ca)^* gg (bd)^* \cup (ab)^*$$

(a) Construct a state-transition graph of a finite automaton that accepts language LL . If such automaton does not exist, prove it.

Answer: See Figure 35.

(b) Construct a state-transition graph of a finite automaton that accepts language L^* . If such automaton does not exist, prove it.

Answer: See Figure 36.

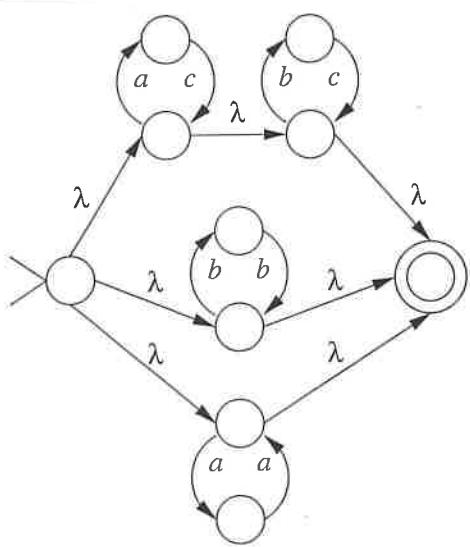


Figure 34:

Problem 422 Let L be the language defined by the regular expression:

$$(ca)^*(bd)^* \cup (aa)^*$$

(a) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 37.

(b) Construct a state-transition graph of a finite automaton that accepts L^* . If such an automaton does not exist, prove it.

Answer: See Figure 38.

Problem 423 Let L be the language defined over alphabet $\{a, b, c, d, g\}$ by the regular expression:

$$(a \cup b \cup g)^* cdc((ab \cup bg)^* \cup dd)$$

(a) Write a complete formal definition of a context-free grammar G that generates L . If such a grammar does not exist, explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, g\}$, $V = \{S, A, B, D, E\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow Acdbc \\ A &\rightarrow AA \mid \lambda \mid a \mid b \mid g \\ B &\rightarrow D \mid E \\ D &\rightarrow DD \mid \lambda \mid ab \mid bg \\ E &\rightarrow dd \end{aligned}$$

(b) Construct a state transition graph of a finite automaton that accepts L . If such an automaton does not exist, explain why.

Answer: See Figure 39.

(c) Write a complete formal definition of a context-free grammar G_1 that generates L^* . If such a grammar does not exist, explain why.

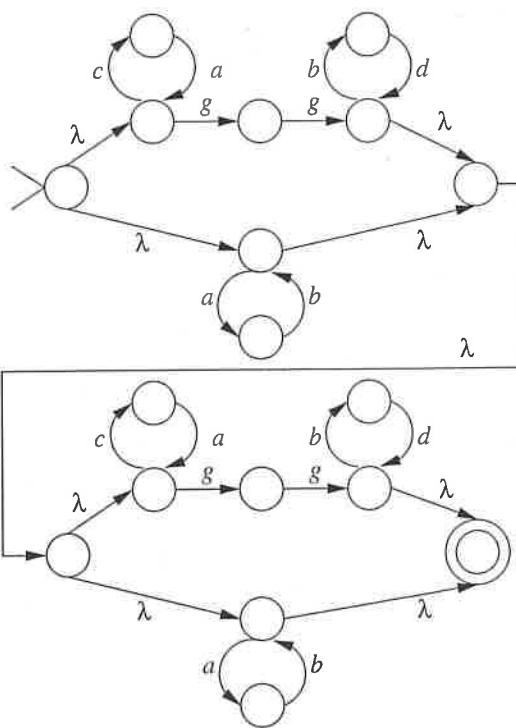


Figure 35:

Answer: $G_1 = (V_1, \Sigma, P_1, S_1)$, where $\Sigma = \{a, b, c, d, g\}$, $V_1 = \{S_1, S, A, B, D, E\}$, and the production set P_1 is:

$$\begin{aligned} S_1 &\rightarrow S_1 S_1 \mid \lambda \mid S \\ S &\rightarrow Acdbc \\ A &\rightarrow AA \mid \lambda \mid a \mid b \mid g \\ B &\rightarrow D \mid E \\ D &\rightarrow DD \mid \lambda \mid ab \mid bg \\ E &\rightarrow dd \end{aligned}$$

Problem 424 Let L be the language defined by the regular expression:

$$(aa \cup dd)(b \cup ee)^*(ad \cup da)$$

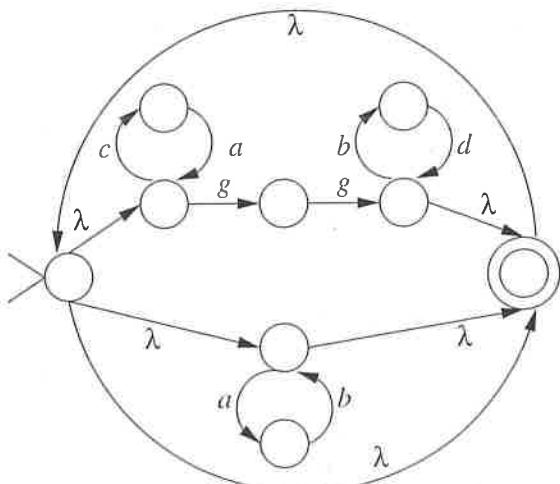


Figure 36:

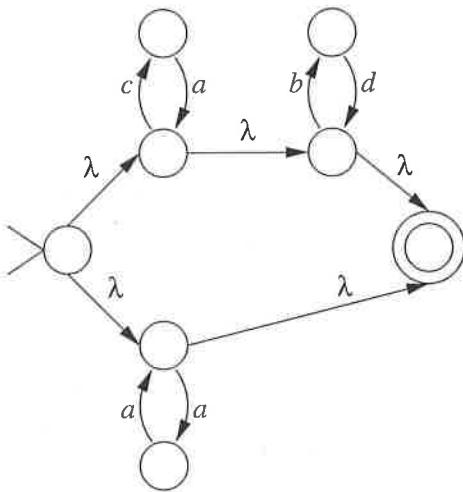


Figure 37:

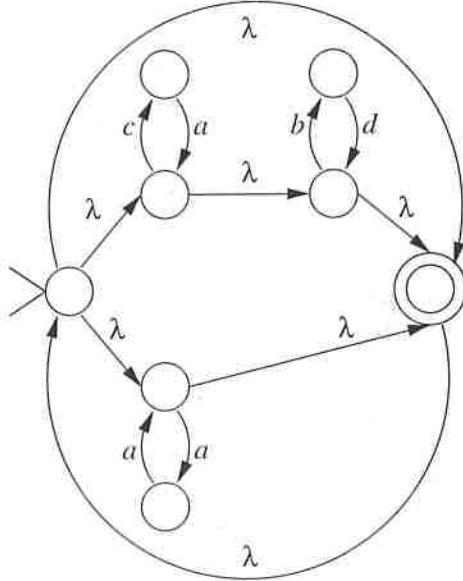


Figure 38:

- (a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, d, e\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ABD \\ A &\rightarrow aa \mid dd \\ D &\rightarrow ad \mid da \\ B &\rightarrow BB \mid \lambda \mid b \mid ee \end{aligned}$$

- (b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 40.

Problem 425 Let L be the language defined by the regular expression:

$$(ab)^*(dd)^* \cup c(a \cup b)^*cc$$

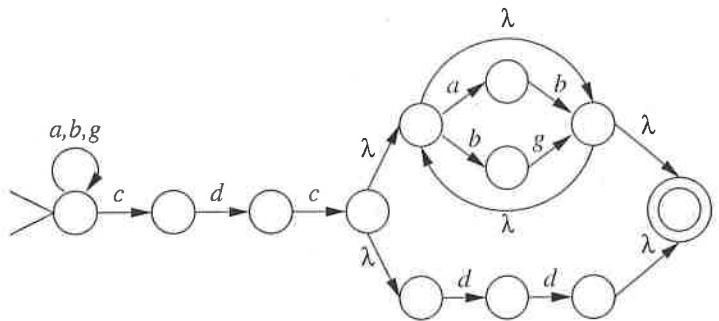


Figure 39:

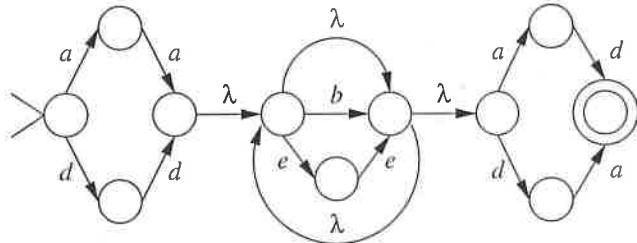


Figure 40:

Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 41.

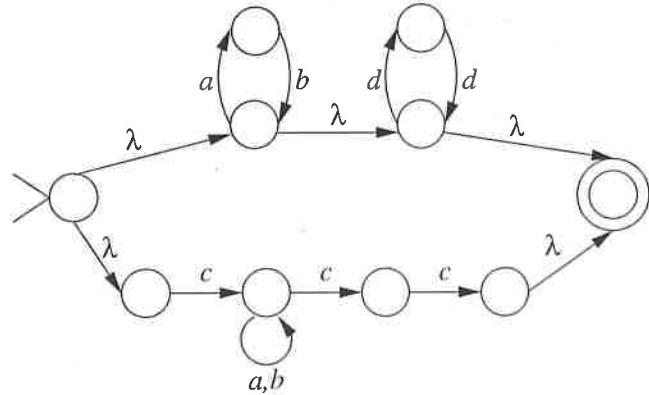


Figure 41:

Problem 426 Let L be the language defined by the regular expression:

$$((ab \cup db)^*(bd \cup ed)^*) \cup abd$$

- (a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b, d, e\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \mid abd \\ A &\rightarrow AA \mid \lambda \mid ab \mid db \\ B &\rightarrow BB \mid \lambda \mid bd \mid ed \end{aligned}$$

- (b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 42.

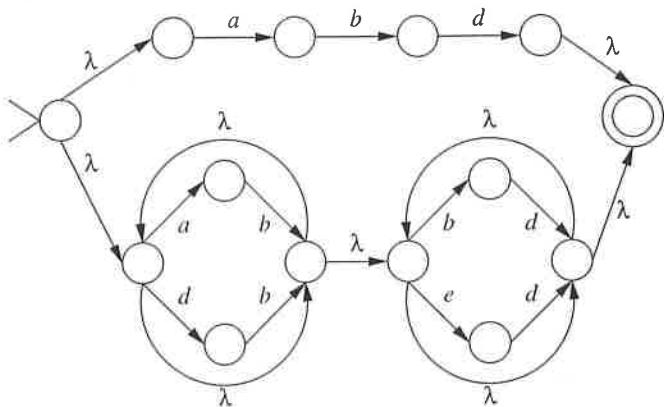


Figure 42:

Problem 427 Let L be the language defined by the regular expression:

$$(ac)^* (bc)^* \cup (b \cup cc)^* \cup (aa)^* bb$$

- (a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D, E\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \mid D \mid Ebb \\ A &\rightarrow AA \mid \lambda \mid ac \\ B &\rightarrow BB \mid \lambda \mid bc \\ D &\rightarrow DD \mid \lambda \mid b \mid cc \\ E &\rightarrow EE \mid \lambda \mid aa \end{aligned}$$

- (b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 43.

Problem 428 Let L be the set of those strings over the alphabet $\{a, b, c\}$ such that all c 's come before any b 's, all b 's come before any a 's, the number of a 's is at least 1, the number of b 's is at least 2, and the number of c 's is at least 3.

- (a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, K\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow KBA \\ A &\rightarrow aA \mid a \\ B &\rightarrow bB \mid bb \\ K &\rightarrow cK \mid ccc \end{aligned}$$

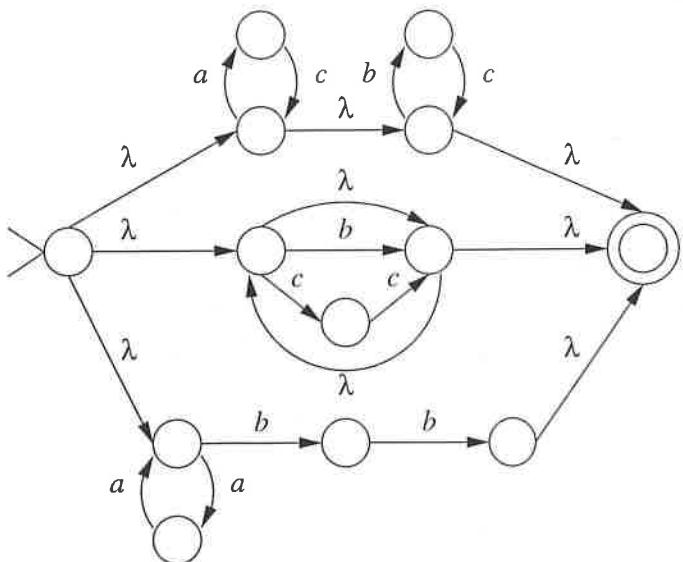


Figure 43:

- (b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 44.

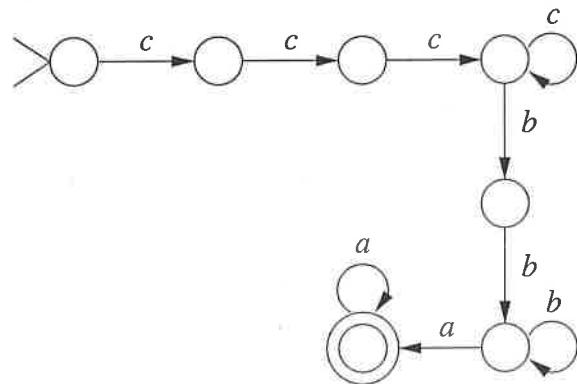


Figure 44:

- (c) State the cardinality of L . If L is finite, state the exact number; if L is infinite, specify whether it is countable or not countable.

Answer: L is infinite and countable.

- (d) State the cardinality of $\mathcal{P}(L)$ (the set of subsets of L). If $\mathcal{P}(L)$ is finite, state the exact number; if $\mathcal{P}(L)$ is infinite, specify whether it is countable or not countable.

Answer: $\mathcal{P}(L)$ is infinite and uncountable.

Problem 429 Let:

$$L = \{c^{n+2}a^m c^j b^p d^{\ell+1} \mid j = n \wedge m = 0\}$$

where $m, n, j, p, \ell \in N$.

- (a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The general template for strings of L is:

$$c^{2n+2}b^pd^{\ell+1} \mid n, p, \ell \geq 0\}$$

whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{b, c, d\}$, $V = \{S, B, K, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow KBD \\ K &\rightarrow ccK \mid cc \\ B &\rightarrow bB \mid \lambda \\ D &\rightarrow dD \mid d \end{aligned}$$

(b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 45.

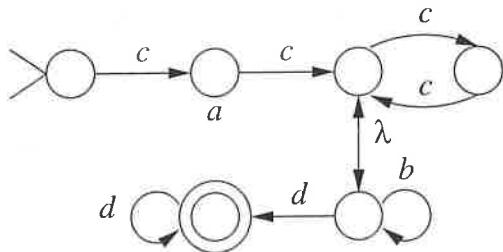


Figure 45:

(c) Is \bar{L} (the complement of L) decidable? Prove your answer.

Answer: Yes. L is context free (as is demonstrated in the answer to part (a)) and every context free language is decidable.

(d) Is \bar{L} (the complement of L) context free? Prove your answer.

Answer: Yes. L is regular (as is demonstrated in the answer to part (b)) and every regular language has a regular complement, which in turn is context free because every regular language is context free.

Problem 430 Let L be the set of all strings over the alphabet $\{a, b, c\}$ that begin with a and end with c .

Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 46.

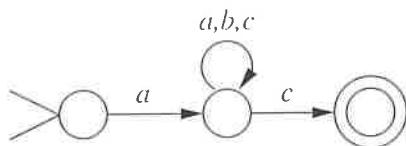


Figure 46:

Problem 431 Let L be the set of all strings over the alphabet $\{a, b, c\}$ that always end with b , always contain at least one b , and exactly two a 's appear before the first b (though not necessarily immediately before it.)

(a) Write a regular expression that represents the language L . If such a regular expression does not exist, state it and explain why.

Answer:

$$c^* a c^* a c^* b (\lambda \cup (a \cup b \cup c)^* b)$$

(b) Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D, K\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow KaKaKb \\ K &\rightarrow cK \mid \lambda \\ B &\rightarrow \lambda \mid Db \\ D &\rightarrow \lambda \mid DD \mid a \mid b \mid c \end{aligned}$$

(c) Draw a state-transition graph of a finite automaton that accepts the language L . If such an automaton does not exist, state it and explain why.

Answer: See Figure 47.

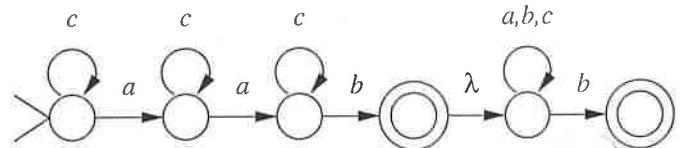


Figure 47:

Problem 432 Let L be the set of strings over the alphabet $\{a, b\}$ that contain exactly one b .

Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 48.

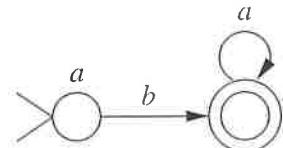


Figure 48:

Problem 433 Let L be the set of strings over the alphabet $\{a, b, c\}$ whose length is divisible by 4.

Draw a state-transition graph of a finite automaton that accepts \bar{L} (the complement of L). If such an automaton does not exist, prove it.

Answer: See Figure 49.

Problem 434 Let L be the set of all strings over the alphabet $\{a, b, c\}$ whose length is not divisible by 3.

Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 50.

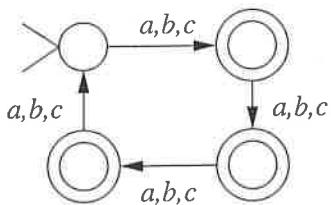


Figure 49:

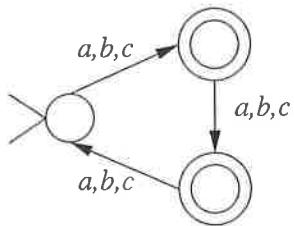


Figure 50:

Problem 435 Let L be the set of all strings over alphabet $\{a, b, c\}$ that have exactly two a 's.

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(b \cup c)^* a (b \cup c)^* a (b \cup c)^*$$

(b) Construct a state transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 51.

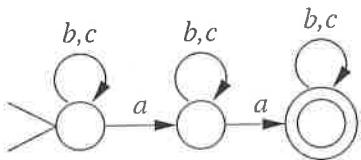


Figure 51:

Problem 436 Let L be the set of all strings over the alphabet $\{a, b, c\}$ that contain at most two b 's.

Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 52.

Problem 437 Let L be the set of all strings over the alphabet $\{a, b, c, d\}$ that contain exactly two d 's.

(a) Write a regular expression that represents the language L . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)^* d (a \cup b \cup c)^* d (a \cup b \cup c)^*$$

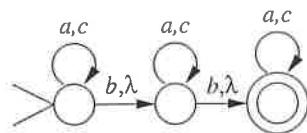


Figure 52:

(b) Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AdAdA \\ A &\rightarrow \lambda \mid AA \mid a \mid b \mid c \end{aligned}$$

(c) Draw a state-transition graph of a finite automaton that accepts the language L . If such an automaton does not exist, prove it.

Answer: See Figure 53.

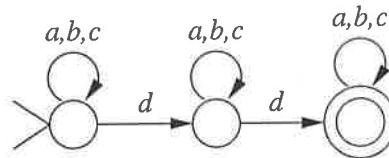


Figure 53:

Problem 438 Let L be the set of strings over alphabet $\{a, b, c\}$ whose total number of a 's and b 's is at least 2.

(a) Write a regular expression that defines L . If such a regular expression does not exist, explain why.

Answer:

$$(a \cup b \cup c)^* (a \cup b) (a \cup b \cup c)^* (a \cup b) (a \cup b \cup c)^*$$

(b) Construct a state transition graph of a finite automaton that accepts L . If such an automaton does not exist, explain why.

Answer: See Figure 54.

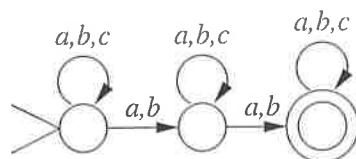


Figure 54:

(c) Write a complete formal definition of a context-free grammar G that generates L . If such a grammar does not exist, explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, B, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow BABAB \\ B &\rightarrow BB \mid \lambda \mid a \mid b \mid c \\ A &\rightarrow a \mid b \end{aligned}$$

Problem 439 Let L be the set of strings over the alphabet $\{a, b, c\}$ whose total number of a 's and c 's is equal to 2.

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$b^* (a \cup c) b^* (a \cup c) b^*$$

(b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 55.

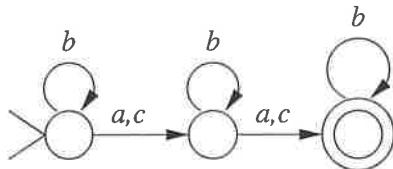


Figure 55:

(c) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, B, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow BZBZB \\ B &\rightarrow bB \mid \lambda \\ Z &\rightarrow a \mid c \end{aligned}$$

Problem 440 Let L_1 be the set of strings over the alphabet $\{a, b, c\}$ that begin with a and end with b .

Let L_2 be the set of strings over the alphabet $\{a, b, c\}$ that begin with b and end with c .

(a) Write a regular expression that defines L_1 . If such a regular expression does not exist, prove it.

Answer:

$$a (a \cup b \cup c)^* b$$

(b) Construct a state-transition graph of a finite automaton that accepts $L_1 \cup L_2$. If such an automaton does not exist, prove it.

Answer: See Figure 56.

(c) Write a complete formal definition of a context-free grammar that generates $L_1 L_2$. If such a grammar does not exist, prove it.

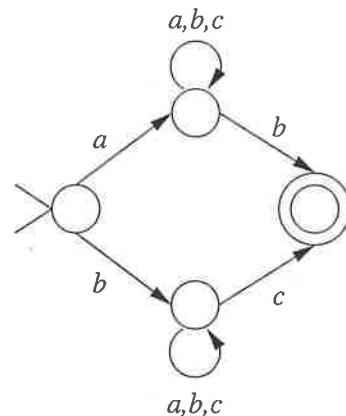


Figure 56:

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, T\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aTb \\ B &\rightarrow bTc \\ T &\rightarrow \lambda \mid TT \mid a \mid b \mid c \end{aligned}$$

Problem 441 Let L be the set of strings over the alphabet $\{a, b\}$ whose length is greater than 1 and the first letter is equal to the last.

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$a (a \cup b)^* a \cup b (a \cup b)^* b$$

(b) Construct a state-transition graph of a finite automaton that accepts LL . If such an automaton does not exist, prove it.

Answer: See Figure 57.

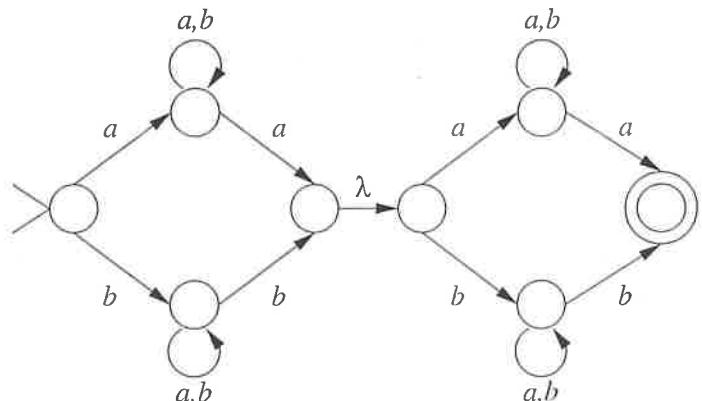


Figure 57:

(c) Write a complete formal definition of a context-free grammar that generates L^* . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, T\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow \lambda \mid SS \mid aTa \mid bTb \\ T &\rightarrow \lambda \mid TT \mid a \mid b \end{aligned}$$

Problem 442 Let L be the set of strings over alphabet $\{a, b, c\}$ whose third symbol is b .

1. Write a regular expression that defines L . If such regular expression does not exist, explain why.

Answer:

$$(a \cup b \cup c)(a \cup b \cup c)b(a \cup b \cup c)^*$$

2. Construct a state transition graph of a finite automaton (possibly non-deterministic) that accepts L . If such automaton does not exist, explain why.

Answer: See Figure 58.

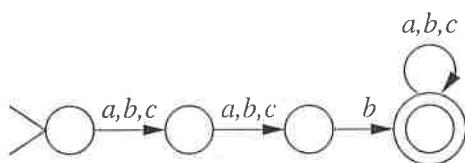


Figure 58:

3. Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ZZbA \\ Z &\rightarrow a \mid b \mid c \\ A &\rightarrow ZA \mid \lambda \end{aligned}$$

Problem 443 Let L be the language defined over alphabet $\{a, b, c, d, g\}$ by the regular expression:

$$(aa \cup b)^*(ccc \cup d)^* \cup gga$$

1. Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, g\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \mid gga \\ A &\rightarrow AA \mid \lambda \mid aa \mid b \\ B &\rightarrow BB \mid \lambda \mid ccc \mid d \end{aligned}$$

2. Construct a state transition graph of a finite automaton (possibly non-deterministic) that accepts L . If such automaton does not exist, explain why.

Answer: See Figure 59.

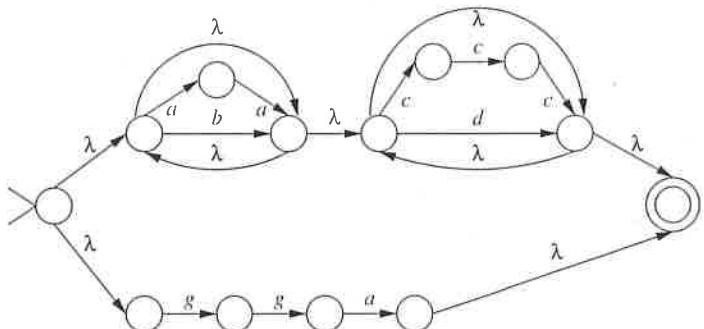


Figure 59:

3. Is the automaton which you constructed in the previous part of this problem deterministic? Explain your answer.

Answer: No—it has λ -moves, for example.

Problem 444 Let L be the set of all strings over alphabet $\{a, b\}$ in which all a 's come before all b 's, and the number of a 's is odd but the number of b 's is even.

- (a) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 60.

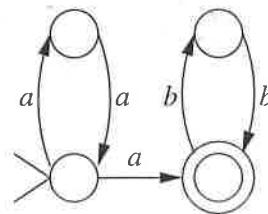


Figure 60:

- (b) Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aaA \mid a \\ B &\rightarrow bbB \mid \lambda \end{aligned}$$

Problem 445 Let L be the set of all nonempty strings over alphabet $\{a, b, d\}$ whose first symbol is equal to the third symbol.

- (a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$\begin{gathered} a(a \cup b \cup d)a(a \cup b \cup d)^* \\ \quad \cup \\ b(a \cup b \cup d)b(a \cup b \cup d)^* \\ \quad \cup \\ d(a \cup b \cup d)d(a \cup b \cup d)^* \end{gathered}$$

- (b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 61.

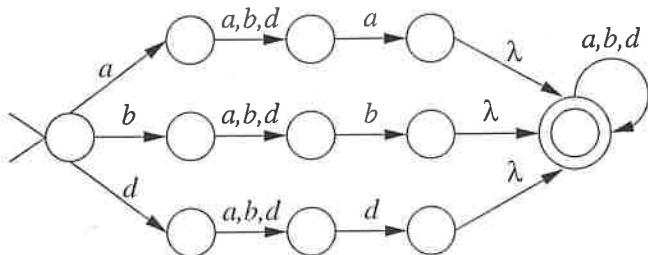


Figure 61:

Problem 446 Let L be the set of strings over alphabet $\{a, b, c\}$ that do not contain the substring aa .

- (a) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 62.

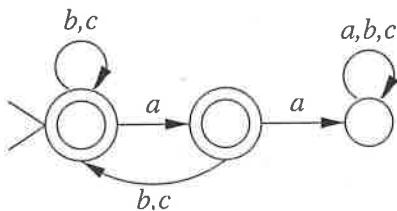


Figure 62:

- (b) Is the complement \overline{L} of the language L countable? Explain your answer briefly.

Answer: Yes—every language is countable, as a subset of the set Σ^* , which is countable because it is the set of all finite-length sequences over a countable set.

Problem 447 (a) Let L be the set of strings over $\{a, b, c\}$ that do not begin with a . Draw a state-transition graph of a deterministic finite automaton that accepts L . If such automaton does not exist, explain why.

Answer: See Figure 63.

- (b) Let L be the set of strings over $\{a, b, c\}$ that contain exactly one a . Draw a state-transition graph of a deterministic finite automaton that accepts L . If such automaton does not exist, explain why.

Answer: See Figure 64.

Problem 448 Let L_1 be the set of strings over alphabet $\{a, b, c\}$ that begin with a . Let L_2 be the set of strings over alphabet $\{a, b, c\}$ whose length is divisible by 3.

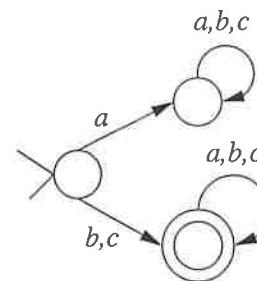


Figure 63:

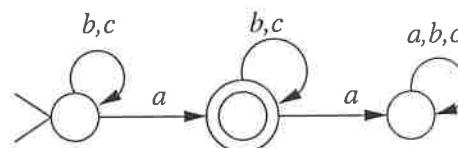


Figure 64:

- (a) Draw a state-transition graph of a finite automaton that accepts $L_1 \cup L_2$. If such automaton does not exist, prove it.

Answer: See Figure 65.

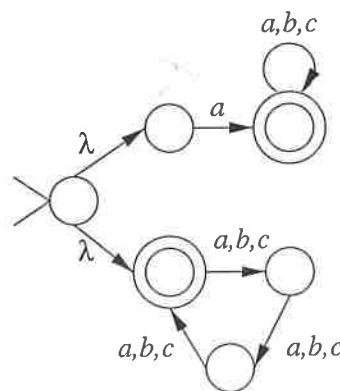


Figure 65:

- (b) Is it possible to write a computer program (algorithm) that operates as follows:

INPUT: An arbitrary string δ consisting of letters $\{a, b, c\}$.

OUTPUT: yes if $\delta \in L_1 \cup L_2$ and no if $\delta \notin L_1 \cup L_2$.

Explain your answer briefly.

Answer: Yes—such an algorithm simulates the finite automaton constructed in the answer to part (a).

Problem 449 (a) Let L_1 be the set of all strings over alphabet $\{a, b\}$ that begin with a . Write 5 distinct strings that belong to L_1 . If such strings do not exist, explain why.

Answer:

$a, aa, ab, abba, abb$

- (b) Draw a state-transition graph of a finite-state automaton M_1 that accepts language L_1 . If such automaton does not exist, explain why.

Answer: See Figure 66.

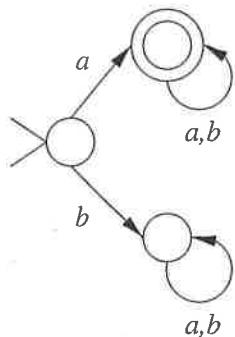


Figure 66:

- (c) Let L_2 be the set of all strings over alphabet $\{a, b\}$ that contain a . Write 5 distinct strings that belong to L_2 . If such strings do not exist, explain why.

Answer:

a, ba, aaa, bba, ab

- (d) Draw a state-transition graph of a finite-state automaton M_2 that accepts language L_2 . If such automaton does not exist, explain why.

Answer: See Figure 67.

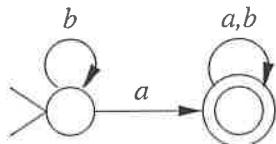


Figure 67:

- Problem 450** Let L be the set of all strings over the alphabet $\{a, b, c\}$ that end with ac .

Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 68.

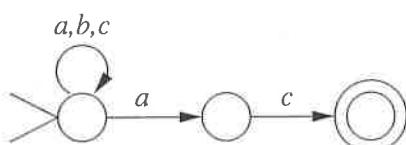


Figure 68:

- Problem 451** Let L be the set of all strings over $\{a, b, c\}$ that end with the substring $abac$. Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 69.

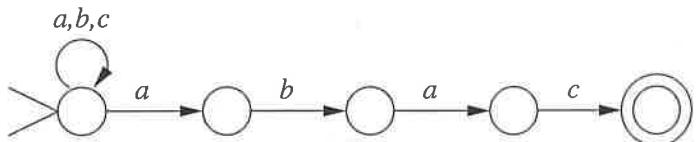


Figure 69:

- Problem 452** Let L be the language over alphabet $\{a, b, c\}$ consisting of all strings that do not contain ab as a substring.

- (a) Construct a finite automaton M that accepts L . If such an automaton M does not exist, explain why.

Answer: See figure Figure 70.

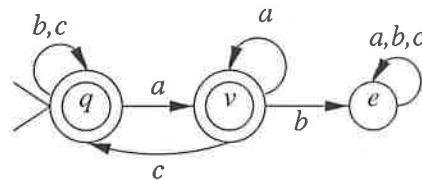


Figure 70:

- (b) If you constructed an automaton M in your answer to part (a), is M deterministic? Justify briefly your answer.

Answer: Yes. In its transition graph, exactly one arc labeled by each alphabet symbol is incident out of every node, while there are no other arcs. This corresponds to a total state-transition function that maps state-symbol pairs to states.

- Problem 453** Let L be the set of strings over the alphabet $\{a, b, c\}$ where every a is immediately followed by c .

- (a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$(ac \cup b \cup c)^*$

- (b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 71.

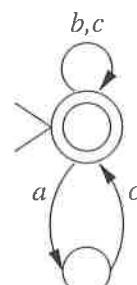


Figure 71:

(c) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow \lambda \mid SS \mid ac \mid b \mid c$$

Problem 454 Construct a state-transition graph of a finite automaton M that accepts the set of all strings over alphabet $\{a, b, c\}$ in which every a is followed immediately by $bbcc$. If such an automaton does not exist, prove it.

Answer: See Figure 72.

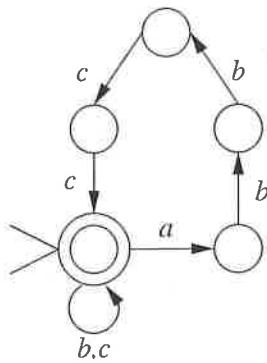


Figure 72:

Problem 455 Let L be the set of all strings over alphabet $\{a, b, c, d\}$ in which every a is immediately followed by bc .

(a) Write a complete formal definition or a state-transition graph of a finite automaton M that accepts L .

Answer: See figure Figure 73.

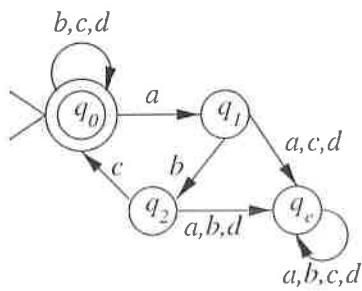


Figure 73:

(b) Is the finite automaton M which you constructed in your answer to part (a) deterministic? Justify briefly your answer.

Answer: Yes. In its transition graph, exactly one arc labeled by each alphabet symbol is incident out of every node, while there are no other arcs. This corresponds to a total state-transition function that maps state-symbol pairs to states.

Problem 456 Let L be the set of all strings over $\{a, b, c\}$ with an even number of a 's or an odd number of b 's. Construct a state-transition graph of a finite automaton M that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 74.

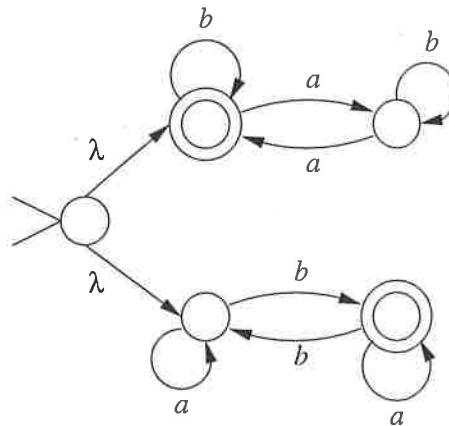


Figure 74:

Problem 457 Let L_1 be the set of strings over alphabet $\{a, b, c\}$ in which the total number of b 's and c 's is one. Let L_2 be the set of strings over alphabet $\{a, b, c\}$ that contain an even number of b 's.

(a) Draw a state-transition graph of a finite automaton M_1 that accepts $L_1 \cup L_2$. If such automaton does not exist, prove it.

Answer: See Figure 75.

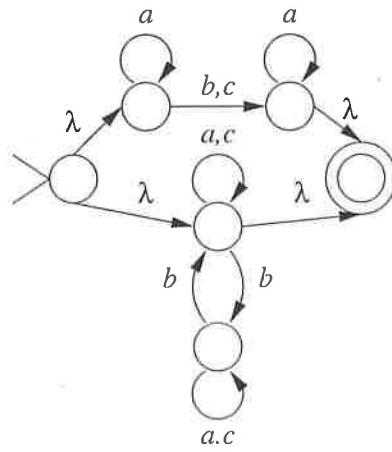


Figure 75:

(b) Draw a state-transition graph of a finite automaton M_2 that accepts $(L_1 L_2)^*$. If such automaton does not exist, prove it.

Answer: See Figure 76.

(c) Is $L_1 \cap L_2$ a context-free language? Explain your answer.

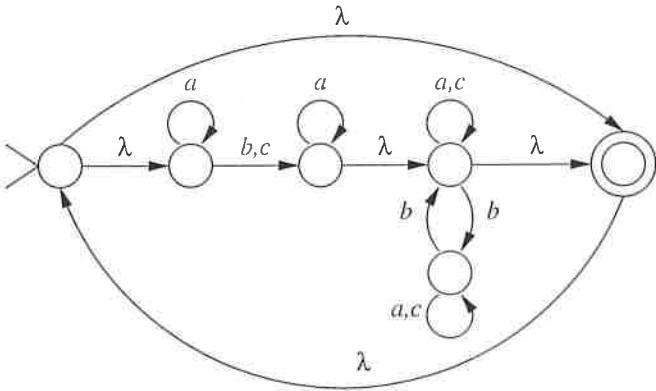


Figure 76:

Answer: Yes— $L_1 \cap L_2$ is regular because the intersection of any two regular languages is regular. Every regular language is context-free.

Problem 458 Let L be the set of all palindromes over alphabet $\{a, b, c, d\}$. (A palindrome is a string that is equal to its reversal.)

(a) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: Such automaton does not exist, since L is not regular.

To prove this, assume the opposite, that L is regular. Let k be the constant as in the Pumping Lemma for L . Let $m > k$; then $a^m b a^m \in L$. In any “pumping” decomposition such that $a^m b a^m = uvx$, we have: $|uv| \leq k < m$. Hence, the “pumping” substring v consists entirely of a 's, say $v = a^\ell$. Recall that $\ell > 0$, since the “pumping” substring cannot be empty. By the pumping, every word of the form $uv^i x$, $i \geq 0$, belongs to L . However, such a word is of the form:

$$w_1 = a^{m+(i-1)\ell} b a^m$$

while its reversal is:

$$w_1^R = a^m b a^{m+(i-1)\ell}$$

Since $m + (i-1)\ell > m$ whenever $i > 1$, word w_1 has more a 's before the single b than w_1^R , and it must be that:

$$w_1 \neq w_1^R$$

Since w_1 is not equal to its reversal w_1^R , we conclude that w_1 is not a palindrome. Hence, $w_1 \notin L$, which is a contradiction.

(b) Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S\}$, and P is:

$$S \rightarrow aSa \mid bSb \mid cSc \mid dSd \mid a \mid b \mid c \mid d \mid \lambda$$

Problem 459 Let L be a language over alphabet $\Sigma = \{a, b\}$, defined as follows:

$$L = \{x \mid (\exists w \in \Sigma^*)(x = ww)\}$$

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer: Such regular expression does not exist, since L is not regular.

To prove this, assume the opposite, that L is regular. Let k be the constant as in the Pumping Lemma for L . Let $m > k$; then $a^n b a^n \in L$. In any “pumping” decomposition such that $a^n b a^n = uvx$, we have: $|uv| \leq k < m$. Hence, the “pumping” substring v consists entirely of a 's, say $v = a^\ell$. Recall that $\ell > 0$, since the “pumping” substring cannot be empty. By the pumping, every word of the form $uv^i x$, $i \geq 0$, belongs to L . Consider the word $uvvx$, obtained by setting $i = 2$:

$$w' = a^{m+\ell} b a^m b$$

The length of w' is:

$$|w'| = m + \ell + 1 + m + 1 = 2m + 2 + \ell$$

Note that ℓ has to be even, since the entire length $|w'|$ has to be even— w' has to be a concatenation of two identical words. However, if we write w' as a concatenation of two words of equal length:

$$w' = w_1 w_2, \text{ where } |w_1| = |w_2|$$

we see that this length is:

$$|w_1| = |w_2| = \frac{1}{2} \cdot |w'| = \frac{1}{2} \cdot (2m + 2 + \ell) = m + 1 + \frac{\ell}{2}$$

However:

$$m + 1 + \frac{\ell}{2} \leq m + \ell, \text{ since } \ell \geq 2$$

and we see that the left-hand word w_1 consists entirely of a 's:

$$\begin{aligned} w_1 &= a^{m+1+(\ell/2)} \\ w_2 &= a^{(\ell/2)-1} b a^m b \end{aligned}$$

In contrast, the right-hand word w_2 contains two b 's. Hence:

$$w_1 \neq w_2$$

meaning that w' is not a concatenation of two identical words, and cannot belong to L .

(b) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: Such automaton does not exist, since L is not regular. This is proved in part (a).

Problem 460 Let L be the set of all strings over alphabet $\{a, b, c\}$ whose length is even and two middle symbols are equal.

- (a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, Z\}$, and P is:

$$\begin{aligned} S &\rightarrow ZSZ \mid aa \mid bb \mid cc \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

- (b) Draw a state-transition graph of a finite automaton M that accepts L . If such an automaton does not exist, prove it.

Answer: The required regular expression does not exist, since this language is not regular. To prove this, assume the opposite, that L is regular. Let η be the constant as in the Pumping Lemma for L . Let $m > \eta$; then the string:

$$(ac)^m bb (ac)^m$$

belongs to L , as its length is equal to $2(m + 1)$ and the two middle symbols are b .

In any “pumping” decomposition such that $(ac)^m bb (ac)^m = uvx$, we have: $|uv| \leq \eta < m$. Hence, the “pumping” substring v is contained entirely in the segment containing letters a and c . Let ℓ be the length of the “pumping” substring v . Recall that $\ell > 0$, since the “pumping” substring cannot be empty. Moreover, it has to be the case that $\ell \geq 2$, since ℓ has to be even, lest “pumping” once would produce a string of odd length, which is invalid. Hence we conclude that the “pumping” substring has one of the following two forms:

$$v = (ac)^k \text{ or } v = (ca)^k$$

for some $k = \ell/2 > 0$, whence the “pumped” part becomes:

$$v^i = (ac)^{ik} \text{ or } v^i = (ca)^{ik}$$

In both cases, letters a and c remain alternating, so that no two adjacent letters are equal in the entire segment to the left of the substring bb .

Let us “pump” once, to produce a word of the form:

$$(ac)^{m+k} bb (ac)^m \text{ where } k \geq 1$$

If $k = 1$, the two-letter substring in the middle of the word is cb ; if $k > 1$, the two-letter substring in the middle of the word is either ca (if k is odd) or ac (if k is even.) In all cases, the two middle symbols are different, whence the contradiction.

Problem 461 Let L be the set of all strings over the alphabet $\{a, b, c\}$ whose number of c 's is equal to 2.

- (a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b)^* c (a \cup b)^* c (a \cup b)^*$$

- (b) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 77.

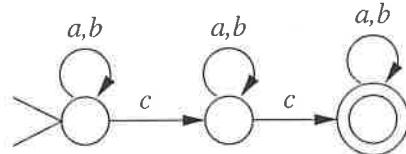


Figure 77:

- (c) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b, c\}$, $V = \{S, A\}$, and P is:

$$\begin{aligned} S &\rightarrow AcAca \\ A &\rightarrow AA \mid \lambda \mid a \mid b \end{aligned}$$

Problem 462 Consider the languages L_1 and L_2 over the alphabet $\{a, b, c\}$, defined as follows.

L_1 is the set of strings whose number of b 's is a (integer) multiple of 2.

L_2 is the set of strings whose number of b 's is a (integer) power of 2.

- (a) Draw a state-transition graph of a finite automaton that accepts L_1 . If such an automaton does not exist, prove it.

Answer: See Figure 78.

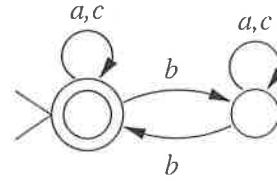


Figure 78:

- (b) Draw a state-transition graph of a finite automaton that accepts L_2 . If such an automaton does not exist, prove it.

Answer: This automaton does not exist, since L_2 is not regular. To prove that L_2 is not regular, assume the opposite—that L_2 is regular.

Let k be the constant as in the Pumping Lemma for L_2 . Consider a word $w \in L_2$, such that $w = b^{2^m}$, where $m > k$ and $m > 1$. Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < k < m$. Hence, $v = b^j$ for some j such that $0 < j < k$. By the pumping, for every $i \geq 0$, we have: $uv^i x \in L_2$. This means that all the words produced by the following template belong to L_2 .

$$uv^i x = uv^{i-1} x = uv^{(i-1)j} x = b^{2^m + (i-1)j}$$

Consider the word:

$$w_2 = b^{2^m+j}$$

obtained by setting $i = 2$ in this template. Let $\beta = 2^m + j$ be the number of b 's in the word w_2 . Recall that $j > 0$ (since j is the length of the non-empty pumping part.) Hence:

$$\beta = 2^m + j > 2^m$$

Next, recall that $j < k$ (since the length of the pumping part j is less than the characteristic constant k .) This implies that: $j < k < m < 2^m$, by the choice of m . (Recall that $2^m > m$ whenever $m > 1$.) Hence:

$$\beta = 2^m + j < 2^m + 2^m = 2 \cdot 2^m = 2^{m+1}$$

This means that β (the number of b 's in the word w_2) satisfies the following inequalities:

$$2^m < \beta < 2^{m+1}$$

We conclude that β cannot be an integer power of 2, since there are no integer powers of 2 between 2^m and 2^{m+1} . This means that:

$$w_2 = b^\beta \notin L_2$$

whence the contradiction.

Problem 463 Let L be the language over alphabet $\{a, b, c\}$ defined as follows:

$$L = \{c^p b^n a^p \mid n, p \geq 0\}$$

Write a complete formal definition or a state-transition graph of a finite automaton M that accepts L . If such automaton does not exist, prove it.

Answer: L is not a regular language—there is no finite automaton that accepts it. To prove this, assume the opposite, that L is regular. By the Pumping Lemma, there exists a constant ℓ such that every word $w \in L$ such that $|w| \geq \ell$ can be written as

$$w = xyz$$

where:

$$\begin{aligned} |y| &> 0 \\ |xy| &\leq \ell \end{aligned}$$

and every word of the form:

$$xy^i z, i \geq 0$$

also belongs to L . Consider a word $c^k b^m a^k \in L$, where k is chosen so that $k > \ell$. By the Pumping Lemma, there exist x, y, z such that:

$$c^k b^m a^k = xyz$$

where $|xy| \leq \ell < k$, which causes both x and y to consist solely of symbols c :

$$y = c^j, j > 0$$

and also:

$$xy^i z \in L, i > 0$$

meaning:

$$c^{k+(i-1)j} b^m a^k \in L$$

which is impossible by the definition of L whenever $k + (i - 1)j \neq k$, which occurs whenever $i > 1$.

Problem 464 Let L be the set of all strings over $\{a, b\}$ with twice as many a 's as b 's. Write a complete formal definition or a state-transition graph of a finite automaton M that accepts L . If such automaton does not exist, prove it.

Answer: Such finite automaton does not exist, since L is not a regular language. To prove this, assume the opposite. Let k be the constant as in the Pumping Lemma. Let $n > k$; then $a^{2n} b^n \in L$. In the “pumping” decomposition: $a^{2n} b^n = uvx$, we have: $|uv| \leq k < n < 2n$, hence the “pumping” substring v consists entirely of a 's, say $v = a^j$. Recall that $j > 0$, since the “pumping” substring cannot be empty. By the pumping, every word of the form $uv^i x$, $i \geq 0$, belongs to L . However, such a word has exactly n occurrences of b and $2n + (i - 1)j$ occurrences of a . Since $2n + (i - 1)j > 2n$ whenever $i > 1$, this word has too many a 's for its number of b 's, and this is a contradiction.

PL

Problem 465 Let L be the set of all strings over alphabet $\{a, b, c\}$ in which at least one of the letters appears at least twice.

(a) Write a complete formal definition of a context-free grammar G that generates L . If such a grammar does not exist, explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, X\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow X a X a X \mid X b X b X \mid X c X c X \\ X &\rightarrow XX \mid \lambda \mid a \mid b \mid c \end{aligned}$$

(b) Construct a state transition graph of a finite automaton that accepts L . If such an automaton does not exist, explain why.

Answer: See Figure 79.

Problem 466 Write a complete formal definition or a state-transition graph of a finite automaton that accepts the set of strings over $\{a, b\}$ that have both or neither aa and bb as substrings. If such automaton does not exist, prove it.

Answer: See Figure 80.

Problem 467 Let:

$$\Sigma = \{a, b, c\}$$

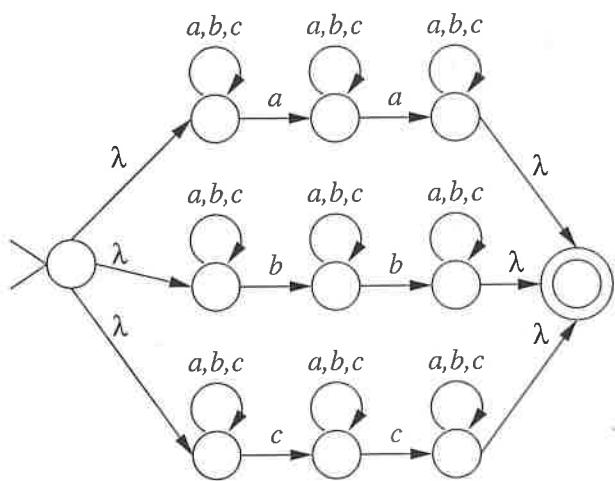


Figure 79:

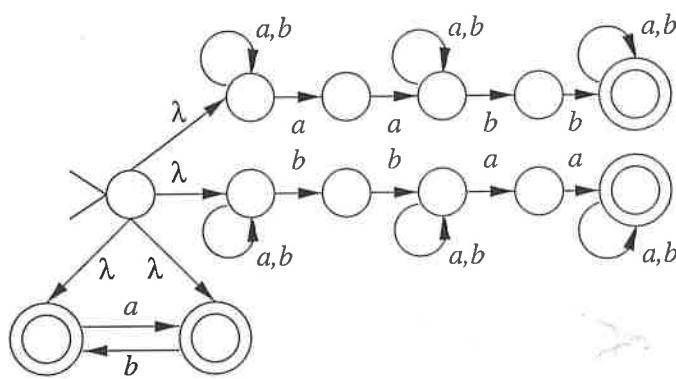


Figure 80:

and let L be the set of all strings over Σ ending with the substring $bacb$. In other words, precisely:

$$L = \{w \mid (\exists x \in \Sigma^*)(w = xbacb)\}$$

(a) Construct a state-transition graph of a finite automaton M that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 81.

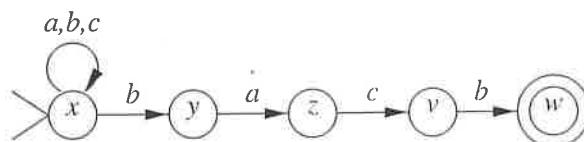


Figure 81:

(b) Construct a state-transition graph of a deterministic finite automaton M' that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 82.

Problem 468 Let:

$$\Sigma = \{a, b, c\}$$

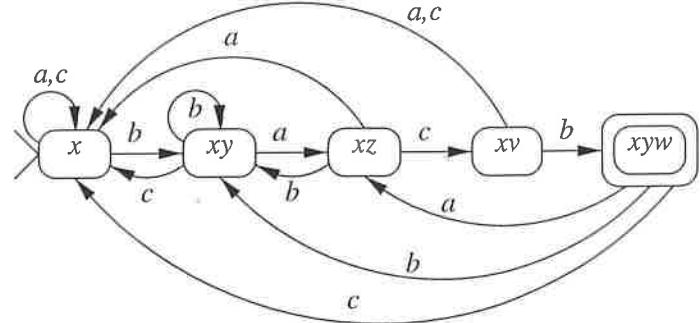


Figure 82:

and let L be the set of all strings over Σ defined by the following regular expression:

$$c(a \cup b)(a \cup b)^*b$$

(a) Construct a state-transition graph of a finite automaton M that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 83.

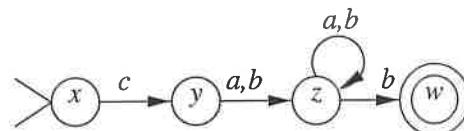


Figure 83:

(b) Construct a state-transition graph of a deterministic finite automaton M' that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 84.

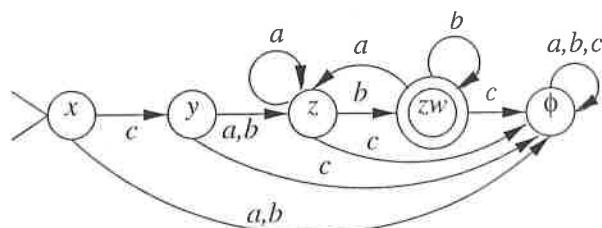


Figure 84:

Problem 469 (a) Let:

$$\Sigma = \{a, b, c\}$$

and let L_1 be the set of all strings over Σ in which every a is either immediately preceded or immediately followed by b .

Construct a state-transition graph of a finite automaton M_1 that accepts L_1 . If such automaton does not exist, prove it.

Answer: See Figure 85.

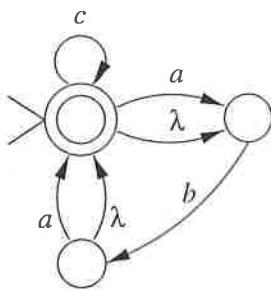


Figure 85:

(b) Let:

$$\Sigma = \{a, b, c\}$$

and let L_2 be the set of all strings over Σ with an even number of a 's or an odd number of b 's.

Write a regular expression that defines L_2 . If such expression does not exist, prove it.

Answer:

$$\begin{aligned} & ((b \cup c)^* a (b \cup c)^* a (b \cup c)^*)^* \\ & \quad \cup \\ & \quad (b \cup c)^* \\ & \quad \cup \\ & \quad (a \cup c)^* b ((a \cup c)^* b (a \cup c)^* b)^* (a \cup c)^* \end{aligned}$$

Problem 470 Let L be the set of all strings over the alphabet $\{a, b, c\}$ that have even length or contain an odd number of b 's.

(a) Write a regular expression that represents the language L . If such a regular expression does not exist, state it and explain why.

Answer:

$$\begin{aligned} & ((a \cup b \cup c) (a \cup b \cup c))^* \\ & \quad \cup \\ & \quad ((a \cup c)^* b (a \cup c)^* b (a \cup c)^*)^* ((a \cup c)^* b (a \cup c)^*)^* \end{aligned}$$

(b) Write a complete formal definition of a context-free grammar that generates the language L : If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D, E, F, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow E \mid D \\ E &\rightarrow \lambda \mid EE \mid ZZ \\ Z &\rightarrow a \mid b \mid c \\ D &\rightarrow FB \\ F &\rightarrow \lambda \mid FF \mid AbAbA \\ B &\rightarrow AbA \\ A &\rightarrow aA \mid cA \mid \lambda \end{aligned}$$

(c) Draw a state-transition graph of a finite automaton that accepts the language L . If such an automaton does not exist, state it and explain why.

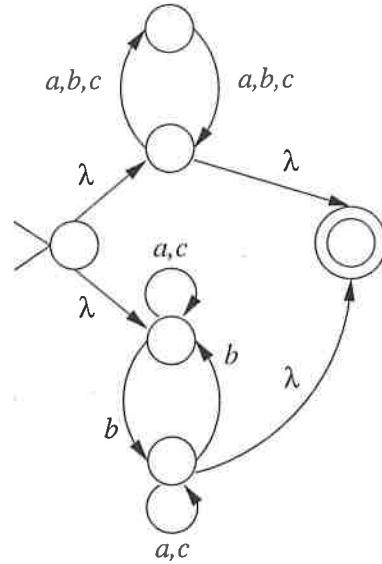
Answer: See Figure 86.

Figure 86:

Problem 471 Let L be the set of all strings over the alphabet $\{a, b, c, d\}$ that begin with d and contain at least one more d (other than this initial one.)

(a) Write a regular expression that represents the language L . If such a regular expression does not exist, state it and explain why.

Answer:

$$d (a \cup b \cup c)^* d (a \cup b \cup c \cup d)^*$$

(b) Write a complete formal definition of a context-free grammar that generates the language L . If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow dAdD \\ A &\rightarrow \lambda \mid AA \mid a \mid b \mid c \\ D &\rightarrow \lambda \mid DD \mid a \mid b \mid c \mid d \end{aligned}$$

(c) Draw a state-transition graph of a finite automaton that accepts the language L . If such an automaton does not exist, state it and explain why.

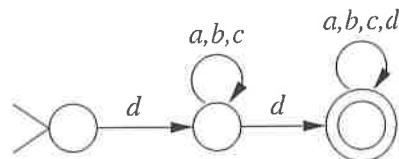
Answer: See Figure 87.

Figure 87:

Problem 472 Recall that for a language L :

$$\bar{L} = \Sigma^* \setminus L$$

(and do not confuse L with \overline{L} in the following text.)

- (a) Let L_1 be the language defined by the regular expression:

$$a^*b^*$$

Construct a state-transition graph of a finite automaton that accepts $\overline{L_1}$. If such automaton does not exist, explain why.

Answer: $\overline{L_1}$ is the set of strings where an a follows a b . See Figure 88.

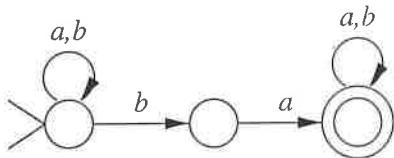


Figure 88:

- (b) Let L_2 be the language defined by the regular expression:

$$((a \cup b)(a \cup b)(a \cup b))^*$$

Construct a state-transition graph of a finite automaton that accepts $\overline{L_2}$. If such automaton does not exist, explain why.

Answer: $\overline{L_2}$ is the set of strings whose length is not divisible by 3. See Figure 89.

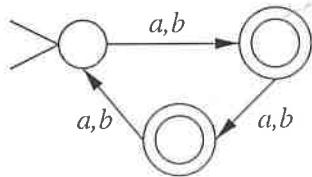


Figure 89:

- (c) Let L_3 be the language defined by the regular expression:

$$(a \cup b)^*bb(a \cup b)^*$$

Construct a state-transition graph of a finite automaton that accepts $\overline{L_3}$. If such automaton does not exist, explain why.

Answer: $\overline{L_3}$ is the set of strings that do not contain bb as a substring. See Figure 90.

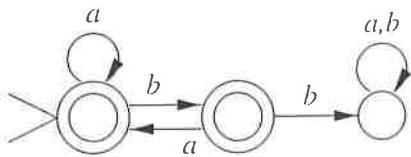


Figure 90:

- Problem 473** Let L be the set of all strings over $\{a, b\}$ whose length is not a prime number. Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: Such a finite automaton does not exist, since L is not a regular language. To prove this, observe that if L was regular, so would be its complement \overline{L} , which is the set of all strings over $\{a, b\}$ whose length is a prime number. To prove that \overline{L} is not regular, assume the opposite. Let k be the constant as in the Pumping Lemma. Let $n > k$ be a prime number; then $a^n \in \overline{L}$. In the “pumping” decomposition: $a^n = uvx$, let $j = |v|$; recall that $j > 0$, since the “pumping” substring cannot be empty. By the pumping, every word of the form $uv^i x$, $i \geq 0$, belongs to \overline{L} . However, $uv^i x = a^{n+(i-1)j}$. Select $i = n + 1$. Then:

$$uv^i x = a^{n+(i-1)j} = a^{n+nj} = a^{n(1+j)}$$

Since $n > k \geq 1$ and $j > 0$, the length of this word is a product of two integer numbers, each greater than 1. Hence, this length is not prime, and this word does not belong to \overline{L} , whence a contradiction.

Problem 474 Let:

$$L = \{b^i c^{2j} a^{3i} d^{5\ell} \mid i, j, \ell \geq 0\}$$

Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: There is no finite automaton that accepts this language. To prove this, assume the opposite, that L is regular. Let k be a constant as in the Pumping Lemma. Consider a word

$$b^n c^{2 \times 0} a^{3n} d^{5 \times 0} = b^n a^{3n} \in L$$

where n is chosen so that $n > k$. By the Lemma, for every decomposition:

$$w = xyz$$

such that:

$$|y| > 0$$

$$|xy| \leq k$$

part xy consists solely of symbols b , since:

$$|xy| \leq k < n = |b^n|$$

and thus:

$$y = b^m, \text{ for some } m > 0$$

If part y is pumping, then every word of the form:

$$xy^p z, p \geq 0$$

also should belong to L , meaning:

$$yb^{n+(p-1)m} c^{3n} \in L$$

which is impossible by the definition of L , since:

$$3(n + (p-1)m) \neq 3n$$

whenever $p > 1$.

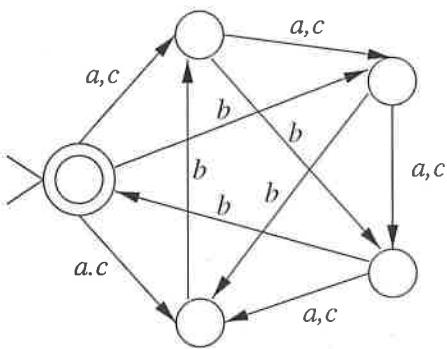


Figure 91:

Problem 475 Let L be the set of strings over $\{a, b, c\}$ in which the number of a 's plus twice the number of b 's plus the number of c 's is divisible by 5. Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 91.

Problem 476 (a) Let L_1 be the set of all strings over the alphabet $\{a, b, c\}$ where the number of a 's is divisible by 3.

Draw a state-transition graph of a finite automaton that accepts the language L_1 . If such an automaton does not exist, state it and explain why.

Answer: See Figure 92.

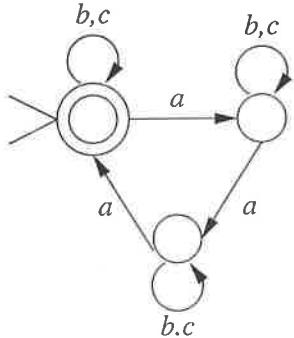


Figure 92:

(b) Let L_2 be the set of all strings over the alphabet $\{a, b, c\}$ where the number of a 's minus the number of b 's is divisible by 3.

Draw a state-transition graph of a finite automaton that accepts the language L_2 . If such an automaton does not exist, state it and explain why.

Answer: See Figure 93.

Problem 477 Let L be the set of strings over alphabet $\{a, b, c\}$ in which at least one of the following strings occurs at least once as a substring:

$abbc, cbca, acba$

Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

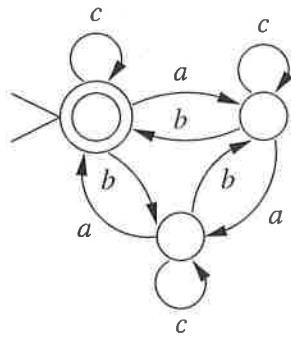


Figure 93:

Answer: See Figure 94.

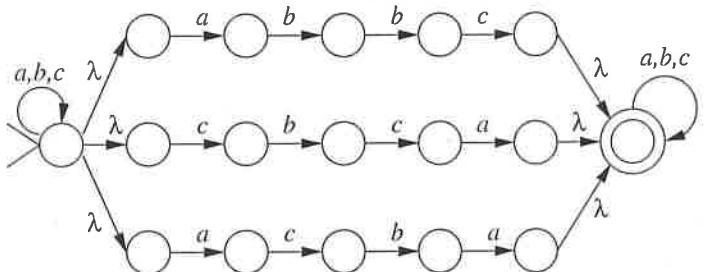


Figure 94:

Problem 478 Let L be the set of those strings over the alphabet $\{a, b, c\}$ that contain as a substring at least one of the two strings: ab, cb .

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)^*(ab \cup cb)(a \cup b \cup c)^*$$

(b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 95.

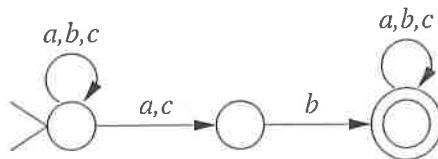


Figure 95:

(c) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A\}$, and P is:

$$\begin{aligned} S &\rightarrow AabA \mid AcbA \\ A &\rightarrow AA \mid \lambda \mid a \mid b \mid c \end{aligned}$$

Problem 479 Write a complete formal definition or a state-transition graph of a finite automaton that accepts the set of strings over $\{a, b\}$ such that the number of a 's in the string equals the number of b 's. If such automaton does not exist, prove it.

Answer: Let L be the set of strings over $\{a, b\}$ such that the number of a 's in the string equals the number of b 's. There is no finite automaton that accepts L . To prove this, assume the opposite, that L is regular. By the Pumping Lemma, there exists a constant ℓ such that every word $w \in L$ with $|w| \geq \ell$ can be written as

$$w = xyz$$

where:

$$\begin{aligned} |y| &> 0 \\ |xy| &\leq \ell \end{aligned}$$

and every word of the form:

$$xy^i z, i \geq 0$$

also belongs to L . Consider a word $b^n a^n \in L$, where n is chosen so that $n > \ell$. By the Pumping Lemma, there exist x, y, z such that:

$$b^n a^n = xyz$$

where y consists solely of symbols b :

$$y = b^k, k > 0$$

and also:

$$xy^i z \in L, i > 0$$

meaning:

$$b^{n+(i-1)k} a^n \in L$$

which is impossible by the definition of L whenever $i > 1$.

Problem 480 Let:

$$L_1 = \{a^{2i}b^j \mid i, j \geq 0\}$$

and

$$L_2 = \{a^i b^{2j} \mid i, j \geq 0\}$$

Draw a state-transition graph of a finite automaton that accepts $L_1 \cup L_2$. If such automaton does not exist, prove it.

Answer: See Figure 96.

Problem 481 Let L be the set of all strings over the alphabet $\{a, b\}$ whose length is divisible by 2 or 3.

(a) Write a regular expression that represents the language L . If such a regular expression does not exist, prove it.

Answer:

$$((a \cup b)(a \cup b))^* \cup ((a \cup b)(a \cup b)(a \cup b))^*$$

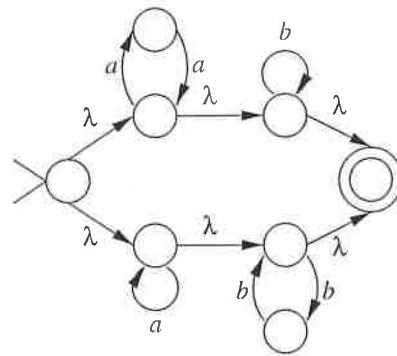


Figure 96:

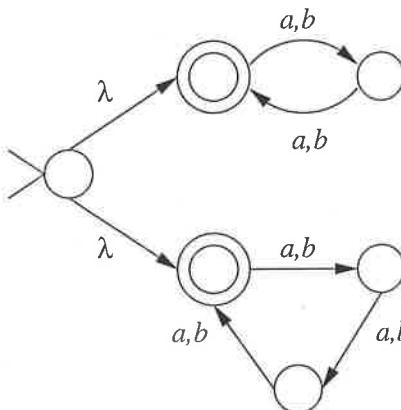


Figure 97:

(b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 97.

(c) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, A, B, Z\}$, and P is:

$$\begin{aligned} S &\rightarrow A \mid B \\ A &\rightarrow ZZ A \mid \lambda \\ B &\rightarrow ZZZ B \mid \lambda \\ Z &\rightarrow a \mid b \end{aligned}$$

Problem 482 Let L be the set of all strings over the alphabet $\{a, b\}$ such that all a 's come before any b 's, and the number of a 's is even but the number of b 's is odd.

(a) Write 5 distinct strings that belong to L . If such strings do not exist, state it and explain why.

Answer:

$$b, aab, bbb, aabbb, bbbbb$$

(b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 98.

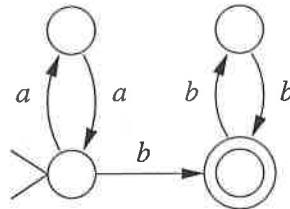


Figure 98:

- (c) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aaA \mid \lambda \\ B &\rightarrow bbB \mid b \end{aligned}$$

Problem 483 Let L be the set of strings over alphabet $\{a, b, c\}$ in which the number of b 's is divisible by 3 or 5.

- (a) Write a regular expression that defines L . If such a regular expression does not exist, explain why.

Answer:

$$\begin{aligned} ((a \cup c)^* b (a \cup c)^* b (a \cup c)^* b (a \cup c)^*)^* \\ \cup \\ ((a \cup c)^* b (a \cup c)^* b (a \cup c)^* b (a \cup c)^* b (a \cup c)^*)^* \\ \cup \\ ((a \cup c)^*)^* \end{aligned}$$

- (b) Construct a state transition graph of a finite automaton that accepts L . If such an automaton does not exist, explain why.

Answer: See Figure 99.

- (c) Let \mathcal{S} be a class of languages over alphabet $\{a, b, c\}$, defined as follows:

Language X is a member of \mathcal{S} if and only if X is represented by some regular expression, but there does not exist a finite automaton that accepts X .

What is the cardinality of the class \mathcal{S} ? (If possible, state the exact number. Otherwise, specify if it is countable or not.) Explain your answer.

Answer:

$$|\mathcal{S}| = 0, \text{ since } \mathcal{S} = \emptyset$$

Class \mathcal{S} is empty, since every language represented by some regular expression is also accepted by some finite automaton. In fact, this finite automaton is obtained by an algorithmic conversion of the original regular expression.

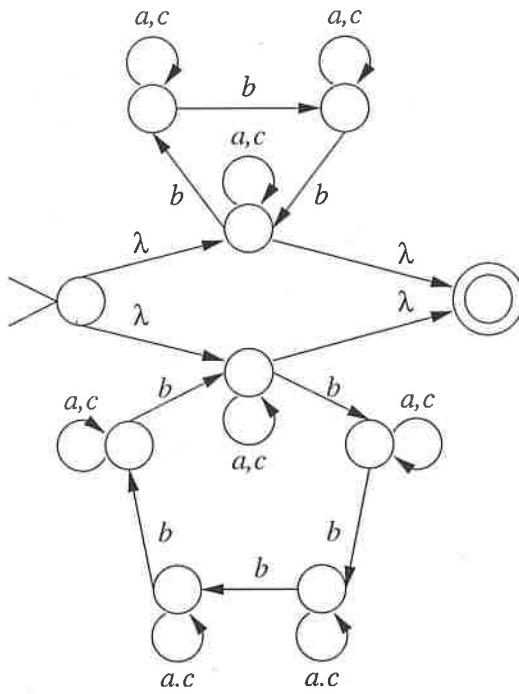


Figure 99:

Problem 484 Let L_1 be the set of strings over alphabet $\{a, b, c\}$ in which the number of b 's is not divisible by 3. Let L_2 be the set of strings over alphabet $\{a, b, c\}$ which begin with a and do not end with a .

- (a) Draw a state-transition graph of a finite automaton M that accepts the language:

$$L_1 \cup L_2 L_2$$

If such an automaton does not exist, prove it.

Answer: See Figure 100.

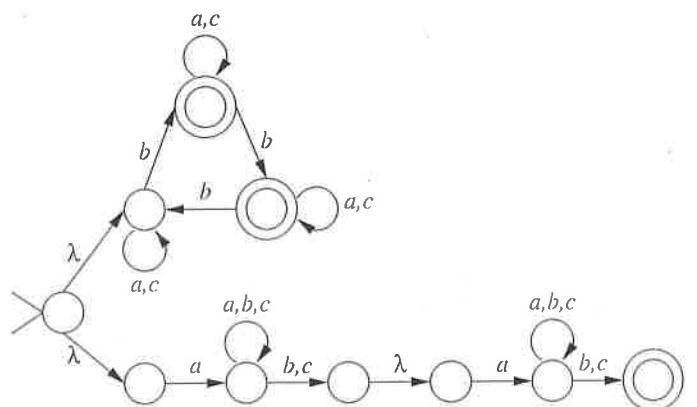


Figure 100:

- (b) Write a complete formal definition of a context-free grammar that generates the language:

$$L_1 \cup L_2 L_2$$

If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, S_1, S_2, X, Y, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow S_1 \mid S_2 S_2 \\ S_1 &\rightarrow aS_1 \mid cS_1 \mid bX \\ X &\rightarrow aX \mid cX \mid bY \mid \lambda \\ Y &\rightarrow aY \mid cY \mid bS_1 \mid \lambda \\ S_2 &\rightarrow aDb \mid aDc \\ D &\rightarrow DD \mid \lambda \mid a \mid b \mid c \end{aligned}$$

Problem 485 Let L_1 be the set of those strings over the alphabet $\{a, b, c\}$ that contain exactly one a .

Let L_2 be the set of those strings over the alphabet $\{a, b, c\}$ that do not end with c .

(a) Construct a state-transition graph of a finite automaton that accepts $L_1 \cup L_2$. If such an automaton does not exist, prove it.

Answer: See Figure 101.

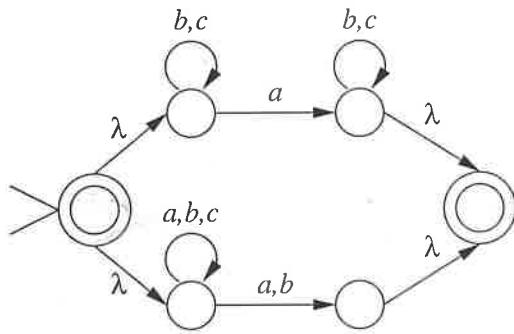


Figure 101:

(b) Write a complete formal definition of a context-free grammar that generates $L_1 \cup L_2$. If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow BaB \mid Da \mid Db \mid \lambda \\ B &\rightarrow bB \mid cB \mid \lambda \\ D &\rightarrow \lambda \mid DD \mid a \mid b \mid c \end{aligned}$$

Problem 486 Let L_1 be the set of strings over alphabet $\{a, b, c\}$ whose next-to-last symbol is a . Let L_2 be the set of strings over alphabet $\{a, b, c\}$ whose length is divisible by 3.

(a) Draw a state-transition graph of a finite automaton that accepts $L_1 L_2$. If such automaton does not exist, prove it.

Answer: See Figure 102.

(b) Write a complete formal definition of a context-free grammar that generates $L_1^* \cup L_2$. If such grammar does not exist, prove it.

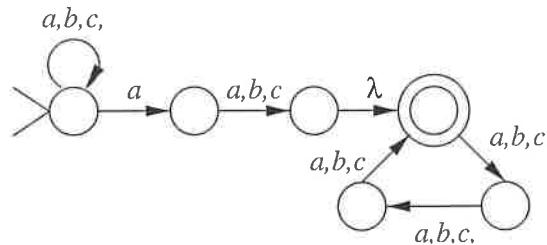


Figure 102:

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, DZ\}$, and P is:

$$\begin{aligned} S &\rightarrow A \mid B \\ A &\rightarrow AA \mid \lambda \mid DaZ \\ Z &\rightarrow a \mid b \mid c \\ D &\rightarrow DD \mid \lambda \mid Z \\ B &\rightarrow BB \mid \lambda \mid ZZZ \end{aligned}$$

Problem 487 Let L be the set of all strings over alphabet $\{a, b\}$ that have even length and end with letter b .

(a) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 103.

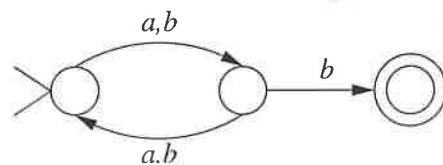


Figure 103:

(b) Let \mathcal{S} be a class of languages over alphabet $\{a, b\}$, defined as follows:

Language L is a member of \mathcal{S} if and only if L is not accepted by any finite automaton.

What is the cardinality of \mathcal{S} ? Explain your answer briefly.

Answer:

$$|\mathcal{S}| > \aleph_0$$

Class \mathcal{S} is infinite and uncountable. The class of all languages over alphabet $\{a, b\}$ is uncountable, while only countably many of them are accepted by some finite automaton.

Problem 488 Let L be the set of all strings over alphabet $\{a, b\}$ in which every substring of length 5 has at least one a .

(a) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

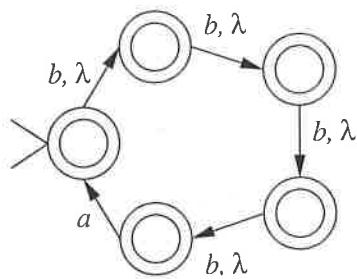


Figure 104:

Answer: See Figure 104.

- (b) Let \mathcal{T} be a set of strings over alphabet $\{a, b\}$, defined as follows:

String w is a member of \mathcal{T} if and only if $w \notin L$.

What is the cardinality of \mathcal{T} ? Explain your answer briefly.

Answer:

$$|\mathcal{T}| = |\overline{L}| = \aleph_0$$

Set $\mathcal{T} = \overline{L}$ is infinite and countable. To see that \mathcal{T} is infinite, observe that it contains (for example) $(bbbbbb)^*$. To see that \mathcal{T} is countable, recall that the entire set of strings $\{a, b\}^*$ is countable.

Problem 489 Let L be the set of strings over alphabet $\{a, b\}$ whose first symbol is different from the next-to-last (penultimate) symbol.

- (a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$(a(a \cup b)^*b \cup b(a \cup b)^*a)(a \cup b)$$

- (b) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 105.

- (c) Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$ is the set of terminals; $V = \{S, A, Z\}$ is the set of variables; S is the start symbol, and the production set P is:

$$\begin{aligned} S &\rightarrow aAbZ \mid bAaZ \\ A &\rightarrow ZA \mid \lambda \\ Z &\rightarrow a \mid b \end{aligned}$$

Problem 490 Let L be the set of strings over alphabet $\{a, b, c\}$ that do not start with c and do not end with a .

- (a) Draw a state-transition graph of a deterministic finite automaton that accepts L . If such automaton does not exist, prove it.

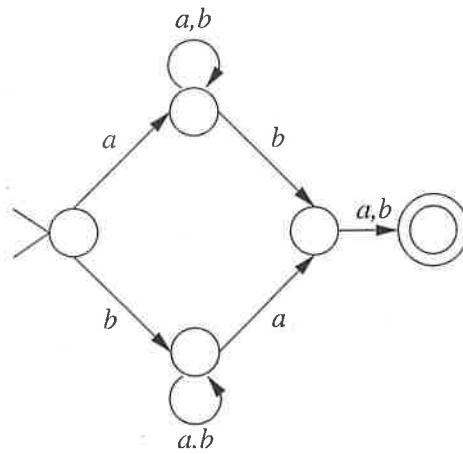


Figure 105:

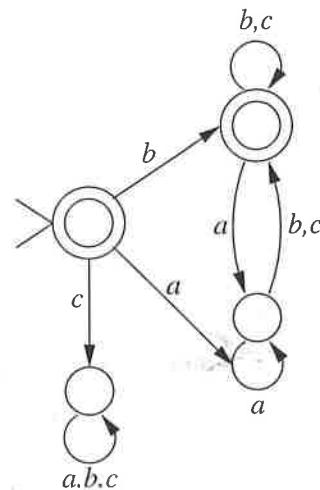


Figure 106:

Answer: See Figure 106.

- (b) Is the complement \overline{L} of the language L decidable? Explain your answer briefly.

Answer: Yes— L is regular, which is demonstrated by the finite automaton constructed in part (a); \overline{L} is regular because L is regular and the class of regular languages is closed under complement; every regular language is decidable as the decision procedure consists of a simulation of the finite automaton that accepts the language.

Problem 491 Let L be the set of strings over alphabet $\{a, b, c\}$ that do not start with c and do not end with a .

- (a) Draw a state-transition graph of a finite automaton that accepts \overline{L} (the complement of L). If such automaton does not exist, prove it.

Answer: See Figure 107.

- (b) Does there exist an algorithm that solves the following problem:

INPUT: An arbitrary regular expression e .

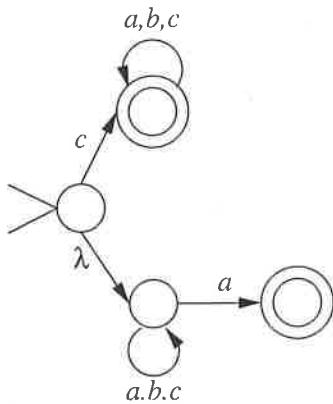


Figure 107:

OUTPUT: A finite automaton M_1 that accepts the complement of the language defined by e .

Explain your answer briefly.

Answer: Yes—such algorithm starts out by converting the regular expression into an equivalent finite automaton; next, this automaton is converted into an equivalent deterministic finite automaton; finally, the deterministic finite automaton is converted into one that accepts the complement.

Problem 492 Let L be the set of all strings over alphabet $\{a, b\}$ that do not end with ab .

(a) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 108.

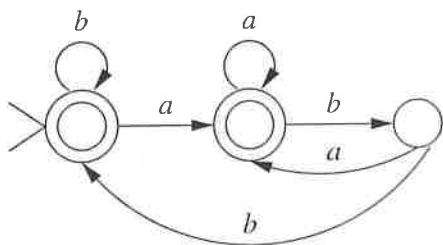


Figure 108:

(b) Dangerous Professor has told her students to write a program that operates as follows:

INPUT: An arbitrary string x over $\{a, b\}$.

OUTPUT: yes if $x \in L$ and no if $x \notin L$.

Explain the algorithm that should be employed by this program, or prove that it does not exist.

Answer: The program should simulate the automaton given in he answer to part (a), and decides exactly as this automaton decides.

Problem 493 Let L be the set of strings defined as follows:

$$L = \{w \mid w \in \{a, b, c\}^* \wedge w \neq aa \wedge w \neq bb\}$$

(a) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 109.

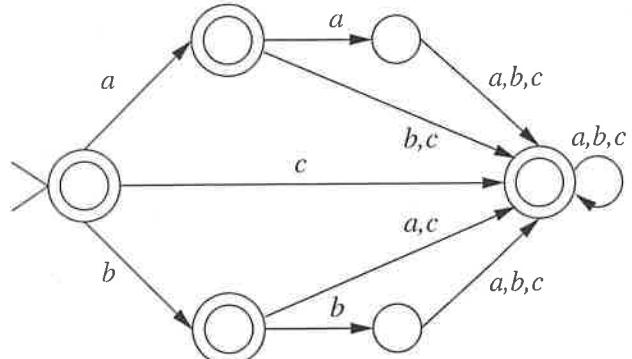


Figure 109:

(b) State the cardinalities of L and \bar{L} (the complement of L) and determine which one is greater. Explain your answer briefly. (For finite sets, state exact numbers. For infinite sets, specify if countable or not.)

Answer: By definition of L :

$$\bar{L} = \{aa, bb\}$$

Hence:

$$|L| = \aleph_0 \text{ and } |\bar{L}| = 2$$

L is infinite and countable, while \bar{L} has only two elements. Thus:

$$|L| > |\bar{L}|$$

Problem 494 Let:

$$L = \{a^{2i+1}b^{j+3}d^{3k}a^{\ell+2} \mid i = 2m, \text{ and } i, j, k, \ell, m \geq 0\}$$

(a) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: The general template for strings on L is:

$$a^{4m+1}b^{j+3}d^{3k}a^{\ell+2} \mid j, k, \ell, m \geq 0$$

whence the automaton represented on Figure 110.

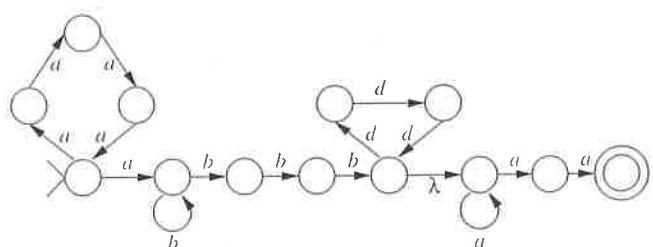


Figure 110:

(b) Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b, d\}$, $V = \{S, A, B, D, E\}$, and P is:

$$\begin{aligned} S &\rightarrow ABDE \\ A &\rightarrow aaaaA \mid a \\ B &\rightarrow bB \mid bbb \\ D &\rightarrow dddD \mid \lambda \\ E &\rightarrow aE \mid aa \end{aligned}$$

(c) Is the automaton (if any) which you constructed in part (a) deterministic? Explain your answer.

Answer: No—for example, it contains a λ -transition.

Problem 495 Let L be the set of all strings over alphabet $\{a, b\}$ that do not end with ab .

(a) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 111.

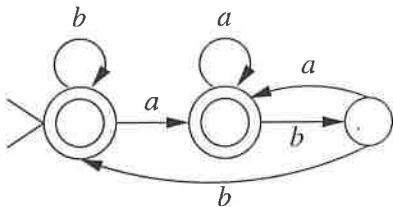


Figure 111:

(b) Dangerous Professor has told her students to write a program that operates as follows:

INPUT: An arbitrary string x over $\{a, b\}$.

OUTPUT: **yes** if $x \in L$ and **no** if $x \notin L$.

Explain the algorithm that should be employed by this program, or prove that it does not exist.

Answer: Simulate the finite automaton constructed in the answer to part (a). If it accepts, output **yes**; if it rejects output **no**.

Problem 496 Let:

$$L = \{a^{i+1}b^{j+3}d^{3k}a^{\ell+2} \mid i = j + k, \text{ and } i, j, k, \ell \geq 0\}$$

(a) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: The general template for strings in L is:

$$a^k a^{j+1} b^{j+3} d^{3k} a^{\ell+2} \mid j, k, \ell \geq 0$$

This language is not regular, and a finite automaton that accepts it does not exist.

To prove this, assume the opposite, that L is regular. Let η be the constant as in the Pumping Lemma for L . Let $m > \eta$; then $a^{m+1}bbb^3aa \in L$, as it is obtained from the general template by setting $j = \ell = 0$, and $k = m$. In any “pumping” decomposition such that

$a^{m+1}bbb^3aa = uvx$, we have: $|uv| \leq \eta < m$. Hence, the “pumping” substring v consists entirely of a 's, say $v = a^\ell$. Recall that $\ell > 0$, since the “pumping” substring cannot be empty. By the pumping, every word of the form $uv^i x$, $i \geq 0$, belongs to L . However, such a word is of the form:

$$w_1 = a^{m+1+(i-1)\ell}bbb^3aa$$

Since $m + 1 + (i - 1)\ell > m + 1$ whenever $i > 1$, word w_1 has more a 's than is appropriate for its number of d 's. Hence, $w_1 \notin L$, which is a contradiction.

(b) Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, d\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aAddd \mid D \\ D &\rightarrow aDb \mid abbb \\ B &\rightarrow aB \mid aa \end{aligned}$$

Problem 497 (a) Let L be the set of strings over the alphabet $\{a, b, c\}$ such that the number of a 's in the string is odd, or the number of b 's in the string is even, or the number of c 's in the string is not divisible by 3. Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 112.

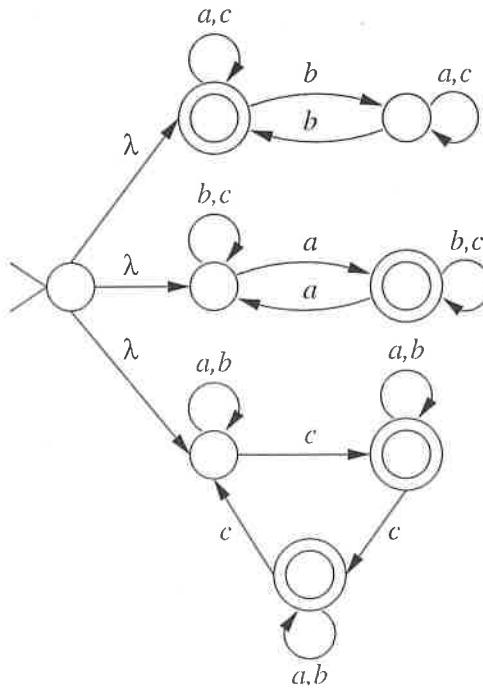


Figure 112:

(b) Let \mathcal{M} be the class of languages that are accepted by a non-deterministic finite automaton, but cannot be

represented by a regular expression. State the cardinality of the class \mathcal{M} , and compare it with the cardinality of the class of regular languages.

Answer: Class \mathcal{M} is empty, since every non-deterministic finite automaton has an equivalent regular expression:

$$|\mathcal{M}| = 0$$

Let \mathcal{R} be the class of regular languages. Class \mathcal{R} is infinite and countable:

$$|\mathcal{R}| = \aleph_0$$

Hence:

$$|\mathcal{M}| < |\mathcal{R}|$$

Problem 498 (a) Let L_1 be a language over the alphabet $\Sigma = \{a, b, c\}$, defined as follows:

$$L_1 = \{a^m b^n c^k b^\ell a^j \mid k=m \wedge n=\ell \wedge j, k, \ell, m, n \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L_1 . If such a grammar does not exist, prove it.

Answer: This grammar does not exist, since L_1 is not context-free. The general template for strings in L_1 is:

$$a^k b^n c^k b^n a^j \text{ where } k, n, j \geq 0\}$$

To prove that L_1 is not context-free, assume the opposite, and let η be the constant as in the Pumping Lemma for L_1 . Select a word:

$$w = a^k b^n c^k b^n \text{ where } k, n > \eta$$

obtained by setting $j = 0$ in the general template. In any “pumping” decomposition: $w = uvxyz$, the length of the “pumping window” vxy is not greater than η : $|vxy| \leq \eta < k$ and $|vxy| \leq \eta < n$. The “pumping” window vxy either falls within one of the four substrings containing a single letter: a^k , b^n , c^k , b^n , or it spans two such adjacent substrings—it is too short to extend through as many as three (or four) of them. If the pumping is only within one of the designated substrings, it will make this substring longer than another substring that should be of the same length. In the other case, if two adjacent letters are pumped, then there will be two other substrings with fewer letters than required (in addition to possible appearance of letters out of order.)

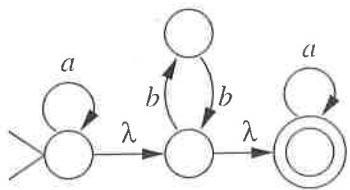


Figure 113:

(b) Let L_2 be a language over the alphabet $\Sigma = \{a, b, c\}$, defined as follows:

$$L_2 = \{a^m b^n c^k b^\ell a^j \mid k=0 \wedge n=\ell \wedge j, k, \ell, m, n \geq 0\}$$

Draw a state-transition graph of a finite automaton that accepts L_2 . If such an automaton does not exist, prove it.

Answer: The general template for strings in L_2 is:

$$a^m b^n b^n a^j \text{ where } m, n, j \geq 0$$

or, equivalently:

$$a^m (bb)^n a^j$$

which is equivalent to the regular expression:

$$a^* (bb)^* a^*$$

whence the automaton represented on Figure 113.

Problem 499 Let L be the set of strings with length greater than 1 over the alphabet $\{a, b\}$, such that the first (leftmost) symbol is different from the second symbol.

(a) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 114.

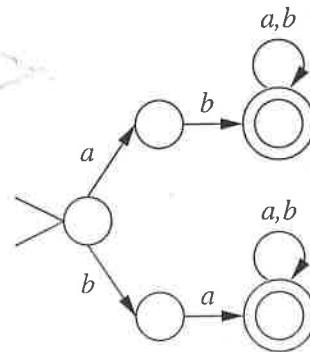


Figure 114:

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b\}$, $V = \{S, A\}$, and P is:

$$\begin{aligned} S &\rightarrow abA \mid baA \\ A &\rightarrow \lambda \mid AA \mid a \mid b \end{aligned}$$

Problem 500 Let L be the set of those strings over the alphabet $\{a, b\}$ that satisfy all of the following properties:

1. The length of the string is equal to $2n$, for some positive integer $n \geq 1$.
2. The first symbol is different from the second symbol (counting from the left.)

- (a) List five different strings that belong to L , or explain why this is impossible.

Answer:

$ab, ba, abaa, baaa, abbaba$

- (c) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(ab \cup ba)((a \cup b)(a \cup b))^*$$

- (b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b\}$, $V = \{S, T, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow abT \mid baT \\ T &\rightarrow ZZT \mid \lambda \\ Z &\rightarrow a \mid b \end{aligned}$$

Problem 501 Let L be the set of those strings over the alphabet $\{a, b\}$ that satisfy all of the following properties:

1. The length of the string is equal to $2n$, for some positive integer $n \geq 1$.
2. The first symbol is different from the $(2n)$ th symbol (counting from the left.)

- (a) List five different strings that belong to L , or explain why this is impossible.

Answer:

$ab, ba, abab, baaa, abbabb$

- (b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The general template for the strings of L is described as follows.

$$L = L_a \cup L_b$$

$$L_a = \{a(\chi\chi)^n b \mid n \geq 0\}$$

$$L_b = \{b(\chi\chi)^n a \mid n \geq 0\}$$

where χ is a place-holder for an arbitrary letter of the alphabet $\{a, b\}$. Hence, L is generated by the grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, T, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aTb \mid bTa \\ T &\rightarrow ZZT \mid \lambda \\ Z &\rightarrow a \mid b \end{aligned}$$

- (c) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 115.

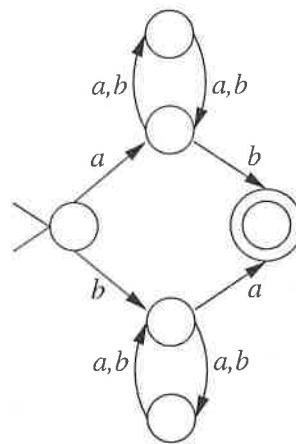


Figure 115:

Problem 502 Let L be the set of those strings over the alphabet $\{a, b\}$ that satisfy all of the following properties:

1. The length of the string is equal to $2n$, for some positive integer $n \geq 1$.
2. The first symbol is different from the $(n + 1)$ st symbol (counting from the left.)

- (a) List five different strings that belong to L , or explain why this is impossible.

Answer:

$ab, ba, aaba, bbbaab, abbabaaa$

- (b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The general template for the strings of L is described as follows.

$$L = L_a \cup L_b$$

$$L_a = \{a\chi^n b\chi^n \mid n \geq 0\}$$

$$L_b = \{b\chi^n a\chi^n \mid n \geq 0\}$$

where χ is a place-holder for an arbitrary letter of the alphabet $\{a, b\}$. Hence, L is generated by the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, A, B, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aB \mid bA \\ A &\rightarrow ZAZ \mid a \\ B &\rightarrow ZBZ \mid b \\ Z &\rightarrow a \mid b \end{aligned}$$

- (c) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: This automaton does not exist, since L is not regular.

Observe that L comprises exactly those nonempty strings that can be written in the form: w_1w_2 where w_1 and w_2 are strings of equal length, but they begin with different letters.

To prove that L is not regular, assume the opposite, that L is regular. Let k be the constant as in the Pumping Lemma for L . Consider a word $w = a^m b^m$, where $m > k$. Evidently, $w \in L$, since $|a^m| = |b^m| = m$, but $a \neq b$.

Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < k < m$. Hence, the pumping part v is entirely within the a -segment, meaning that $v = a^j$ for some j such that $0 < j \leq k$. After pumping once, we obtain a word: $w_p = a^{m+j}b^m$. If j is odd, then $|w_p|$ is also odd, and $w_p \notin L$. Otherwise, if j is even, then $j = 2\ell$, for some $\ell > 0$, and $w_p = a^\ell a^m a^\ell b^m$. Hence, w_p is a concatenation of two words of equal length: $a^\ell a^m$ and $a^\ell b^m$. However, these two words begin with the same letter, namely a , meaning that $w_p \notin L$.

Problem 503 Let L be the language over the alphabet $\Sigma = \{a, b, c, d\}$ that contains exactly those strings whose form is:

$$a^i b^j c^k a^\ell d^m b^n c^p d^q$$

where $i, j, k, \ell, m, n, p, q \geq 0$ are natural numbers such that: $i = m$, $j = k$, $\ell = 0$, $n = p$, $q = 0$.

If L is regular, then use part (a) of the answer space below to draw a state transition graph of a finite automaton that accepts L , and do not write anything in part (b). If L is not regular, then do not write anything in part (a) of the answer space, but complete the missing parts of the text given in part (b) so as to obtain a proof that L is not regular.

(a) Finite automaton that accepts L :

Answer:

(b) Proof that L is not regular (answer underlined):

Observe that all words of L satisfy the following characteristic property:

template $a^m b^k c^k d^m b^p c^p$

hence number of a 's is equal to number of d 's.

Assume the opposite, that L is regular. Let π be the constant as in the Pumping Lemma for L . Let $w_0 \in L$ be a string defined as follows:

$w_0 = \underline{a^n d^n}$ where $n > \pi$

w_0 belongs to L because

it is obtained from the template with $k = 0, p = 0$

w_0 must pump because $|w_0| = 2n > \pi$

In any “pumping” decomposition of w_0 , the pumping window satisfies the following property:

it is equal to a^j for some $j > 0$

because it is located within the first π symbols and $\pi < n$.

By pumping 1 times, we obtain a string $w_1 = a^{n+j} d^n$

which violates the stated characteristic property because number of a 's is equal to $n + j$ and $n + j \neq n$ (since $j > 0$) while number of d 's is equal to n .

hence $w_1 \notin L$.

Since L violates the Pumping Lemma, it is not regular.

PL fill in

Problem 504 Let L be the language over the alphabet $\Sigma = \{a, b, c, d\}$ that contains exactly those strings whose form is:

$$a^i c^j b^k a^\ell d^m c^n b^p d^q$$

where $i, j, k, \ell, m, n, p, q \geq 0$ are natural numbers such that: $m = n$, $j = k$, $i = 0$, $\ell = q$, $p = 0$.

If L is regular, then use part (a) of the answer space below to draw a state transition graph of a finite automaton that accepts L , and do not write anything in part (b). If L is not regular, then do not write anything in part (a) of the answer space, but complete the missing parts of the text given in part (b) so as to obtain a proof that L is not regular.

(a) Finite automaton that accepts L :

Answer:

(b) Proof that L is not regular (answer underlined):

Observe that all words of L satisfy the following characteristic property:

template $c^k b^k a^q d^m c^n d^q$, hence

number of b 's is equal to number of c 's to the left of b 's.

Assume the opposite, that L is regular. Let π be the constant as in the Pumping Lemma for L . Let $w_0 \in L$ be a string defined as follows:

$w_0 = \underline{c^k b^k}$ where $k > \pi$

w_0 belongs to L because

it is obtained from the template with $q = 0, n = 0$

w_0 must pump because $|w_0| = 2k > \pi$

In any “pumping” decomposition of w_0 , the pumping window satisfies the following property:

it is equal to c^j for some $j > 0$

because it is located within the first π symbols and $\pi < k$.

By pumping 1 times, we obtain a string $w_1 = c^{k+j} b^k$

which violates the stated characteristic property because

number of c 's is equal to $k + j$ and $k + j \neq k$ (since $j > 0$)

while number of b 's is equal to k .

hence $w_1 \notin L$.

Since L violates the Pumping Lemma, it is not regular.

Problem 505 Let L be the language over the alphabet $\Sigma = \{a, b, c, d\}$ that contains exactly those strings whose form is:

$$a^{3j+1} c^{2k+3} d^{2m+2} b^{\ell+4} a^{p+1}$$

where $j, k, \ell, m, p \geq 0$ are natural numbers.

If L is regular, then use part (a) of the answer space below to write a regular expression that defines L , and do not write anything in part (b).

If L is not regular, then do not write anything in part (a) of the answer space, but complete the missing parts of the text given in part (b) so as to obtain a proof that L is not regular.

(a) Regular expression for L :

Answer:

$$(aaa)^* a (cc)^* ccc (dd)^* dd b^* bbbb a^* a$$

(b) Proof that L is not regular (no answer allowed):

Observe that all words of L satisfy the following characteristic property:

Assume the opposite, that L is regular. Let α be the constant as in the Pumping Lemma for L . Let $w_0 \in L$ be a string defined as follows:

$$w_0 =$$

w_0 belongs to L because

w_0 must pump because

In any “pumping” decomposition of w_0 , the pumping window satisfies the following property:

because

By pumping times, we obtain a string:

which violates the stated characteristic property because and thus does not belong to L . Since L violates the Pumping Lemma, L is not regular.

Problem 506 Let L be the language over the alphabet $\Sigma = \{a, b, c, d\}$ that contains exactly those strings whose form is:

$$a^\ell c^{3m} d^k b^n a^\ell$$

where $k, \ell, m, n \geq 0$ are natural numbers.

If L is regular, then use part (a) of the answer space below to write a regular expression that generates L , and do not write anything in part (b).

If L is not regular, then do not write anything in part (a) of the answer space, but complete the missing parts of the text given in part (b) so as to obtain a proof that L is not regular.

(a) regular expression for L :

Answer:

(b) Proof that L is not regular:

Observe that all words of L satisfy the following characteristic property:

number of a 's on the left of c 's is equal to the number of a 's on the right of c 's

Assume the opposite, that L is regular.

Let β be the constant as in the Pumping Lemma for L .

Let $w_0 \in L$ be a string defined as follows:

$$a^\ell ccc a^\ell, \text{ where } \ell > \beta$$

w_0 belongs to L because

it is obtained from the template by setting

$$k = n = 0 \text{ and } m = 1 .$$

w_0 must pump because

$$|w_0| = 2\ell + 3 > \ell > \beta$$

In any “pumping” decomposition of w_0 , the pumping window satisfies the following property:

pumping window y is: $y = a^j$ for some $0 < j < \beta$

because

pumping window is shorter than the constant β , so $|y| < \beta < \ell$, hence y is entirely within the a -segment.

By pumping 1 times, we obtain a string

$$w_1 = a^{\ell+j} ccc a^\ell$$

which violates the stated characteristic property because w_1 has $\ell + j$ a 's on the left of c 's but only ℓ a 's on the right of c 's, and $\ell + j \neq \ell$ since $j > 0$ (as the length of the pumping window.)

and thus does not belong to L . Since L violates the Pumping Lemma, L is not regular.

Problem 507 Let L be the language over the alphabet $\Sigma = \{a, b, c, d\}$ that contains exactly those strings whose form is:

$$b^{3\ell} c^{2p} a^{4m+k+1} d^\ell c^{j+1}$$

where $j, k, \ell, m, p \geq 0$ are natural numbers.

If L is regular, then use part (a) of the answer space below to write a regular expression that defines L , and do not write anything in part (b).

If L is not regular, then do not write anything in part (a) of the answer space, but complete the missing parts of the text given in part (b) so as to obtain a proof that L is not regular.

(a) Regular expression for L :

Answer:

(b) Proof that L is not regular (answer underlined):

Observe that all words of L satisfy the following characteristic property:

number of b 's = 3 times number of d 's

Assume the opposite, that L is regular. Let σ be the constant as in the Pumping Lemma for L . Let $w_0 \in L$ be a string defined as follows:

$$w_0 = b^{3\ell} ad^\ell c, \ell > \sigma$$

w_0 belongs to L because

it is obtained from the template by setting

$$p = m = k = j = 0$$

w_0 must pump because

$$\|w_0\| = 4\ell + 2 > \ell > \sigma$$

In any “pumping” decomposition of w_0 , the pumping window satisfies the following property:

its form is b^n for some $n < \sigma < \ell$

because

it is located within the first σ symbols.

By pumping 1 times, we obtain a string:

$$w_1 = b^{3\ell+n}ad^\ell c$$

which violates the stated characteristic property because

$$3\ell + n \neq 3\ell \text{ since } n > 0.$$

and thus does not belong to L . Since L violates the Pumping Lemma, L is not regular.

Problem 508 Let L be the language over the alphabet $\Sigma = \{a, b, c\}$ that contains exactly those strings which satisfy all of the following properties:

1. the three symbols a, b, c occur in the string in the alphabetic order;
2. the length of the substring which consists of a 's is even;
3. the length of the substring which consists of b 's is odd;
4. the length of the substring which consists of c 's is equal to twice length of the substring which consists of b 's.

If L is regular, then use part (a) of the answer space below to write a regular expression that defines L , and do not write anything in part (b).

If L is not regular, then do not write anything in part (a) of the answer space, but complete the missing parts of the text given in part (b) so as to obtain a proof that L is not regular.

(a) Regular expression for L :

Answer:

(b) Proof that L is not regular:

Observe that all words of L satisfy the following characteristic property:

template:

$$a^{2n}b^{2k+1}c^{4k+2}$$

In particular, the 4th item of the definition: number of c 's is equal to twice the number of b 's.

Assume the opposite, that L is regular. Let π be the constant as in the Pumping Lemma for L . Let $w_0 \in L$ be a string defined as follows:

$$w_0 = b^{2k+1}c^{4k+2} \text{ where } k > \pi$$

w_0 belongs to L because

it is obtained from the template (which in turn is equivalent to the definition) by setting $n = 0$.

w_0 must pump because

$$|w_0| = 6k + 3 > k > \pi$$

In any “pumping” decomposition of w_0 , the pumping window satisfies the following property:

it is equal to

$$b^j \text{ for some } j > 0$$

because

the pumping window is contained within the leftmost π symbols, and the length of the b -segment is $2k + 1 > \pi$.

By pumping 1 times, we obtain a string:
 $w_1 = b^{2k+1+j}c^{4k+2}$

which violates the stated characteristic property because

the number of c 's remains $4k + 2$ instead of $4k + 2 + 2j = 2 \cdot (2k + 1 + j)$ as is required by the template,

and thus does not belong to L . Since L violates the Pumping Lemma, L is not regular.

Problem 509 Let L be the language over the alphabet $\Sigma = \{a, b, c\}$ that contains exactly those strings which satisfy all of the following properties:

1. the three symbols a, b, c occur in the string in the alphabetic order;
2. the length of the substring which consists of a 's is divisible by 3;
3. the length of the substring which consists of b 's is divisible by 4;
4. the length of the substring which consists of c 's is divisible by 2 or 5.

If L is regular, then use part (a) of the answer space below to draw a state transition graph of a finite automaton that generates L , and do not write anything in part (b).

If L is not regular, then do not write anything in part (a) of the answer space, but complete the missing parts of the text given in part (b) so as to obtain a proof that L is not regular.

(a) Finite automaton for L :

Answer: See Figure 116.

(b) Proof that L is not regular (no answer allowed):

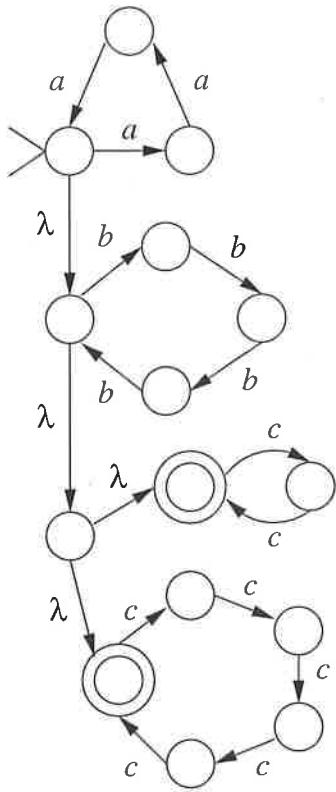


Figure 116:

Observe that all words of L satisfy the following characteristic property:

Assume the opposite, that L is regular. Let τ be the constant as in the Pumping Lemma for L . Let $w_0 \in L$ be a string defined as follows:

$w_0 =$

w_0 belongs to L because

w_0 must pump because

In any “pumping” decomposition of w_0 , the pumping window satisfies the following property:

because

By pumping times, we obtain a string:

which violates the stated characteristic property because

and thus does not belong to L . Since L violates the Pumping Lemma, L is not regular.

Problem 510 Let L be the language over the alphabet $\Sigma = \{a, b, c, g\}$ that contains exactly those strings whose form is:

$$g^m c^{m+3} b^n a^{n+4}$$

where $m, n \geq 0$ are natural numbers.

If L is regular, then use part (a) of the answer space below to write a regular expression that generates L , and do not write anything in part (b).

If L is not regular, then do not write anything in part (a) of the answer space, but complete the missing parts of the text given in part (b) so as to obtain a proof that L is not regular.

(a) regular expression for L :

Answer:

(b) Proof that L is not regular:

Observe that all words of L satisfy the following characteristic property:

number of c 's is by 3 greater than the number of g 's.

Assume the opposite, that L is regular.

Let τ be the constant as in the Pumping Lemma for L . Let $w_0 \in L$ be a string defined as follows:

$$w_0 = g^n c^{m+3} aaaa$$

where $m > \tau$.

w_0 belongs to L because

it is obtained from the template by setting $n = 0$.

w_0 must pump because

$$\|w_0\| = 2m + 7 > 2\tau + 7 > \tau$$

In any “pumping” decomposition of w_0 , the pumping window satisfies the following property:

it is equal to g^j for some $j > 0$

because

the pumping window is within the first τ symbols, and there are more than τ g 's.

By pumping 1 times, we obtain a string

$$w_1 = g^{m+j} c^{m+3} aaaa$$

which violates the stated characteristic property because $m + 3 \neq m + j + 3$, since $j > 0$

and thus does not belong to L . Since L violates the Pumping Lemma, L is not regular.

Problem 511 Let:

$$L = \{a^i c^j a^k b^\ell g^m b^n a^x f^y\}$$

where $i, j, k, \ell, m, n, x, y \geq 0$

are natural numbers such that:

$$i = k, n = 3i + 1, j = 0, \ell = m + 1, x = y$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The template for L is:

$$L = \{a^{2k} b^{m+1} g^m b^{3k+1} a^x f^x\}$$

whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, g, f\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aaAbbb \mid Db \\ D &\rightarrow bDg \mid b \\ B &\rightarrow aBf \mid \lambda \end{aligned}$$

(b) Draw a state transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: This automaton does not exist, since L is not regular. To prove this, we show that Pumping Lemma does not hold for L .

Observe that all words of L satisfy the following characteristic property: number of g 's is by one less than the number of b 's to the left of g 's.

Assume the opposite, that L is regular. Let ξ be the constant as in the Pumping Lemma for L . Let $m > \xi$; then the word $w = b^{m+1}g^mb$ belongs to L , as it is obtained from the template by setting $k = x = 0$.

In any “pumping” decomposition such that $b^{m+1}g^mb = uvx$, we have: $|uv| \leq \xi < m$. Hence, the “pumping” substring v consists entirely of b 's, say $v = b^\ell$. Recall that $\ell > 0$, since the “pumping” substring cannot be empty. Pump up once, obtaining the word:

$$w_1 = b^{m+1+\ell}g^mb$$

Since $m + 1 + \ell > m + 1$, word w_1 violates the stated characteristic property and thus $w_1 \notin L$, in violation of the Pumping Lemma.

Problem 512 Let:

$$L = \{a^n c^k a^\ell b^j a^m \mid k = j + 1, n = 0, m = 2p\}$$

where $j, k, \ell, m, n, p \geq 0$.

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: Observe that the general template is:

$$c^{j+1} a^\ell b^j a^{2p} \text{ where } j, \ell, p \geq 0$$

whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow cAb \mid cD \\ D &\rightarrow aD \mid \lambda \\ B &\rightarrow aaB \mid \lambda \end{aligned}$$

(b) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: This automaton does not exist, since L is not regular.

To prove this, assume the opposite—that L is regular. Observe that the template guarantees that every word of L satisfies the following property:

The number of c 's is by one greater than the number of b 's.

Let η be the constant as in the Pumping Lemma for L . Consider a word $w = c^{j+1}b^j$, where j has been selected so that $j > \eta$. Word w is obtained from the general template by setting $\ell = p = 0$.

Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < \eta < j$. Hence, the pumping part v is entirely within the c -segment, meaning that $v = c^q$ for some q such that $0 < q < \eta$.

After pumping once (up) we obtain a word: $w_1 = c^{j+1+q}b^j$, where the number of c 's and the number of b 's differ by more than one. This means that w_1 violates the stated property of L and $w_1 \notin L$, whence the contradiction.

Problem 513 Let:

$$L = \{a^n c^k a^\ell b^j a^m \mid j = p + 1, k = 0, \ell = 2n\}$$

where $j, k, \ell, m, n, p \geq 0$.

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The general template is:

$$a^n a^{2n} b^{p+1} a^m \text{ where } n, p, m \geq 0$$

which in turn is:

$$a^{3n} b^{p+1} a^m \mid n, p, m \geq 0$$

This language is generated by the grammar:

$G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow ABD \\ A &\rightarrow aaaA \mid \lambda \\ B &\rightarrow bB \mid b \\ D &\rightarrow aD \mid \lambda \end{aligned}$$

(b) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 117.

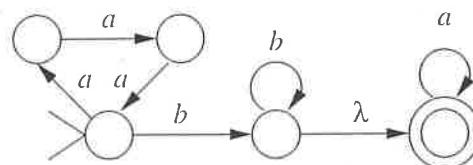


Figure 117:

Problem 514 Let:

$$L = \{a^n c^k a^\ell b^j a^m \mid n = j + 1, \ell = 0, k = 2m\}$$

where $j, k, \ell, m, n, p \geq 0$.

- (a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: This grammar does not exist, since L is not context-free.

To prove this, observe that the general template for all strings that belong to L is:

$$a^{j+1} c^{2m} b^j a^m \text{ where } j, m \geq 0$$

and every such string has the following two properties:

1. the number of a 's that are at the left of c 's is by one greater than the number of b 's;
2. the number of c 's is twice the number of a 's after the b 's.

Now, assume that L is context-free, and let η be the constant as in the Pumping Lemma for L . Consider a word:

$$w = a^{j+1} c^{2m} b^j a^m$$

where j, m have been selected so that $j, m > \eta$. In any “pumping” decomposition: $w = uvxyz$, the length of the “pumping window” vxy is not greater than η : $|vxy| \leq \eta < m$ and $|vxy| \leq \eta < j$. The “pumping” window vxy either falls within one of the four substrings containing a single letter: a^{j+1} , c^{2m} , b^j , a^m , or it spans two such adjacent substrings—it is too short to extend through as many as three (or four) of them. If the pumping is only within one of the designated substrings, it will make this substring longer than is appropriate with respect to the length of another substring. In the other case, if two adjacent letters are pumped, then there will be two other substrings with fewer letters than required (in addition to possible appearance of letters out of order.) This means that the word obtained by pumping does not honor the required template and cannot belong to L , which is a contradiction.

- (b) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: Such an automaton does not exist—if it existed, L would be regular and thereby context-free, since every regular language is context-free. However, the answer to the part (a) shows that L is not context-free,

Problem 515 Let:

$$L = \{a^n c^k a^\ell b^j a^m \mid k=0, \ell=3n, j=2p, j, k, \ell, m, n, p \geq 0\}$$

- (a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The template is:

$$\{a^n a^{3n} b^{2p} a^m \mid m, n, p \geq 0\}$$

or, equivalently,

$$\{(aaaa)^n (bb)^p a^m \mid m, n, p \geq 0\}$$

which yields the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ABD \\ A &\rightarrow \lambda \mid AA \mid aaaa \\ B &\rightarrow \lambda \mid BB \mid bb \\ D &\rightarrow \lambda \mid DD \mid a \end{aligned}$$

- (b) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 118.

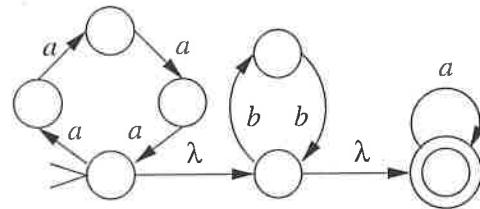


Figure 118:

Problem 516 Let:

$$L = \{a^n c^k a^\ell b^j a^m \mid j=0, \ell=3n, k=m, j, k, \ell, m, n, p \geq 0\}$$

- (a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The template is:

$$\{a^n c^k a^{3n} a^k \mid k, n \geq 0\}$$

or, equivalently,

$$\{a^n c^k a^k (aaa)^n \mid k, n \geq 0\}$$

which yields the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aSaaa \mid A \\ A &\rightarrow cAa \mid \lambda \end{aligned}$$

- (b) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: This automaton does not exist, since L is not regular.

To prove this, assume the opposite—that L is regular. First, observe that every word of L satisfies the following property:

The number of c 's is not greater than the number of a 's.

To verify that the property always holds, observe that when the number of c 's is equal to k , then the number of a 's is equal to $k + 4n$, for some $n \geq 0$.

Let η be the constant as in the Pumping Lemma for L . Consider a word $w = c^k a^k$, where $k > \eta$, obtained from the general template by setting $n = 0$.

Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < \eta < k$. Hence, the pumping part v is entirely within the c -segment, meaning that $v = c^q$ for some q such that $0 < q < \eta$.

After pumping once (up) we obtain a word: $w_1 = c^{k+q} a^k$, where the number of c 's is greater than the number of a 's. This means that w_1 violates the stated property of L and $w_1 \notin L$, whence the contradiction.

Problem 517

$$L = \{a^n c^k a^\ell b^j a^m \mid n=0, \ell=p+1, j=\ell, j, k, \ell, m, n, p \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The template is:

$$\{c^k a^{p+1} b^{p+1} a^m \mid k, m, p \geq 0\}$$

which yields the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, K\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow KBA \\ K &\rightarrow cK \mid \lambda \\ B &\rightarrow aBb \mid ab \\ A &\rightarrow aA \mid \lambda \end{aligned}$$

(b) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: This automaton does not exist, since L is not regular.

To prove this, assume the opposite—that L is regular. First, observe that the template guarantees that every word of L satisfies the following property:

The number of a 's to the left of the (single) segment of b 's is equal to the number of b 's.

Let η be the constant as in the Pumping Lemma for L . Consider a word $w = a^{p+1} b^{p+1}$, where $p > \eta$, obtained from the general template by setting $k = m = 0$.

Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < \eta < p$. Hence, the pumping part v is entirely within the a -segment, meaning that $v = a^q$ for some q such that $0 < q < \eta$.

After pumping once (up) we obtain a word: $w_1 = a^{p+1+q} b^{p+1}$, where the number of a 's is greater

than the number of b 's (and these a 's are to the left of the b 's.) This means that w_1 violates the stated property of L and $w_1 \notin L$, whence the contradiction.

Problem 518 Let:

$$L = \{c^{n+2} a^m b^j a^p d^{\ell+1} \mid n = j + p \wedge m = 0\}$$

where $m, n, j, p, \ell \in N$ and $N = \{0, 1, \dots\}$ is the set of natural numbers.

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The general template for strings of L is:

$$c^p c^j cc b^j a^p d^{\ell+1}$$

Hence, L is generated by the grammar:

$G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow AD \\ A &\rightarrow cAa \mid B \\ B &\rightarrow cBb \mid cc \\ D &\rightarrow dD \mid d \end{aligned}$$

(b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: This automaton does not exist, since L is not regular. To prove this, first observe that in the general template $c^{p+j+2} b^j a^p d^{\ell+1}$ the number of c 's is greater by 2 than the sum of numbers of b 's and a 's.

Assume the opposite—that L is regular. Let k be the constant as in the Pumping Lemma for L . Consider a word $w = c^{n+2} a^n d$, where $n > k$, obtained from the general template for L by setting $j = \ell = 0$.

Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < k < n$. Hence, the pumping part v is entirely within the c -segment, meaning that $v = c^m$ for some m such that $0 < m < k$. After pumping once, we obtain a word: $w_1 = c^{n+m+2} a^n d$. However, since $m > 0$ (as the length of the pumping substring), we conclude that $n + m + 2 > n + 0 + 2$. Yet, w_1 has exactly n occurrences of a and zero b 's, which means that it should have exactly $n + 0 + 2$ occurrences of c , but this is false, since w_1 has more c 's than this—precisely: $n + m + 2$. This means that $w_1 \notin L$, whence the contradiction.

Problem 519 Let:

$$L = \{b^{n+2} a^m d^j a^p c^{\ell+1} \mid n = j + p \wedge m = \ell\}$$

where $m, n, j, p, \ell \in N$, and $N = \{0, 1, \dots\}$ is the set of natural numbers.

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: Observe that the general template for strings that belong to L is:

$$b^{j+p+2}a^\ell d^j a^p c^{\ell+1}$$

Observe that the number of b 's is always by 2 greater than the number of d 's plus the number of a 's between d 's and c 's; also, the number of c 's is by one greater than the number of a 's before the d 's.

To prove that L is not context-free, assume the opposite; let k be the constant as in the Pumping Lemma for L . Select a word:

$$w = b^{\ell+2}a^\ell d^\ell c^{\ell+1} \text{ where } \ell > k$$

obtained by setting $p = 0, j = \ell$ in the general template.

In any “pumping” decomposition: $w = uvxyz$, the length of the “pumping window” vxy is not greater than k : $|vxy| \leq k < \ell$. The “pumping” window vxy either falls within one of the four substrings composed of (multiple) occurrences of a single letter, or it spans two such adjacent substrings—it is too short to extend through as many as three (or four) of them.

If the pumping is only within one of the four designated substrings, it will make this substring longer than is appropriate with respect to the length of another substring. If two different letters are pumped, then there will be at least one other substring with fewer letters than required (in addition to possible appearance of letters out of order.)

Since every pumping decomposition produces illegal words, we conclude that the Pumping Lemma does not hold for L , and L is not context-free.

(b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: This automaton does not exist, since L is not a regular language. If L was regular, then it would be context-free, since every regular language is context-free. However, part (a) proves that L is not context-free. This means that L cannot be regular.

Problem 520 Let:

$$L = \{c^{n+2}a^m b^j a^p d^{\ell+1} \mid j = n \wedge m = p\}$$

where $m, n, j, p, \ell \in N$ and $N = \{0, 1, \dots\}$ is the set of natural numbers.

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The required grammar does not exist, since L is not context-free.

Observe that the general template for strings that belong to L is:

$$c^{j+2}a^p b^j a^p d^{\ell+1}$$

To prove that L is not context-free, assume the opposite. Since L is, by assumption, context-free, its intersection

with any regular language is also context-free. Consider the regular language

$$L_1 = c^* a^* b^* a^* d$$

Then, the general template for string that belong to $L \cap L_1$ is:

$$c^{j+2}a^p b^j a^p d$$

Observe that every string contains exactly one d , the number of c 's is equal to the number of b 's plus two, while the two a -substrings have the same length. We show that $L \cap L_1$ is not context-free.

Assume the opposite, and let k be the constant as in the Pumping Lemma for $L \cap L_1$. Select a word:

$$w = c^{p+2}a^p b^p a^p d \text{ where } p > k$$

obtained by setting $j = p$ in the general template.

In any “pumping” decomposition: $w = uvxyz$, the length of the “pumping window” vxy is not greater than k : $|vxy| \leq k < p$. The “pumping” window vxy either falls within one of the five substrings composed of (multiple) occurrences of a single letter, or it spans two such adjacent substrings—it is too short to extend through as many as three (or four or five) of them.

If the trailing d is pumped, then the string will have more than one d , while all strings in $L \cap L_1$ have exactly one d . If the pumping is only within one of the leftmost four designated substrings, it will make this substring longer than is appropriate with respect to the length of another substring. In the last case, if two different letters are pumped, then there will be at least one other substring with fewer letters than required (in addition to possible appearance of letters out of order.)

Since every pumping decomposition produces illegal words, we conclude that the Pumping Lemma does not hold for $L \cap L_1$, and $L \cap L_1$ is not context-free. Since L_1 is regular, it means that L cannot be context-free.

(b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: This automaton does not exist, since L is not a regular language. If L was regular, then it would be context-free, since every regular language is context-free. However, part (a) proves that L is not context-free. This means that L cannot be regular.

Problem 521 Let:

$$L = \{b^{n+2}a^m d^j a^p c^{\ell+1} \mid j = 0 \wedge m = p\}$$

where $m, n, j, p, \ell \in N$, and $N = \{0, 1, \dots\}$ is the set of natural numbers.

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The general template for strings of L is:

$$b^{n+2}a^p a^p c^{\ell+1}$$

or, equivalently:

$$b^{n+2}(aa)^p c^{\ell+1}$$

which is generated by the context-free grammar:

$G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$,
 $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow BAD \\ B &\rightarrow bB \mid bb \\ A &\rightarrow aaA \mid \lambda \\ D &\rightarrow cD \mid c \end{aligned}$$

- (b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 119.

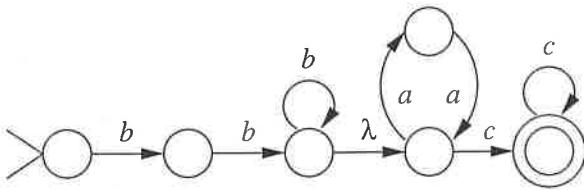


Figure 119:

Problem 522 Let: $L = \{a^n c^k a^\ell b^j a^m\}$ where $n = 2j + 1$, $\ell = p + 1$, $k = 2\ell$, $m = 0$, $j, k, \ell, m, n, p \geq 0$.

- (a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The template is:

$$\{a^{2j+1} c^{2p+2} a^{p+1} b^j \mid j, p \geq 0\}$$

or, equivalently,

$$\{(aa)^j a (cc)^{p+1} a^{p+1} b^j \mid j, p \geq 0\}$$

whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aaSb \mid aA \\ A &\rightarrow ccAa \mid cca \end{aligned}$$

- (b) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: This automaton does not exist, since L is not regular.

To prove this, assume the opposite—that L is regular. First, observe that every word of L satisfies the following property:

If the number of b 's is equal to j , then the number of a 's to the left of (all) c 's is equal to exactly $2j + 1$.

Let k be the constant as in the Pumping Lemma for L . Consider a word $w = (aa)^j a c c a b^j$, where $j > k$, obtained from the general template by setting $p = 0$.

Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < k < j$. Hence, the pumping part v is entirely within the a -segment, meaning that $v = a^m$ for some m such that $0 < m < k$, and w does not contain any a 's.

After pumping once (up) we obtain a word: $w_1 = (aa)^j a^m a c c a b^j$, where there are exactly $2j + m + 1$ a 's to the left of the c 's. However, w_1 still has exactly j occurrences of b . This means that w_1 violates the stated property of L and $w_1 \notin L$, whence the contradiction.

Problem 523 Let: $L = \{a^n c^k a^\ell b^j a^m\}$ where $n = \ell + 2$, $\ell = p + 1$, $k = 0$, $j, k, \ell, m, n, p \geq 0$.

- (a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The template is:

$$\{a^{p+3} a^{p+1} b^j a^m \mid j, p, m \geq 0\}$$

or, equivalently,

$$\{(aa)^p aaaa b^j a^m \mid j, p, m \geq 0\}$$

which is:

$$\{(aa)^p aaaa b^j a^m \mid j, p, m \geq 0\}$$

whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow DaaaaBA \\ A &\rightarrow aA \mid \lambda \\ B &\rightarrow bB \mid \lambda \\ D &\rightarrow aaD \mid \lambda \end{aligned}$$

- (b) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 120.

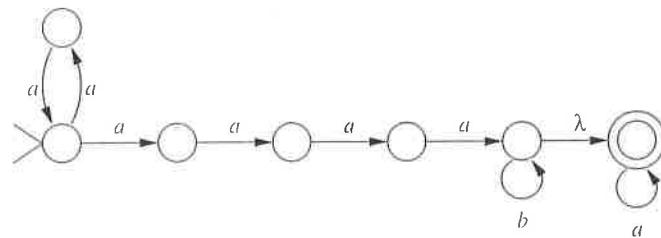


Figure 120:

Problem 524 Let: $L = \{a^n c^k a^\ell b^j a^m\}$ where $n = k + 2$, $m = 2p + 1$, $\ell = 2n + 1$, $j = 0$, $j, k, \ell, m, n, p \geq 0$.

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: This grammar does not exist, since L is not context-free.

The template is: $\{a^{k+2}c^ka^{2k+5}a^{2p+1} \mid k, p \geq 0\}$ or, equivalently, $\{a^{k+2}c^ka^{2k+2p+6} \mid k, p \geq 0\}$.

To prove that L is not context-free, assume the opposite, that L is context-free. First, observe that every word of L satisfies the following property:

If the number of c 's is equal to k , then the number of a 's to the left of (all) c 's is equal to exactly $k+2$, while the number of a 's to the right of (all) c 's is no less than $2k+6$.

Let μ be the constant as in the Pumping Lemma for L . Consider a word $w = a^{k+2}c^ka^{2k+6}$, where $k > \mu$, obtained from the general template by setting $p = 0$.

In any “pumping” decomposition: $w = uvxyz$, the length of the “pumping window” vxy is not greater than μ , and thereby not greater than k . The “pumping” window vxy either falls within one of the three substrings containing a single letter: a^{k+2} , c^k , a^{2k+6} , or it spans two such substrings—it is too short to extend through as many as three of them.

If any part of the pumping window is in the first segment, pump once up, and the new word will contain too few a 's in the third segment. If any part of the pumping window is in the third segment, pump once down, and the new word will contain too many a 's in the first segment. (Note that in this case, pumping up produces no useful result.) Finally, if any part of the pumping window is in the second segment, pump once up, and the new word will contain too few a 's in whichever of the two a -segments is not pumped.

Hence, the mandatory property does not hold for the new word, meaning that the new word is not in L , which is a contradiction.

(b) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: This automaton does not exist, since L is not regular. As is known from the answer to part (a), L is not context-free, and thereby not regular (since every regular language is context-free.)

Problem 525 Let:

$$L = \{a^kb^n c^j d^\ell \mid n \leq 3, j \leq 2, \ell = 2m, j, k, \ell, m, n \geq 0\}$$

Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 121.

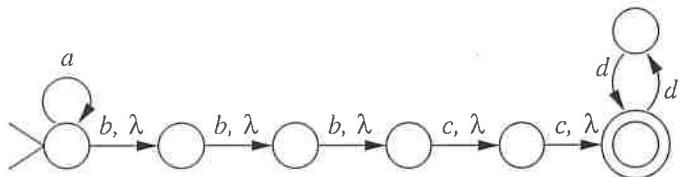


Figure 121:

Problem 526 Let:

$$L = \{a^n c^k d^\ell b^j d^m \mid k > 0, \ell = m, j = 0\}$$

where $n, k, \ell, j, p, m \geq 0$.

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The general template for the strings of L is:

$$L = \{a^n c^k d^{2m} \mid n, k, m \geq 0\}$$

but $k > 0$, whence the grammar:

$G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, K, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AKD \\ A &\rightarrow aA \mid \lambda \\ K &\rightarrow cK \mid c \\ D &\rightarrow ddD \mid \lambda \end{aligned}$$

(b) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 122.

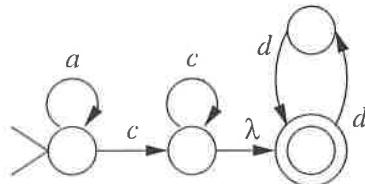


Figure 122:

Problem 527 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aA \parallel cD \mid dD \mid \lambda \\ A &\rightarrow aA \mid bA \mid cB \mid a \\ B &\rightarrow dB \mid aS \mid cA \mid b \\ D &\rightarrow dD \mid aB \mid c \mid d \mid \lambda \end{aligned}$$

Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: The given grammar G is regular. Hence, the automaton represented on Figure 123 is obtained from G straightforwardly, by algorithmic conversion.

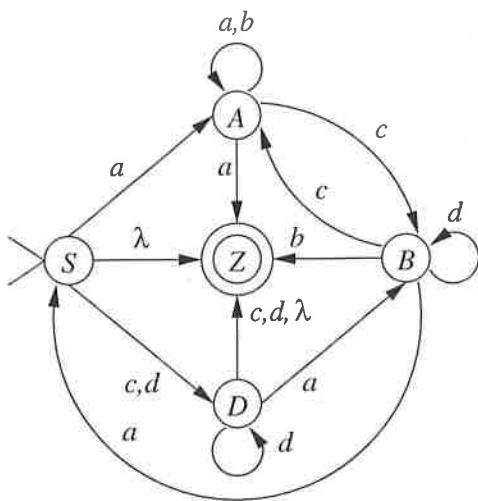


Figure 123:

Problem 528 (a) Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow AA \mid \lambda \mid DDD \\ B &\rightarrow DD \\ D &\rightarrow a \mid b \mid c \end{aligned}$$

Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$((a \cup b \cup c)(a \cup b \cup c)(a \cup b \cup c))^* (a \cup b \cup c)(a \cup b \cup c)$$

(b) Let L' be the language generated by the context-free grammar $G' = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow aS \mid bA \mid cS \mid a \mid c \mid \lambda \\ A &\rightarrow aA \mid bB \mid cA \\ B &\rightarrow aB \mid bS \mid cB \mid b \end{aligned}$$

Draw a state-transition graph of a finite automaton that accepts L' . If such automaton does not exist, prove it.

Answer: This grammar is regular, hence the construction of the automaton is algorithmic. See Figure 124.

Problem 529 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, B, D, E\}$, and P is:

$$\begin{aligned} S &\rightarrow aB \mid bD \mid a \\ B &\rightarrow bD \mid a \mid b \\ D &\rightarrow a \mid b \\ E &\rightarrow aS \mid bE \mid \lambda \end{aligned}$$

(a) Construct a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

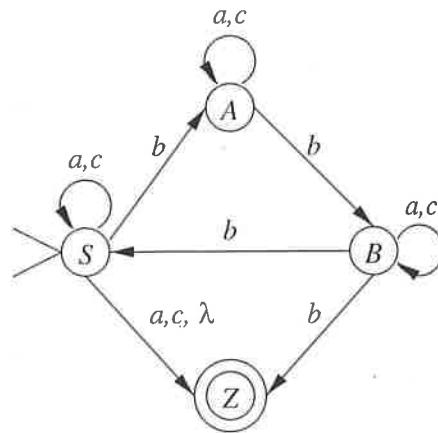


Figure 124:

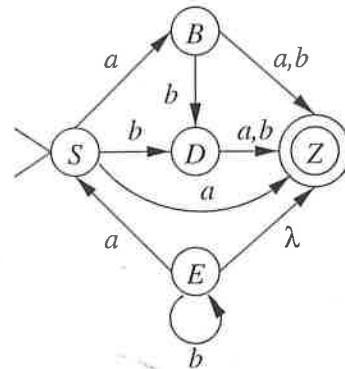


Figure 125:

Answer: See Figure 125.

(b) Does there exist an algorithm that solves the following problem:

INPUT: An arbitrary context-free grammar G .

QUESTION: Is G a regular grammar?

Explain your answer briefly.

Answer: Yes—by definition, a grammar is regular exactly when each of its productions has one of the following forms:

$$\begin{aligned} A &\rightarrow \lambda \\ A &\rightarrow a \\ A &\rightarrow aB \end{aligned}$$

where A and B stand for arbitrary non-terminals, while a stands for an arbitrary terminal.

Problem 530 Let L be the language generated by the grammar $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b, c\}$, $V = \{S, B, D\}$, and P comprises:

$$\begin{aligned} S &\rightarrow bS \mid cS \mid aB \\ B &\rightarrow aB \mid cS \mid bD \mid \lambda \\ D &\rightarrow aB \mid bS \mid cD \mid \lambda \end{aligned}$$

(a) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: The grammar is regular, and the construction of the equivalent finite automaton is algorithmic. See Figure 126.

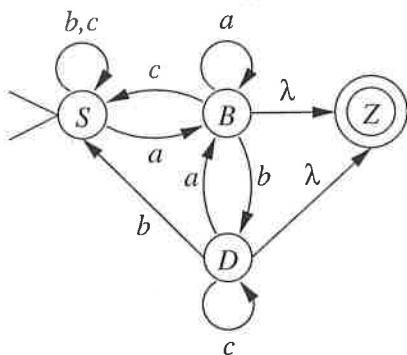


Figure 126:

(b) Does there exist a context-free grammar G' such that G' is not a regular grammar, but G' generates L ? Prove your answer.

Answer: Yes—there exist infinitely (countably) many such grammars. One of them, for instance, is the grammar $G_1 = (V_1, \Sigma, P_1, S)$, where:

$\Sigma = \{a, b, c\}$, $V_1 = \{S, B, D, E\}$, and P comprises:

$$\begin{aligned} S &\rightarrow bS \mid cS \mid aB \mid ES \\ B &\rightarrow aB \mid cS \mid bD \mid \lambda \\ D &\rightarrow aB \mid bS \mid cD \mid \lambda \\ E &\rightarrow \lambda \end{aligned}$$

Grammar G_1 is not regular, since the production $S \rightarrow ES$ violates the definition of regular grammars. Precisely, this production has two variables on the right side, while the right side of any production in a regular grammar is restricted to the empty string or a single terminal, possibly followed by a single variable. However, G_1 and G are equivalent, since the new production—the only one in which the two grammars differ—does not generate any terminal strings other than those already generated by S .

Problem 531 Let L_1 be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, B, D, E\}$, and P is:

$$\begin{aligned} S &\rightarrow aB \mid aD \mid bE \mid a \\ B &\rightarrow bS \mid bB \mid cE \mid b \\ D &\rightarrow aD \mid bB \mid cS \mid c \\ E &\rightarrow aS \mid aB \mid aD \end{aligned}$$

Construct a state-transition graph of a finite automaton M_1 that accepts L_1 . If such automaton does not exist, explain why.

Answer: See Figure 127.

Problem 532 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where

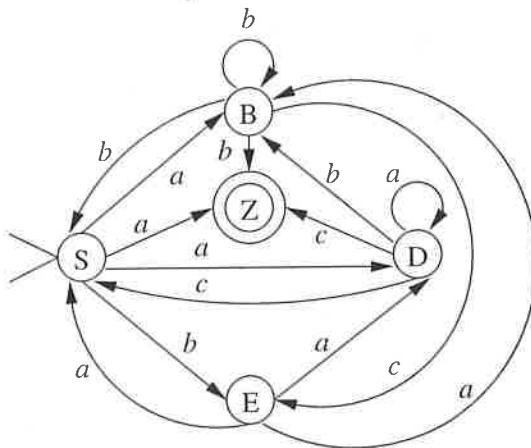


Figure 127:

$\Sigma = \{a, b, c\}$, $V = \{S, B, D, E\}$, and P is:

$$\begin{aligned} S &\rightarrow aB \mid bD \mid cE \mid a \\ B &\rightarrow bD \mid cE \mid a \mid b \\ D &\rightarrow a \mid b \mid c \\ E &\rightarrow aS \mid bE \mid c \mid \lambda \end{aligned}$$

(a) Construct a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Advice for Answer: G is regular and the construction is algorithmic.

(b) Is the complement \bar{L} of the language L regular? Explain your answer briefly.

Advice for Answer: The complement of every regular language is regular.

Problem 533 (a) Let L_1 be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, B, D, E\}$, and P is:

$$\begin{aligned} S &\rightarrow aB \mid aD \mid bD \mid c \\ B &\rightarrow bD \mid bE \mid cS \mid a \\ D &\rightarrow cD \mid aS \mid bB \mid b \\ E &\rightarrow cS \mid aB \mid cD \end{aligned}$$

Construct a state-transition graph of a finite automaton M_1 that accepts L_1 . If such an automaton does not exist, explain why.

Answer: The grammar is regular, and the construction of the equivalent finite automaton is algorithmic. See Figure 128.

(b) Let L_2 be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, B, D, E\}$, and P is:

$$\begin{aligned} S &\rightarrow BDE \\ B &\rightarrow BB \mid \lambda \mid b \mid c \\ D &\rightarrow bcaB \\ E &\rightarrow aE \mid bE \mid c \end{aligned}$$

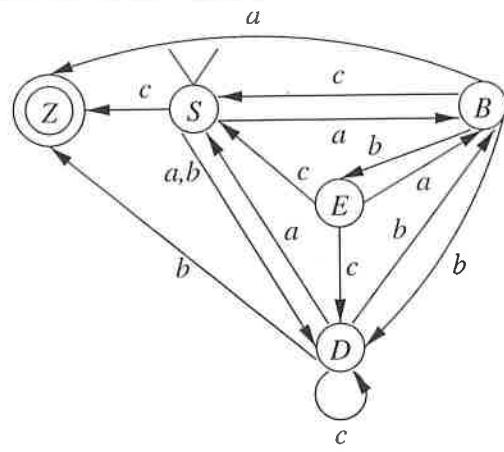


Figure 128:

Construct a state-transition graph of a finite automaton M_2 that accepts L_2 . If such an automaton does not exist, explain why.

Answer: See Figure 129.

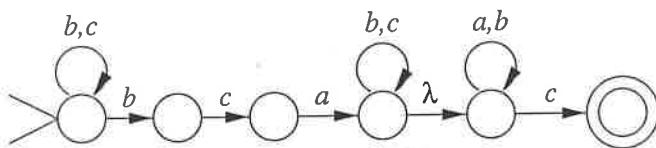


Figure 129:

(c) Is L_2 recursive (decidable)? Explain your answer.

Answer: L_2 is decidable because it is context-free. All context-free languages are decidable.

Problem 534 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, T)$, where $\Sigma = \{a, b, c\}$, $V = \{Q, R, T\}$, and the production set P is:

$$\begin{aligned} Q &\rightarrow bQ \mid cQ \mid \lambda \\ R &\rightarrow aR \mid cR \mid \lambda \\ T &\rightarrow bT \mid cQ \mid aR \end{aligned}$$

Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: Apply the conversion algorithm directly, to obtain the automaton represented on Figure 130..

Problem 535 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aA \mid bD \mid dD \mid \lambda \\ A &\rightarrow aA \mid bA \mid cB \mid a \\ B &\rightarrow dB \mid aS \mid cA \mid b \\ D &\rightarrow dD \mid \lambda \end{aligned}$$

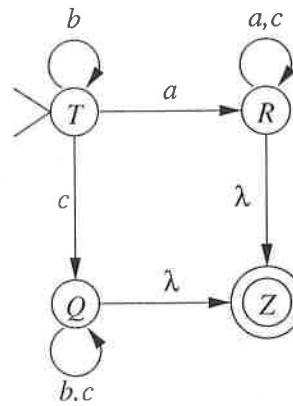


Figure 130:

(a) Construct a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: The automaton represented on Figure 131 is obtained by algorithmic conversion of the given grammar.

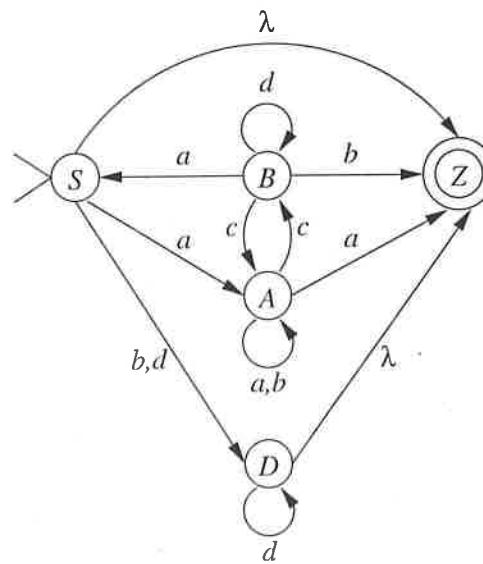


Figure 131:

(b) Is G a regular grammar? Prove your answer.

Answer: Yes—its form complies with that given in the definition of regular grammars. The right-hand side of every rule is either λ or a single terminal symbol, possibly followed by a single variable.

Problem 536 Let L be the language accepted by the automaton represented on Figure 132.

Write a complete formal definition of a regular context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: Apply the conversion algorithm directly, to obtain the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and the production

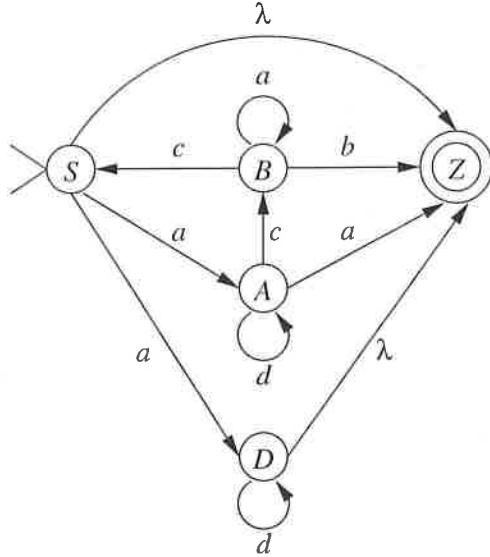


Figure 132:

set P is:

$$\begin{aligned} S &\rightarrow aA \mid aD \mid \lambda \\ A &\rightarrow dA \mid cB \mid a \\ B &\rightarrow aB \mid cS \mid b \\ D &\rightarrow dD \mid \lambda \end{aligned}$$

Problem 537 Let M be the finite automaton represented by the state-transition diagram on Figure 133, and let L be the language accepted by M .

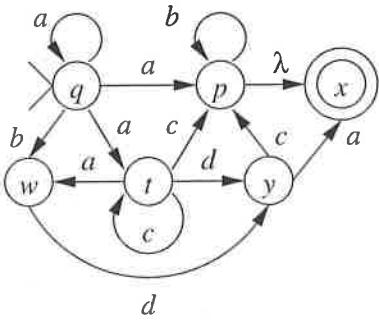


Figure 133:

Write a complete formal definition of a context-free grammar G that generates L . If such a grammar does not exist, prove it.

Answer: The automaton given on Figure 133 is compliant with the form of the input to the conversion algorithm—hence, the following regular grammar is obtained straightforwardly, by an application of the conversion algorithm.

$G = (V, \Sigma, P, q)$, where $\Sigma = \{a, b, c, d\}$, $V = \{q, p, w, t, y\}$, and the production set P is:

$$\begin{aligned} q &\rightarrow aq \mid ap \mid at \mid bw \\ p &\rightarrow bp \mid \lambda \\ w &\rightarrow dy \\ t &\rightarrow ct \mid dy \mid cp \mid aw \\ y &\rightarrow cp \mid a \end{aligned}$$

Problem 538 Let L be the language accepted by the automaton represented on Figure 134.

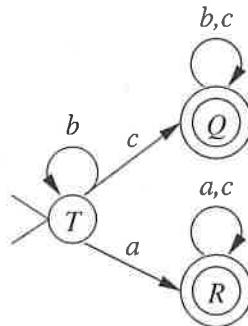


Figure 134:

Write a complete formal definition of a regular context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The original automaton is equivalent to that given on Figure 135, which is, in turn, compliant with the form required for the input to the conversion algorithm—hence, the following regular grammar is obtained straightforwardly, by an application of the conversion algorithm.

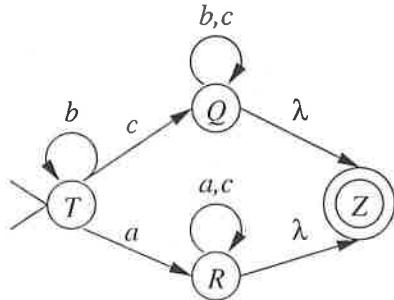


Figure 135:

$G = (V, \Sigma, P, T)$, where $\Sigma = \{a, b, c, d\}$, $V = \{T, Q, R\}$, and the production set P is:

$$\begin{aligned} T &\rightarrow bT \mid cQ \mid aR \\ Q &\rightarrow bQ \mid cQ \mid \lambda \\ R &\rightarrow aR \mid cR \mid \lambda \end{aligned}$$

Problem 539 Let L be the set of strings over alphabet $\{a, b, c\}$ whose length gives remainder 1 if divided by 3.

(b) Draw a state-transition graph of a finite automaton M that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 136.

(b) Write a complete formal definition of a regular context-free grammar G that generates L . If such grammar does not exist, prove it.

Answer: We employ the algorithmic conversion of the automaton constructed in part (a) to obtain the following grammar:

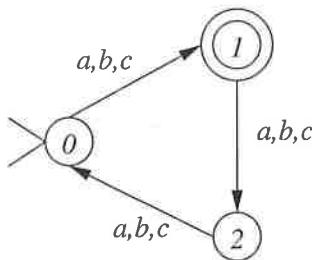


Figure 136:

 $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b, c\}$, $V = \{S, N, T\}$,and the production set P is:

$$\begin{aligned} S &\rightarrow aN \mid bN \mid cN \\ N &\rightarrow aT \mid bT \mid cT \mid \lambda \\ T &\rightarrow aS \mid bS \mid cS \end{aligned}$$

Problem 540 Let L be the set of strings over alphabet $\{a, b, c\}$ in which the number of b 's is not divisible by 4.

(a) Draw a state-transition graph of a finite automaton M that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 137,

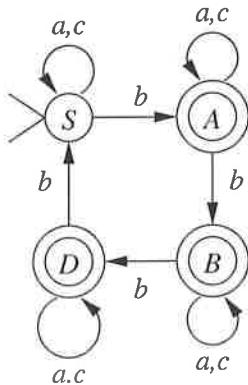


Figure 137:

(b) Write a complete formal definition of a regular context-free grammar G that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b, c\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aS \mid cS \mid bA \\ A &\rightarrow aA \mid cA \mid bB \mid \lambda \\ B &\rightarrow aB \mid cB \mid bD \mid \lambda \\ D &\rightarrow aD \mid cD \mid bS \mid \lambda \end{aligned}$$

Problem 541 Let M be the finite automaton represented by the state diagram on Figure 138, and let L be the language accepted by M .

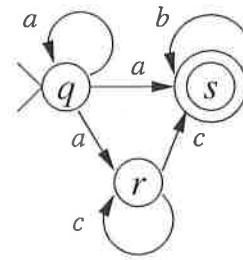


Figure 138:

(a) Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, prove it.

Answer: Apply the algorithm to convert M into a regular context-free grammar: $G = (V, \Sigma, P, q)$, where $\Sigma = \{a, b, c\}$, $V = \{q, s, r\}$, and the production set P is:

$$\begin{aligned} q &\rightarrow aq \mid as \mid ar \\ s &\rightarrow bs \mid \lambda \\ r &\rightarrow cr \mid cs \end{aligned}$$

(b) Let \mathcal{S} be a class of languages over alphabet $\{a, b, c\}$, defined as follows:

Language L is a member of \mathcal{S} if and only if L is accepted by some non-deterministic finite automaton, but there does not exist a context-free grammar that generates L .

What is the cardinality of \mathcal{S} ? Explain your answer briefly.

Answer:

$$|\mathcal{S}| = 0, \text{ since } \mathcal{S} = \emptyset$$

Class \mathcal{S} is empty, since every language accepted by a finite automaton is also generated by some context-free grammar. In fact, this context-free grammar is obtained by an algorithmic conversion of the original finite automaton.

Problem 542 Let M be the finite automaton represented by the state diagram on Figure 139, and let L be the language accepted by M .

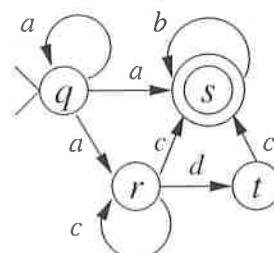


Figure 139:

Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, prove it.

Answer: Apply the algorithm to convert M into a regular context-free grammar: $G = (V, \Sigma, P, q)$, where $\Sigma = \{a, b, c\}$, $V = \{q, s, r, t\}$, and the production set P is:

$$\begin{aligned} q &\rightarrow aq \mid as \mid ar \\ s &\rightarrow bs \mid \lambda \\ r &\rightarrow cr \mid cs \mid dt \\ t &\rightarrow cs \end{aligned}$$

Problem 543 Let L be the language defined by the regular expression:

$$(ac \cup bc)^*(bd \cup dd)^* \cup a^*d^*$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D, E\}$, and P is:

$$\begin{aligned} S &\rightarrow BE \mid AD \\ B &\rightarrow BB \mid \lambda \mid ac \mid bc \\ E &\rightarrow EE \mid \lambda \mid bd \mid dd \\ A &\rightarrow aA \mid \lambda \\ D &\rightarrow dD \mid \lambda \end{aligned}$$

(b) Does there exist an algorithm to convert an arbitrary regular expression into an equivalent regular context-free grammar? Explain your answer briefly.

Answer: Yes—the regular expression is converted into a (nondeterministic) finite automaton, this automaton is converted into a deterministic equivalent, and the deterministic finite automaton into an equivalent regular context-free grammar.

(c) Does there exist an algorithm that solves the following problem:

INPUT: An arbitrary regular expression e .

OUTPUT: A finite automaton M_1 that accepts the complement of the language defined by e .

Explain your answer briefly.

Answer: Yes; automaton M_1 is constructed as follows. First, employ the algorithmic conversion of e into a (nondeterministic) finite automaton M' . Next, convert M' , again by algorithmic conversion, into its deterministic equivalent M'' . Finally, M_1 is obtained from M'' by declaring every final state of M'' to be non-final, and every non-final state to be final.

Problem 544 Write a complete formal definition of a regular context-free grammar that generates the set of all strings over alphabet $\{a, b, c\}$ whose length is not a multiple of 3. If such a regular context-free grammar does not exist, prove it.

Answer: This language is accepted by the finite automaton represented of Figure 140, which is algorithmically convertible into the following regular grammar:

$G = (V, \Sigma, P, Z)$, where $\Sigma = \{a, b\}$, $V = \{Z, N, T\}$, and P is:

$$\begin{aligned} Z &\rightarrow aN \mid bN \mid cN \\ N &\rightarrow aT \mid bT \mid cT \mid \lambda \\ T &\rightarrow aZ \mid bZ \mid cZ \mid \lambda \end{aligned}$$

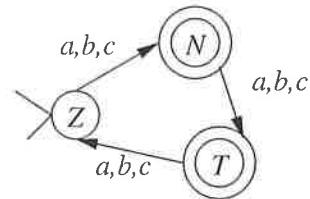


Figure 140:

Problem 545 Let L be a set of non-empty strings over alphabet $\{a, b\}$, defined as follows:

If $w \in L$ and w begins with a , then the length of w is even. If $w \in L$ and w begins with b , then w contains at least one a .

(a) Write a complete formal definition of a regular context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: L is accepted by the finite automaton represented of Figure 141, which is algorithmically convertible into the following regular grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, A, D, E, F\}$, and P is:

$$\begin{aligned} S &\rightarrow aD \mid bA \\ D &\rightarrow aE \mid bE \\ E &\rightarrow aD \mid bD \mid \lambda \\ A &\rightarrow bA \mid aF \\ F &\rightarrow aF \mid bF \mid \lambda \end{aligned}$$

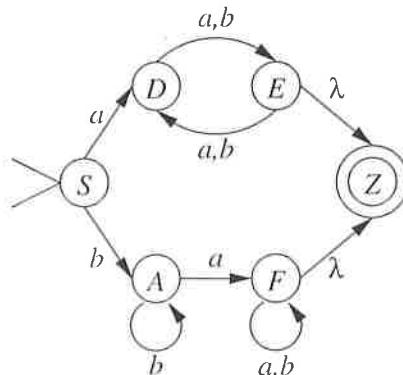


Figure 141:

(b) Is \overline{L} (the complement of L) regular? Prove your answer.

Answer: Yes—the answer to the part (a) shows that L is regular, and the complement of every regular language is regular.

Problem 546 Write a complete formal definition of a regular context-free grammar that generates the set of all strings over the alphabet $\{a, b, c\}$ that contain the substring bb . If such a regular context-free grammar does not exist, prove it.

Answer: The required language is accepted by the finite automaton represented on Figure 142, which is in a form that admits an algorithmic conversion into the following regular grammar.

$G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, Z\}$, and P is:

$$\begin{aligned} S &\rightarrow aS \mid bS \mid cS \mid bA \\ A &\rightarrow bB \\ B &\rightarrow aB \mid bB \mid cB \mid \lambda \end{aligned}$$

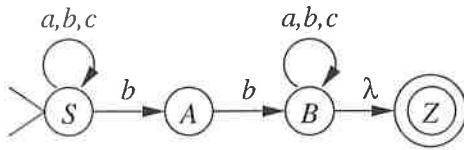


Figure 142:

Problem 547 Let L be the set of strings over the alphabet a, b, d that contain string dba as a substring.

(a) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 143.

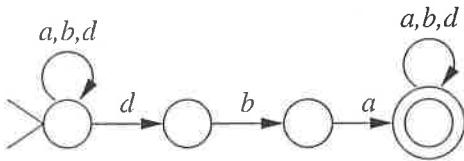


Figure 143:

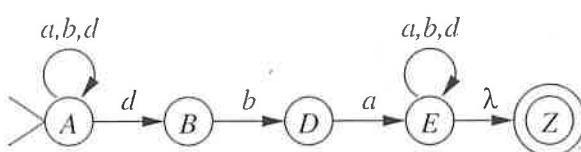


Figure 144:

(b) Write a complete formal definition of a regular context-free grammar that generates the language L . If such a grammar does not exist, prove it.

Answer: The automaton represented on Figure 143 is evidently equivalent to the automaton represented on Figure 144, which is in turn in a form appropriate for the algorithmic conversion into a regular context-free grammar. Hence, the required grammar is: $G = (V, \Sigma, P, A)$,

where $\Sigma = \{a, b, d\}$, $V = \{A, B, D, E\}$, and the production set P is:

$$\begin{aligned} A &\rightarrow aA \mid bA \mid dA \mid dB \\ B &\rightarrow bD \\ D &\rightarrow aE \\ E &\rightarrow aE \mid bE \mid dE \mid \lambda \end{aligned}$$

Problem 548 (a) Let L_1 be the language generated by the context-free grammar $G_1 = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \mid D \mid \lambda \mid SS \\ A &\rightarrow \lambda \mid AA \mid aaa \\ B &\rightarrow bb \mid b \\ D &\rightarrow ccccD \mid \lambda \end{aligned}$$

Draw a state-transition graph of a finite automaton that accepts L_1 . If such an automaton does not exist, prove it.

Answer: By inspection of the grammar G_1 , we conclude that it generates a regular language, represented by the regular expression:

$$((aaa)^*(b \cup bb) \cup (cccc)^*)^*$$

This language is accepted by the automaton represented on Figure 145.

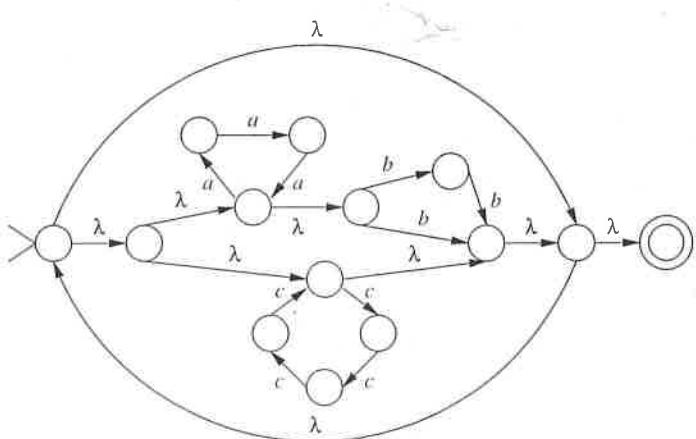


Figure 145:

(b) Let L_2 be the language generated by the context-free grammar $G_2 = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow cS \mid aA \mid bB \\ A &\rightarrow \lambda \mid aa \mid bd \\ B &\rightarrow bS \mid ba \mid cd \\ D &\rightarrow aA \mid bB \mid cD \mid \lambda \end{aligned}$$

Draw a state-transition graph of a finite automaton that accepts L_2 . If such an automaton does not exist, prove it.

Answer: Grammar G_2 is regular. Hence, its conversion to a finite automaton is algorithmic. The result is represented on Figure 146.

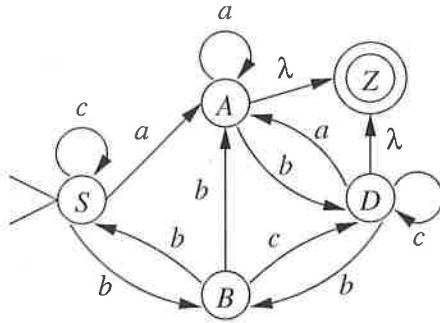


Figure 146:

Problem 549 Let L be the language accepted by the automaton represented on Figure 147.

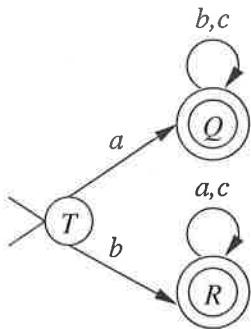


Figure 147:

Write a complete formal definition of a regular context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The automaton given on Figure 147 is evidently equivalent to that given on Figure 148, which in turn is in a form that admits algorithmic conversion into a regular grammar: $G = (V, \Sigma, P, T)$, where $\Sigma = \{a, b, c\}$ is the set of terminals; $V = \{T, Q, R, Z\}$ is the set of variables; T is the start symbol, and the production set P is:

$$\begin{aligned} T &\rightarrow aQ \mid bR \\ Q &\rightarrow bQ \mid cQ \mid \lambda \\ R &\rightarrow aR \mid cR \mid \lambda \end{aligned}$$

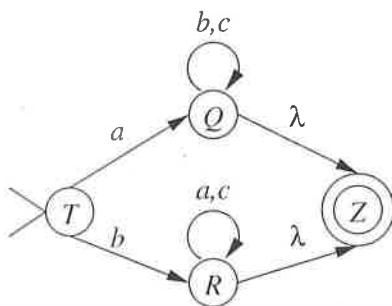


Figure 148:

Problem 550 Let L_1 be the set of all strings over alphabet $\{a, b\}$ whose first and last letters are equal. Let

L_2 be the set of strings over alphabet $\{a, b\}$ that contain at least one b .

(a) Draw a state-transition graph of a finite automaton that accepts $L_1 \cup L_2$. If such automaton does not exist, prove it.

Answer: See Figure 149.

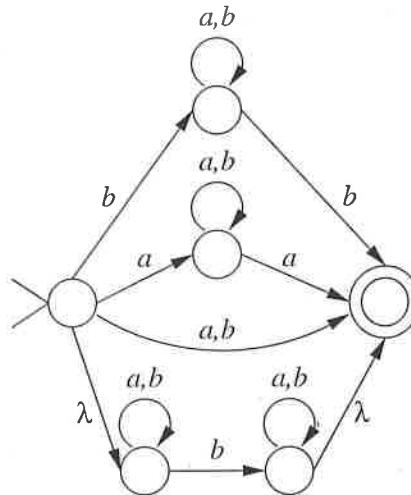


Figure 149:

(b) Draw a state-transition graph of a finite automaton that accepts $L_1 \cap L_2$. If such automaton does not exist, prove it.

Answer: See Figure 150.

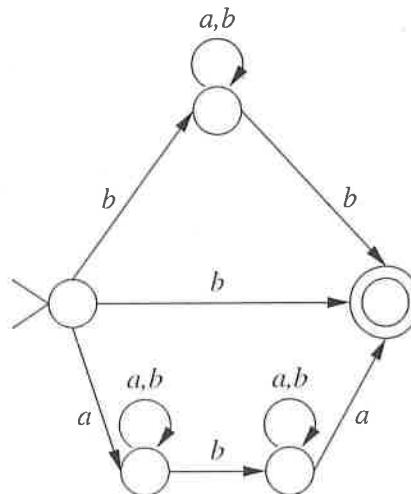


Figure 150:

(c) Write a complete formal definition of a context-free grammar that generates $L_1 L_2$. If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, L, R, A\}$, and the production set

P is:

$$\begin{aligned} S &\rightarrow LR \\ L &\rightarrow aAa \mid bAb \mid a \mid b \\ R &\rightarrow AbA \\ A &\rightarrow AA \mid \lambda \mid a \mid b \end{aligned}$$

Problem 551 (a) Let L_1 be the set of strings over alphabet $\{a, b, c\}$ whose length is equal to $3k$, for some non-negative integer k .

Draw a state-transition graph of a finite automaton that accepts L_1 . If such automaton does not exist, prove it.

Answer: See Figure 151.

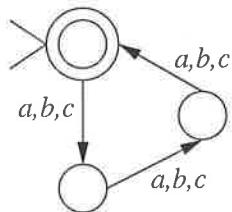


Figure 151:

(b) Let L_2 be the set of strings over alphabet $\{a, b, c\}$ whose length is equal to $3k + 5\ell$, for some non-negative integers k, ℓ .

Draw a state-transition graph of a finite automaton that accepts L_2 . If such automaton does not exist, prove it.

Answer: See Figure 152.

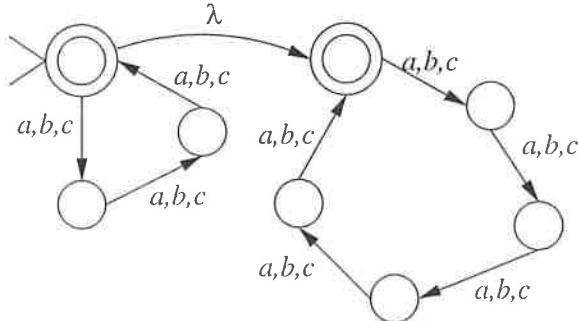


Figure 152:

Problem 552 Let L be the set of all strings over the alphabet $\{a, b, c\}$ whose length is greater than 2.

(a) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 153.

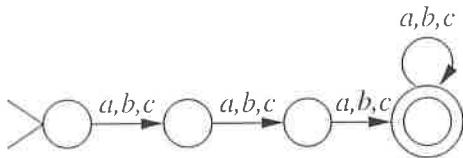


Figure 153:

(b) Draw a state-transition graph of a finite automaton that accepts \overline{L} (the complement of L). If such an automaton does not exist, prove it.

Answer:

Answer: See Figure 154.

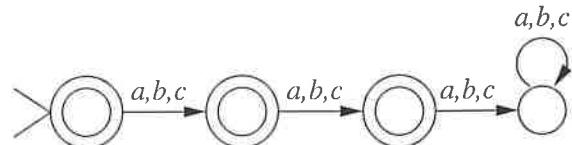


Figure 154:

Problem 553 Let L be the set of strings over $\{a, b, c\}$ whose number of b 's is divisible by 3.

(a) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 155 (a).

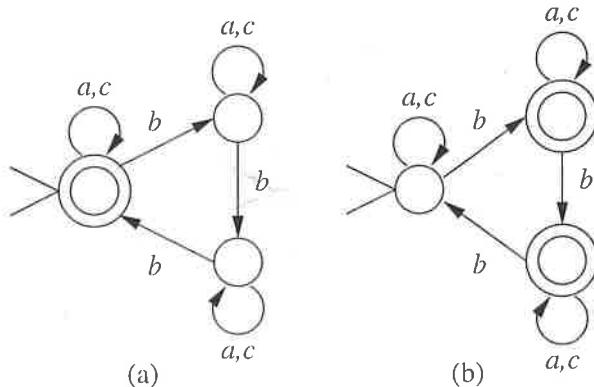


Figure 155:

(b) Construct a state-transition graph of a finite automaton that accepts \overline{L} (the complement of L). If such an automaton does not exist, prove it.

Answer: See Figure 155 (b).

Problem 554 Let L_1 be the set of all strings over the alphabet $\{a, b, c, d\}$ that contain an even number of d 's. Let L_2 be the set of all strings over the alphabet $\{a, b, c, d\}$ that do not contain b .

(a) Draw a state-transition graph of a finite automaton that accepts the language $L_1 L_2$. If such an automaton does not exist, explain why.

Answer: See Figure 156.

(b) Draw a state-transition graph of a finite automaton that accepts the language $\overline{L_1 L_2}$ (the complement of $L_1 L_2$). If such an automaton does not exist, explain why.

Answer: Since the automaton constructed in the answer to part (a) is small, a reasonable approach is to convert it into a deterministic automaton, which then directly yields the complement construction.

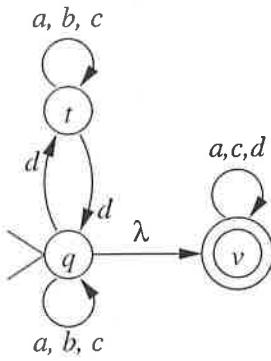


Figure 156:

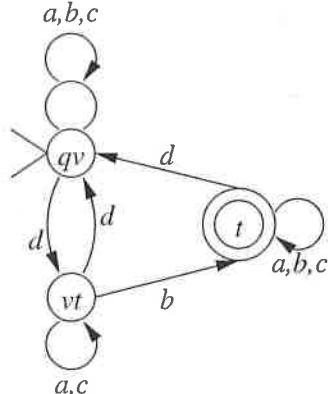


Figure 157:

Let

$$M = (\{q, t, v\}, \{a, b, c, d\}, \delta, q, \{v\})$$

be the automaton constructed in the answer to part (a), which accepts $L_1 L_2$, and let

$$M' = (Q', \{a, b, c, d\}, \delta', q'_0, F')$$

where $Q' \subseteq \mathcal{P}(\{q, t, v\})$, be a deterministic automaton equivalent to M . We apply the conversion algorithm.

The old transition function:

δ	a	b	c	d	λ
q	q	q	q	t	v
t	t	t	t	q	\emptyset
v	v	\emptyset	v	v	\emptyset

The λ -closures of the states of M are:

x	$C(x)$
q	{q, v}
t	{t}
v	{v}

The new initial state: $q'_0 = C(q) = \{q, v\}$.

The transition function δ' :

δ'	a	b	c	d
{q, v}	{q, v}	{q, v}	{q, v}	{v, t}
{v, t}	{v, t}	{t}	{v, t}	{q, v}
{t}	{t}	{t}	{t}	{q, v}

The new set of states:

$$Q' = \{\{q, v\}, \{v, t\}, \{t\}\}.$$

The new set of final states:

$$F' = \{\{q, v\}, \{v, t\}\}.$$

Finally, let M_1 be an automaton that accepts $\overline{L_1 L_2}$. M_1 is different from M' only in that its set of final states F_1 is the complement in Q' of the set of final states of M' :

$$F_1 = Q' \setminus F' = \{\{t\}\}$$

The required automaton is given on Figure 157.

Problem 555 Let L be the set of strings over the alphabet $\Sigma = \{a, b, c\}$ defined as follows:

- (a) Draw a state-transition graph of a finite automaton that accepts the language L . If such an automaton does not exist, explain why.

Answer: See Figure 158.

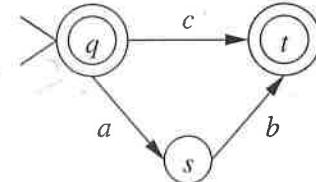


Figure 158:

- (b) Draw a state-transition graph of a *deterministic* finite automaton that accepts the language \overline{L} (the complement of L .) If such an automaton does not exist, explain why.

Answer: Since the automaton constructed in the answer to part (a) is small, a reasonable approach is to convert it into a deterministic automaton, which then directly yields the complement construction.

Let

$$M = (\{q, s, t\}, \{a, b, c\}, \delta, q, \{t\})$$

be the automaton constructed in the answer to part (a), which accepts L , and let

$$M' = (Q', \{a, b, c\}, \delta', q'_0, F')$$

where $Q' \subseteq \mathcal{P}(\{q, s, t\})$, be a deterministic automaton equivalent to M . We apply the conversion algorithm.

The old transition function:

δ	a	b	c
q	s	\emptyset	t
s	\emptyset	t	\emptyset
t	\emptyset	\emptyset	\emptyset

Since the old automaton does not have any λ -transitions, the λ -closure of any state of M is the singleton containing that state.

The new initial state: $q'_0 = \{q\}$.

The transition function δ' :

δ'	a	b	c
{q}	{s}	\emptyset	{t}
\emptyset	\emptyset	\emptyset	\emptyset
{s}	\emptyset	{t}	\emptyset
{t}	\emptyset	\emptyset	\emptyset

The new set of states:

$$Q' = \{\{q\}, \{s\}, \{t\}, \emptyset\}.$$

The new set of final states:

$$F' = \{\{q\}, \{t\}\}.$$

Finally, let M_1 be an automaton that accepts \overline{L} . M_1 is different from M' only in that its set of final states F_1 is the complement in Q' of the set of final states of M' :

$$F_1 = Q' \setminus F' = \{\{s\}, \emptyset\}$$

The required automaton is given on Figure 159.

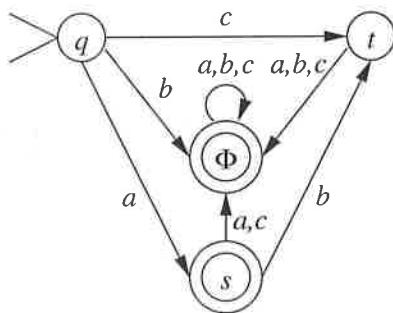


Figure 159:

(c) Write a complete formal definition of a context-free grammar that generates the language \overline{L} (the complement of L .) If such a grammar does not exist, explain why.

Answer: To construct the grammar for \overline{L} , observe:

- (1) All strings of length 3 or more belong to \overline{L} ;
- (2) Among the 9 strings of length 2, only one (ab) is not in \overline{L} ;
- (3) Among the 3 strings of length 1, only one (c) is not in \overline{L} ;
- (4) Empty string is not in \overline{L} .

The grammar represents a union of the just enumerated subsets: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, S_3, S_2, S_1, Z, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow S_3 \mid S_2 \mid S_1 \\ S_3 &\rightarrow ZZZA \\ A &\rightarrow \lambda \mid AA \mid Z \\ Z &\rightarrow a \mid b \mid c \\ S_2 &\rightarrow bZ \mid cZ \mid aa \mid ac \\ S_1 &\rightarrow a \mid b \end{aligned}$$

Problem 556 Consider the finite automaton M represented by the state transition graph given on Figure 160. Let L be the language over the alphabet $\{a, b, c\}$ accepted by M .

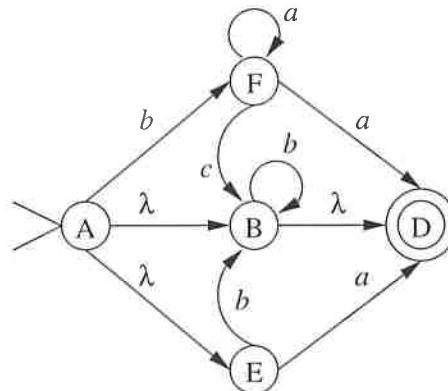


Figure 160:

(a) Construct a state-transition graph of a deterministic finite automaton that accepts L , and explain your construction. If such an automaton does not exist, prove it.

Answer: We apply the algorithm for conversion of a nondeterministic automaton to a deterministic equivalent, to construct a deterministic automaton $M_1 = (Q, \{a, b, c\}, \delta, q_0, F)$, equivalent to M .

The transition function of M is given by the following table.

	a	b	c	λ
A	\emptyset	{F}	\emptyset	{B, E}
B	\emptyset	{B}	\emptyset	{D}
E	{D}	{B}	\emptyset	\emptyset
F	{F, D}	\emptyset	{B}	\emptyset
D	\emptyset	\emptyset	\emptyset	\emptyset

The λ -closures of the states of M are:

q	$\mathcal{C}(q)$
A	{A, B, D, E}
B	{B, D}
E	{E}
F	{F}
D	{D}

The initial state: $q_0 = \mathcal{C}(A) = \{A, B, D, E\}$.

The transition function δ of M_1 is given by the following table.

δ	a	b	c
{A, B, D, E}	{D}	{B, D, F}	\emptyset
{D}	\emptyset	\emptyset	\emptyset
{B, D, F}	{D, F}	{B, D}	{B, D}
\emptyset	\emptyset	\emptyset	\emptyset
{B, D}	\emptyset	{B, D}	\emptyset
{D, F}	{D, F}	\emptyset	{B, D}

The set of states of M_1 :

$$Q = \{\{A, B, D, E\}, \{D\}, \{B, D, F\}, \{B, D\}, \{D, F\}, \emptyset\}.$$

The set of final states:

$$F = \{\{A, B, D, E\}, \{D\}, \{B, D, F\}, \{B, D\}, \{D, F\}\}.$$

The state diagram of M_1 is given on Figure 161.

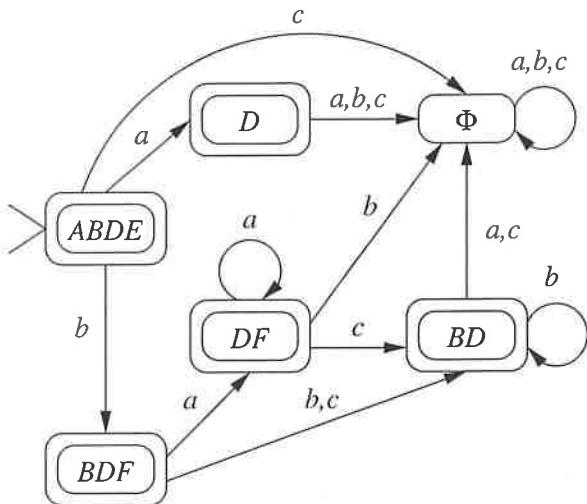


Figure 161:

(b) Construct a state-transition graph of a deterministic finite automaton that accepts \bar{L} (the complement of L). If such an automaton does not exist, prove it.

Answer: Since M_1 is deterministic, the algorithm for conversion to a complement is applied directly, yielding the result given on Figure 162.

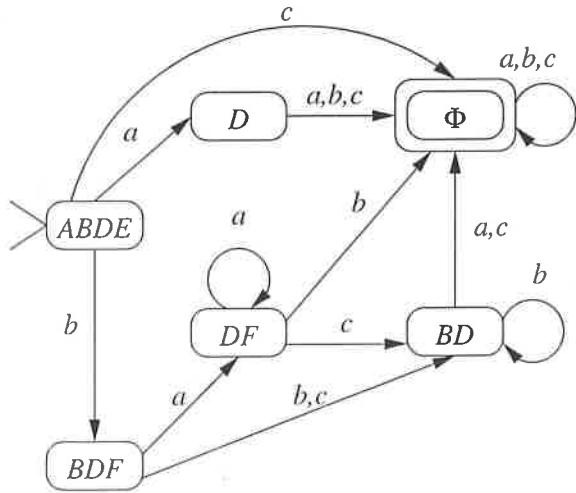


Figure 162:

Problem 557 (a) Let L_1 be the set of non-empty strings over the alphabet $\{a, b, c\}$ with the following additional properties:

If the length of the string is odd, then the middle letter is b .

If the length of the string is even, then the string ends with c .

Write a complete formal definition of a context-free grammar that generates L_1 . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, D, E, Z\}$, and P is:

$$\begin{aligned} S &\rightarrow D \mid E \\ D &\rightarrow ZDZ \mid b \\ E &\rightarrow ZZE \mid Zc \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

(b) Let L_2 be the set of non-empty strings over the alphabet $\{a, b, c\}$ the following additional properties:

If the length of the string is odd, then the first letter is b .

If the length of the string is even, then the string does not end with c .

Draw a state-transition graph of a finite automaton that accepts L_2 . If such an automaton does not exist, prove it.

Answer: See Figure 163.

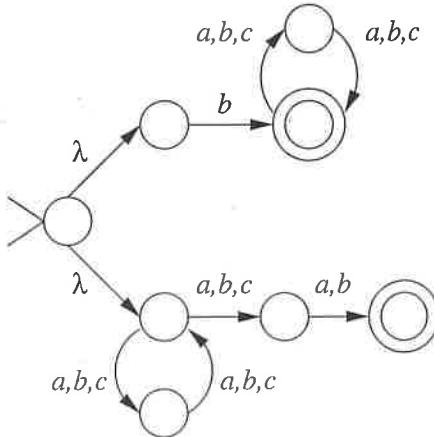


Figure 163:

Problem 558 Let L_1 be the set of all strings over the alphabet $\{a, b\}$ whose length is not divisible by 3.

Let L_2 be the set of all strings over the alphabet $\{a, b\}$ whose length is equal to 4 and the second letter is a .

(a) Write a regular expression that represents the language

$$L_2^*$$

If such a regular expression does not exist, state it and explain why.

Answer:

$$((a \cup b)a(a \cup b)(a \cup b))^*$$

(b) Write a complete formal definition of a context-free grammar that generates the language

$$L_1 \cup L_2$$

If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, A, B, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow A \mid B \\ A &\rightarrow ZZZA \mid Z \mid ZZ \\ B &\rightarrow ZaZ \\ Z &\rightarrow a \mid b \end{aligned}$$

(c) Draw a state-transition graph of a finite automaton that accepts the language

$$L_1 L_2$$

If such an automaton does not exist, state it and explain why.

Answer: See Figure 164.

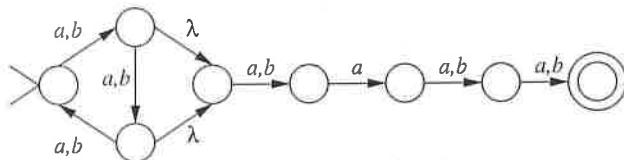


Figure 164:

Problem 559 Let L_1 be the set of all strings over the alphabet $\{a, b\}$ that contain $babb$ as a substring.

Let L_2 be the set of all strings over the alphabet $\{a, b\}$ that contain exactly one a .

(a) Write a regular expression that represents the language

$$L_2^*$$

If such a regular expression does not exist, state it and explain why.

Answer:

$$(b^*ab^*)^*$$

(b) Write a complete formal definition of a context-free grammar that generates the language

$$L_1 \cup L_2$$

If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow DbabbD \mid BaB \\ D &\rightarrow aD \mid bD \mid \lambda \\ B &\rightarrow bB \mid \lambda \end{aligned}$$

(c) Draw a state-transition graph of a finite automaton that accepts the language

$$L_1 L_2$$

If such an automaton does not exist, state it and explain why.

Answer: See Figure 165.

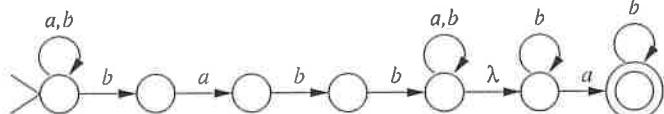


Figure 165:

Problem 560 Let L_1 be the set of all strings over the alphabet $\{a, b, c\}$ that begin with b and end with a .

Let L_2 be the set of all strings over the alphabet $\{a, b, c\}$ that contain an even number of c 's.

(a) Draw a state-transition graph of a finite automaton that accepts the language:

$$L_1 \cup L_2$$

If such an automaton does not exist, state it and explain why.

Answer: See Figure 166.

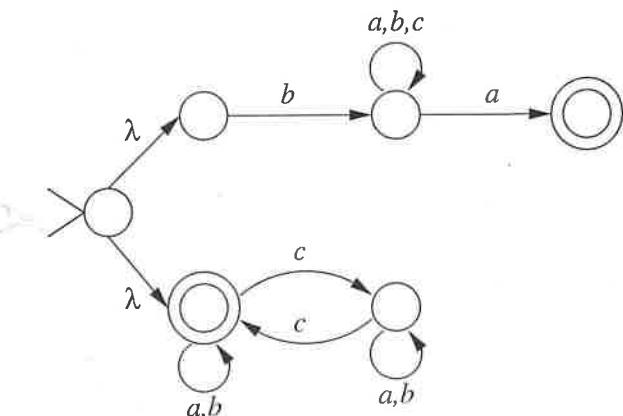


Figure 166:

(b) Write a complete formal definition of a context-free grammar that generates the language

$$L_1 L_2$$

If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D, E\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow bDa \\ D &\rightarrow DD \mid \lambda \mid a \mid b \mid c \\ B &\rightarrow EcEcEB \mid E \\ E &\rightarrow aE \mid bE \mid \lambda \end{aligned}$$

Problem 561 Let L_1 be the set of all strings over the alphabet $\{a, b, c\}$ where every c is immediately followed by b .

Let L_2 be the set of all strings over the alphabet $\{a, b, c\}$ that contain exactly one a .

- (a) Draw a state-transition graph of a finite automaton that accepts the language:

$$L_1 \cap L_2$$

If such an automaton does not exist, state it and explain why.

Answer: See Figure 167.

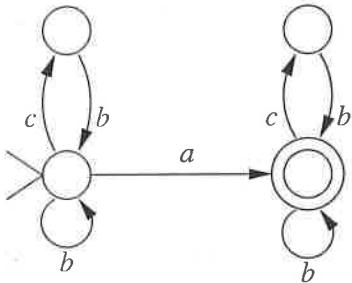


Figure 167:

- (b) Write a complete formal definition of a context-free grammar that generates the language

$$(L_1 L_2)^*$$

If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow \lambda \mid SS \mid AB \\ A &\rightarrow \lambda \mid AA \mid a \mid b \mid cb \\ B &\rightarrow DaD \\ D &\rightarrow \lambda \mid DD \mid b \mid c \end{aligned}$$

Problem 562 Let L_1 be the set of all strings over the alphabet $\{a, b, c\}$ where the number of a 's minus the number of b 's is divisible by 3.

Let L_2 be the set of all strings over the alphabet $\{a, b, c\}$ which end with b .

- (a) Draw a state-transition graph of a finite automaton that accepts the language L_1 . If such an automaton does not exist, state it and explain why.

Answer: See Figure 168.

- (b) Draw a state-transition graph of a finite automaton that accepts the language L_2 . If such an automaton does not exist, state it and explain why.

Answer: See Figure 169.

- (c) Draw a state-transition graph of a finite automaton that accepts the language $L_1 \cup L_2$. If such an automaton does not exist, state it and explain why.

Answer: See Figure 170.

- (d) Draw a state-transition graph of a finite automaton that accepts the language $L_1 \cap L_2$. If such an automaton does not exist, state it and explain why.

Answer: See Figure 171.

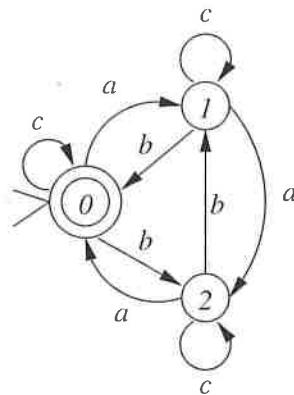


Figure 168:

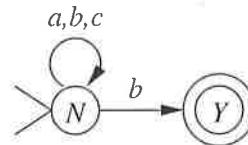


Figure 169:

Problem 563 Let L_1 be the set of all strings over the alphabet $\{a, b, c\}$ that begin with b .

Let L_2 be the set of all strings over the alphabet $\{a, b, c\}$ that contain an odd number of c 's.

Construct a finite automaton that accept the language $L_1 \cap L_2$. If such an automaton does not exist, prove it.

Answer: See Figure 172.

Problem 564 Let L be the set of all strings over alphabet $\{a, b, c\}$ that contain a substring cc .

Let L_1 be the set of all strings over alphabet $\{a, b, c\}$ that have an odd length.

- (a) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 173.

- (b) Construct a state-transition graph of a finite automaton that accepts $L \cap L_1$. If such an automaton does not exist, prove it.

Answer: See Figure 174.

Problem 565 Let L be the set of all strings over alphabet $\{a, b, c\}$ that contain at least one a .

Let L_1 be the set of all strings over alphabet $\{a, b, c\}$ that contain at least one b .

- (a) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 175.

- (b) Construct a state-transition graph of a finite automaton that accepts $L \cap L_1$. If such an automaton does not exist, prove it.

Answer: See Figure 176.

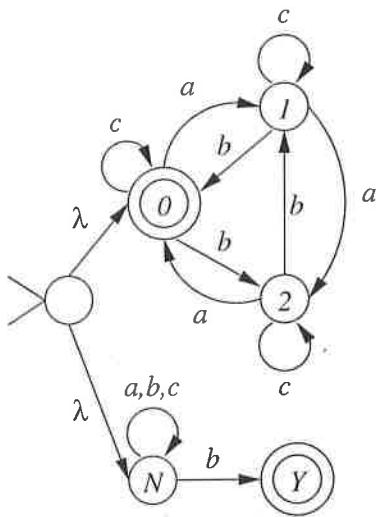


Figure 170:

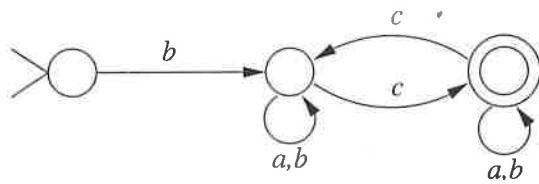


Figure 172:

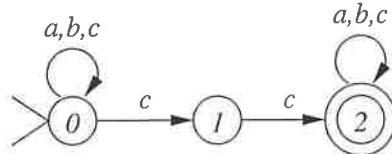


Figure 173:

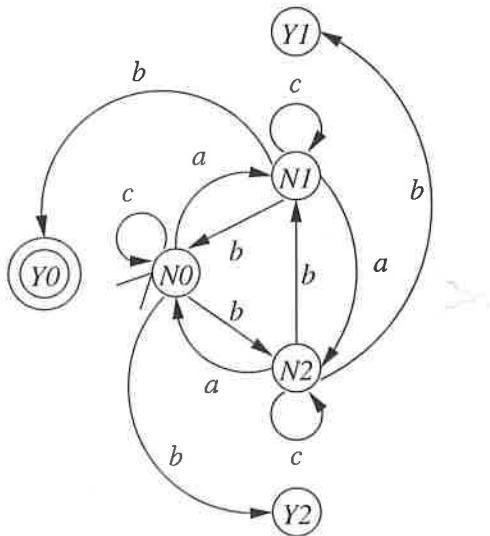


Figure 171:

Problem 566 Let L be the set of all strings over the alphabet $\{a, b, c, d, e\}$ that contain exactly two a 's and exactly two b 's.

Draw a state-transition graph of a finite automaton M that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 177.

Problem 567 Let L_1 be the set of all strings over the alphabet $\{a, b, c\}$ that end with b .

Let L_2 be the set of all strings over the alphabet $\{a, b, c\}$ that contain an odd number of a 's.

Draw a state-transition graph of a finite automaton that accepts the language $L_1 \cap L_2$. If such an automaton does not exist, prove it.

Answer: See Figure 178 for a deterministic, product construction, and Figure 179 for an equivalent, more compact but non-deterministic one.

Problem 568 Let L_1 be the set of all strings over the alphabet $\{a, b, c\}$ that contain exactly one b .

Let L_2 be the set of all strings over the alphabet $\{a, b, c\}$ that contain an even number of a 's.

Draw a state-transition graph of a finite automaton that accepts the language $L_1 \cap L_2$. If such an automaton does not exist, prove it.

Answer: See Figure 180.

Problem 569 Let L_1 be the set of all strings over the alphabet $\{a, b, c\}$ that contain at least one b .

Let L_2 be the set of all strings over the alphabet $\{a, b, c\}$ that have odd length.

(a) Write a complete formal definition of a context-free grammar that generates the language $L_1 \cup L_2$. If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow B \mid D \\ B &\rightarrow AbA \\ A &\rightarrow \lambda \mid AA \mid Z \\ D &\rightarrow ZZD \mid Z \\ Z &\rightarrow a \mid b \mid c \end{aligned}$$

(b) Draw a state-transition graph of a finite automaton that accepts the language $L_1 \cap L_2$. If such an automaton does not exist, prove it.

Answer: See Figure 181.

Problem 570 (a) Construct a state-transition graph of a finite automaton that accepts the set of strings over

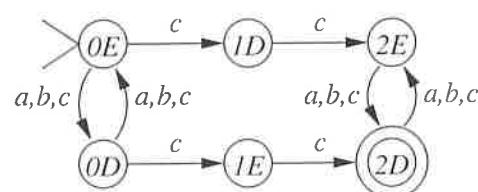


Figure 174:

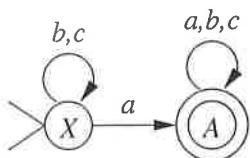


Figure 175:

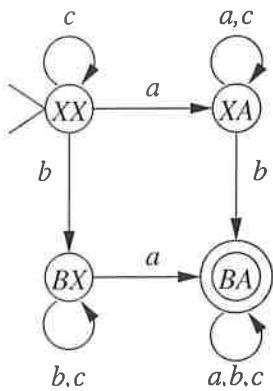


Figure 176:

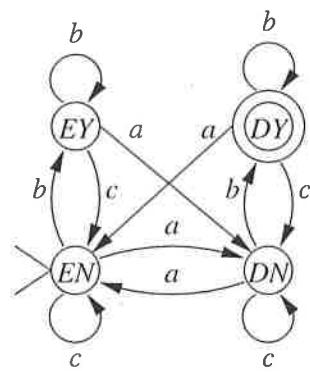


Figure 178:

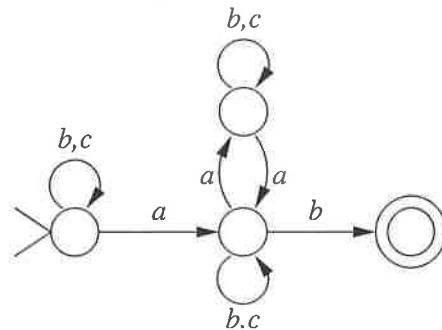


Figure 179:

$\{a, b, c\}$ that contain an even number of a 's or at least two c 's. If such an automaton does not exist, prove it.

Answer: See Figure 182.

(b) Construct a state-transition graph of a finite automaton that accepts the set of strings over $\{a, b, c\}$ that contain an even number of a 's and at least two c 's. If such an automaton does not exist, prove it.

Answer: See Figure 183.

Problem 571 Let L_1 be the set of strings over alphabet $\{a, b\}$ which contain at most two b 's. Let L_2 be the set of strings over alphabet $\{a, b\}$ which have odd length.

(a) Draw a state-transition graph of a finite automaton that accepts L_1 . If such automaton does not exist, prove it.

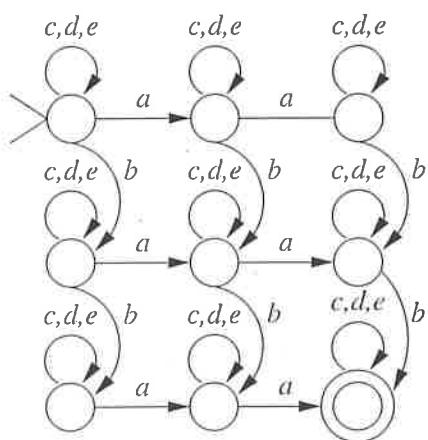


Figure 177:

Answer: See Figure 184.

(b) Draw a state-transition graph of a finite automaton that accepts L_2 . If such automaton does not exist, prove it.

Answer: See Figure 185.

(c) Draw a state-transition graph of a finite automaton that accepts $L_1 \cup L_2$. If such automaton does not exist, prove it.

Answer: The algorithmic construction based on the closure properties of regular languages, applied to automata obtained in the answers to parts (a) and (b), gives the automaton represented on Figure 186.

(d) Draw a state-transition graph of a finite automaton that accepts $L_1 \cap L_2$. If such automaton does not exist,

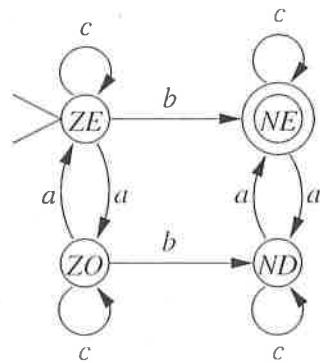


Figure 180:

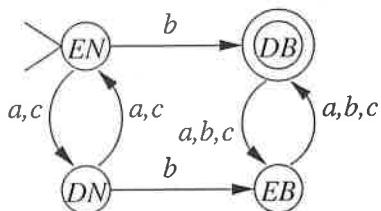


Figure 181:

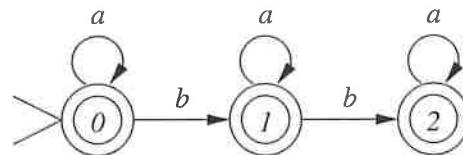


Figure 184:

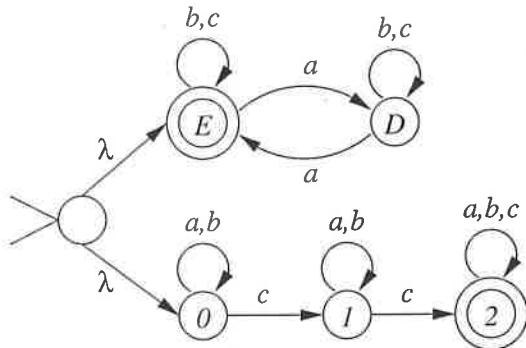


Figure 182:

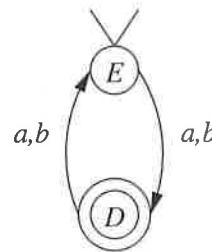


Figure 185:

- (a) Draw a state-transition graph of a finite automaton that accepts \bar{L} (the complement of L). If such automaton does not exist, prove it.

Answer: See Figure 189.

- (b) Does there exist an algorithm that solves the following problem:

INPUT: An arbitrary finite automaton M .

OUTPUT: A finite automaton M_1 that accepts the complement of the language accepted by M .

Explain your answer briefly.

Answer: Yes. This algorithm starts out by converting M into an equivalent deterministic automaton M' . To obtain M_1 from M' , make every final state of M' non-final, and make every non-final state of M' final.

- Problem 574** Let L be the set of all strings over $\{a, b\}$ in which the number of a 's is not divisible by 3 or the number of b 's is not divisible by 2. Draw a state-transition graph of a finite automaton that accepts L . If

prove it.

Answer: The algorithmic construction based on the simultaneous simulation, applied to automata obtained in the answers to parts (a) and (b), gives the automaton represented on Figure 187.

Problem 572 Let L_1 be the set of all strings over alphabet $\{a, b\}$ that contain at least two b 's. Let L_2 be the set of all strings over alphabet $\{a, b\}$ that have even length.

Draw a state-transition graph of a finite automaton M that accepts:

$$L_1 \cap L_2$$

If such an automaton does not exist, prove it.

Answer: See Figure 188.

Problem 573 Let L be the language defined by the regular expression:

$$(a \cup c)^* b^*$$

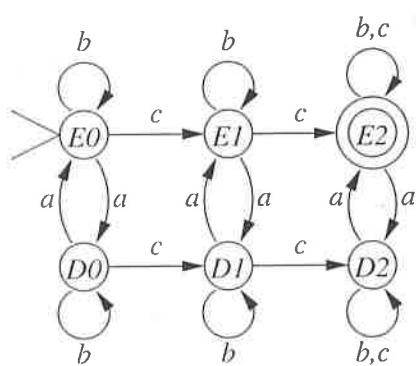


Figure 183:

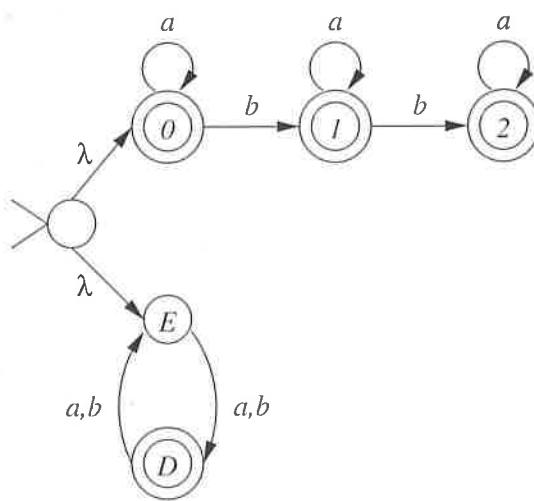


Figure 186:

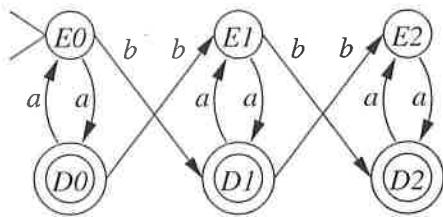


Figure 187:

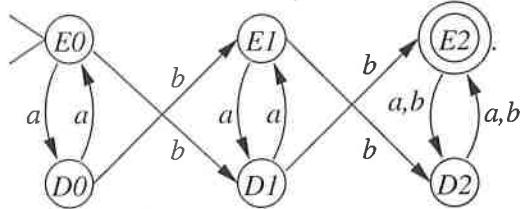


Figure 188:

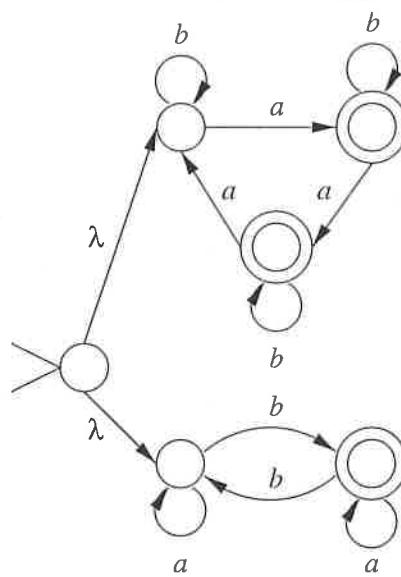


Figure 190:

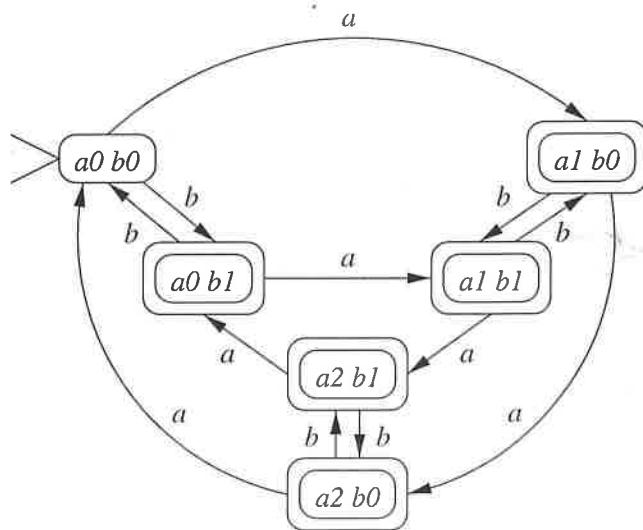


Figure 191:

such automaton does not exist, prove it.

Answer: See Figure 190 for a straightforward non-deterministic construction, and Figure 191 for a deterministic one.

Problem 575 (a) Let L_1 be the set of all strings over $\{a, b\}$ in which the number of a 's is divisible by 2 or the number of b 's is divisible by 3.

Construct a state-transition graph of a finite automaton that accepts L_1 . If such an automaton does not exist, prove it.

Answer: See Figure 192 for a straightforward non-deterministic construction, and Figure 193 for a deterministic one.

(b) Let L_2 be the set of all strings over $\{a, b\}$ in which the number of a 's is divisible by 2 and the number of b 's is divisible by 3.

Construct a state-transition graph of a finite automaton that accepts L_2 . If such an automaton does not exist, prove it.

Answer: See Figure 193.

Problem 576 Let L_1 be the set of strings over alphabet $\{a, b\}$ that contain an even number of b 's. Let L_2 be the set of strings over alphabet $\{a, b\}$ that contain aab as a substring.

(a) Draw a state-transition graph of a finite automaton that accepts L_1 . If such automaton does not exist,

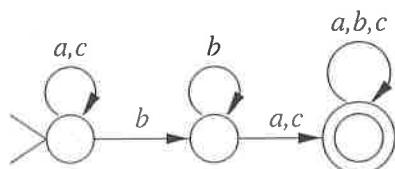


Figure 189:

prove it.

Answer: See Figure 194.

(b) Draw a state-transition graph of a finite automaton that accepts L_2 . If such automaton does not exist, prove it.

Answer: See Figure 195.

(c) Draw a state-transition graph of a finite automaton that accepts $L_1 \cup L_2$. If such automaton does not exist, prove it.

Answer: See Figure 196.

(d) Draw a state-transition graph of a finite automaton that accepts $L_1 \cap L_2$. If such automaton does not exist, prove it.

Answer: See Figure 197.

Problem 577 (a) Let L_1 be the set of all non-

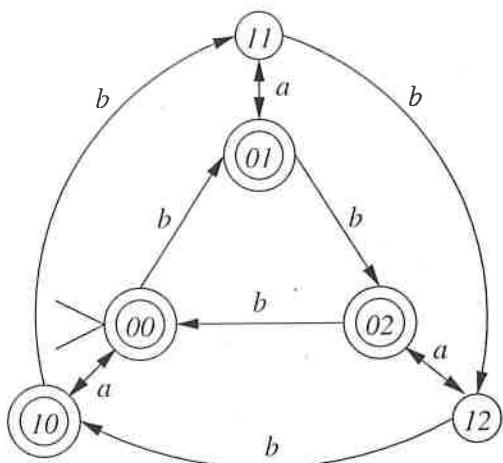


Figure 192:

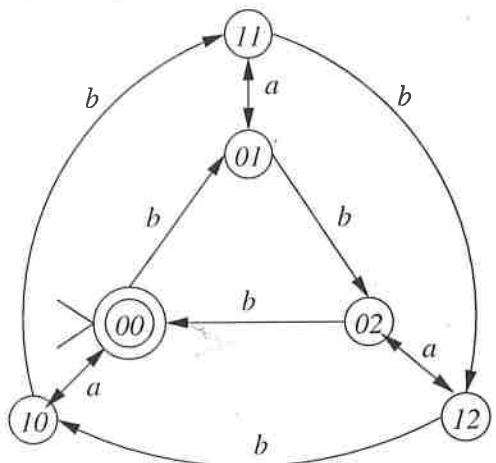


Figure 193:

empty strings over the alphabet $\{a, b, c\}$ that do not begin with a .

Draw a state-transition graph of a finite automaton that accepts the language L_1 . If such an automaton does not exist, state it and explain why.

Answer: See Figure 198.

(b) Let L_2 be the set of all non-empty strings over the alphabet $\{a, b, c\}$ that do not end with c .

Draw a state-transition graph of a finite automaton that accepts the language L_2 . If such an automaton does not exist, state it and explain why.

Answer: See Figure 199.

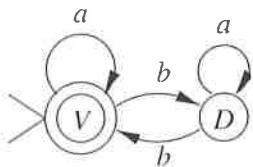


Figure 194:

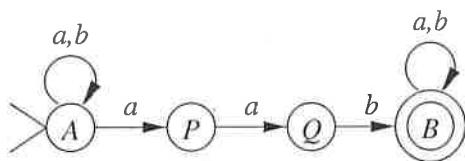


Figure 195:

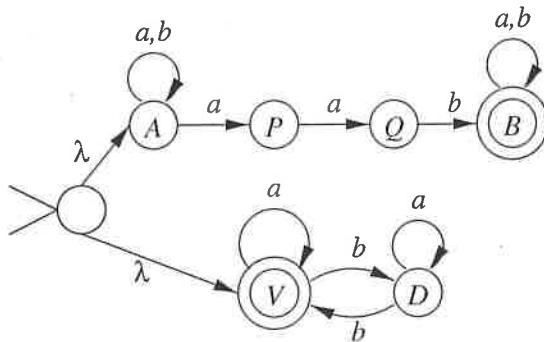


Figure 196:

(c) Let L_3 be the set of all non-empty strings over the alphabet $\{a, b, c\}$ that neither begin with a nor end with c . Draw a state-transition graph of a finite automaton that accepts the language L_3 . If such an automaton does not exist, state it and explain why.

Answer: Since $L_3 = L_1 \cap L_2$, the required automaton, given of Figure 200, can be obtained by algorithmic conversion of the two automata produced in the answers to parts (a) and (b).

Problem 578 Let L_1 be the set of all strings over the alphabet $\{a, b, c\}$ that begin with a , end with b , and contain at least one c .

Let L_2 be the set of all strings over the alphabet $\{a, b, c\}$ whose length is less than or equal to 4.

(a) Draw a state-transition graph of a finite automaton that accepts the language L_1 . If such an automaton does not exist, state it and explain why.

Answer: See Figure 201.

(b) Draw a state-transition graph of a finite automaton

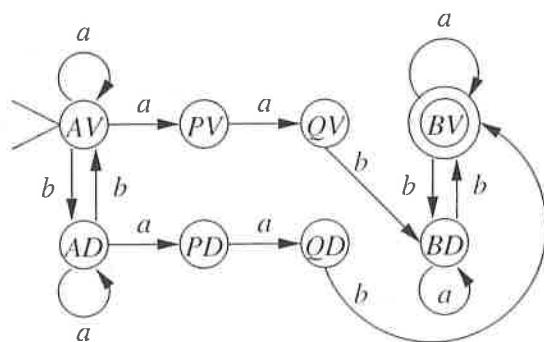


Figure 197:

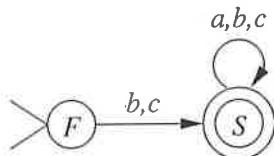


Figure 198:

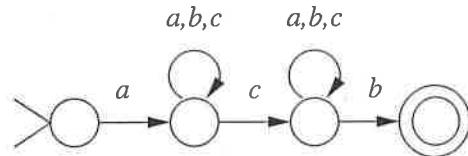


Figure 201:

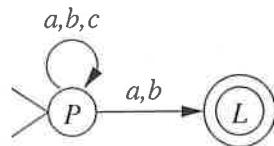


Figure 199:

that accepts the language L_1^* . If such an automaton does not exist, state it and explain why.

Answer: See Figure 202.

(c) Write a complete formal definition of a context-free grammar that generates the language $L_1 \cup L_2$. If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow A \mid B \\ A &\rightarrow aDcDb \\ D &\rightarrow \lambda \mid DD \mid a \mid b \mid c \\ B &\rightarrow ZZZZ \\ Z &\rightarrow a \mid b \mid c \mid \lambda \end{aligned}$$

(d) Write a complete formal definition of a context-free grammar that generates the language $L_1 \cap L_2$. If such a grammar does not exist, state it and explain why.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, Z\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aZcb \mid acZb \\ Z &\rightarrow a \mid b \mid c \mid \lambda \end{aligned}$$

Problem 579 Let L_1 be the set of strings over alphabet $\{a, b, c\}$ that contain at least one b . Let L_2 be the set of strings over alphabet $\{a, b, c\}$ whose length gives remainder 2 when divided by 3.

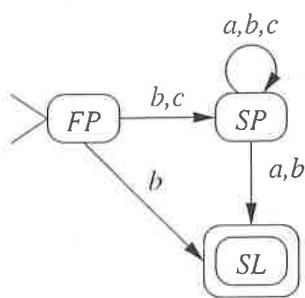


Figure 200:

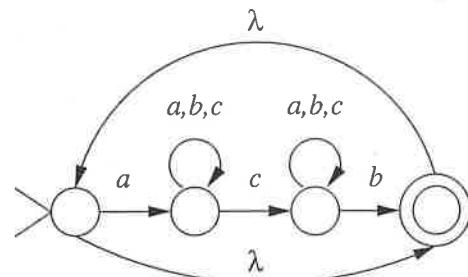


Figure 202:

(a) Draw a state-transition graph of a finite automaton that accepts L_1 . If such automaton does not exist, prove it.

Answer: See Figure 203.

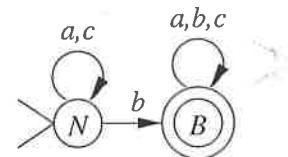


Figure 203:

(b) Draw a state-transition graph of a finite automaton that accepts L_2 . If such automaton does not exist, prove it.

Answer: See Figure 204.

(c) Draw a state-transition graph of a finite automaton that accepts $L_1 \cup L_2$. If such automaton does not exist, prove it.

Answer: See Figure 205.

(d) Draw a state-transition graph of a finite automaton that accepts $L_1 \cap L_2$. If such automaton does not exist, prove it.

Answer: See Figure 206.

Problem 580 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b\}$, $V = \{S\}$, and P is:

$$S \rightarrow aSbb \mid \lambda$$

(a) Write a complete formal definition of a context-free grammar that generates the language L^* . If such a grammar does not exist, prove it.

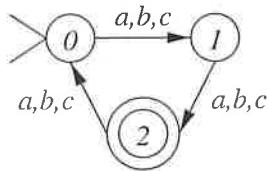


Figure 204:

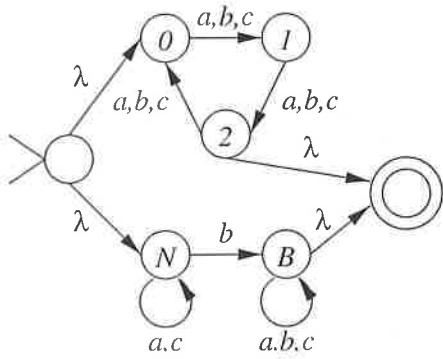


Figure 205:

Answer: $G = (V, \Sigma, P, T)$, where:
 $\Sigma = \{a, b, c\}$, $V = \{T, S\}$, and P is:

$$\begin{aligned} T &\rightarrow TT \mid \lambda \mid S \\ S &\rightarrow aSbb \mid \lambda \end{aligned}$$

(b) Draw a state-transition graph of a finite automaton that accepts the language L^* . If such an automaton does not exist, prove it.

Answer: The required automaton does not exist, since L^* is not regular.

Observe that the general template for the strings that belong to L is as follows.

$$L = \{a^n b^{2n} \mid n \geq 0\}$$

This means that each string that belongs to L^* consists of alternating segments of a 's and b 's, where the length of each b -segment is equal to twice the length of the a -segment that immediately precedes it.

To prove that L^* is not regular, assume the opposite—that L^* is regular.

Let k be the constant as in the Pumping Lemma for L^* . Consider a word $w \in L$ (which implies that $w \in L^*$),

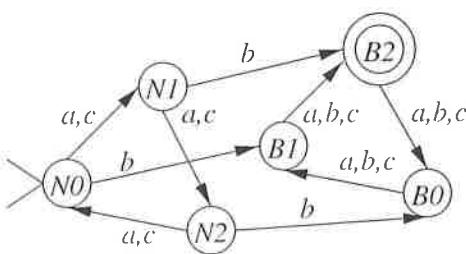


Figure 206:

such that $w = a^m b^{2m}$, where $m > k$. Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < k < m$. Hence, the pumping part v is entirely within the a -segment, meaning that $v = a^j$ for some j such that $0 < j < k$. By the pumping, for every $i \geq 0$, we have: $uv^i x \in L$. This means that all the words produced by the following template belong to L .

$$uv^i x = uvv^{i-1} x = uva^{(i-1)j} x = a^{m+(i-1)j} b^{2m}$$

Consider the word:

$$w_2 = a^{m+j} b^{2m}$$

obtained by setting $i = 2$ in this template. Since $j > 0$ (as the length of the non-empty pumping part) we see that:

$$2m < 2m + 2j = 2(m + j)$$

meaning that the length of the b -segment of w_2 is less than twice the length of the preceding a -segment. This means that:

$$w_2 \notin L^*$$

whence the contradiction.

Problem 581 Let L be a language over the alphabet $\{a, b, c\}$, defined as follows:

$$L = \{a^\ell b^n c^j b^m a^p \mid \ell, n, m \geq 0 \wedge p > 1 \wedge j = 0 \wedge n = 2m\}$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The general template for strings in L is:

$$a^\ell b^{2m} b^m a^p, \text{ where } \ell, m \geq 0, p > 1$$

or, equivalently:

$$a^\ell b^{3m} a^{k+2}, \text{ where } \ell, m, k \geq 0$$

which is represented by the regular expression: $a^*(bbb)^*a^*aa$, which yields the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow ABAAa \\ A &\rightarrow aA \mid \lambda \\ B &\rightarrow bbbB \mid \lambda \end{aligned}$$

(b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 207.

Problem 582 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow A \mid BD \\ A &\rightarrow Aa \mid b \\ B &\rightarrow c \mid cc \\ D &\rightarrow DD \mid \lambda \mid bb \end{aligned}$$

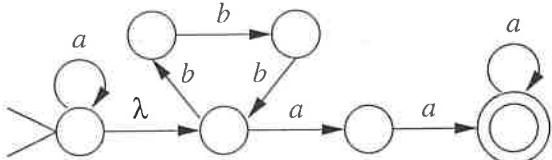


Figure 207:

- (a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$ba^* \cup (c \cup cc)(bb)^*$$

- (b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 208.

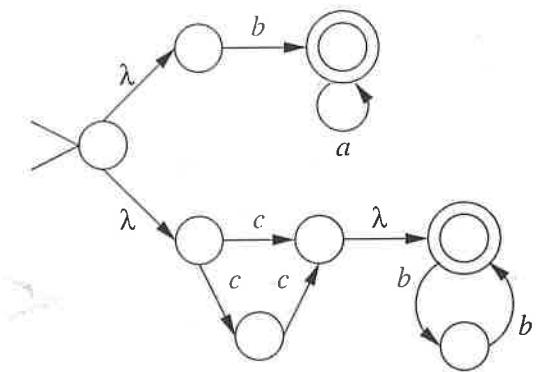


Figure 208:

- Problem 583** Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow ABD \\ A &\rightarrow aAa \mid b \\ B &\rightarrow c \mid cc \\ D &\rightarrow \lambda \mid bb \end{aligned}$$

- (a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: This regular expression does not exist, since L is not regular. To prove this, observe that the general template for strings in L is:

$$L = a^nba^ncc^m(bb)^\ell \text{ where } n \geq 0, 1 \leq m \leq 2, 0 \leq \ell \leq 1$$

Assume the opposite—that L is regular. Let k be the constant as in the Pumping Lemma for L . Consider a word $w = a^nba^ncc^m$, where $n > k$, obtained from the general template for L by setting $m = 1, \ell = 0$. Observe that every word in L that does not end in b contains two substrings of equal length containing a 's, separated by a single b . Let $w = uvw$ be a decomposition of w , where v is the pumping part. By the Lemma, it must

be that $|uv| < k < n$. Hence, the pumping part v is entirely within the first a -segment, meaning that $v = a^j$ for some j such that $0 < j < k$. After pumping once, we obtain a word: $w_1 = a^{n+j}ba^nc$. However, since $j > 0$ (as the length of the pumping substring), we conclude that $m + j > m$, meaning that the two a -segments in w_1 do not have equal length. Hence, $w_1 \notin L$, whence the contradiction.

- (b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: This language is not regular, as is proved in the answer to part (a). Since it is not regular, there does not exist a finite automaton that accepts it.

- Problem 584** (a) Let L_1 be the language generated by the context-free grammar $G_1 = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow A \mid B \\ A &\rightarrow DAD \mid aa \\ B &\rightarrow DBD \mid cc \\ D &\rightarrow bd \end{aligned}$$

Write a regular expression that defines L_1 . If such a regular expression does not exist, prove it.

Answer: This regular expression does not exist, since L_1 is not regular. To prove this, observe that the general template for strings in L is:

$$(bd)^n \xi \xi (bd)^n \text{ where } n \geq 0 \wedge \xi \in \{aa, cc\}$$

In other words, every word of L_1 contains a substring aa or cc , with an equal number of occurrences of bd on either side of it.

Assume the opposite—that L_1 is regular. Let k be the constant as in the Pumping Lemma for L_1 . Consider a word $w = (bd)^m aa (bd)^m$, where $m > k$.

Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < k < m$. Hence, the pumping part v is entirely within the bd -segment, meaning that $|v| = j$ for some j such that $0 < j < k$, and w does not contain any a 's. After pumping once, we obtain a word: $w_1 = xaa(bd)^{m-j}$, where $|x| = 2m + j > 2m$, and x does not contain any a 's. Hence, there are exactly $2m + j$ b 's or d 's to the left of the substring aa , and exactly $2m$ to the right of it. This means that w_1 does not contain equal numbers of occurrences of bd at the two sides of aa . Since all words of L_1 that do not contain c have to be in this form, we conclude that $w_1 \notin L_1$, whence the contradiction.

- (b) Let L_2 be the language generated by the context-free grammar $G_2 = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow aA \mid bB \mid dD \\ A &\rightarrow dA \mid a \\ B &\rightarrow cB \mid b \\ D &\rightarrow aD \mid \lambda \end{aligned}$$

Construct a state-transition graph of a finite automaton that accepts L_2 . If such an automaton does not exist, prove it.

Answer: This grammar is regular, hence the construction of the automaton is algorithmic. See Figure 209.

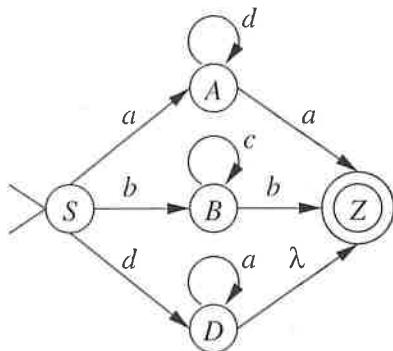


Figure 209:

Problem 585 Let L be a language over the alphabet $\{a, b, c, d\}$, defined as follows:

$$L = \{a^\ell b^k c^j d^m a^p \mid j, k, \ell, m, p \geq 0 \wedge \ell = j = p\}$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The required grammar does not exist, since L is not context-free. To prove this, assume the opposite, and let ξ be the constant as in the Pumping Lemma for L . Observe that the template for all words of L that contain neither b 's nor d 's is: $w = a^p c^p a^p$, $p \geq 0$, and is obtained by setting $k = m = 0$ in the general template for L . Select a word $w = a^p c^p a^p$, $p > \xi$. In any “pumping” decomposition: $w = uvxyz$, the length of the “pumping window” vxy is not greater than ξ : $|vxy| \leq \xi < p$. The “pumping” window vxy either falls within one of the three substrings containing a single letter: a^p , c^p , a^p , or it spans two such substrings—it is too short to extend through as many as three of them. In all cases, it omits at least one of the letters, so that pumping results in the deficit of that letter with respect to others.

(b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: Such a finite automaton does not exist, because L is not regular. If L was regular, then L would be context-free, since every regular language is context-free. However, we know from the answer to part (a) that L is not context-free. Hence, L is not regular either.

Problem 586 Let L be a language over the alphabet $\{a, b, c, d\}$, defined as follows:

$$L = \{a^\ell b^{2k} c^{3j+1} d^{m+2} a^{2p} \mid j, k, \ell, m, p \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: Observe that L is given by the regular expression:

$$a^*(bb)^*c(ccc)^*dd\ d^*(aa)^*$$

whence the answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, K, D, H\}$, and P is:

$$\begin{aligned} S &\rightarrow ABcKddDH \\ A &\rightarrow aA \mid \lambda \\ B &\rightarrow bbB \mid \lambda \\ K &\rightarrow cccK \mid \lambda \\ D &\rightarrow dD \mid \lambda \\ H &\rightarrow aaH \mid \lambda \end{aligned}$$

(b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 210.

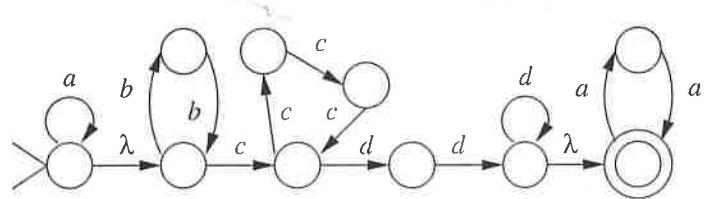


Figure 210:

Problem 587 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow BA \mid BD \mid AD \\ A &\rightarrow aaA \mid bbA \mid \lambda \\ B &\rightarrow dbaB \mid \lambda \\ D &\rightarrow DD \mid \lambda \mid cd \end{aligned}$$

(a) Construct a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 211.

(b) Is L a regular language? Prove your answer.

Answer: Yes—the finite automation which accepts L is given in the answer to the previous part.

Problem 588 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where

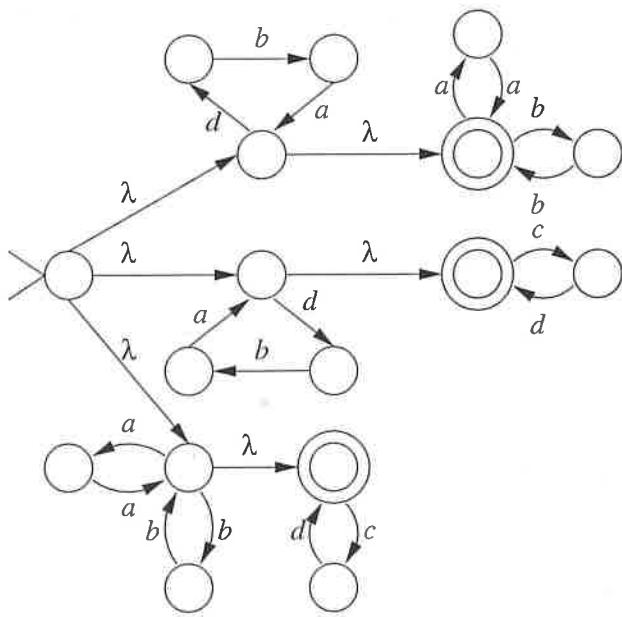


Figure 211:

$\Sigma = \{a, b, c\}$, $V = \{S, A, B\}$,
and the production set P is:

$$\begin{aligned} S &\rightarrow BAB \mid ABA \\ A &\rightarrow aA \mid bA \mid \lambda \\ B &\rightarrow ccB \mid \lambda \end{aligned}$$

(a) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 212.

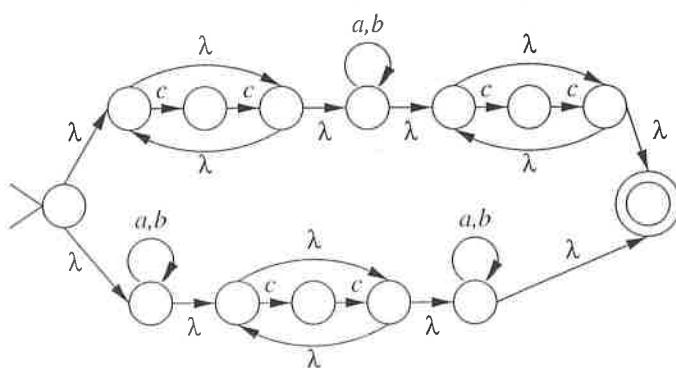


Figure 212:

(b) Does there exist an algorithm to convert an arbitrary context-free grammar into an equivalent regular expression? Explain your answer briefly.

Answer: No—some context-free languages are not regular. In fact, it is undecidable for an arbitrary context-free grammar whether the language derived by it is regular.

Problem 589 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b, c\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow A \mid B \\ A &\rightarrow aDb \\ D &\rightarrow DD \mid \lambda \mid c \\ B &\rightarrow bbB \mid \lambda \end{aligned}$$

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$ac^*b \cup (bb)^*$$

(b) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 213.

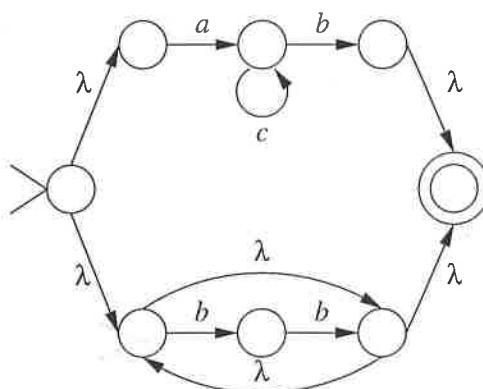


Figure 213:

Problem 590 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AcBcAcB \\ A &\rightarrow aA \mid \lambda \\ B &\rightarrow abbB \mid \lambda \end{aligned}$$

(a) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 214.

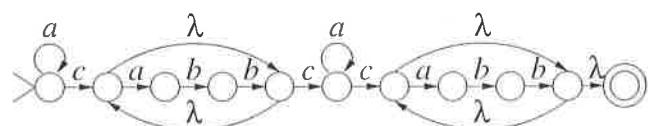


Figure 214:

(b) Is L regular? Explain your answer briefly.

Answer: Yes— L is accepted by the finite automaton constructed in the answer to part (a).

Problem 591 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, B, D, E\}$, and P comprises:

$$\begin{aligned} S &\rightarrow aS \mid aB \\ B &\rightarrow bB \mid bD \\ D &\rightarrow cD \mid a \end{aligned}$$

Write a complete formal definition or a state-transition graph of a finite automaton M' that accepts L . If such automaton does not exist, prove it.

Answer: Observe that G is a regular grammar. The state diagram of M' is given on Figure 215.

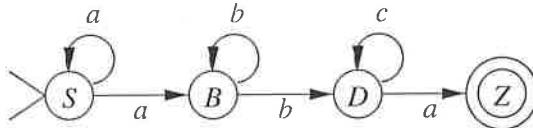


Figure 215:

Problem 592 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B\}$, and P comprises:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aA \mid \epsilon \\ B &\rightarrow bB \mid Bc \mid bc \end{aligned}$$

(a) Write 10 distinct strings that belong to L . If such strings do not exist, explain why.

(b) Write 10 distinct strings over alphabet $\{a, b, c\}$ that do not belong to L . If such strings do not exist, explain why.

Answer:

$\in L$	$\notin L$
bac	a
bcc	b
$bbbc$	c
$bccc$	ab
bc	ac
abc	cb
$abbc$	ca
$abbcc$	ba
$aabbcc$	bca
$aaabcccc$	bac

(c) Write a complete formal definition or a state-transition graph of a finite automaton M that accepts L . If such automaton M does not exist, explain why.

Answer: The language generated by grammar G is defined by the regular expression:

$$a^*b^*bcc^*$$

and accepted by the finite automaton whose state-transition graph is given on Figure 216.

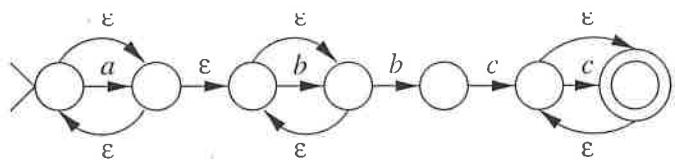


Figure 216:

Problem 593 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$ is the set of terminals; $V = \{S, A, B, C\}$ is the set of variables; S is the start symbol; and the set of productions P comprises:

$$\begin{aligned} S &\rightarrow ABC \\ A &\rightarrow aA \mid a \\ B &\rightarrow b \\ C &\rightarrow cC \mid \lambda \end{aligned}$$

(a) Write a complete formal definition or a state-transition graph of a finite automaton M that accepts L .

Answer: See figure Figure 217. Note that the grammar

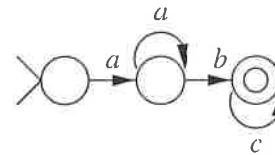


Figure 217:

G generates the language defined by regular expression:

$$aa^*bc^*$$

(b) Is the finite automaton M which you constructed in your answer to part (a) deterministic? Justify briefly your answer.

Answer: No. While the alphabet has 3 symbols, each node in the transition graph has out-degree less than 3, showing that the next state is undefined for some state-symbol pairs. Thus, the transition function is not total.

Problem 594 (a) Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$ is the set of terminals; $V = \{S, A, B, D, E, F, H, J\}$ is the set of variables; S is the start symbol; and the set of productions P comprises:

$$\begin{aligned} S &\rightarrow A \mid B \mid D \\ A &\rightarrow aA \mid E \\ E &\rightarrow b \mid c \\ B &\rightarrow bF \mid cF \mid \lambda \\ F &\rightarrow aB \\ D &\rightarrow bH \\ H &\rightarrow cJ \\ J &\rightarrow b \end{aligned}$$

Write a complete formal definition or a state-transition graph of a finite automaton M that accepts L . If such an automaton does not exist, explain why.

Advice for Answer: L is regular.

- (b) Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$ is the set of terminals; $V = \{S\}$ is the set of variables; S is the start symbol; and the set of productions P comprises:

$$S \rightarrow SS \mid a \mid ba \mid bba \mid \lambda$$

Write a complete formal definition or a state-transition graph of a finite automaton M that accepts L . If such automaton does not exist, explain why.

Advice for Answer: L is regular.

- Problem 595** Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B\}$, and P comprises:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aA \mid \epsilon \\ B &\rightarrow bBc \mid bc \end{aligned}$$

- (a) Write 10 distinct strings that belong to L . If such strings do not exist, explain why.
 (b) Write 10 distinct strings over alphabet $\{a, b, c\}$ that do not belong to L . If such strings do not exist, explain why.

Answer:

$\in L$	$\notin L$
bc	a
$bbcc$	b
$bbbccc$	c
abc	$abbc$
$abcc$	$abcc$
$aaabc$	cb
$aabbcc$	cba
$aaaaabbc$	bcc
$aaabbccc$	bca
$aaabbbcccc$	bac

- (c) Write a complete formal definition or a state-transition graph of a finite automaton M that accepts L . If such automaton M does not exist, explain why.

Answer: The language L , generated by grammar G is:

$$L = \{a^m b^n c^n \mid m \geq 0, n > 0\}$$

There is no finite automaton that accepts this language. To prove this, assume the opposite, that L is regular. By the Pumping Lemma, there exists a constant ℓ such that every word $w \in L$ such that $|w| \geq \ell$ can be written as

$$w = xyz$$

where:

$$\begin{aligned} |y| &> 0 \\ |xy| &\leq \ell \end{aligned}$$

and every word of the form:

$$xy^i z, i \geq 0$$

also belongs to L . Consider a word $b^n c^n \in L$, where n is chosen so that $n > \ell$. By the Pumping Lemma, there exist x, y, z such that:

$$b^n c^n = xyz$$

where y consists solely of symbols b :

$$y = b^k, k > 0$$

and also:

$$xy^i z \in L, i > 0$$

meaning:

$$b^{n+(i-1)k} c^n \in L$$

which is impossible by the definition of L whenever $i > 1$.

- Problem 596** Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, B, D, E\}$, and P comprises:

$$\begin{aligned} S &\rightarrow BD \mid E \\ B &\rightarrow BB \mid ab \mid \lambda \\ D &\rightarrow DD \mid c \mid \lambda \\ E &\rightarrow EE \mid b \mid \lambda \end{aligned}$$

- (a) Write a regular expression that defines L . If such regular expression does not exist, explain why.

Answer:

$$(ab)^* c^* \cup b^*$$

- (b) Write a complete formal definition or a state-transition graph of a finite automaton M that accepts L . If such automaton does not exist, explain why.

Answer: The state transition graph of the finite automaton that accepts L is given on Figure 218.

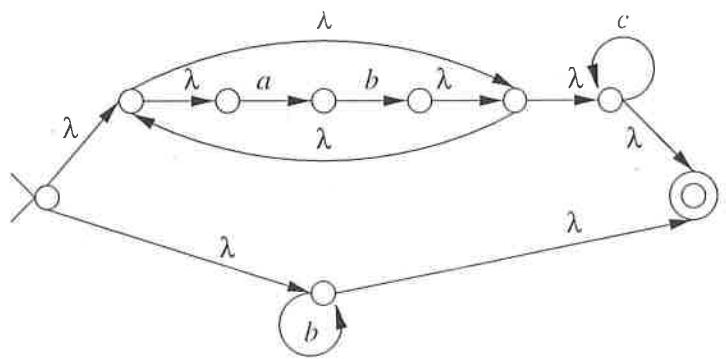


Figure 218:

- Problem 597** (a) Let:

$$\Sigma = \{a, b, c\}$$

and let L_1 be the set of all strings over Σ defined by the following regular expression:

$$(cb^*a)^*$$

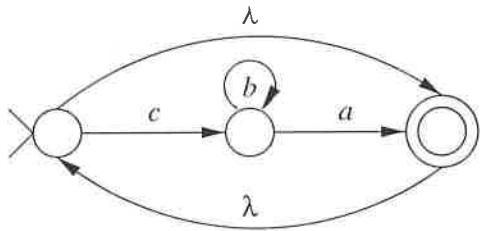


Figure 219:

Construct a state-transition graph of a finite automaton M_1 that accepts L_1 . If such automaton does not exist, prove it.

Answer: See Figure 219.

(b) Let L_2 be the set of all strings generated by the following grammar:

$$G = (V, \Sigma, P, S)$$

where: $\Sigma = \{a, b, c\}$, $V = \{S, A, E, O\}$, and P is:

$$\begin{aligned} S &\rightarrow EaO \mid OaE \\ E &\rightarrow EE \mid \lambda \mid AA \\ O &\rightarrow AE \\ A &\rightarrow b \mid c \end{aligned}$$

Construct a state-transition graph of a finite automaton M_2 that accepts L_2 . If such automaton does not exist, prove it.

Answer: See Figure 220.

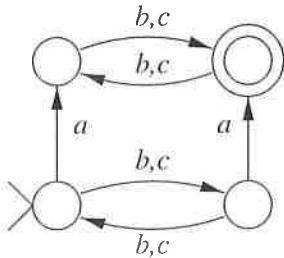


Figure 220:

Problem 598 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aSbbb \mid A \\ A &\rightarrow a \end{aligned}$$

Write a complete formal definition or a state-transition graph of a finite automaton M that accepts L . If such automaton does not exist, prove it.

Advice for Answer: The general template for strings of L is:

$$a^{n+1}b^{3n}$$

and L is not regular.

Problem 599 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aA \mid bB \\ A &\rightarrow aS \mid bD \\ B &\rightarrow bS \mid aD \mid \lambda \\ D &\rightarrow aB \mid bA \end{aligned}$$

Write a complete formal definition or a state-transition graph of a finite automaton M that accepts L . If such automaton does not exist, prove it.

Advice for Answer: G is a regular grammar, and thereby algorithmically convertible to a finite automaton.

Problem 600 (a) Let L_1 be the language generated by the context-free grammar $G_1 = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, P, Q, R\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow PQQR \\ A &\rightarrow aA \mid cA \mid \lambda \\ P &\rightarrow aA \\ Q &\rightarrow bA \\ R &\rightarrow c \end{aligned}$$

Write a complete formal definition or a state-transition graph of a finite automaton M_1 that accepts L_1 . If such automaton does not exist, prove it.

Advice for Answer: L_1 is represented by the following regular expression:

$$a(a \cup c)^* b(a \cup c)^* b(a \cup c)^* c$$

(b) Let L_2 be the language generated by the context-free grammar $G_2 = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, P, Q, R\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow PQR \\ P &\rightarrow a \mid \lambda \\ Q &\rightarrow bQa \mid \lambda \\ R &\rightarrow c \end{aligned}$$

Write a complete formal definition or a state-transition graph of a finite automaton M_2 that accepts L_2 . If such automaton does not exist, prove it.

Advice for Answer: The general template for strings of L_2 is:

$$(a \cup \lambda) b^n a^n c$$

and L_2 is not regular.

Problem 601 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aaAbbb \mid \lambda \\ B &\rightarrow aB \mid \lambda \end{aligned}$$

Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: Such a finite automaton does not exist, since L is not a regular language. By inspection of the grammar, we conclude that:

$$L = \{a^{2n}b^{3n}a^n \mid m, n \geq 0\}$$

To prove that L is not regular, assume the opposite. Let k be the constant as in the Pumping Lemma. Let $n > k$; then $a^{2n}b^{3n} \in L$. In the “pumping” decomposition: $a^{2n}b^{3n} = uvx$, we have that $|uv| \leq k < n < 2n$, hence the “pumping” substring v consists entirely of a 's, say $v = a^j$. Recall that $j > 0$, since the “pumping” substring cannot be empty. By the pumping, every word of the form uv^ix , $i \geq 0$, belongs to L . However, such a word has $3n$ occurrences of b and $2n + (i - 1)j$ occurrences of a , whereas it should have $2n$ occurrences of a for $3n$ occurrences of b . Since $2n + (i - 1)j > 2n$ whenever $i > 1$, this is a contradiction.

Problem 602 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ABAB \\ A &\rightarrow abA \mid \lambda \\ B &\rightarrow cB \mid c \end{aligned}$$

(a) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Advice for Answer: L is represented by the following regular expression:

$$(ab)^* c c^* (ab)^* c c^*$$

(b) Is the complement \overline{L} of the language L context-free? Explain your answer briefly.

Answer: Yes— L is regular and must have a regular complement.

Problem 603 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow ABD \\ A &\rightarrow aAb \mid \lambda \\ B &\rightarrow bB \mid \lambda \\ D &\rightarrow cD \mid \lambda \end{aligned}$$

Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: Such automaton does not exist, since the language generated by grammar G is:

$$L = \{a^m b^m b^n c^j \mid m, n, j \geq 0\}$$

or, equivalently:

$$L = \{a^m b^p c^j \mid m \leq p, m, p, j \geq 0\}$$

and this language is not regular.

To prove this, assume the opposite, that L is regular. Let k be the constant as in the Pumping Lemma for L . Let $m > k$; then $a^m b^m \in L$, since $a^m b^m = a^m b^m c^0$ and $m \leq m$. In any “pumping” decomposition such that: $a^m b^m = uvx$, we have: $|uv| \leq k < m$, hence the “pumping” substring v consists entirely of a 's, say $v = a^\ell$. Recall that $\ell > 0$, since the “pumping” substring cannot be empty. By the pumping, every word of the form uv^ix , $i \geq 0$, belongs to L . However, such a word has exactly m occurrences of b and $m + (i - 1)\ell$ occurrences of a . Whenever $i > 1$, it is true that $m + (i - 1)\ell > m$. Hence, in this case, such a word has more a 's than b 's, and cannot belong to L , which is a contradiction.

Problem 604 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, e\}$, $V = \{S, A, B, D, E\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \mid D \\ A &\rightarrow aE \\ E &\rightarrow bE \mid \lambda \\ B &\rightarrow eeB \mid \lambda \\ D &\rightarrow DD \mid \lambda \mid ca \mid da \end{aligned}$$

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$ab^*(ee)^* \cup (ca \cup da)^*$$

(b) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 221.

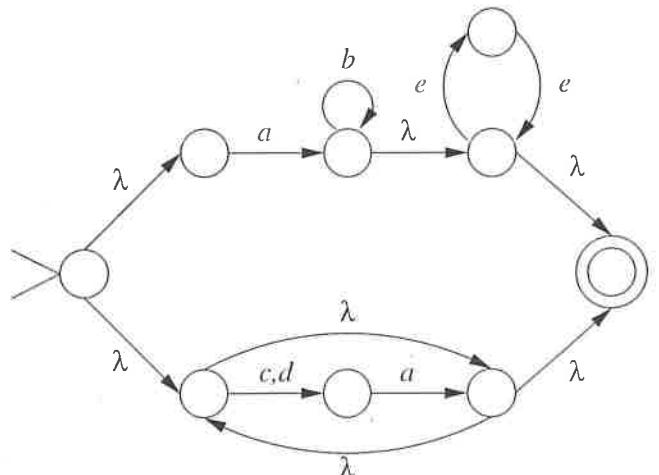


Figure 221:

Problem 605 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, e\}$, $V = \{S, A, B, D, E\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow A \mid BD \\ A &\rightarrow abeE \\ E &\rightarrow bE \mid \lambda \\ B &\rightarrow beeB \mid \lambda \\ D &\rightarrow DD \mid \lambda \mid ca \mid da \end{aligned}$$

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$abeb^* \cup (bee)^*(ca \cup da)^*$$

(b) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 222.

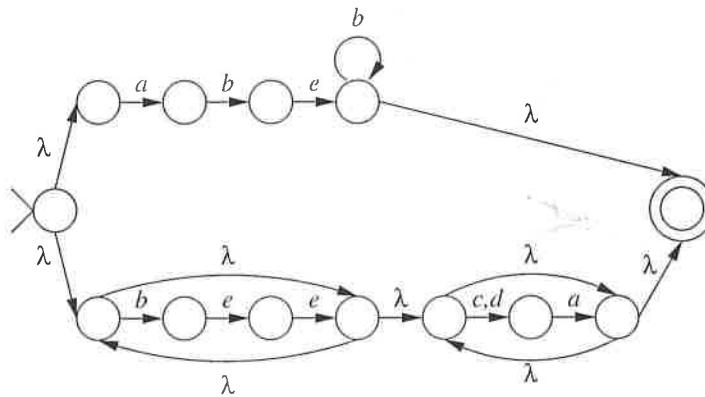


Figure 222:

Problem 606 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow BA \mid AB \mid DA \mid AD \\ A &\rightarrow aA \mid bA \mid \lambda \\ B &\rightarrow dB \mid \lambda \\ D &\rightarrow DD \mid \lambda \mid cd \end{aligned}$$

(a) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: Note that L is defined by the regular expression:

$$d^*(a \cup b)^* \cup (a \cup b)^*d^* \cup (cd)^*(a \cup b)^* \cup (a \cup b)^*(cd)^*$$

whence the automaton represented on Figure 223.

(b) Is the grammar G regular? Explain your answer briefly.

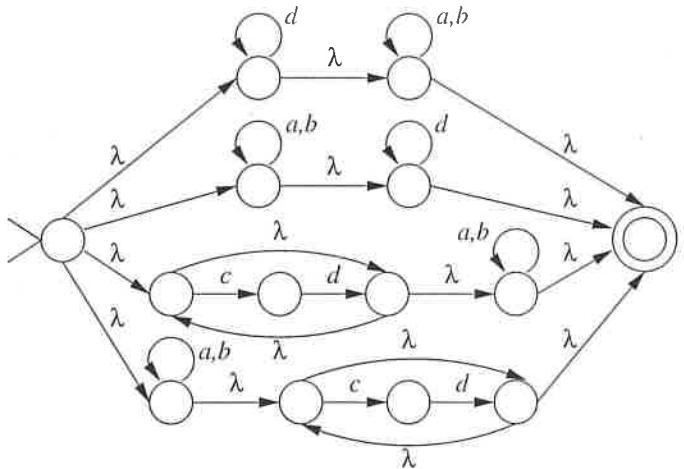


Figure 223:

Answer: No. All rules in a regular grammar fit into one of the following three forms:

$$\begin{aligned} A &\rightarrow aB \\ A &\rightarrow a \\ A &\rightarrow \lambda \end{aligned}$$

where a is an arbitrary terminal, and B is an arbitrary variable. In the given grammar, the four rules with S on the left side (for example) violate the prescribed form.

Problem 607 Let L be the set of strings over alphabet $\{a, b, c\}$ in which the number of b 's is not greater than 3.

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$(a \cup c)^*(b \cup \lambda)(a \cup c)^*(b \cup \lambda)(a \cup c)^*(b \cup \lambda)(a \cup c)^*$$

(b) Does there exist an algorithm to convert an arbitrary regular context-free grammar into a regular expression? Explain your answer briefly.

Answer: Yes. First, we convert such regular grammar into a finite automaton; next, we convert this automaton into a regular expression.

Problem 608 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow BAD \\ A &\rightarrow aaA \mid bbA \mid \lambda \\ B &\rightarrow dbaB \mid \lambda \\ D &\rightarrow DD \mid \lambda \mid cd \end{aligned}$$

(a) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

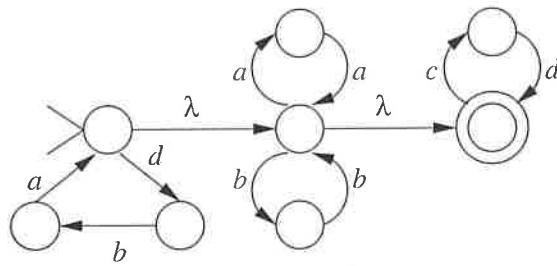


Figure 224:

Answer: See Figure 224.

(b) Describe an algorithm that solves the following problem:

INPUT: An arbitrary context-free grammar \mathcal{G} .

OUTPUT: A finite automaton M that accepts the language generated by \mathcal{G} .

Explain your answer briefly. If such algorithm does not exist, explain why.

Answer: Such algorithm does not exist. Some context-free languages are not regular—in fact, it is undecidable for an arbitrary context-free grammar whether the language derived by it is regular.

Problem 609 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow BAB \\ A &\rightarrow aAa \mid bAb \mid \lambda \\ B &\rightarrow dcB \mid \lambda \end{aligned}$$

(a) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: Such regular expression does not exist, since:

$$L = \{(dc)^j w (dc)^\ell \mid j, \ell \geq 0\}$$

where w is an even-length palindrome over $\{a, b\}$. L is not a regular language.

To prove this, assume the opposite, that L is regular. Let k be the constant as in the Pumping Lemma for L . Let $m > k$; then the word $w = a^m b b a^m$ can be obtained from the general template for the words of L by setting $j = \ell = 0$. Since w is an even-length palindrome over $\{a, b\}$, it follows that $w \in L$.

In any “pumping” decomposition such that $a^m b b a^m = uvx$, we have: $|uv| \leq k < m$. Hence, the “pumping” substring v consists entirely of a 's, say $v = a^\ell$. Recall that $\ell > 0$, since the “pumping” substring cannot be empty. By the pumping, every word of the form $uv^i x$, $i \geq 0$, belongs to L . However, such a word is of the form:

$$w_1 = a^{m+(i-1)\ell} b b a^m$$

while its reversal is:

$$w_1^R = a^m b b a^{m+(i-1)\ell}$$

Since $m + (i - 1)\ell > m$ whenever $i > 1$, word w_1 has more a 's before the pair bb than w_1^R , and it must be that:

$$w_1 \neq w_1^R$$

Since w_1 is not equal to its reversal w_1^R , we conclude that w_1 is not a palindrome. Hence, $w_1 \notin L$, which is a contradiction.

(b) Does there exist an algorithm that solves the following problem:

INPUT: An arbitrary context-free grammar \mathcal{G} .

OUTPUT: A pushdown automaton M that accepts the language generated by \mathcal{G} .

Explain your answer briefly.

Answer: Yes—the algorithmic conversion of an arbitrary context-free grammar into an equivalent pushdown automaton is described in our textbook.

Problem 610 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow A \mid D \\ A &\rightarrow \lambda \mid AA \mid ab \mid B \\ B &\rightarrow ccB \mid \lambda \\ D &\rightarrow dd \mid d \end{aligned}$$

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(ab \cup (cc)^*)^* \cup d^* d$$

(b) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 225.

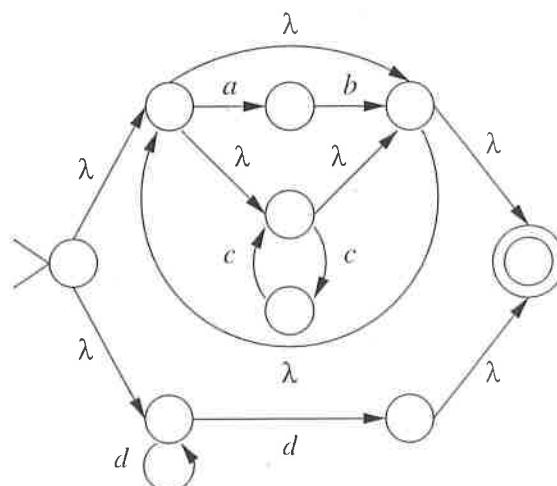


Figure 225:

Problem 611 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow A \mid D \\ A &\rightarrow \lambda \mid aAb \mid B \\ B &\rightarrow ccB \mid \lambda \\ D &\rightarrow dD \mid d \end{aligned}$$

- (a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: This regular expression does not exist, since L is not regular.

Observe that $L = L_1 \cup L_2$, where:

$$L_1 = \{a^n c^{2\ell} b^n \mid n, \ell \geq 0\}$$

$$L_2 = d^*d$$

Those elements of L that do not belong to L_1 are exactly the non-empty sequences of d 's.

To prove that L is not regular, assume the opposite—that L is regular.

Let k be the constant as in the Pumping Lemma for L . Consider a word $w \in L$, such that $w = a^n b^n$, where $n > k$, obtained from the general template for L_1 by setting $\ell = 0$. Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < k < n$. Hence, the pumping part v is entirely within the a -segment, meaning that $v = a^j$ for some j such that $0 < j < k$. By the pumping, for every $i \geq 0$, we have: $uv^i x \in L$. This means that all the words produced by the following template belong to L .

$$uv^i x = uvv^{i-1} x = uva^{(i-1)j} x = a^{n+(i-1)j} b^n$$

Consider the word:

$$w_2 = a^{n+j} b^n$$

obtained by setting $i = 2$ in this template. Since $j > 0$ (as the length of the non-empty pumping part) we see that: $n < n + j$, meaning that the length of the b -segment of w_2 is less than the length of the preceding a -segment. This means that $w_2 \notin L_1$. However, since w_2 is not a non-empty sequence of d 's, we also conclude that $w_2 \notin L_2$, which means that $w_2 \notin L$, whence the contradiction.

(b) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: Such a finite automaton does not exist, since L is not regular, as is proved in the answer to the part (a).

Problem 612 Let L_1 be the language generated by the context-free grammar $G_1 = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow aA \mid aD \\ A &\rightarrow \lambda \mid aA \mid bB \\ B &\rightarrow cB \mid \lambda \\ D &\rightarrow dD \mid dB \end{aligned}$$

Let L_2 be the language generated by the context-free grammar $G_2 = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow AD \\ A &\rightarrow \lambda \mid aaAb \mid B \\ B &\rightarrow cB \mid \lambda \\ D &\rightarrow dD \mid \lambda \end{aligned}$$

- (a) Draw a state-transition graph of a finite automaton M_1 that accepts L_1 . If such an automaton does not exist, prove it.

Answer: Since G_1 is a regular grammar, the automaton represented on Figure 226 is obtained by an algorithmic conversion.

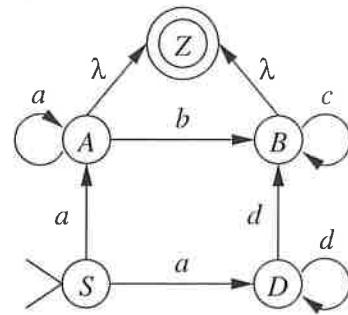


Figure 226:

- (b) List five different strings that belong to L_2 , or explain why this is impossible.

Answer: Observe that the general template for the strings of $L(M)$ is:

$$a^{2m} c^n b^m d^\ell \text{ for } m, n, \ell \geq 0$$

whence the answer:

$$\lambda, aab, c, d, aacbd$$

- (c) Draw a state-transition graph of a finite automaton M_2 that accepts L_2 . If such an automaton does not exist, prove it.

Answer: Such a finite automaton does not exist, because L_2 is not regular.

To prove that L_2 is not regular, assume the opposite, that L_2 is regular. Let k be the constant as in the Pumping Lemma for L_2 . Consider a word $w = a^{2m} b^m$, where $m > k$, obtained from the general template by setting $n = \ell = 0$. In the word w , as in any word of L_2 , the number of a 's is equal twice the number of b 's.

Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < k < m$. Hence, the pumping part v is entirely within the a -segment, meaning that $v = a^j$ for some j such that $0 < j \leq k$. After pumping once, we obtain a word: $w_p = a^{2m+j} b^m \in L_2$. Since w_p belongs to L_2 , the number of a 's in the word has to be equal to twice the number of b 's, precisely: $2m$. However, w_p has as

many as $2m+j$ occurrences of a , and $2m < 2m + j$, since $j > 0$ (because j is the length of the non-empty pumping substring.) Hence, $w_p \notin L_2$, which is a contradiction.

Problem 613 Let \mathcal{S} be a class of languages over alphabet $\{a, b, c\}$, defined as follows:

Language L is a member of \mathcal{S} if and only if L is accepted by some non-deterministic finite automaton, but there does not exist a deterministic finite automaton that accepts L .

What is the cardinality of \mathcal{S} ? Explain your answer briefly.

Answer:

$$|\mathcal{S}| = 0, \text{ since } \mathcal{S} = \emptyset$$

Class \mathcal{S} is empty, since every language accepted by a non-deterministic finite automaton is also accepted by some deterministic finite automaton. In fact, this deterministic finite automaton is obtained by an algorithmic conversion of the original non-deterministic finite automaton.

Problem 614 Let L be the language accepted by the finite automaton $M = (Q, \Sigma, \delta, q, \{f\})$, where $\Sigma = \{a\}$, $Q = \{p, q, r, s, t, v, w, x, y, z, f\}$, and δ is given by the following table:

	a	λ
p	$\{z\}$	\emptyset
q	$\{t, r\}$	$\{s\}$
r	\emptyset	$\{q, t\}$
s	\emptyset	$\{w\}$
t	$\{z, y\}$	$\{p, w\}$
v	$\{x\}$	$\{r\}$
w	$\{y\}$	\emptyset
x	$\{p\}$	$\{v\}$
y	$\{p\}$	$\{f\}$
z	\emptyset	$\{v\}$
f	\emptyset	\emptyset

Compute the λ -closure of state v .

Answer:

$$\mathcal{C}(v) = \{v, r, q, t, s, p, w\}$$

Problem 615 Let L be the language accepted by the finite automaton $M = (Q, \Sigma, \delta, q, \{f\})$, where $\Sigma = \{a\}$, $Q = \{f, h, p, q, r, s, t, v, w, x, y, z\}$,

and δ is given by the following table:

	a	λ
f	\emptyset	$\{w\}$
h	\emptyset	$\{p\}$
p	$\{z\}$	\emptyset
q	$\{t, r\}$	$\{s\}$
r	\emptyset	$\{q\}$
s	\emptyset	$\{t, v\}$
t	$\{z, y\}$	$\{p\}$
v	$\{y\}$	$\{h\}$
w	$\{x\}$	$\{r\}$
x	$\{p\}$	$\{w\}$
y	$\{p\}$	$\{f\}$
z	\emptyset	$\{w\}$

Compute the λ -closure of state w .

Answer:

$$\mathcal{C}(w) = \{w, r, q, s, t, v, p, h\}$$

Problem 616 Let M be the finite automaton represented by the state diagram on Figure 227, and let L be the language accepted by M .

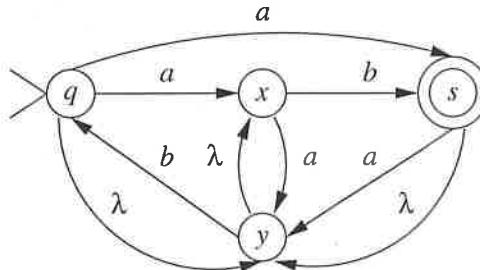


Figure 227:

(a) Is M deterministic? Explain your answer.

Answer: No—the automaton has λ -arcs, two transitions out of state q on symbol a ; no transitions out of state s on symbol b , and more.

(b) If your answer to part (a) is NO, construct a state transition graph of a deterministic finite automaton M_1 that accepts L , and show your work.

Answer: Apply the known algorithm to convert M into a deterministic automaton M_1 .

Let $M_1 = (Q_1, \{a, b\}, \delta_1, q_1, F_1)$, where $Q_1 \in \mathcal{P}(Q)$.

Transition function of M :

δ	a	b	λ
q	$\{x, s\}$	\emptyset	$\{y\}$
x	$\{y\}$	$\{s\}$	\emptyset
y	\emptyset	$\{q\}$	$\{x\}$
s	$\{y\}$	\emptyset	$\{y\}$

λ -closure:

$\mathcal{C}(\cdot)$	q	x	y	s
q	$\{q, y, x\}$			
x		$\{x\}$		
y		$\{y, x\}$		
s		$\{s, y, x\}$		

The initial state: $q_1 = \mathcal{C}(q) = \{q, y, x\}$.

The transition function δ_1 :

δ_1	a	b
$\{q, y, x\}$	$\{s, y, x\}$	$\{q, y, x, s\}$
$\{s, y, x\}$	$\{y, x\}$	$\{q, y, x, s\}$
$\{q, y, x, s\}$	$\{s, y, x\}$	$\{q, y, x, s\}$
$\{y, x\}$	$\{y, x\}$	$\{q, y, x, s\}$

The set of states:

$$Q_1 = \{\{q, y, x\}, \{s, y, x\}, \{q, y, x, s\}, \{y, x\}\}.$$

The set of final states:

$$F_1 = \{\{s, y, x\}, \{q, y, x, s\}\}.$$

The state diagram of M_1 is given on Figure 228.

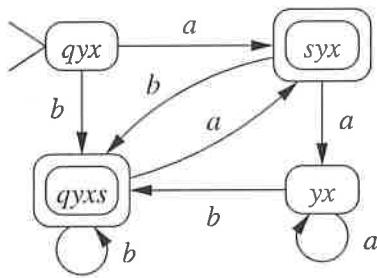


Figure 228:

- (c) If your answer to part (a) is YES, construct the configuration of M after it processes the input string aab , and show your work.

Answer: The question does not apply. Indeed, after it processes the input string aab , the automaton M can be in any one of its four states. Hence, there cannot be “the” configuration of M after aab .

Problem 617 Let M be the finite automaton represented by the state diagram on Figure 229, and let L be the language accepted by M .

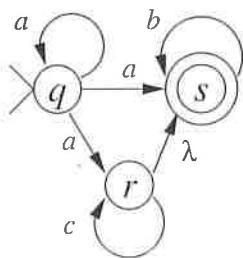


Figure 229:

Construct a state-transition graph of a deterministic finite automaton M_1 that accepts L , and show your work. If such automaton does not exist, prove it.

Answer:

Let $M_1 = (Q', \{a, b, c\}, \delta', q', F')$, where $Q' \in \mathcal{P}(Q)$.

Transition function of M :

δ	a	b	c	λ
q	$\{q, s, r\}$	\emptyset	\emptyset	\emptyset
s	\emptyset	$\{s\}$	\emptyset	\emptyset
r	\emptyset	\emptyset	$\{r\}$	$\{s\}$

λ -closure:

	$\mathcal{C}(q)$
q	$\{q\}$
s	$\{s\}$
r	$\{r, s\}$

The initial state: $q' = \{q\}$.

The transition function δ' :

δ'	a	b	c
$\{q\}$	$\{q, s, r\}$	\emptyset	\emptyset
$\{q, s, r\}$	$\{q, s, r\}$	$\{s\}$	$\{s, r\}$
$\{s\}$	\emptyset	$\{s\}$	\emptyset
$\{s, r\}$	\emptyset	$\{s\}$	$\{s, r\}$
\emptyset	\emptyset	\emptyset	\emptyset

The set of states:

$$Q' = \{\{q\}, \{s\}, \{s, r\}, \{q, s, r\}, \emptyset\}.$$

The set of final states:

$$F' = \{\{s\}, \{s, r\}, \{q, s, r\}\}.$$

The state diagram of M_1 is given on Figure 230.

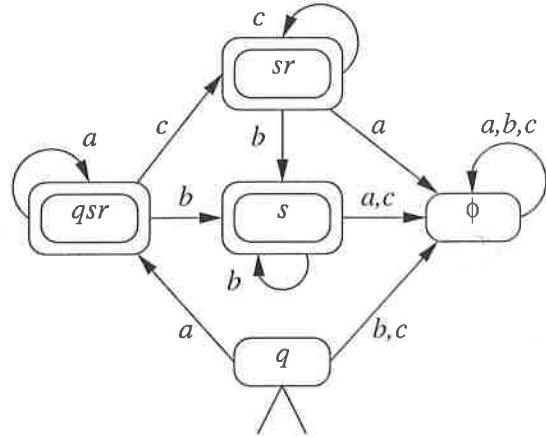


Figure 230:

Problem 618 Let M be the finite automaton represented by the state diagram on Figure 231, and let L be the language accepted by M .

Write a complete formal definition or a state-transition graph of a deterministic finite automaton M' that accepts L and show your work. If such automaton does not exist, prove it.

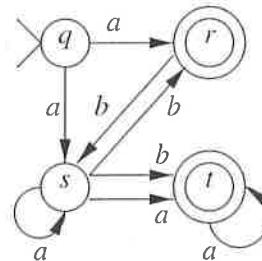


Figure 231:

Answer: There are no λ -transitions; therefore, the λ -closure of every state is the singleton containing that

state. Hence, $q' = \{q\}$. Furthermore, the transition function δ of M and the input transition function t of M' are identical:

$t = \delta$	a	b
q	$\{r, s\}$	\emptyset
r	\emptyset	$\{s\}$
s	$\{s, t\}$	$\{r, t\}$
t	$\{t\}$	\emptyset

The transition function δ' :

δ'	a	b
$\{q\}$	$\{r, s\}$	\emptyset
$\{r, s\}$	$\{s, t\}$	$\{r, s, t\}$
$\{s, t\}$	$\{s, t\}$	$\{r, t\}$
$\{r, s, t\}$	$\{s, t\}$	$\{r, s, t\}$
$\{r, t\}$	$\{t\}$	$\{s\}$
$\{t\}$	$\{t\}$	\emptyset
$\{s\}$	$\{s, t\}$	$\{r, t\}$
\emptyset	\emptyset	\emptyset

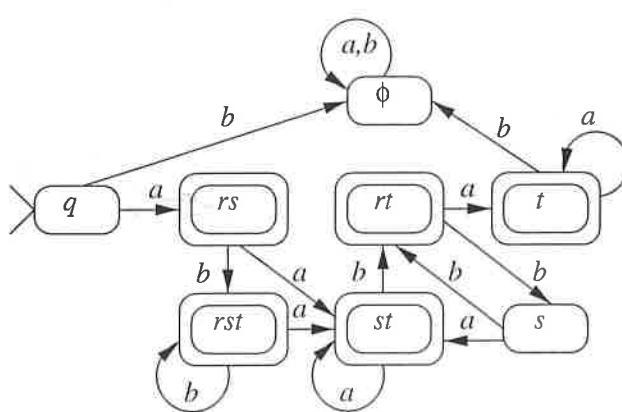


Figure 232:

The set of states:

$$Q' = \{\{q\}, \{r, s\}, \{s, t\}, \{r, s, t\}, \{r, t\}, \{t\}, \{s\}, \emptyset\}.$$

The set of final states:

$$F' = \{\{r, s\}, \{s, t\}, \{r, s, t\}, \{r, t\}, \{t\}\}.$$

The state diagram of M' is given on Figure 232.

Problem 619 Let M be the finite automaton represented by the state diagram on Figure 233, and let L be the language accepted by M .

Write a complete formal definition or a state-transition graph of a deterministic finite automaton M' that accepts L and show your work. If such automaton does not exist, prove it.

Answer:

Transition function of M :

δ	a	b	λ
x	$\{z\}$	\emptyset	$\{y, w\}$
y	$\{y\}$	\emptyset	\emptyset
z	\emptyset	$\{y\}$	$\{w\}$
w	$\{y\}$	\emptyset	\emptyset

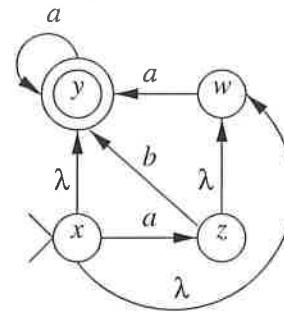


Figure 233:

λ -closure:

q	$C(q)$
x	$\{x, y, w\}$
y	$\{y\}$
z	$\{z, w\}$
w	$\{w\}$

The initial state: $q'_0 = C(x) = \{x, y, w\}$. Input transition function of M' :

t	a	b
x	$\{z, y, w\}$	\emptyset
y	$\{y\}$	\emptyset
z	$\{y\}$	$\{y\}$
w	$\{y\}$	\emptyset

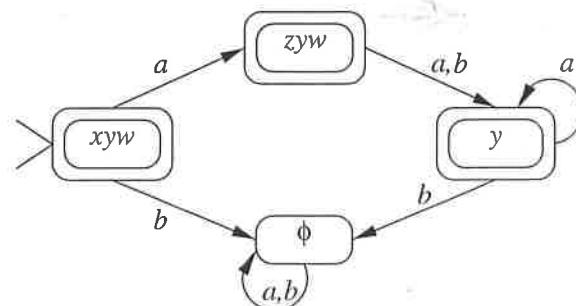


Figure 234:

The transition function δ' :

δ'	a	b
$\{x, y, w\}$	$\{z, y, w\}$	\emptyset
$\{z, y, w\}$	$\{y\}$	$\{y\}$
$\{y\}$	$\{y\}$	\emptyset
\emptyset	\emptyset	\emptyset

The set of states:

$$Q' = \{\{x, y, w\}, \{z, y, w\}, \{y\}, \emptyset\}.$$

The set of final states:

$$F' = \{\{x, y, w\}, \{z, y, w\}, \{y\}\}.$$

The state diagram of M' is given on Figure 234.

Problem 620 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow aB \mid bA \mid bB \mid \lambda \\ A &\rightarrow bB \\ B &\rightarrow bA \mid a \end{aligned}$$

- (a) Draw a state-transition graph of a finite automaton M_1 that accepts L . If such automaton does not exist, prove it.

Answer: The grammar is regular, and the construction is algorithmic. See Figure 235.

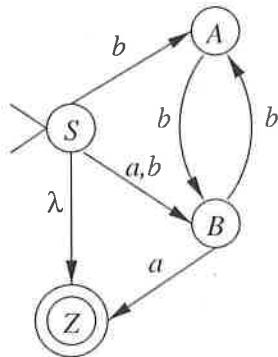


Figure 235:

- (b) Draw a state-transition graph of a deterministic finite automaton M_2 that accepts L . If such automaton does not exist, prove it.

Answer: Transition function of M_1 :

δ	a	b	λ
S	{B}	{A, B}	{Z}
A	\emptyset	{B}	\emptyset
B	{Z}	{A}	\emptyset
Z	\emptyset	\emptyset	\emptyset

λ -closure:

x	$C(x)$
S	{S, Z}
A	{A}
B	{B}
Z	{Z}

Let $M_2 = (Q_2, \{a, b\}, \delta_2, q_0, F_2)$.

The initial state: $q_0 = \{S, Z\}$.

The transition function δ_2 :

δ_2	a	b
{S, Z}	{B}	{A, B}
{B}	{Z}	{A}
{A, B}	{Z}	{A, B}
{Z}	\emptyset	\emptyset
{A}	\emptyset	{B}
\emptyset	\emptyset	\emptyset

The set of states:

$$Q_2 = \{\{S, Z\}, \{B\}, \{A, B\}, \{Z\}, \{A\}, \emptyset\}.$$

The set of final states:

$$F_2 = \{\{S, Z\}, \{Z\}\}.$$

The state diagram of M_2 is given on Figure 236.

Problem 621 Let M be the finite automaton represented by the state diagram on Figure 237, and let L be the language accepted by M .

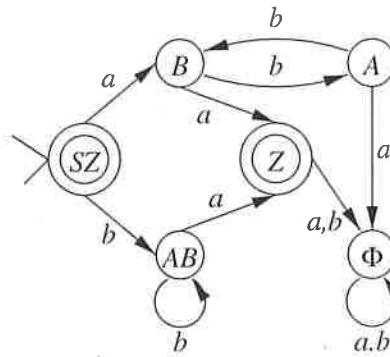


Figure 236:

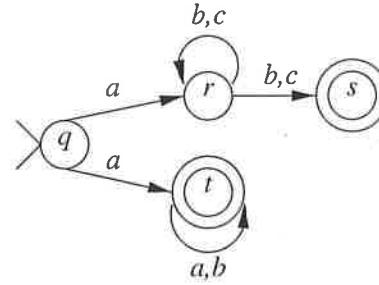


Figure 237:

- (a) Is the finite automaton M deterministic? Justify briefly your answer.

Answer: No—for example, there are two states, r and s , reachable from state r on input symbol b (or c), etc.

- (b) If M is not deterministic, construct a deterministic finite automaton M' that accepts L and show your work. If such an automaton M' does not exist, explain why.

Answer: Let $M' = (Q', \{a, b, c\}, \delta', q', F')$, where $Q' \in \mathcal{P}(Q)$.

There are no ϵ -transitions; therefore, the ϵ -closure of every state is the singleton containing that state. Hence, $q' = \{q\}$. The transition function δ :

δ'	a	b	c
q	{r, t}	\emptyset	\emptyset
r	\emptyset	{r, s}	{r, s}
s	\emptyset	\emptyset	\emptyset
t	{t}	{t}	\emptyset

The transition function δ' :

δ'	a	b	c
{q}	{r, t}	\emptyset	\emptyset
{r, t}	{t}	{r, s, t}	{r, s}
{t}	{t}	{t}	\emptyset
{r, s, t}	{t}	{r, s, t}	{r, s}
{r, s}	\emptyset	{r, s}	{r, s}
\emptyset	\emptyset	\emptyset	\emptyset

The set of states:

$$Q' = \{\{q\}, \{r, t\}, \{t\}, \{r, s, t\}, \{r, s\}, \emptyset\}.$$

The set of final states:

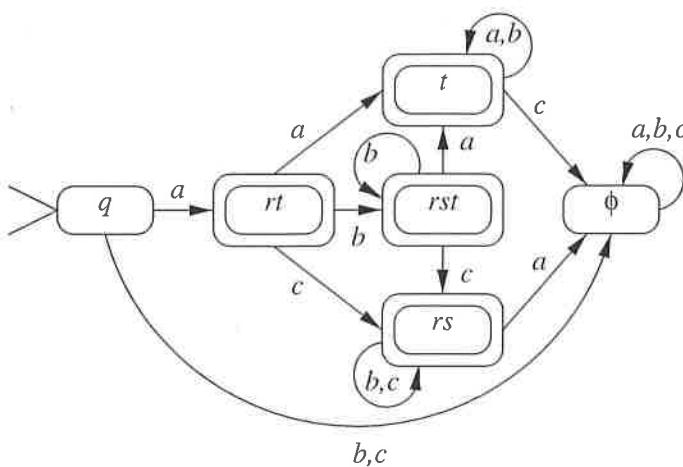


Figure 238:

$$F' = \{\{t\}, \{r, t\}, \{r, s, t\}, \{r, s\}\}.$$

The state diagram of M' is given on Figure 238.

Problem 622 Let M be the finite automaton represented by the state diagram on Figure 239, and let L be the language accepted by M .

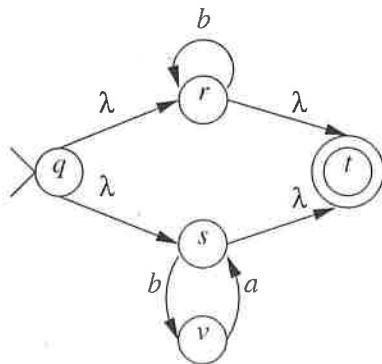


Figure 239:

(a) Is the finite automaton M deterministic? Justify your answer briefly.

Answer: No—for example, M has λ -transitions; there is no transition from state s on symbol a , etc.

(b) If M is not deterministic, construct a deterministic finite automaton M_1 that accepts L and show your work. If such an automaton M_1 does not exist, explain why.

Answer: Let $M_1 = (Q_1, \{a, b\}, \delta_1, q_1, F_1)$, where $Q_1 \in \mathcal{P}(Q)$.

Transition function of M :

δ	a	b	λ
q	\emptyset	\emptyset	$\{r, s\}$
r	\emptyset	$\{r\}$	$\{t\}$
s	\emptyset	$\{v\}$	$\{t\}$
t	\emptyset	\emptyset	\emptyset
v	$\{s\}$	\emptyset	\emptyset

λ -closure:

$-$	$C(-)$
q	$\{q, r, s, t\}$
r	$\{r, t\}$
s	$\{s, t\}$
t	$\{t\}$
v	$\{v\}$

The initial state: $q_1 = C(q) = \{q, r, s, t\}$.

The transition function δ_1 :

δ_1	a	b
$\{q, r, s, t\}$	\emptyset	$\{r, t, v\}$
$\{r, t, v\}$	$\{s, t\}$	$\{r, t\}$
$\{s, t\}$	\emptyset	$\{v\}$
$\{r, t\}$	\emptyset	$\{r, t\}$
$\{v\}$	$\{s, t\}$	\emptyset
\emptyset	\emptyset	\emptyset

The set of states:

$$Q_1 = \{\{q, r, s, t\}, \{r, t, v\}, \{s, t\}, \{r, t\}, \{v\}, \emptyset\}.$$

The set of final states:

$$F_1 = \{\{q, r, s, t\}, \{r, t, v\}, \{s, t\}, \{r, t\}\}.$$

The state diagram of M_1 is given on Figure 240.

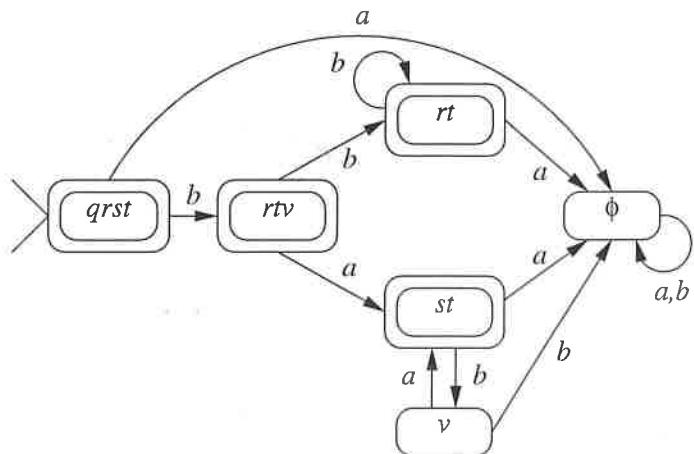


Figure 240:

(c) Construct a state transition graph of a deterministic finite automaton M_2 that accepts \bar{L} (the complement of L), and explain your construction briefly. If such an automaton does not exist, explain why.

Answer: Since M_1 is deterministic, a conversion to an automation that accepts the complement of L is algorithmic— M_2 is obtained from M_1 by turning every final state into a non-final and every non-final state to a final one. See Figure 241.

Problem 623 Let M be the finite automaton represented by the state diagram given on Figure 242, and let L be the language accepted by M .

Construct a deterministic finite automaton M' that accepts L and show your work. If such M' does not exist, explain why.

Answer: Let $M' = (Q', \{a, b\}, \delta', q'_0, F')$, where $Q' \in \mathcal{P}(Q)$.

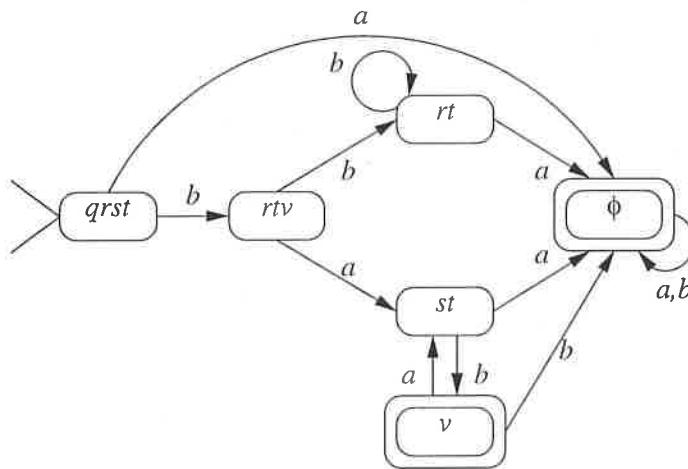


Figure 241:

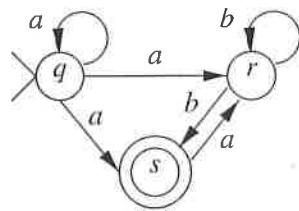


Figure 242:

There are no λ -transitions; therefore, the λ -closure of every state is the singleton containing that state. Hence, $q' = \{q\}$. Furthermore, the transition function of M and the input transition function t of M' are identical:

t	a	b
q	$\{q, r, s\}$	\emptyset
r	\emptyset	$\{r, s\}$
s	$\{r\}$	\emptyset

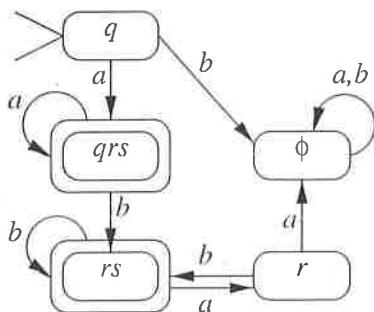


Figure 243:

The transition function δ' :

δ'	a	b
{q}	$\{q, r, s\}$	\emptyset
{q, r, s}	$\{q, r, s\}$	$\{r, s\}$
{r, s}	$\{r\}$	$\{r, s\}$
{r}	\emptyset	$\{r, s\}$
\emptyset	\emptyset	\emptyset

The set of states:

$$Q' = \{\{q\}, \{q, r, s\}, \{r\}, \{r, s\}, \emptyset\}.$$

The set of final states:

$$F' = \{\{q, r, s\}, \{r, s\}\}.$$

The state diagram of M' is given on Figure 243.

Problem 624 Let M be the finite automaton represented by the state diagram given on Figure 244, and let L be the language accepted by M .

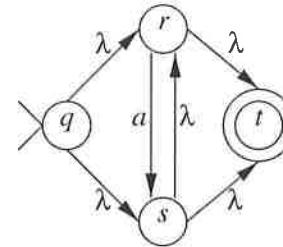


Figure 244:

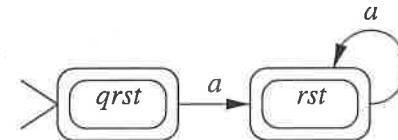


Figure 245:

Construct a deterministic finite automaton M' that accepts L and show your work. If such M' does not exist, explain why.

Answer: Let $M' = (Q', \{a\}, \delta', q'_0, F')$, where $Q' \in \mathcal{P}(Q)$.

Transition function of M :

δ	a	λ
q	\emptyset	$\{r, s\}$
r	$\{s\}$	$\{t\}$
s	\emptyset	$\{r, t\}$
t	\emptyset	\emptyset

λ -closure:

w	$\mathcal{C}(w)$
q	$\{q, r, s, t\}$
r	$\{r, t\}$
s	$\{r, s, t\}$
t	$\{t\}$

The initial state: $q'_0 = \mathcal{C}(q) = \{q, r, s, t\}$.
Input transition function of M' :

t	a
q	$\{r, s, t\}$
r	$\{r, s, t\}$
s	$\{r, s, t\}$
t	\emptyset

The transition function δ' :

δ'	a
$\{q, r, s, t\}$	$\{r, s, t\}$
$\{r, s, t\}$	$\{r, s, t\}$

The set of states:

$$Q' = \{\{q, r, s, t\}, \{r, s, t\}\}.$$

The set of final states:

$$F' = \{\{q, r, s, t\}, \{r, s, t\}\}.$$

The state diagram of M' is given on Figure 245.

Problem 625 Let M be the finite automaton represented by the state diagram of Figure 246, and let L be the language accepted by M .

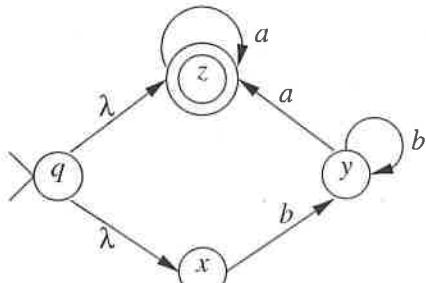


Figure 246:

(a) Construct a deterministic finite automaton M_1 that accepts L . Show your work. If such an automaton does not exist, explain why.

Answer: Let $M_1 = (Q_1, \{a, b\}, \delta_1, q_1, F_1)$, where $Q_1 \in \mathcal{P}(Q)$.

Transition function of M :

δ	a	b	λ
q	\emptyset	\emptyset	$\{x, z\}$
x	\emptyset	$\{y\}$	\emptyset
y	$\{z\}$	$\{y\}$	\emptyset
z	$\{z\}$	\emptyset	\emptyset

λ -closure:

w	$C(w)$
q	$\{q, x, z\}$
x	$\{x\}$
y	$\{y\}$
z	$\{z\}$

The initial state: $q_1 = \{q, x, z\}$.

The transition function δ_1 :

δ_1	a	b
$\{q, x, z\}$	$\{z\}$	$\{y\}$
$\{z\}$	$\{z\}$	\emptyset
$\{y\}$	$\{z\}$	$\{y\}$
\emptyset	\emptyset	\emptyset

The set of states:

$$Q_1 = \{\{q, x, z\}, \{z\}, \{y\}, \emptyset\}.$$

The set of final states:

$$F_1 = \{\{q, x, z\}, \{z\}\}.$$

The state diagram of M_1 is given on Figure 247.

(b) Write a complete formal definition of a *regular context-free grammar* G that accepts L , and explain your construction briefly. If such a grammar does not exist, explain why.

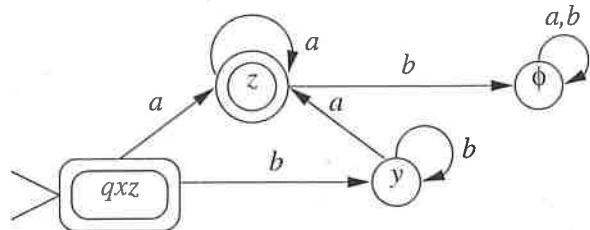


Figure 247:

Answer: The automaton M_1 is deterministic, and thereby algorithmically convertible to a regular grammar. For convenience, M_1 can be renamed and drawn as on Figure 248, yielding directly an equivalent regular grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, Y, Z, F\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aZ \mid bY \mid \lambda \\ Z &\rightarrow aZ \mid bF \mid \lambda \\ Y &\rightarrow aZ \mid bY \\ F &\rightarrow aF \mid bF \end{aligned}$$

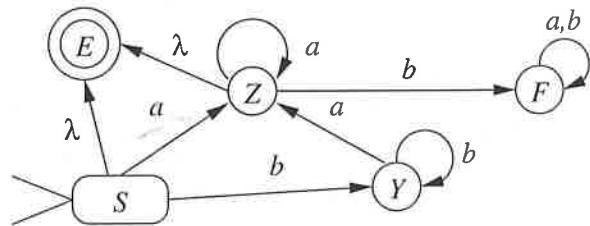


Figure 248:

Problem 626 Let M be the finite automaton represented by the state diagram on Figure 249, and let L be the language accepted by M .

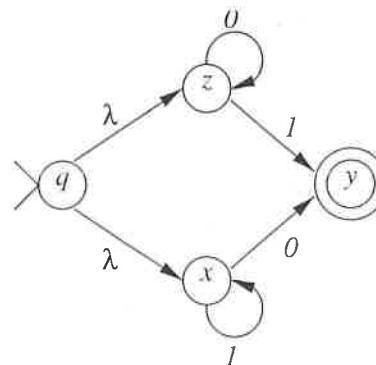


Figure 249:

Construct a deterministic finite automaton M' that accepts L . Show your work.

Answer: Let $M' = (Q', \{0, 1\}, \delta', q'_0, F')$, where $Q' \in \mathcal{P}(Q)$.

Transition function of M :

δ	0	1	λ
q	\emptyset	\emptyset	$\{x, z\}$
x	$\{y\}$	$\{x\}$	\emptyset
y	\emptyset	\emptyset	\emptyset
z	$\{z\}$	$\{y\}$	\emptyset

λ -closure:

w	$C(w)$
q	$\{q, x, z\}$
x	$\{x\}$
y	$\{y\}$
z	$\{z\}$

The initial state: $q'_0 = \{q, x, z\}$.

Input transition function:

t	0	1
q	$\{y, z\}$	$\{x, y\}$
x	$\{y\}$	$\{x\}$
y	\emptyset	\emptyset
z	$\{z\}$	$\{y\}$

The transition function δ' :

δ'	0	1
$\{q, x, z\}$	$\{y, z\}$	$\{x, y\}$
$\{x, y\}$	$\{y\}$	$\{x\}$
$\{y, z\}$	$\{z\}$	$\{y\}$
$\{y\}$	\emptyset	\emptyset
$\{x\}$	$\{y\}$	$\{x\}$
$\{z\}$	$\{z\}$	$\{y\}$
\emptyset	\emptyset	\emptyset

The set of states:

$$Q' = \{\{q, x, z\}, \{x, y\}, \{y, z\}, \{x\}, \{y\}, \{z\}, \emptyset\}.$$

The set of final states:

$$F' = \{\{y, z\}, \{x, y\}, \{y\}\}.$$

The state diagram of M' is given on Figure 250.

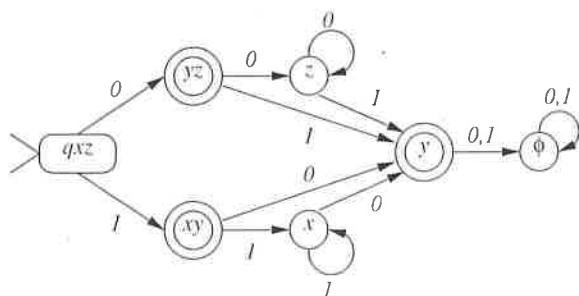


Figure 250:

Problem 627 Let M be the finite automaton represented by the state diagram on Figure 251, and let L be the language accepted by M .

(a) Is the finite automaton M deterministic? Justify briefly your answer.

Answer: No. It has 2 a -transitions out of state q_0 .

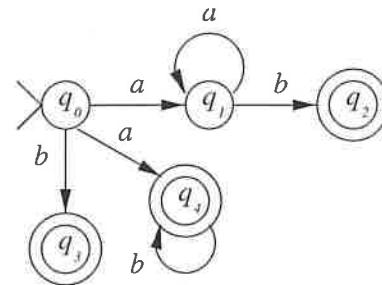


Figure 251:

(b) If M is not deterministic, construct a deterministic finite automaton M' that accepts L . Show your work.

Answer: Let $M' = (Q', \{a, b\}, \delta', q'_0, F')$, where $Q' \in \mathcal{P}(Q)$.

There are no λ -transitions; therefore, the λ -closure of every state is the singleton containing that state. Hence, $q' = \{q_0\}$. Furthermore, the transition function δ and the input transition function of M' are identical:

$t = \delta$	a	b
q_0	$\{q_1, q_4\}$	$\{q_3\}$
q_1	$\{q_1\}$	$\{q_2\}$
q_2	\emptyset	\emptyset
q_3	\emptyset	\emptyset
q_4	\emptyset	$\{q_4\}$

The transition function δ' :

δ'	a	b
$\{q_0\}$	$\{q_1, q_4\}$	$\{q_3\}$
$\{q_1, q_4\}$	$\{q_1\}$	$\{q_2, q_4\}$
$\{q_3\}$	\emptyset	\emptyset
$\{q_1\}$	$\{q_1\}$	$\{q_2\}$
$\{q_2, q_4\}$	\emptyset	$\{q_4\}$
$\{q_2\}$	\emptyset	\emptyset
$\{q_4\}$	\emptyset	$\{q_4\}$
\emptyset	\emptyset	\emptyset

The set of states:

$$Q' = \{\{q_0\}, \{q_1\}, \{q_2\}, \{q_3\}, \{q_4\}, \{q_1, q_4\}, \{q_2, q_4\}, \emptyset\}.$$

The set of final states:

$$F' = \{\{q_2\}, \{q_3\}, \{q_4\}, \{q_1, q_4\}, \{q_2, q_4\}\}.$$

The state diagram of M' is given on Figure 252.

Problem 628 Let M be the finite automaton represented by the state diagram on Figure 253, and let L be the language accepted by M .

(a) Is the finite automaton M deterministic? Justify briefly your answer.

Answer: No. It has 2 a -transitions out of state q_1 .

(b) If M is not deterministic, construct a deterministic finite automaton M' that accepts L . Show your work.

Answer: Let $M' = (Q', \{a, b, c\}, \delta', q'_0, F')$, where $Q' \in \mathcal{P}(Q)$.

There are no λ -transitions; therefore, the λ -closure of every state is the singleton containing that state. Hence,

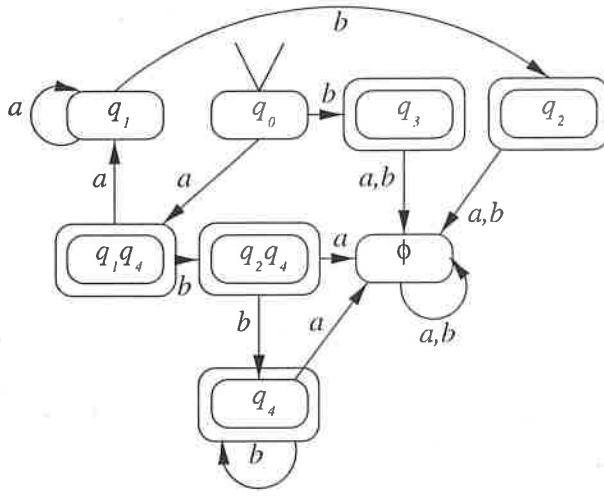


Figure 252:

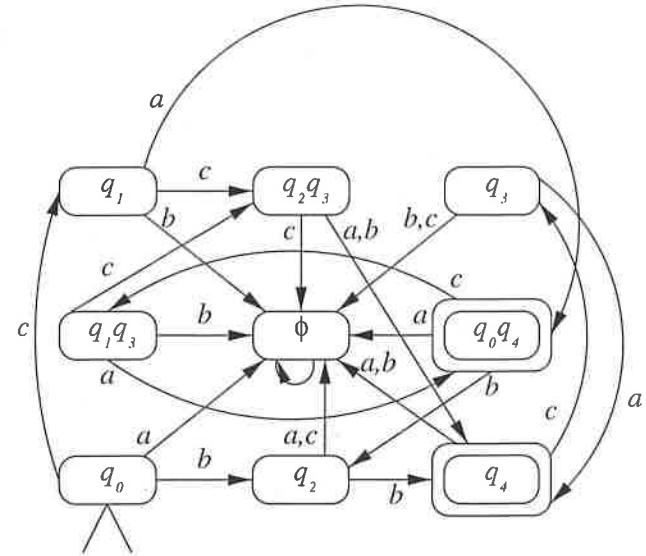


Figure 254:

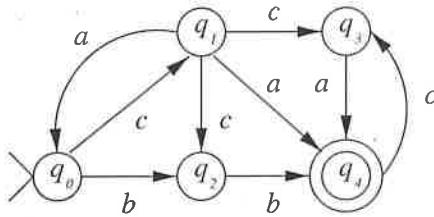


Figure 253:

$q' = \{q_0\}$. Furthermore, the transition function δ and the input transition function of M' are identical:

$t = \delta$	a	b	c
q_0	\emptyset	$\{q_2\}$	$\{q_1\}$
q_1	$\{q_0, q_4\}$	\emptyset	$\{q_2, q_3\}$
q_2	\emptyset	$\{q_4\}$	\emptyset
q_3	$\{q_4\}$	\emptyset	\emptyset
q_4	\emptyset	\emptyset	$\{q_3\}$

The transition function δ' :

δ'	a	b	c
$\{q_0\}$	\emptyset	$\{q_2\}$	$\{q_1\}$
$\{q_1\}$	$\{q_0, q_4\}$	\emptyset	$\{q_2, q_3\}$
$\{q_2\}$	\emptyset	$\{q_4\}$	\emptyset
$\{q_3\}$	$\{q_4\}$	\emptyset	\emptyset
$\{q_4\}$	\emptyset	\emptyset	$\{q_3\}$
$\{q_2, q_3\}$	$\{q_4\}$	$\{q_4\}$	\emptyset
$\{q_0, q_4\}$	\emptyset	$\{q_2\}$	$\{q_1, q_3\}$
$\{q_1, q_3\}$	$\{q_0, q_4\}$	\emptyset	$\{q_2, q_3\}$
\emptyset	\emptyset	\emptyset	\emptyset

The set of states:

$$Q' = \{\{q_0\}, \{q_1\}, \{q_2\}, \{q_3\}, \{q_4\}, \{q_2, q_3\}, \{q_0, q_4\}, \{q_1, q_3\}, \emptyset\}.$$

The set of final states:

$$F' = \{\{q_4\}, \{q_0, q_4\}\}.$$

The state diagram of M' is given on Figure 254.

Problem 629 Consider the finite automaton M represented by the following state transition graph.

Construct a deterministic finite automaton M' that is equivalent to M , and show your work. If such an automaton does not exist, prove it. If M is already deterministic, prove it.

Answer: Let $M' = (Q', \{a, b, c\}, \delta', q'_0, F')$. Transition function of M :

δ	a	b	c	λ
s	$\{s\}$	\emptyset	$\{t\}$	$\{w\}$
t	\emptyset	\emptyset	$\{t\}$	\emptyset
w	\emptyset	$\{t, w\}$	\emptyset	\emptyset

λ -closure:

q	$C(q)$
s	$\{s, w\}$
t	$\{t\}$
w	$\{w\}$

The initial state: $q'_0 = \{s, w\}$.

The transition function δ' :

δ'	a	b	c
$\{s, w\}$	$\{s, w\}$	$\{t, w\}$	$\{t\}$
$\{t, w\}$	\emptyset	$\{t, w\}$	$\{t\}$
$\{t\}$	\emptyset	\emptyset	$\{t\}$
\emptyset	\emptyset	\emptyset	\emptyset

The set of states:

$$Q' = \{\{s, w\}, \{t, w\}, \{t\}, \emptyset\}.$$

The set of final states:

$$F' = \{\{s, w\}, \{t, w\}, \{t\}\}.$$

The state diagram of M' is given on Figure 256.

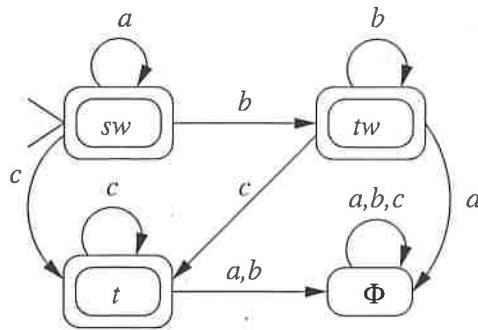


Figure 256:

Problem 630 Let M be the finite automaton represented by the state diagram on Figure 257, and let L be the language accepted by M .

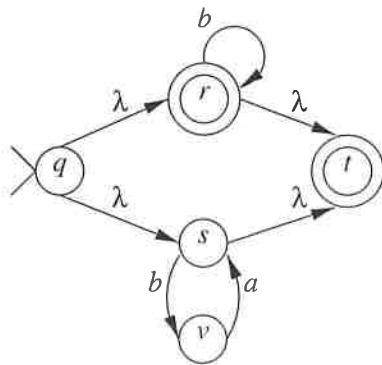


Figure 257:

(a) Is the finite automaton M deterministic? Justify briefly your answer.

Answer: No— M has λ -transitions out of states q, s, r , etc.

(b) If M is not deterministic, construct a deterministic finite automaton M' that accepts L and show your work. If such automaton does not exist, prove it.

Answer: Let $M' = (Q', \{a, b\}, \delta', q'_0, F')$, where $Q' \in \mathcal{P}(Q)$.

Transition function of M :

δ	a	b	λ
q	\emptyset	\emptyset	$\{r, s\}$
r	\emptyset	$\{r\}$	$\{t\}$
s	\emptyset	$\{v\}$	$\{t\}$
t	\emptyset	\emptyset	\emptyset
v	$\{s\}$	\emptyset	\emptyset

λ -closure:

x	$C(x)$
q	$\{q, r, s, t\}$
r	$\{r, t\}$
s	$\{s, t\}$
t	$\{t\}$
v	$\{v\}$

The initial state: $q'_0 = \{q, r, s, t\}$.

The transition function δ' :

δ'	a	b
$\{q, r, s, t\}$	\emptyset	$\{r, t, v\}$
$\{r, t, v\}$	$\{s, t\}$	$\{r, t\}$
$\{s, t\}$	\emptyset	$\{v\}$
$\{r, t\}$	\emptyset	$\{r, t\}$
$\{v\}$	$\{s, t\}$	\emptyset
\emptyset	\emptyset	\emptyset

The set of states:

$$Q' = \{\{q, r, s, t\}, \{r, t, v\}, \{s, t\}, \{r, t\}, \{v\}, \emptyset\}.$$

The set of final states:

$$F' = \{\{q, r, s, t\}, \{r, t, v\}, \{s, t\}, \{r, t\}\}.$$

The state diagram of M' is given on Figure 258.

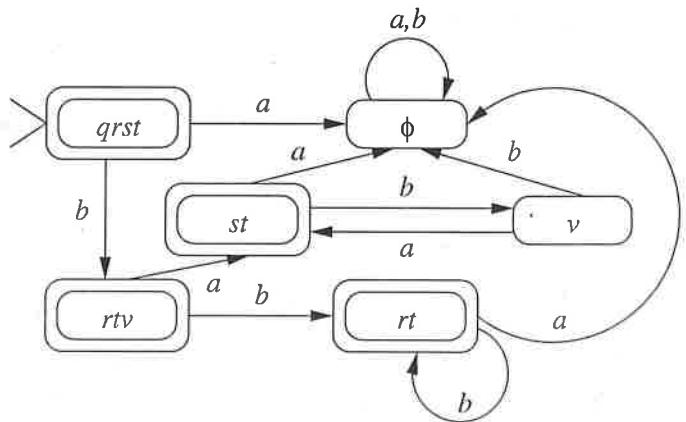


Figure 258:

Problem 631 Let M be the finite automaton represented on Figure 259, and let L be the language accepted by M .

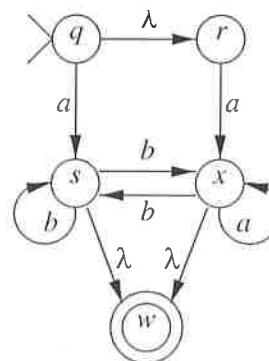


Figure 259:

- (a) Is the finite automaton M deterministic? Explain your answer.

Answer: No—for example, it contains λ -transitions.

- (b) If M is not deterministic, construct a deterministic finite automaton M' that accepts L and show your work. If such automaton does not exist, prove it.

Answer: Let $M' = (Q', \{a, b\}, \delta', q'_0, F')$, where $Q' \in \mathcal{P}(Q)$.

Transition function of M :

δ	a	b	λ
q	$\{s\}$	\emptyset	$\{r\}$
r	$\{x\}$	\emptyset	\emptyset
s	\emptyset	$\{s, x\}$	$\{w\}$
x	$\{x\}$	$\{s\}$	$\{w\}$
w	\emptyset	\emptyset	\emptyset

λ -closure:

y	$C(y)$
q	$\{q, r\}$
r	$\{r\}$
s	$\{s, w\}$
x	$\{x, w\}$
w	$\{w\}$

The initial state: $q'_0 = \{q, r\}$.

The transition function δ' :

δ'	a	b
$\{q, r\}$	$\{s, x, w\}$	\emptyset
$\{s, x, w\}$	$\{x, w\}$	$\{s, x, w\}$
$\{x, w\}$	$\{x, w\}$	$\{s, w\}$
$\{s, w\}$	\emptyset	$\{s, x, w\}$
\emptyset	\emptyset	\emptyset

The set of states:

$$Q' = \{\{q, r\}, \{s, x, w\}, \{x, w\}, \{s, w\}, \emptyset\}.$$

The set of final states:

$$F' = \{\{s, x, w\}, \{x, w\}, \{s, w\}\}.$$

The state diagram of M' is given on Figure 260.

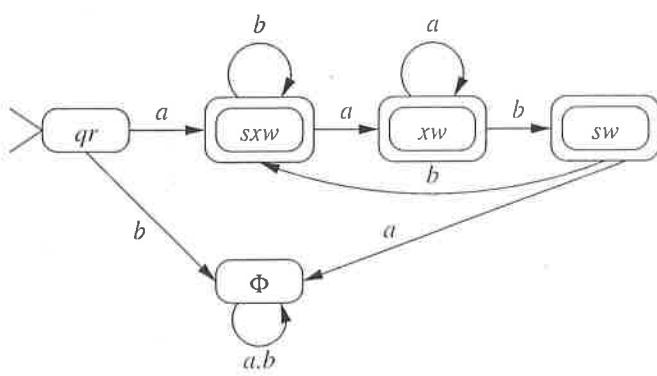


Figure 260:

- (c) Let \mathcal{S} be a class of languages over alphabet $\{a, b\}$, defined as follows:

Language L is a member of \mathcal{S} if and only if L is accepted by some non-deterministic finite

automaton, but is not accepted by any deterministic finite automaton.

What is the cardinality of \mathcal{S} ? Explain your answer briefly.

Answer:

$$|\mathcal{S}| = 0 \text{ since } \mathcal{S} = \emptyset$$

Every language accepted by some non-deterministic finite automaton is also accepted by some deterministic finite automaton. Hence, class \mathcal{S} is empty, and thereby of cardinality zero.

Problem 632 Let M be the finite automaton represented by the state diagram on Figure 261 and let L be the language accepted by M .

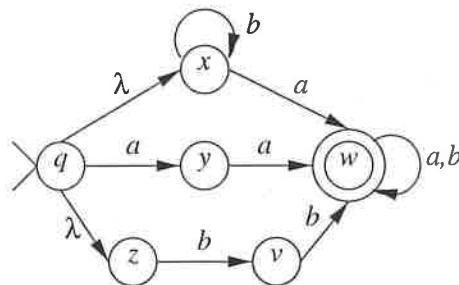


Figure 261:

Construct a deterministic finite automaton M' that accepts L and show your work. If such M' does not exist, explain why.

Problem 633 Let M be the finite automaton represented by the state diagram on Figure 262 and let L be the language accepted by M .

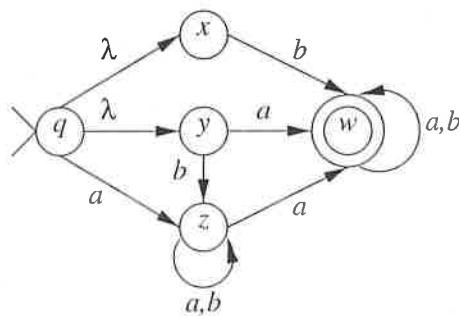


Figure 262:

- (a) Construct a deterministic finite automaton M' that accepts L and show your work. If such M' does not exist, prove it.

Answer: Let $M' = (Q', \{a, b\}, \delta', q'_0, F')$, where $Q' \in \mathcal{P}(\{q, w, x, y, z\})$.

The transition function δ of the original automaton M

is given by the following table:

δ	a	b	λ
q	$\{z\}$	\emptyset	$\{x, y\}$
x	\emptyset	$\{w\}$	\emptyset
y	$\{w\}$	$\{z\}$	\emptyset
z	$\{z, w\}$	$\{z\}$	\emptyset
w	$\{w\}$	$\{w\}$	\emptyset

The λ -closures of the states of M are:

ξ	$C(\xi)$
q	$\{q, x, y\}$
w	$\{w\}$
x	$\{x\}$
y	$\{y\}$
z	$\{z\}$

The initial state: $q'_0 = C(q) = \{q, x, y\}$.

The transition function δ' :

δ'	a	b
$\{q, x, y\}$	$\{z, w\}$	$\{z, w\}$
$\{z, w\}$	$\{z, w\}$	$\{z, w\}$

The set of states:

$$Q' = \{\{q, x, y\}, \{z, w\}\}.$$

The set of final states:

$$F' = \{\{z, w\}\}.$$

The state diagram of M' is given on Figure 263.

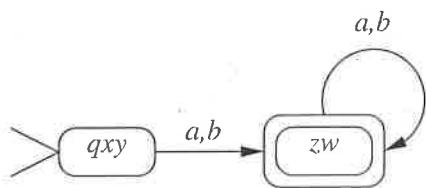


Figure 263:

(b) List 10 distinct strings over alphabet $\{a, b\}$ that are rejected by automaton M . If such strings do not exist, prove it.

Answer: By inspection of the diagram given on Figure 263, we see that every non-empty string from $\{a, b\}^*$ takes M' to its final state. There is only one string, λ , rejected by M' . By construction, M and M' are equivalent. Hence, M rejects only the empty string, and the other nine do not exist.

Problem 634 Let M be the finite automaton represented by the state diagram on Figure 264, and let L be the language accepted by M .

(a) Is the finite automaton M deterministic? Justify briefly your answer.

Answer: No—for example, there is no transition from state x on input symbol b .

(b) If M is not deterministic, construct a deterministic finite automaton M' that accepts L and show your work. If such automaton M' does not exist, explain why.

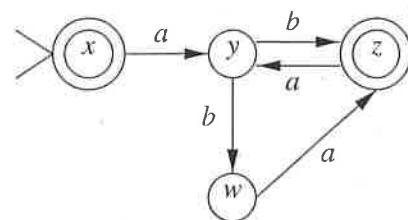


Figure 264:

Answer: Let $M' = (Q', \{a, b\}, \delta', q'_0, F')$, where $Q' \in \mathcal{P}(Q)$.

There are no λ -transitions; therefore, the λ -closure of every state is the singleton containing that state. Hence, $q'_0 = \{x\}$. Furthermore, the transition function of M and the input transition function t of M' are identical:

t	a	b
x	$\{y\}$	\emptyset
y	\emptyset	$\{w, z\}$
w	$\{z\}$	\emptyset
z	$\{y\}$	\emptyset

The transition function δ' :

δ'	a	b
$\{x\}$	$\{y\}$	\emptyset
$\{y\}$	\emptyset	$\{w, z\}$
$\{w, z\}$	$\{y, z\}$	\emptyset
$\{y, z\}$	$\{y\}$	$\{w, z\}$
\emptyset	\emptyset	\emptyset

The set of states:

$$Q' = \{\{x\}, \{y\}, \{w, z\}, \{y, z\}, \emptyset\}.$$

The set of final states:

$$F' = \{\{x\}, \{w, z\}, \{y, z\}\}.$$

The state diagram of M' is given on Figure 265.

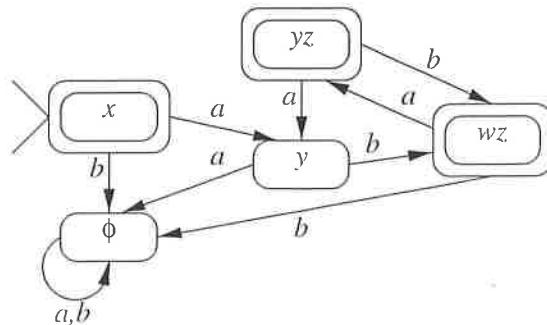


Figure 265:

Problem 635 Let L be the language defined by the regular expression:

$$a^*(ab)^*a^*$$

(a) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

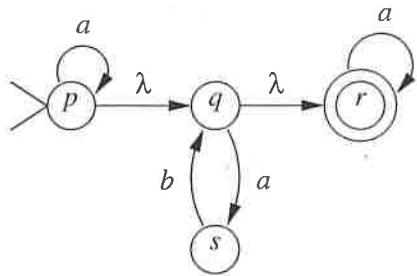


Figure 266:

Answer: See Figure 266.

(b) Draw a state-transition graph of a deterministic finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: Let M be the automaton constructed in the answer to part (a). We apply the conversion algorithm to construct a deterministic finite automaton M' that accepts L .

Answer: Let $M' = (Q', \{a, b\}, \delta', q'_0, F')$, where $Q' \in \mathcal{P}(Q)$. Transition function of M :

δ	a	b	λ
p	$\{p\}$	\emptyset	$\{q\}$
q	$\{s\}$	\emptyset	$\{r\}$
r	$\{r\}$	\emptyset	\emptyset
s	\emptyset	$\{q\}$	\emptyset

λ -closure:

x	$C(x)$
p	$\{p, q, r\}$
q	$\{q, r\}$
r	$\{r\}$
s	$\{s\}$

The initial state: $q'_0 = C(p) = \{p, q, r\}$.

The transition function δ' :

δ	a	b
$\{p, q, r\}$	$\{p, q, r, s\}$	\emptyset
$\{p, q, r, s\}$	$\{p, q, r, s\}$	$\{q, r\}$
$\{q, r\}$	$\{r, s\}$	\emptyset
$\{r, s\}$	$\{r\}$	$\{q, r\}$
$\{r\}$	$\{r\}$	\emptyset
\emptyset	\emptyset	\emptyset

The set of states:

$$Q' = \{\{p, q, r\}, \{p, q, r, s\}, \{q, r\}, \{r, s\}, \{r\}, \emptyset\}.$$

The set of final states:

$$F' = \{\{p, q, r\}, \{p, q, r, s\}, \{q, r\}, \{r, s\}, \{r\}\}.$$

The state diagram of M' is given on Figure 267.

Problem 636 Let L be the language accepted by the finite automaton $M = (\{w, x, y, z\}, \{a, b\}, \delta, x, \{z\})$, where δ is given by the following table:

	a	λ	
		b	
w	\emptyset	$\{x, w\}$	\emptyset
x	$\{y\}$	\emptyset	$\{z\}$
y	\emptyset	$\{z\}$	$\{y\}$
z	$\{z, w\}$	\emptyset	\emptyset

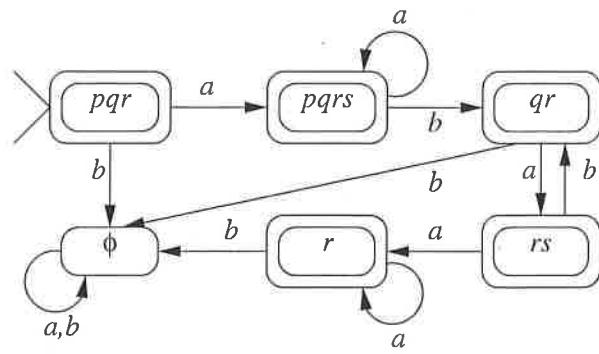


Figure 267:

Write a complete formal definition or a state-transition graph of a deterministic finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: Let $M' = (Q', \{a, b\}, \delta', q'_0, F')$, where $Q' \in \mathcal{P}(\{w, x, y, z\})$. The λ -closures of the states of M are:

q	$C(q)$
w	$\{w\}$
x	$\{x, z\}$
y	$\{y\}$
z	$\{z\}$

The initial state: $q'_0 = C(x) = \{x, z\}$.

The transition function δ' :

δ'	a	b
$\{x, z\}$	$\{y, z, w\}$	\emptyset
$\{y, z, w\}$	$\{z, w\}$	$\{x, z, w\}$
$\{z, w\}$	$\{z, w\}$	$\{x, z, w\}$
$\{x, z, w\}$	$\{y, z, w\}$	$\{x, z, w\}$
\emptyset	\emptyset	\emptyset

The set of states:

$$Q' = \{\{x, z\}, \{y, z, w\}, \{z, w\}, \{x, z, w\}, \emptyset\}.$$

The set of final states:

$$F' = \{\{x, z\}, \{y, z, w\}, \{z, w\}, \{x, z, w\}\}.$$

The state diagram of M' is given on Figure 268.

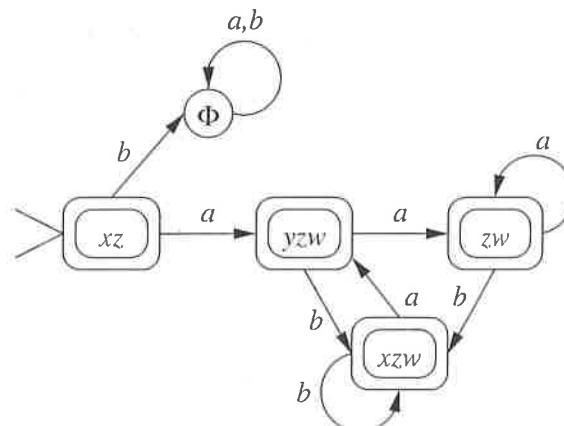


Figure 268:

Problem 637 Let L be the set of all strings over alphabet $\{a, b, c\}$ that begin with cc .

(a) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 269.

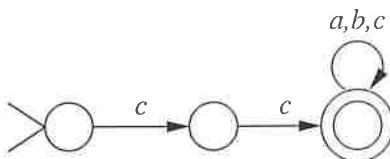


Figure 269:

(b) Construct a state-transition graph of a finite automaton that accepts \overline{L} (the complement of L .) If such an automaton does not exist, prove it.

Answer: See Figure 270.

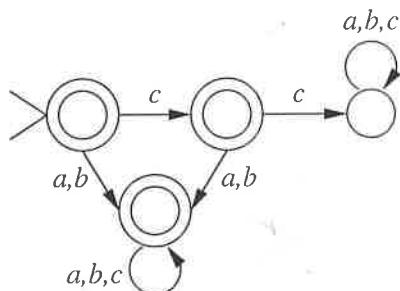


Figure 270:

(c) Explain how to construct an algorithm that solves the following problem:

INPUT: A regular expression e .

OUTPUT: A finite state automaton that accepts the complement of the language defined by e .

If this algorithm does not exist, prove it.

Answer: The construction has three steps. First, we apply the known algorithm to convert the regular expression e into a finite automaton, say F ; next, we apply the known algorithm to convert the automaton F into a deterministic finite automaton, say D ; and, finally, we apply the known algorithm to convert D into a finite automaton that accepts the complement of $L(D)$.

Problem 638 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \mid BA \\ A &\rightarrow ab \\ B &\rightarrow Bc \mid d \end{aligned}$$

(a) List 5 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: See part (b).

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$abdc^* \cup dc^*ab$$

(c) Dangerous Professor has told her students to construct a context-free grammar for \overline{L} (the complement of L .) Explain how to construct this grammar, or prove that it does not exist.

Answer: The construction proceeds as a sequence of four algorithmic conversions, as follows.

1. convert the regular expression obtained in the answer to part (a) to a finite automaton, say F ;
2. convert the finite automaton F to a deterministic finite automaton F_1 ;
3. convert the finite automaton F_1 into another deterministic finite automaton F_2 , which accepts the complement \overline{L} .
4. convert the deterministic finite automaton F_2 into a context-free grammar, which is the result.

Problem 639 Let M be the finite automaton represented by the state diagram on Figure 271, and let L be the language accepted by M .

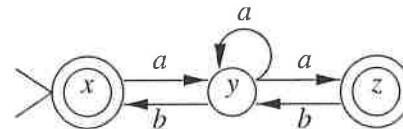


Figure 271:

Construct a regular expression that defines L and show your work. If such regular expression does not exist, prove it.

Answer:

$$a(a \cup ab \cup ba)^*(a \cup b) \cup \lambda$$

The sequence of expression graphs equivalent to M is represented on Figure 272.

Problem 640 Let M be the finite automaton represented by the state diagram on Figure 273, and let L be the language accepted by M .

Construct a regular expression that defines L and show your work. If such a regular expression does not exist, explain why.

Answer:

$$(a \cup b)a^*b(a(a \cup b)a^*b \cup ba^*b)^* \cup \epsilon$$

The sequence of generalized finite automata equivalent to M is represented on Figure 274.

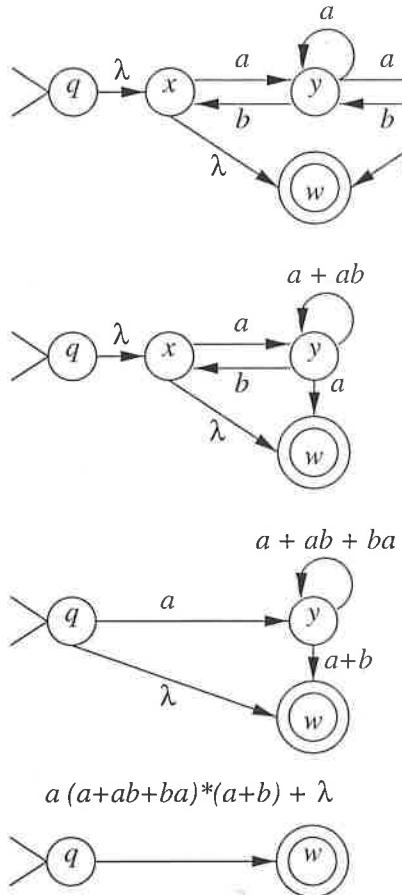


Figure 272:

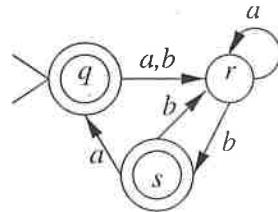


Figure 273:

Problem 641 Let L be the language accepted by the finite automaton M represented on Figure 275.

(a) Draw a generalized expression graph obtained from the automaton M by the elimination of node t , according to the algorithm for conversion of finite automata to regular expressions. If it is impossible to eliminate node t , prove it.

Answer: Apply the known conversion algorithm to produce the generalized expression graph given on Figure 276.

(b) Write a complete formal definition of a *regular context-free grammar* that generates L . If such a grammar does not exist, prove it.

Answer: Apply the known conversion algorithm to produce the following grammar.

$$G = (V, \Sigma, P, y), \text{ where } \Sigma = \{a, b, c, d\}, V = \{y, q, s, t\},$$

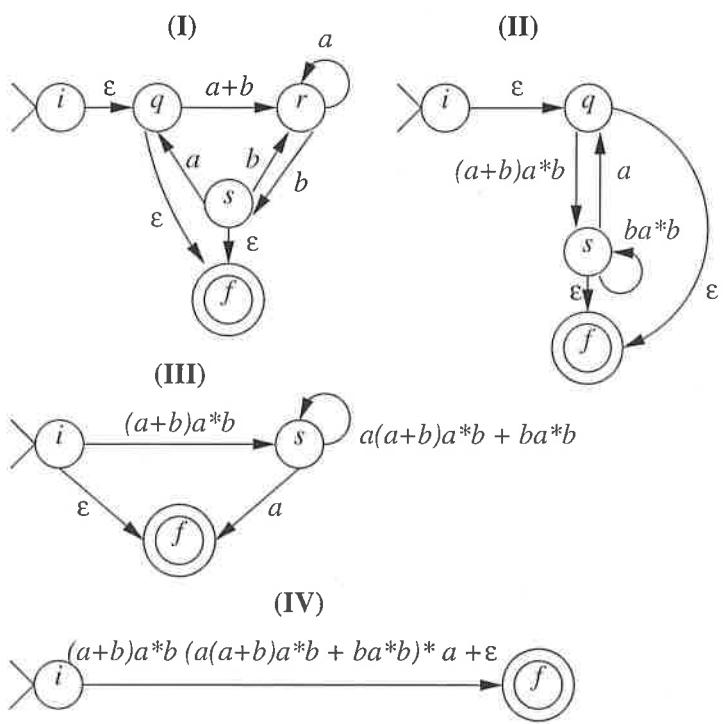


Figure 274:

and the production set P is:

$$\begin{aligned} y &\rightarrow cq \mid as \mid ct \mid \lambda \\ q &\rightarrow aq \mid cs \mid b \\ s &\rightarrow ds \mid bq \mid at \mid a \\ t &\rightarrow dt \mid ay \mid as \mid b \mid \lambda \end{aligned}$$

Problem 642 Let L be the language accepted by the finite automaton M represented on Figure 277.

(a) Write a complete formal definition of a *regular context-free grammar* that generates L . If such a grammar does not exist, prove it.

Answer: The algorithm for conversion of finite automata into regular grammars directly applies to the automaton given on Figure 277 (because this automaton is already in the required form) yielding the grammar: $G = (V, \Sigma, P, y)$, where $\Sigma = \{a, b, c\}$, $V = \{y, q, s, t, w\}$, and P is:

$$\begin{aligned} y &\rightarrow cq \mid as \mid \lambda \\ q &\rightarrow aq \mid b \mid cs \\ s &\rightarrow bq \mid a \mid at \\ t &\rightarrow dt \mid as \mid ay \mid \lambda \end{aligned}$$

(b) Draw a generalized expression graph obtained from the automaton M by the elimination of node q , according to the algorithm for conversion of finite automata to regular expressions. If it is impossible to eliminate node q , prove it.

Answer: See Figure 278.

Problem 643 Consider the finite automaton M represented by the state transition graph given on Figure 279.

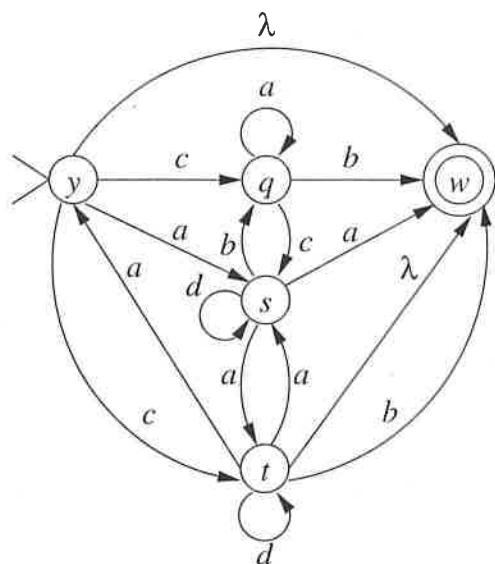


Figure 275:

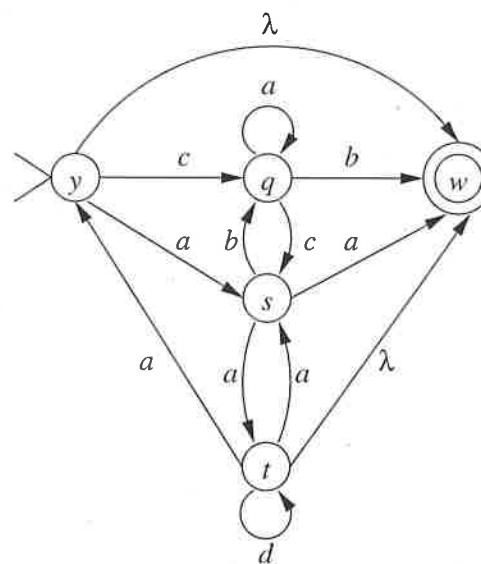


Figure 277:

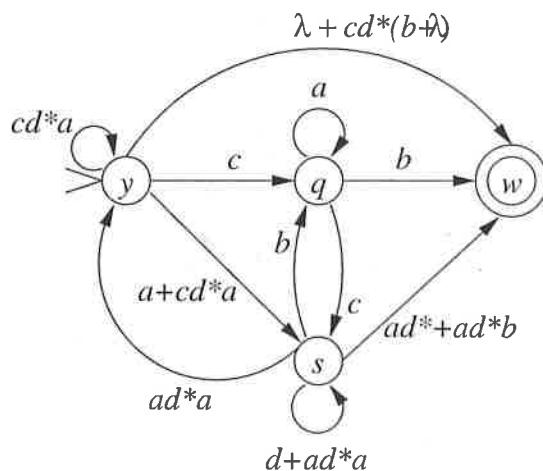


Figure 276:

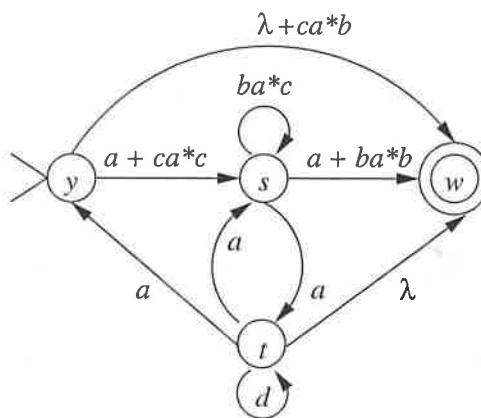


Figure 278:

Draw the generalized expression graph obtained from the automaton M after the elimination of node Z , according to the algorithm for conversion of finite automata to regular expressions. If it is impossible to eliminate node Z , state it and explain why.

Answer: See Figure 280.

Problem 644 Consider the finite automaton M represented by the state-transition graph given on Figure 281.

Draw a generalized expression graph obtained from the automaton M by the elimination of node B , according to the algorithm for conversion of finite automata to regular expressions. If it is impossible to eliminate node B , prove it.

Answer: See Figure 282.

Problem 645 Let L be the language accepted by the finite automaton M represented on Figure 283.

- (a) Write a complete formal definition of a *regular* context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: Apply the algorithm for conversion of finite automata to regular context-free grammars. Conveniently, the automaton is already in the required canonical form, whence the grammar: $G = (V, \Sigma, P, y)$, where $\Sigma = \{a, b, c, d\}$, $V = \{y, q, s, t\}$, and P is:

$$\begin{aligned} y &\rightarrow cq \mid as \mid \lambda \\ q &\rightarrow aq \mid cs \mid b \\ s &\rightarrow at \mid bq \mid a \\ t &\rightarrow ay \mid dt \mid as \mid \lambda \end{aligned}$$

- (b) Draw a generalized expression graph obtained from the automaton M by the elimination of node s , according to the algorithm for conversion of finite automata to regular expressions. If it is impossible to eliminate node s , prove it.

Answer: See Figure 284.

Problem 646 Consider the finite automaton M rep-

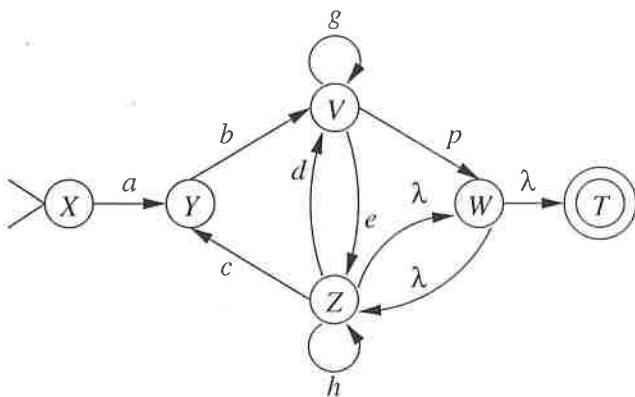


Figure 279:

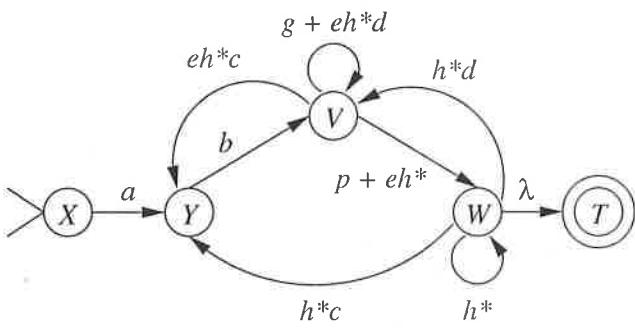


Figure 280:

resented by the state transition graph given on Figure 285.

Draw the generalized expression graph obtained from the automaton M by the elimination of node X , according to the algorithm for the conversion of finite automata to regular expressions. If it is impossible to eliminate node X , prove it.

Answer: See Figure 286.

Problem 647 Consider the finite automaton M represented by the state transition graph given on Figure 287.

Draw the generalized expression graph obtained from the automaton M by the elimination of node X , according to the algorithm for the conversion of finite automata to regular expressions. If it is impossible to eliminate node X , prove it.

Answer: See Figure 288.

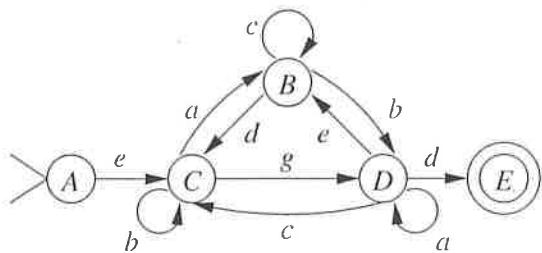


Figure 281:

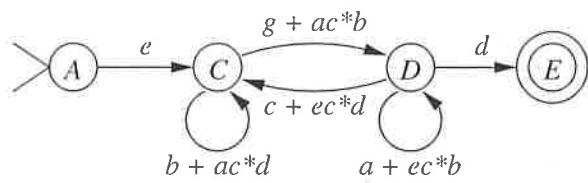


Figure 282:

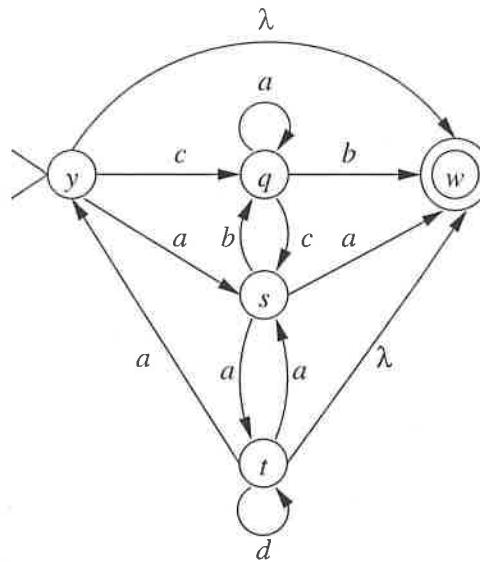


Figure 283:

Problem 648 Consider the finite automaton M represented by the state transition graph given on Figure 289.

Draw the generalized expression graph obtained from the automaton M by the elimination of node X , according to the algorithm for the conversion of finite automata to regular expressions. If it is impossible to eliminate node X , prove it.

Answer: See Figure 290.

Problem 649 Consider the finite automaton M represented by the state transition graph given on Figure 291.

(a) Draw the generalized expression graph obtained from the automaton M after elimination of node B , according to the algorithm for conversion of finite automata to regular expressions. If it is impossible to eliminate node B , prove it.

Answer: See Figure 292.

(b) Write a complete formal definition of a regular context-free grammar that generates the language accepted by M . If such a grammar does not exist, prove it.

Answer: The automaton is compliant with the required input form, hence the algorithm for conversion of a finite automaton to a regular grammar is applied directly, yielding the following result.

$G = (V, \Sigma, P, A)$, where $\Sigma = \{a, b, c, d, e, g, h\}$,

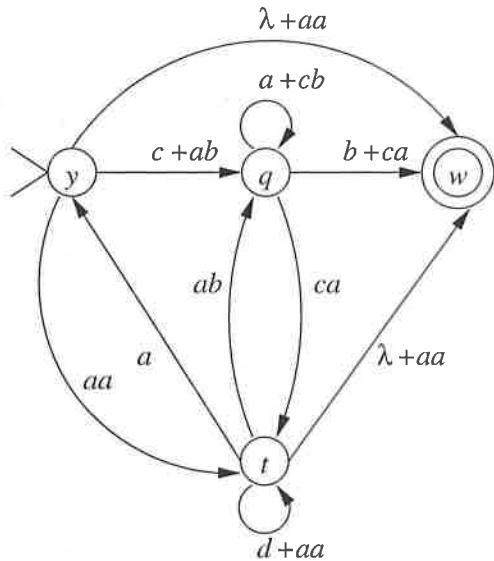


Figure 284:

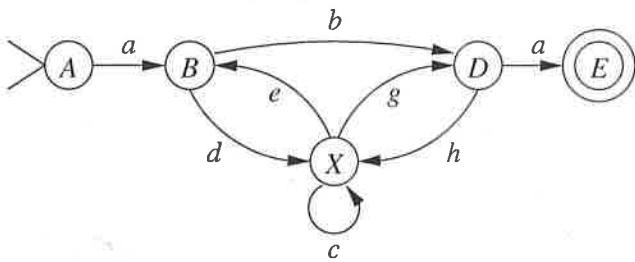


Figure 285:

$V = \{A, B, E, F\}$, and the production set P is:

$$\begin{aligned} A &\rightarrow aF \mid bB \mid cE \\ B &\rightarrow eB \mid \lambda \\ E &\rightarrow hB \mid \lambda \\ F &\rightarrow dF \mid gB \mid \lambda \end{aligned}$$

Problem 650 Consider the finite automaton M represented by the state transition graph given on Figure 293.

(a) Draw the generalized expression graph obtained from the automaton M after elimination of node B , according to the algorithm for conversion of finite automata to regular expressions. If it is impossible to eliminate node B , prove it.

Answer: See Figure 294.

(b) Write a complete formal definition of a regular

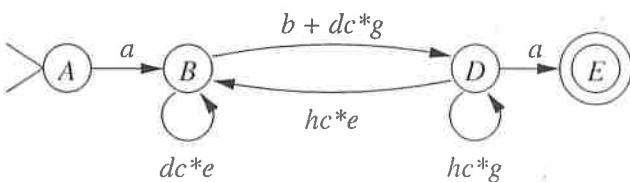


Figure 286:

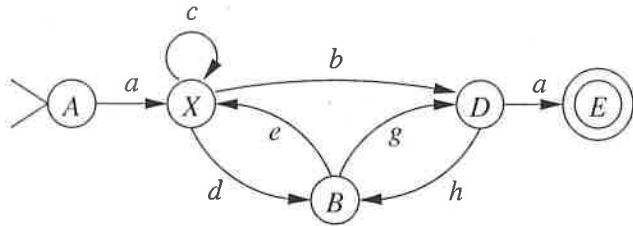


Figure 287:

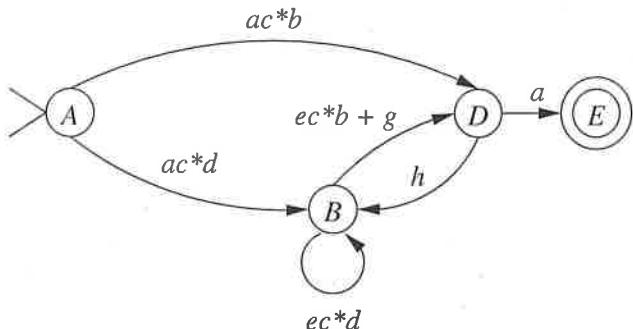


Figure 288:

context-free grammar that generates the language accepted by M . If such a grammar does not exist, prove it.

Answer: The automaton is compliant with the required input form, hence the algorithm for conversion of a finite automaton to a regular grammar is applied directly, yielding the following result.

$G = (V, \Sigma, P, A)$, where $\Sigma = \{a, b, c, d, e, g, h\}$, $V = \{A, B, E, F\}$, and the production set P is:

$$\begin{aligned} A &\rightarrow aF \mid bB \mid cE \\ B &\rightarrow eB \mid pE \mid \lambda \\ E &\rightarrow hB \mid \lambda \\ F &\rightarrow dF \mid gB \mid \lambda \end{aligned}$$

Problem 651 Consider the finite automaton given on Figure 295.

Draw the generalized expression graph obtained from this automaton by an elimination of the node X , according to the algorithm for conversion of a finite automaton to a regular expression, and explain your answer. If such an algorithm does not exist, prove it.

Advice for Answer: See Figure 296.

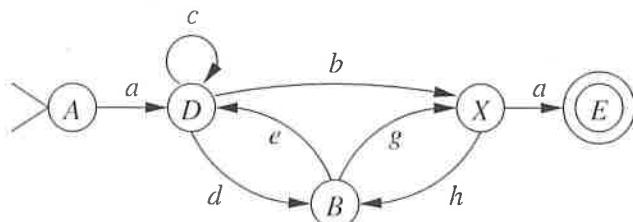


Figure 289:

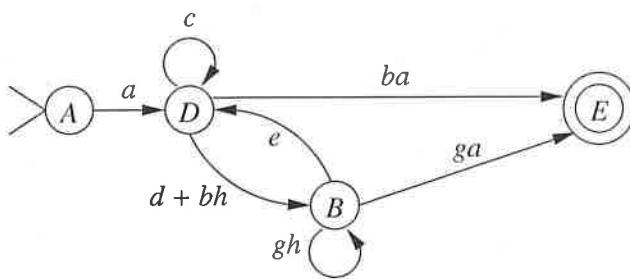


Figure 290:

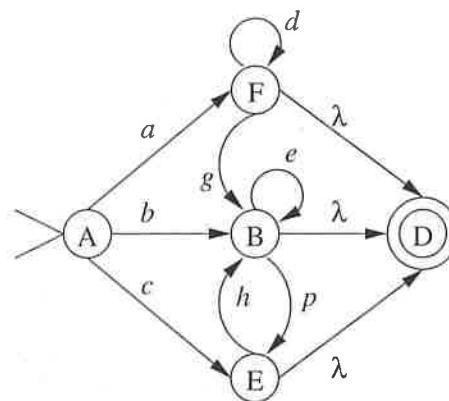


Figure 293:

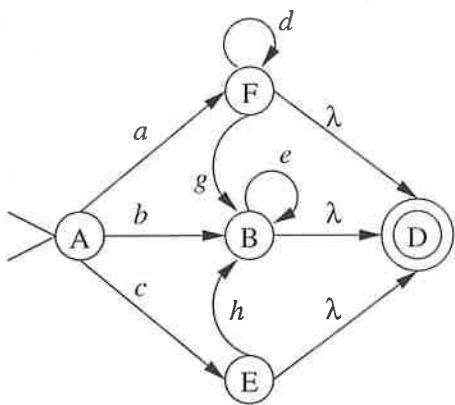


Figure 291:

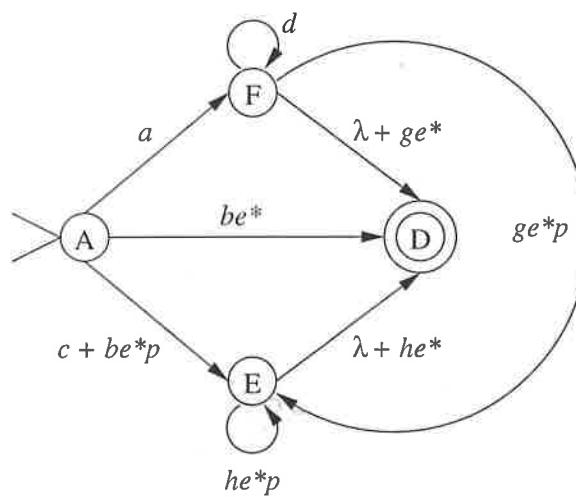


Figure 294:

Problem 652 For each of the following claims, circle the word “yes” that follows the claim if the claim is correct, and circle the word “no” that follows the claim if the claim is not correct.

1. The class of context-free languages is closed under concatenation. **yes**
2. The Kleene star of any regular language is regular. **yes**
3. The Kleene star of any regular language is context-free. **yes**
4. The union of any two context-free languages is context-free. **yes**

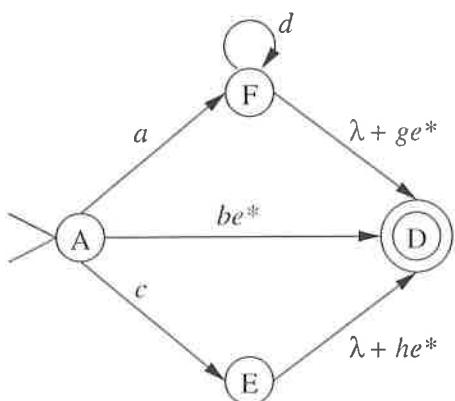


Figure 292:

5. Every finite language is regular. **yes**
6. There exists an algorithm to convert any regular expression into an equivalent finite automaton. **yes**
7. There exists an algorithm to convert any regular expression into an equivalent context-free grammar. **yes**
8. Every deterministic finite automaton is equivalent to some non-deterministic finite automaton. **yes**
9. Every non-deterministic finite automaton is equivalent to some deterministic finite automaton. **yes**
10. Every context-free grammar is equivalent to some regular expression. **no**
11. Set of strings of the form $\{a^k b^k c^k \mid k \geq 0\}$ is regular. **no**
12. Set of strings of the form $\{a^k b^k c^k \mid k \geq 0\}$ is context-free. **no**
13. Set of strings of the form $\{a^k b^k \mid k \geq 0\}$ is regular. **no**

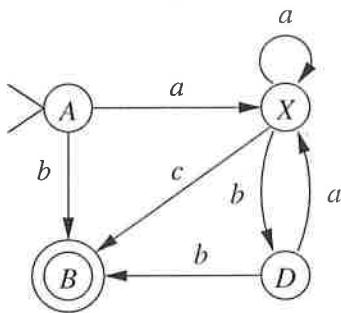


Figure 295:

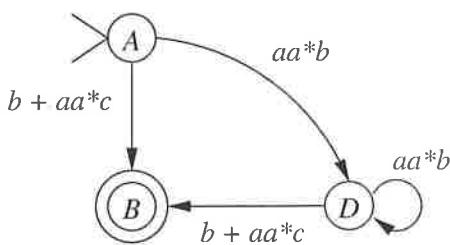


Figure 296:

14. Set of strings of the form $\{a^k b^k \mid k \geq 0\}$ is context-free. **yes**
15. Every language has a finite description. **no**
16. Every finite language has a finite description. **yes**
17. Set $\{a, b\}^*$ is regular. **yes**
18. Every subset of $\{a, b\}^*$ is regular. **no**
19. Every subset of $\{a, b\}^*$ is context-free. **no**

Problem 653 For each of the following claims, circle the word “yes” that follows the claim if the claim is correct, and circle the word “no” that follows the claim if the claim is not correct.

1. The concatenation of any two regular languages is context-free. **yes**
2. The Kleene star of any regular language is context-free. **yes**
3. Every subset of a regular language is regular. **no**
4. Some context-free languages are not regular. **yes**
5. Some finite languages are not regular. **no**
6. Some regular languages are not finite. **yes**
7. The union of any two regular languages is regular. **yes**
8. Language $\{a^n b^m c^k \mid k = m + n, \quad k, m, n \geq 0\}$ is context-free. **yes**

9. Language $\{a^n b^m c^k \mid k, m, n \geq 0\}$ is not regular. **no**
10. Language $\{a^n b^m c^k d^\ell \mid n = k \wedge m = \ell, \quad k, m, n, \ell \geq 0\}$ is context-free. **no**
11. Language $\{a^n b^m c^k d^\ell \mid n = k \vee m = \ell, \quad k, m, n, \ell \geq 0\}$ is context-free. **yes**
12. Every regular language has a context-free subset. **yes**
13. Every infinite context-free language has an infinite regular superset. **yes**
14. Language $\{a^n b^m c^k d^\ell \mid n = k \wedge m = \ell, \quad k, m, n, \ell \geq 0\}$ has a proper subset which is context-free and infinite. **yes**
15. Every infinite language has a finite description. **no**
16. Every finite language is generated by some context-free grammar. **yes**
17. Some infinite languages cannot be described by a regular expression. **yes**
18. Set $\{a, b\}^*$ has a proper subset which is infinite and not context-free. **yes**
19. The intersection of $\{a^n b^m c^m \mid m, n \geq 0\}$ and $\{a^n b^n c^m \mid m, n \geq 0\}$ is context-free. **no**
20. The intersection of $\{a^n b^m c^m \mid m, n \geq 0\}$ and $\{a^n b^n c^m \mid m, n \geq 0\}$ has a context-free complement. **yes**

Specific instructions for Problems 654–658:

Fill the empty box at the end of each of the numbered claims with one of the following three signs:

✓ — ?

so that the signs have the following meaning.

- ✓ means that the preceding claim is always true;
- — means that the preceding claim is always false;
- ? means that the preceding claim may be true but may be false, depending on the values of the variable(s) appearing in the claim.

Problem 654 Problem variables:

e is a regular expression.

L is a language represented by e .

Claims:

1. L is context-free. _____

2. L is finite. _____ [?]
3. The complement \bar{L} of L is finite. _____ [?]
4. There exists a non-deterministic finite automaton that accepts L . _____ [✓]
5. There exists a deterministic finite automaton that accepts L , but there does not exist an algorithm to construct this automaton. _____ [-]
6. There exists an algorithm to convert the expression e into an equivalent regular context-free grammar. _____ [✓]

Problem 655 Problem variables:

M is an arbitrary non-deterministic finite automaton. L is the language accepted by M .

Claims:

1. There exists a context-free grammar equivalent to M . _____ [✓]
2. There exists a regular context-free grammar equivalent to M , but there does not exist an algorithm to construct it. _____ [-]
3. There exists a deterministic finite automaton equivalent to M , and there exists an algorithm to construct it. _____ [✓]
4. L is infinite and countable. _____ [?]
5. \bar{L} (the complement of L) is regular. _____ [✓]
6. \bar{L} (the complement of L) is infinite and countable. _____ [?]

Problem 656 Problem variables:

L_1 is an arbitrary regular language; L_2 is an arbitrary context-free language; e_1 is a regular expression that represents L_1 ; G_2 is a context-free grammar that generates L_2 .

Claims:

1. L_1 is context-free. _____ [✓]
2. L_2 is regular. _____ [?]
3. L_2^* is context-free. _____ [✓]
4. L_1 has a regular complement. _____ [✓]
5. $L_1 \cap L_2$ is regular. _____ [?]
6. $L_1 \cap L_2$ is context-free. _____ [✓]

7. L_2 has a context-free complement. _____ [?]
8. $L_1 L_2$ is context-free. _____ [✓]
9. There exists an algorithm which on input e_1 outputs a regular context-free grammar for L_1 . _____ [✓]
10. There exists an algorithm which on input G_2 outputs a regular context-free grammar for L_2 . _____ [-]

Problem 657 Problem variables:

G is a regular grammar over the alphabet $\Sigma = \{a, b, c\}$. $L(G)$ is the language generated by G (and $\bar{L}(G)$ is the complement of $L(G)$.)

Claims:

1. $L(G)$ is infinite. _____ [?]
2. $L(G)$ is regular. _____ [✓]
3. $L(G)$ is context-free. _____ [✓]
4. $\bar{L}(G)$ is not infinite. _____ [?]
5. $\bar{L}(G)$ is not regular. _____ [-]
6. $\bar{L}(G)$ is not context-free. _____ [-]
7. $\bar{L}(G)$ need not be regular, but there exists an algorithm to determine whether it is regular. _____ [-]
8. $\bar{L}(G)$ need not be context-free, and there is no algorithm to determine whether $\bar{L}(G)$ is context-free. _____ [-]

Problem 658 Problem variables:

F is a finite automaton over the alphabet $\Sigma = \{a, b, c\}$. $L(F)$ is the language accepted by F .

e is a regular expression over the alphabet $\Sigma = \{a, b, c\}$. L_e is the language represented by e .

Claims:

1. $L(F) \cup L_e$ is equivalent to some regular expression, and there exists an algorithm to construct this regular expression. _____ [✓]
2. $L(F) \cap L_e$ is regular. _____ [✓]
3. $L(F) \cap L_e$ may be regular, but there is no algorithm to determine whether $L(F) \cap L_e$ is regular. _____ [-]
4. $\bar{L}(F)$ (the complement of $L(F)$) need not be regular but there exists an algorithm to determine whether $\bar{L}(F)$ is regular. _____ [-]

5. $L(F) \cap L_e$ may be non-regular but it is context-free.

6. F is deterministic. _____ ?

7. F need not be deterministic but there exists an algorithm to convert it into a deterministic equivalent.

8. F need not have an equivalent deterministic finite automaton, but there exists an algorithm to determine whether it has a deterministic equivalent.

5 Push-Down Automata

Problem 659 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, s\}$; $\Sigma = \{a, b\}$; $\Gamma = \{B\}$; $F = \{s\}$; and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, s, B] \\ & [s, a, \lambda, s, \lambda] \\ & [s, b, B, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack)

- (a) List 4 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer: $ab, aab, aba, aaba$

- (b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$a a^* b a^*$$

Problem 660 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, s\}$; $\Sigma = \{a, b\}$; $\Gamma = \{B\}$; $F = \{s\}$; and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, q, B] \\ & [q, a, \lambda, s, B] \\ & [s, b, B, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack)

- (a) List 4 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: The template for L is:

$$a^{n+1} b^{n+1}, n \geq 0$$

- (b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSb \mid ab$$

Problem 661 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, t, p, s\}$; $\Sigma = \{a, b, c, d\}$; $\Gamma = \{A, D\}$; $F = \{s\}$ and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, \lambda, q, A] \\ & [t, c, A, t, \lambda] \\ & [p, d, \lambda, p, D] \\ & [s, b, D, s, \lambda] \\ & [q, \lambda, \lambda, t, \lambda] \\ & [t, \lambda, \lambda, p, \lambda] \\ & [p, \lambda, \lambda, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

- (a) Write 6 distinct strings that belong to L . If such strings do not exist, state it and prove it.

Advice for Answer: Template:

$$a^n c^n d^k b^k$$

- (b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b, c, d\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S & \rightarrow AB \\ A & \rightarrow aAc \mid \lambda \\ B & \rightarrow dBb \mid \lambda \end{aligned}$$

Problem 662 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, t, s\}$; $\Sigma = \{a, b, c, d\}$; $\Gamma = \{A, D\}$; $F = \{s\}$ and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, \lambda, q, A] \\ & [t, d, \lambda, t, D] \\ & [s, b, D, s, \lambda] \\ & [s, c, A, s, \lambda] \\ & [q, \lambda, \lambda, t, \lambda] \\ & [t, \lambda, \lambda, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

- (a) Write 6 distinct strings that belong to L . If such strings do not exist, state it and prove it.

Advice for Answer: Template:

$$a^n d^k b^k c^n$$

- (b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b, c, d\}$, $V = \{S, A\}$, and P is:

$$\begin{aligned} S & \rightarrow aSc \mid A \\ A & \rightarrow dAb \mid \lambda \end{aligned}$$

Problem 663 Let L_5 be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where:
 $Q = \{q, s, t, x, y, v\}$;
 $\Sigma = \{a, b, c, g, h, f\}$; $\Gamma = \{E, M, O, P, T, Y\}$; $F = \{v\}$ and δ is defined by the following transition set:

$[q, b, \lambda, x, \lambda]$	$[t, a, O, t, \lambda]$	$[s, a, O, s, \lambda]$
$[q, a, \lambda, y, \lambda]$	$[t, b, T, t, \lambda]$	$[s, b, T, s, \lambda]$
$[x, f, \lambda, t, EMPTY]$	$[t, c, E, x, \lambda]$	$[s, c, E, s, \lambda]$
$[y, h, \lambda, s, POT]$	$[t, g, Y, t, \lambda]$	$[s, g, Y, s, \lambda]$
$[x, \lambda, \lambda, v, \lambda]$	$[t, h, M, t, \lambda]$	$[s, h, M, s, \lambda]$
$[y, \lambda, \lambda, v, \lambda]$	$[t, f, P, t, \lambda]$	$[s, f, P, y, \lambda]$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write a regular expression that represents L_5 . If such a regular expression does not exist, prove it.

Answer:

$$b(fgbfhc)^* \cup a(hbaf)^*$$

(b) Write a complete formal definition of a context-free grammar that generates L_5 . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, g, h, f\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow bA \mid aB \\ A &\rightarrow \lambda \mid AA \mid fgbfhc \\ B &\rightarrow \lambda \mid BB \mid hbaf \end{aligned}$$

Problem 664 Let L_5 be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where:
 $Q = \{q, t\}$; $\Sigma = \{a, b, c, g, h, f\}$;
 $\Gamma = \{A, D, I, N, R, Y\}$; $F = \{q\}$ and δ is defined by the following transition set:

$[q, c, \lambda, t, RAINY]$	$[t, a, A, t, \lambda]$	$[t, g, Y, t, \lambda]$
$[q, h, \lambda, t, DAY]$	$[t, b, R, q, \lambda]$	$[t, h, N, t, \lambda]$
	$[t, c, D, q, \lambda]$	$[t, f, I, t, \lambda]$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write a regular expression that represents L_5 . If such a regular expression does not exist, prove it.

Answer:

$$(cghfab \cup hgac)^*$$

(b) Write a complete formal definition of a context-free grammar that generates L_5 . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, g, h, f\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow \lambda \mid SS \mid cghfab \mid hgac$$

Problem 665 Let L be the language accepted by the pushdown automaton:

$M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, p\}$;
 $\Sigma = \{a, b, c, g\}$; $\Gamma = \{A, D, E, L, P, R\}$; $F = \{q\}$ and δ is defined by the following transition set:

$[q, a, \lambda, p, RED]$	$[p, a, A, q, \lambda]$
$[q, g, \lambda, p, APPLE]$	$[p, g, D, p, \lambda]$
	$[p, c, E, p, \lambda]$
	$[p, b, L, p, \lambda]$
	$[p, g, P, p, \lambda]$
	$[p, b, R, q, \lambda]$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write a regular expression that represents L . If such a regular expression does not exist, prove it.

Answer:

$$(agcb \cup gcbgga)^*$$

(b) Draw a state-transition graph of a finite-state automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 297.

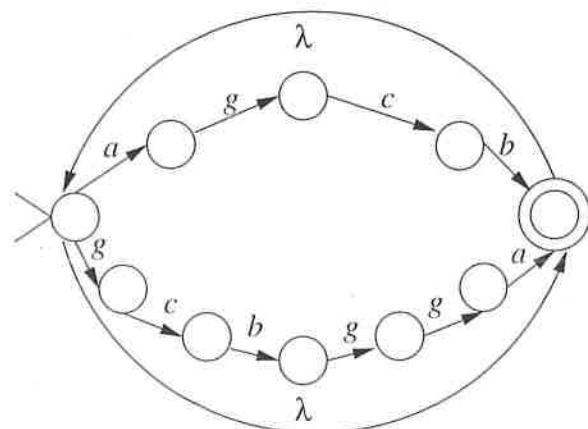


Figure 297:

(c) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, g\}$, $V = \{S\}$, and P is:

$$S \rightarrow SS \mid \lambda \mid agcb \mid gcbgga$$

Problem 666 Let L be the language accepted by the pushdown automaton:

$M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, p\}$;
 $\Sigma = \{a, b, c, g\}$; $\Gamma = \{A, B, D, E, F, H, R, S\}$; $F = \{p\}$ and δ is defined by the following transition set:

$[q, g, \lambda, q, FRESH]$	$[p, a, A, p, \lambda]$
$[q, a, \lambda, q, BREAD]$	$[p, b, B, p, \lambda]$
$[q, b, \lambda, p, \lambda]$	$[p, g, D, p, \lambda]$
	$[p, c, E, p, \lambda]$
	$[p, b, F, p, \lambda]$
	$[p, g, H, p, \lambda]$
	$[p, b, R, p, \lambda]$
	$[p, c, S, p, \lambda]$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, g\}$, $V = \{S\}$, and P is:

$$S \rightarrow gSgccbb \mid aSgacbb \mid b$$

(b) Explain how to construct an algorithm that solves the following problem:

INPUT: Pushdown automaton P_1 that accepts a language $L(P_1)$ and a Turing Machine T_2 that accepts a language $L(T_2)$;

OUTPUT: Turing Machine T_3 that accepts the language $L(T_2) \setminus L(P_1)$;

If this algorithm does not exist, prove it.

Answer: T_3 will simulate P_1 and T_2 in parallel, and halt if and only if P_1 rejects and T_2 halts.

Problem 667 Let L be the language accepted by the pushdown automaton:

$M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, p, s, t\}$;
 $\Sigma = \{a, b, c, g\}$; $\Gamma = \{A, C, D, E, L, O, T\}$; $F = \{s\}$ and δ is defined by the following transition set:

$[q, a, \lambda, q, COLD]$	$[s, a, A, s, \lambda]$
$[t, b, \lambda, t, TEA]$	$[s, c, E, s, \lambda]$
	$[s, g, T, s, \lambda]$
$[q, a, \lambda, p, \lambda]$	$[p, c, C, p, \lambda]$
$[p, b, \lambda, t, \lambda]$	$[p, g, D, p, \lambda]$
$[t, c, \lambda, s, \lambda]$	$[p, g, L, p, \lambda]$
	$[p, a, O, p, \lambda]$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on

the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, g\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow AbB \\ A &\rightarrow aAgac | a \\ B &\rightarrow bBacg | c \end{aligned}$$

(b) Explain how to construct an algorithm that solves the following problem:

INPUT: Pushdown automaton P_1 that accepts a language $L(P_1)$ and a pushdown automaton P_2 that accepts a language $L(P_2)$;

OUTPUT: Pushdown automaton P_3 that accepts the language $L(P_1) \cap L(P_2)$.

If this algorithm does not exist, prove it.

Answer: Impossible—there exist pairs of context free languages whose intersection is not context free.

Problem 668 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$Q = \{q, t\}; \Sigma = \{a, b, c\}; \Gamma = \{A\}; F = \{t\};$$

and the transition function δ is defined as follows:

$[q, a, \lambda, q, A]$
$[q, a, \lambda, t, \lambda]$
$[t, b, A, t, \lambda]$
$[t, c, A, t, \lambda]$

(Recall that M is defined so as to accept by final state and empty stack.)

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$ is the set of terminals; $V = \{S\}$ is the set of variables; S is the start symbol, and the production set P is:

$$S \rightarrow aSb \mid aSc \mid a$$

Problem 669 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$Q = \{q, t\}; \Sigma = \{a, b, c\}; \Gamma = \{A\}; F = \{t\};$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, q, A] \\ & [q, b, \lambda, q, A] \\ & [q, c, \lambda, t, \lambda] \\ & [t, c, A, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSc \mid bSc \mid c$$

Problem 670 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$Q = \{q, t\}; \Sigma = \{a, b, c\}; \Gamma = \{A\}; F = \{t\};$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, t, A] \\ & [t, b, A, t, \lambda] \\ & [t, c, \lambda, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$a c^* b c^*$$

Problem 671 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$Q = \{q, t\}; \Sigma = \{a, b, c\}; \Gamma = \{A\}; F = \{t\};$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, t, AA] \\ & [t, b, A, t, \lambda] \\ & [t, c, \lambda, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$a c^* b c^* b c^*$$

Problem 672 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F) \text{ where: } Q = \{q, p, t\};$$

$\Sigma = \{a, b, c, d\}; \Gamma = \{A, E, K, D\}; F = \{t\}$ and δ is defined by the following transition set:

$$\begin{array}{ll} [q, a, \lambda, q, EEK] & [t, a, A, t, \lambda] \\ [q, \lambda, \lambda, p, \lambda] & [t, b, E, t, \lambda] \\ [p, b, \lambda, p, DA] & [t, c, K, t, \lambda] \\ [p, \lambda, \lambda, t, \lambda] & [t, d, D, t, \lambda] \end{array}$$

(Recall that M is defined so as to accept by final state and empty stack.) Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A\}$, and P is:

$$\begin{aligned} S &\rightarrow aScbb \mid A \\ A &\rightarrow bAad \mid \lambda \end{aligned}$$

(b) Write a complete formal definition of a context-free grammar that generates L^* . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, T)$, where $\Sigma = \{a, b, c, d\}$, $V = \{T, S, A\}$, and P is:

$$\begin{aligned} T &\rightarrow \lambda \mid TT \mid S \\ S &\rightarrow aScbb \mid A \\ A &\rightarrow bAad \mid \lambda \end{aligned}$$

(c) (Requires Rice's Theorem.) State a nontrivial property of recursively enumerable languages that is true for L and true for \emptyset . Explain why the property is nontrivial and show that it indeed is true for L and true for \emptyset . If such a property does not exist, state it and explain your answer.

Answer: One such property is:

does not contain string a as element.

This property is true for \emptyset since \emptyset does not contain any elements. It is true for L , by inspection of the context-free grammar given in the answer to part (a)—no sequence of rule applications can lead to derivation of a single-symbol terminal string. On the other hand, this property is false for say a^* (which contains a as element.) Straightforwardly, both \emptyset and a^* are recursively enumerable languages (because they are regular.) Hence, our property is nontrivial because it is true for \emptyset but false for a^* .

Problem 673 Let L be the language accepted by the pushdown automaton:

$M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, p, t, s\}$;
 $\Sigma = \{a, b, c, d\}; \Gamma = \{A, E, K, D\}; F = \{t\}$ and δ is defined by the following transition set:

$$\begin{array}{lll} [q, a, \lambda, q, DEE] & [p, b, E, p, \lambda] & [t, a, A, t, \lambda] \\ [q, \lambda, \lambda, p, \lambda] & [p, d, D, p, \lambda] & [t, c, K, t, \lambda] \\ [s, b, \lambda, s, AKK] & [p, \lambda, \lambda, s, \lambda] & \\ [s, \lambda, \lambda, t, \lambda] & & \end{array}$$

(Recall that M is defined so as to accept by final state and empty stack.) Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, D\}$, and P is:

$$\begin{array}{l} S \rightarrow AD \\ A \rightarrow aAbbd \mid \lambda \\ D \rightarrow bDcca \mid \lambda \end{array}$$

(b) Write a complete formal definition of a context-free grammar that generates LLL . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, T)$, where $\Sigma = \{a, b, c, d\}$, $V = \{T, S, A, D\}$, and P is:

$$\begin{array}{l} T \rightarrow SSS \\ S \rightarrow AD \\ A \rightarrow aAbbd \mid \lambda \\ D \rightarrow bDcca \mid \lambda \end{array}$$

(c) (*Requires Rice's Theorem.*) State a trivial property of recursively enumerable languages that is false for L but true for Σ^* . Explain why the property is trivial and show that it indeed is false for L and true for Σ^* . If such a property does not exist, state it and explain your answer.

Answer: Such a property does not exist. By definition, a trivial property has the same value for all recursively enumerable languages—either always true or always false. Straightforwardly, both L and Σ^* are recursively enumerable languages (Σ^* is regular while L is context free, according to the answer given in the part (a).) Hence, a trivial property cannot be true for Σ^* but false for L .

Problem 674 Let L be the language accepted by the pushdown automaton:

$M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, t\}$;
 $\Sigma = \{a, b, c, d\}; \Gamma = \{A, E, K, D\}; F = \{t\}$ and δ is defined by the following transition set:

$$\begin{array}{ll} [q, a, \lambda, q, AEE] & [t, a, A, t, \lambda] \\ [q, b, \lambda, q, KKKD] & [t, b, E, t, \lambda] \\ [q, \lambda, \lambda, t, \lambda] & [t, c, K, t, \lambda] \\ & [t, d, D, t, \lambda] \end{array}$$

(Recall that M is defined so as to accept by final state and empty stack.) Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S\}$, and P is:

$$S \rightarrow aSbba \mid bSdccc \mid \lambda$$

(b) Let R be the set of exactly those strings over the alphabet $\Sigma = \{a, b, c, d\}$ whose length is equal to 4. Write a complete formal definition of a context-free grammar that generates $L \cap R$. If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S\}$, and P is:

$$S \rightarrow abba$$

(c) Explain how to construct an algorithm that solves the following problem:

INPUT: Pushdown automaton P_1 that accepts a language L_1 , and a pushdown automaton P_2 that accepts a language L_2 .

OUTPUT: Pushdown automaton P that accepts the language $L_1 \cap L_2$.

If this algorithm does not exist, prove it.

Answer: Such algorithm does not exist, since there exist pairs of context free languages whose intersection is not context free.

Problem 675 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, p\}; \Sigma = \{a, b, c, d, g, h\}$;
 $\Gamma = \{H, O, R, S, T, Y\}; F = \{q\}$ and δ is defined by the following transition set:

$$\begin{array}{ll} [q, d, \lambda, p, SHORT] & [p, h, H, p, \lambda] \\ [q, h, \lambda, p, STORY] & [p, a, O, p, \lambda] \\ & [p, b, R, p, \lambda] \\ & [p, g, S, q, \lambda] \\ & [p, d, T, p, \lambda] \\ & [p, c, Y, p, \lambda] \end{array}$$

(Recall that M is defined so as to accept by final state and empty stack.) Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write a regular expression that represents L . If such a regular expression does not exist, prove it.

Answer:

$$(ddbahg \cup hcbadg)^*$$

(c) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, g, h\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow \lambda \mid SS \mid A \mid B \\ A &\rightarrow ddbahg \\ B &\rightarrow hcbadg \end{aligned}$$

Problem 676 Let L be the language accepted by the pushdown automaton:

$M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, p\}$; $\Sigma = \{a, b, c, d, g, h\}$; $\Gamma = \{A, E, G, N, R, S\}$; $F = \{p\}$ and δ is defined by the following transition set:

$$\begin{array}{ll} [q, c, \lambda, q, GREEN] & [p, a, A, p, \lambda] \\ [q, a, \lambda, q, GRASS] & [p, c, E, p, \lambda] \\ [q, \lambda, \lambda, p, \lambda] & [p, g, G, p, \lambda] \\ & [p, h, N, p, \lambda] \\ & [p, b, R, p, \lambda] \\ & [p, d, S, p, \lambda] \end{array}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, g, h\}$, $V = \{S\}$, and P is:

$$S \rightarrow cShccbg \mid aSddabg \mid \lambda$$

Problem 677 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, p, s, t\}$; $\Sigma = \{a, b, c, d, g, h\}$; $\Gamma = \{A, E, H, L, R, T\}$; $F = \{s\}$ and δ is defined by the following transition set:

$$\begin{array}{ll} [q, a, \lambda, q, HEAR] & [p, a, A, p, \lambda] \\ [t, b, \lambda, t, TELL] & [p, c, E, p, \lambda] \\ [q, \lambda, \lambda, p, \lambda] & [p, b, R, p, \lambda] \\ [p, c, \lambda, t, \lambda] & [p, h, H, p, \lambda] \\ [t, \lambda, \lambda, s, \lambda] & [s, g, L, s, \lambda] \\ & [s, c, E, s, \lambda] \\ & [s, d, T, s, \lambda] \end{array}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost

symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, g, h\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow AcB \\ A &\rightarrow aAbach \mid \lambda \\ B &\rightarrow bBggcd \mid \lambda \end{aligned}$$

(b) Explain how to construct an algorithm that solves the following problem:

INPUT: Pushdown automaton P_1 that accepts a language L_1 .

OUTPUT: Pushdown automaton P_2 that accepts the language $\overline{L_1}$.

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist: there exist context-free languages whose complement is not context free.

Problem 678 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, t\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A, B\} \\ F &= \{t\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{array}{l} [q, a, \lambda, q, A] \\ [q, a, \lambda, t, B] \\ [t, c, A, t, \lambda] \\ [t, b, B, t, \lambda] \end{array}$$

(Recall that M is defined so as to accept by final state and empty stack.)

Write a complete formal definition of a context-free grammar that generates the language:

$$L^*$$

If such a grammar does not exist, prove it.

Answer: Observe that:

$$L = \{a^{n+1}bc^n \mid n \geq 0\}$$

Hence, L^* is generated by the grammar:

$G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$ is the set of terminals; $V = \{S, A\}$ is the set of variables; S is the start symbol, and the production set P is:

$$\begin{aligned} S &\rightarrow \lambda \mid SS \mid A \\ A &\rightarrow aAc \mid ab \end{aligned}$$

Problem 679 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$Q = \{q, t\}; \Sigma = \{a, b\}; \Gamma = \{A\}; F = \{t\};$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, t, AA] \\ & [q, a, \lambda, q, \lambda] \\ & [t, b, A, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

- (a) List five different strings that belong to L , or explain why this is impossible.

Answer:

$$abb, aabb, aaabb, aaaabb, aaaaabb$$

- (c) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$a^*abb$$

- (b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aS \mid abb$$

Problem 680 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$Q = \{q, t\}; \Sigma = \{a, b, c\}; \Gamma = \{A, B\}; F = \{q\};$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, t, AA] \\ & [q, a, \lambda, q, A] \\ & [t, b, A, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

- (a) List five different strings that belong to L , or explain why this is impossible.

Answer:

$$abb, aabbb, aaabbbb, aaaabbbbb, aaaaaabbbbb,$$

- (b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The general template for the strings of L is as follows.

$$L = \{a^{n_1+1}b^{n_2}c \mid n_1, n_2 \geq 0\}$$

Hence, L is generated by the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSb \mid abb$$

- (c) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: This regular expression does not exist, since L is not regular.

To prove that L is not regular, assume the opposite, that L is regular. Let k be the constant as in the Pumping Lemma for L . Consider a word $w = a^{m+1}b^{m+2}$, where $m > k$.

Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < k < m$. Hence, the pumping part v is entirely within the a -segment, meaning that $v = a^j$ for some j such that $0 < j \leq k$. After pumping once, we obtain a word: $w_p = a^{m+j+1}b^{m+2}$. For w_p to be in L , it must be that the number of b 's in the word is by one greater than the number of a 's: $m + 2 = (m + j + 1) + 1 = m + j + 2$. However, since $j > 0$, we have that: $m + 2 < m + j + 2$, meaning that $w_p \notin L$.

Problem 681 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$Q = \{q, t, s\}; \Sigma = \{a, b, c\}; \Gamma = \{A, B\}; F = \{q\};$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, t, A] \\ & [t, c, \lambda, s, \lambda] \\ & [t, a, \lambda, t, B] \\ & [s, c, A, q, \lambda] \\ & [s, b, B, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

- (a) List five different strings that belong to L , or explain why this is impossible.

Answer:

$$acc, aacbc, aaaacbbbc, aacbcaaacbbc, accacccaacbc$$

- (b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The general template for the strings of L is as follows.

$$L = \left\{ (a^{n_1+1}cb^{n_2}c)^\ell \mid n_1, n_2, \ell \geq 0 \right\}$$

Hence, L is generated by the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, T\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow \lambda \mid SS \mid aTc \\ T &\rightarrow aTb \mid c \end{aligned}$$

(c) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: This finite automaton does not exist, since L is not regular.

To prove that L is not regular, assume the opposite, that L is regular. Let k be the constant as in the Pumping Lemma for L . Consider a word $w = a^{m+1}cb^mc$, where $m > k$, obtained from the general template by setting $\ell = 0$.

Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < k < m$. Hence, the pumping part v is entirely within the a -segment, meaning that $v = a^j$ for some j such that $0 < j \leq k$. After pumping once, we obtain a word: $w_p = a^{m+1+j}cb^mc$. For w_p to be in L , it must be that the number of b 's in the word is by one less than the number of a 's: $m = (m + 1 + j) - 1 = m + j$. However, since $j > 0$, we have that: $m < m + j$, meaning that $w_p \notin L$.

Problem 682 Let L be the language accepted by the pushdown automaton:

$M = (Q, \Sigma, \Gamma, \delta, p, F)$ where: $Q = \{p, v\}$;

$\Sigma = \{a, b, c, d\}; \Gamma = \{A, R, S, T\}; F = \{p, v\}$ and δ is defined by the following set of five transitions:

- (1) $[p, d, \lambda, v, START]$
- (2) $[v, a, A, v, \lambda]$
- (3) $[v, b, T, v, \lambda]$
- (4) $[v, c, R, v, \lambda]$
- (5) $[v, d, S, p, \lambda]$

(Recall that M is defined so as to accept by final state and empty stack.) Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(dbcabd)^*$$

(b) Is the language L context-free? Prove your answer.

Answer: Yes, it is accepted by pushdown automaton M , and all languages accepted by pushdown automata are context-free.

In the following parts, your task is to modify exactly one of the five transition tuples of the automaton M , so as to obtain an automaton M_1 that accepts a *finite* language. In more words, select one of the tuples labelled (1) – (5)

in the definition of the automaton M , and substitute another tuple in its place (leaving all other parts of the definition intact), so that the new automaton accepts a finite language. If this modification is possible, explain it in parts (c), (d), (e) of the answer space, as directed, and do not write anything in the answer space of part (f). If this modification is impossible, then prove its impossibility in the answer space of part (f), and do not write anything in the answer space of parts (c), (d), (e).

(c) State the label (one of (1) – (5)) of the transition tuple which you propose to modify.

Answer: (5)

(d) Write the new transition tuple which you propose to substitute in place of the tuple which you named in your answer to part (c).

Answer:

$$[v, d, S, v, \lambda]$$

(e) Write a regular expression that defines the language accepted by the new automaton, obtained by your modification.

Answer:

$$dbcabd \cup \lambda$$

(f) Proof that the required modification is impossible:

Answer:

Problem 683 Let L be the language accepted by the pushdown automaton:

$M = (Q, \Sigma, \Gamma, \delta, p, F)$ where: $Q = \{p, t\}$;

$\Sigma = \{a, b, c, d\}; \Gamma = \{O, P, S, T\}; F = \{t\}$ and δ is defined by the following set of five transitions:

- $[p, d, \lambda, p, STOP]$
- $[p, a, \lambda, t, T]$
- $[t, a, O, t, \lambda]$
- $[t, b, P, t, \lambda]$
- $[t, c, S, t, \lambda]$
- $[t, d, T, t, \lambda]$

(Recall that M is defined so as to accept by final state and empty stack.) Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S\}$, and P is:

$$S \rightarrow dSbadc \mid ad$$

(b) Explain how to construct an algorithm that solves the following problem:

INPUT: Pushdown automaton M_P that accepts a language L_P .

OUTPUT: Regular expression e that defines the language L_P .

If this algorithm does not exist, prove it.

Answer: The algorithm does not exist: some context-free languages are not regular.

(c) Explain how to construct an algorithm that solves the following problem:

INPUT: Pushdown automaton M_P that accepts a language L_P .

OUTPUT: Pushdown automaton M_Q that accepts the complement $\overline{L_P}$ of the language L_P .

If this algorithm does not exist, prove it.

Answer: The algorithm does not exist: the complement of some context-free languages is not context-free.

Problem 684 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, t\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, B, D\} \\ F &= \{t\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, c, \lambda, t, BAD] \\ [t, a, A, t, \lambda] \\ [t, b, B, t, \lambda] \\ [t, d, D, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) List 6 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer: It is impossible to list six distinct strings that belong to L because L contains exactly one string: $cdab$.

(b) Draw a state transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 298.



Figure 298:

(c) What is the cardinality of the set L ? If it is finite, state the exact number; if it is infinite, state whether it is countable or uncountable.

Answer: Cardinality of L is equal to 1.

(d) What is the cardinality of the set L^* ? If it is finite, state the exact number; if it is infinite, state whether it is countable or uncountable.

Answer: L^* is infinite and countable.

Problem 685 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where: $Q = \{q, s\}$; $\Sigma = \{a, b, c, d\}$; $\Gamma = \{A, B, K, D\}$; $F = \{s\}$; and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, BKA] \\ [q, \lambda, \lambda, s, \lambda] \\ [q, d, \lambda, s, DDD] \\ [s, a, A, s, \lambda] \\ [s, b, B, s, \lambda] \\ [s, c, K, s, \lambda] \\ [s, d, D, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) List 4 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer:

$$\lambda, dddd, aacb, addddacb$$

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSacb \mid \lambda \mid dddd$$

(c) State the cardinality of the language L . If it is finite, state the exact number; if it is infinite specify whether it is countable or uncountable.

Answer: L is infinite and countable.

(d) State the cardinality of the language \overline{L} (the complement of L in Σ^*). If it is finite, state the exact number; if it is infinite specify whether it is countable or uncountable.

Answer: \overline{L} is infinite and countable.

(e) State the cardinality of the class that contains all context-free languages over Σ . If it is finite, state the exact number; if it is infinite specify whether it is countable or uncountable.

Answer: Class of all context-free languages over Σ is infinite and countable.

(f) State the cardinality of the class that contains all languages over Σ that are not context-free. If it is finite, state the exact number; if it is infinite specify whether it is countable or uncountable.

Answer: Class of all languages over Σ that are not context-free is infinite and uncountable.

Problem 686 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$Q = \{q, p, t\}; \Sigma = \{a, b, c\}; \Gamma = \{Z, A\}; F = \{t\};$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, b, \lambda, p, ZAA] \\ & [p, c, A, p, \lambda] \\ & [p, a, Z, q, \lambda] \\ & [q, a, \lambda, t, ZZ] \\ & [t, b, Z, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: By inspection of the automaton M , we conclude that L is represented by the regular expression:

$$(bcca)^*abb$$

whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S\}$, and P is:

$$S \rightarrow bccaaS \mid abb$$

(b) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 299.

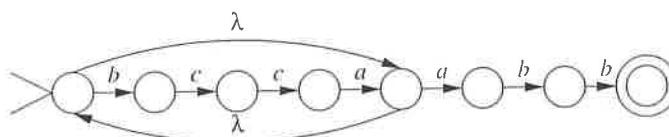


Figure 299:

Problem 687 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$Q = \{q, p, t\}; \Sigma = \{a, b\}; \Gamma = \{Z, A\}; F = \{t\};$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, p, Z] \\ & [p, a, \lambda, p, A] \\ & [p, \lambda, \lambda, t, \lambda] \\ & [t, b, A, t, \lambda] \\ & [t, b, Z, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: By inspection of the automaton M , we conclude that:

$$L = \{a^{n+1}b^{n+1} \mid n \geq 0\}$$

whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S\}$, and P is:

$$S \rightarrow aSb \mid ab$$

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: This regular expression does not exist, since L is not regular. To prove this, assume the opposite—that L is regular. Observe that all words of L contain an equal (positive) number of a 's and b 's (and all a 's precede all b 's.) Let k be the constant as in the Pumping Lemma for L . Consider a word $w = a^m b^m$, where $m > k + 1$. Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < k < m$. Hence, the pumping part v is entirely within the a -segment, meaning that $v = a^j$ for some j such that $0 < j < k$. After pumping once, we obtain a word: $w_1 = a^{m+j} b^m$. However, since $j > 0$ (as the length of the pumping substring), we conclude that $m + j > m$, meaning that the a -segment and the b -segment in w_1 do not have equal length. Hence, $w_1 \notin L$, whence the contradiction.

Problem 688 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$Q = \{q, t\}; \Sigma = \{a, b\}; \Gamma = \{A, B\}; F = \{t\};$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, q, BA] \\ & [q, \lambda, \lambda, t, \lambda] \\ & [t, b, A, t, \lambda] \\ & [t, b, B, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: This regular expression does not exist, since L is not regular. To prove this, observe that the general template for strings in L is:

$$L = a^n b^{2n} \text{ where } n \geq 0$$

Assume the opposite—that L is regular. Let k be the constant as in the Pumping Lemma for L . Consider a word $w = a^n b^{2n}$, where $n > k$. Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < k < n$. Hence, the pumping part v is entirely within the a -segment, meaning that $v = a^j$ for some j such that $0 < j < k$. After pumping once, we obtain a word: $w_1 = a^{n+j} b^{2n}$. However, since $j > 0$ (as the length of the pumping substring), we conclude that $2(n+j) > 2n$, meaning that the b -segment in w_1 has too few b 's for its number of a 's. Hence, $w_1 \notin L$, whence the contradiction.

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSbb \mid \lambda$$

Problem 689 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$Q = \{q, t, s\}; \Sigma = \{a, b, c, d\}; \Gamma = \{A\}; F = \{s\};$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, a, \lambda, t, ABB] \\ &[t, c, A, q, \lambda] \\ &[t, b, B, t, \lambda] \\ &[q, c, \lambda, s, \lambda] \\ &[q, d, \lambda, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost

symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(abbc)^* (c \cup d)$$

(b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 300.

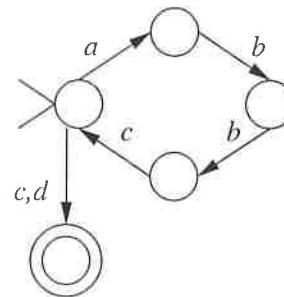


Figure 300:

Problem 690 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s, t\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A, B, Z\} \\ F &= \{q\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, c, \lambda, r, Z] \\ &[r, b, \lambda, t, ABB] \\ &[t, a, A, t, \lambda] \\ &[t, b, B, t, \lambda] \\ &[t, \lambda, Z, s, \lambda] \\ &[s, \lambda, \lambda, q, \lambda] \\ &[s, d, \lambda, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

Construct a state-transition graph of a finite automaton that accepts the language:

$$LL$$

If such an automaton does not exist, prove it.

Answer: Observe that L is represented by the regular expression:

$$(cbbaad^*)^*$$

Hence: $LL = L$, and LL is accepted by the finite automaton given on Figure 301.

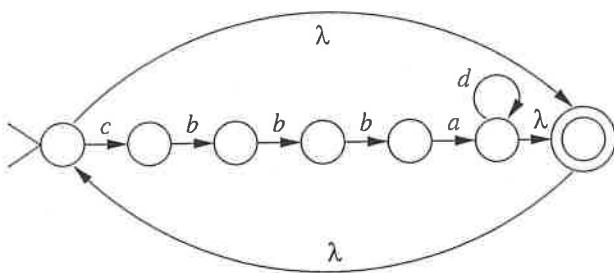


Figure 301:

Problem 691 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s, t, x\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, B, Z\} \\ F &= \{x\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, c, \lambda, r, Z] \\ &[q, c, \lambda, x, \lambda] \\ &[r, b, \lambda, t, ABB] \\ &[t, a, A, t, \lambda] \\ &[t, b, B, t, \lambda] \\ &[t, \lambda, Z, s, \lambda] \\ &[s, \lambda, \lambda, q, \lambda] \\ &[s, d, \lambda, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

Construct a state-transition graph of a finite automaton that accepts the language:

$$L^*$$

If such an automaton does not exist, prove it.

Answer: See Figure 302.

Problem 692 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s, t, x\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, Z\} \\ F &= \{x\} \end{aligned}$$

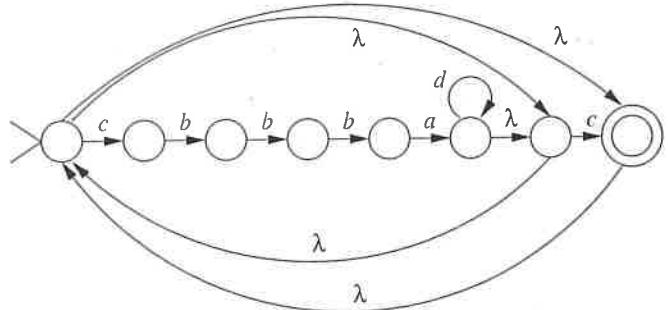


Figure 302:

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, c, \lambda, r, Z] \\ &[q, c, \lambda, x, \lambda] \\ &[r, b, \lambda, t, A] \\ &[t, a, A, t, \lambda] \\ &[t, b, A, t, \lambda] \\ &[t, \lambda, Z, s, \lambda] \\ &[s, \lambda, \lambda, q, \lambda] \\ &[s, d, \lambda, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

Construct a state-transition graph of a finite automaton that accepts the language:

$$LL$$

If such an automaton does not exist, prove it.

Advice for Answer:

$$L = (cb(a \cup b)d^*)^*c$$

Problem 693 Consider the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$Q = \{q, t\}; \Sigma = \{a, b\}; \Gamma = \{A\}; F = \{t\};$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, \lambda, \lambda, t, \lambda] \\ &[q, a, \lambda, q, AAA] \\ &[t, b, A, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

Let $L(M)$ be the language accepted by M .

(a) List five different strings that belong to $L(M)$, or explain why this is impossible.

Answer:

$\lambda, abbb, aabbffff, aaabbbbbbbb, aaaaabbbbbbbbbb$

(b) Write a complete formal definition of a context-free grammar G such that G generates $L(M)$:

$$L(M) = L(G)$$

If such a grammar does not exist, prove it.

Answer: Observe that the general template for the strings of $L(M)$ is:

$$a^m b^{3m} \text{ for } m \geq 0$$

whence the answer: $G = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b\}$, $V = \{S\}$, and P is:

$$S \rightarrow aSbbb \mid \lambda$$

(c) Write a complete formal definition (or draw a state transition graph) of a finite automaton F such that F accepts $L(M)$:

$$L(M) = L(F)$$

If such a finite automaton does not exist, prove it.

Answer: Such a finite automaton does not exist, because $L(M)$ is not regular.

To prove that $L(M)$ is not regular, assume the opposite, that $L(M)$ is regular. Let k be the constant as in the Pumping Lemma for $L(M)$. Consider a word $w = a^m b^{3m}$, where $m > k$.

Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < k < m$. Hence, the pumping part v is entirely within the a -segment, meaning that $v = a^j$ for some j such that $0 < j \leq k$. After pumping once, we obtain a word: $w_p = a^{m+j} b^{3m} \in L(M)$. Since w_p belongs to $L(M)$, the number of b 's in the word has to be equal to three times the number of a 's, precisely: $3(m+j) = 3m + 3j$. However, w_p has only $3m$ occurrences of b , and $3m < 3m + 3j$, since $j > 0$ (because j is the length of the non-empty pumping substring.) Hence, $w_p \notin L(M)$, which is a contradiction.

Problem 694 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where: $Q = \{q, t, s, v, x, y, z\}$; $\Sigma = \{a, b, c, d\}$;

$\Gamma = \{A, B\}$; $F = \{z\}$;

and the transition function δ is defined as follows:

$[q, a, \lambda, t, BA]$
$[t, c, A, s, \lambda]$
$[t, a, \lambda, t, A]$
$[s, c, A, s, \lambda]$
$[s, \lambda, B, v, \lambda]$
$[v, b, \lambda, x, BA]$
$[x, d, A, y, \lambda]$
$[x, b, \lambda, x, A]$
$[y, d, A, y, \lambda]$
$[y, \lambda, B, z, \lambda]$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) List five different strings that belong to L , or explain why this is impossible.

Answer:

$acbd, aaccbd, aaaccbbdd, acbbbddddd, aaaaacccbd$

(c) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The general template for the strings of L is as follows.

$$L = \{a^{n+1}c^{n+1}b^{\ell+1}d^{\ell+1} \mid n, \ell \geq 0\}$$

Hence, L is generated by the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aAc \mid ac \\ B &\rightarrow bBd \mid bd \end{aligned}$$

(b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: This finite automaton does not exist, since L is not regular.

To prove that L is not regular, assume the opposite, that L is regular. Let k be the constant as in the Pumping Lemma for L . Consider a word $w = a^{m+1}c^{m+1}bd$, where $m > k$, obtained from the general template by setting $\ell = 0$.

Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < k < m$. Hence, the pumping part v is entirely within the a -segment, meaning that $v = a^j$ for some j such that $0 < j \leq k$. After pumping once, we obtain a word: $w_p = a^{m+1+j}c^{m+1}bd$. For w_p to be in L , it must be that the number of c 's in the word is equal to the number of a 's: $m+1 = m+1+j$. However, since $j > 0$, we have that: $m+1 < m+1+j$, meaning that $w_p \notin L$.

Problem 695 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, s, t, v, x\}$; $\Sigma = \{a, b, c, d\}$; $\Gamma = \{A, Z\}$; $F = \{v\}$; and the transition function δ is

defined as follows:

$$[q, \lambda, \lambda, t, Z]$$

$$\begin{aligned} & [t, d, \lambda, t, AA] \\ & [t, \lambda, \lambda, s, \lambda] \end{aligned}$$

$$\begin{aligned} & [s, b, A, x, \lambda] \\ & [s, a, Z, v, \lambda] \end{aligned}$$

$$[x, c, A, s, \lambda]$$

$$[v, d, \lambda, v, \lambda]$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) List 8 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer:

$$\begin{aligned} & a, ad, addd, dbca, dbcadd, ddcbca, \\ & ddcbcacddd, dddbcbcba \end{aligned}$$

(The general template is: $d^n(bc)^n a d^j$, $j, n \geq 0$.)

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: This regular expression does not exist, since L is not regular.

To prove that L is not regular, assume the opposite, that L is regular. Let k be the constant as in the Pumping Lemma for L . Consider a word $w \in L$, such that $w = d^m(bc)^m a$, for some $m > k$; w is obtained from the general template by setting $j = 0$.

Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < k < m$. Hence, the pumping part v is entirely within the d -segment, meaning that $v = d^\ell$ for some ℓ such that $0 < \ell \leq k$. After pumping once, we obtain a word: $w_p = d^{m+\ell}(bc)^m$. However, $m + \ell > m$, meaning that w_p has more d 's than bc -pairs, which in turn implies that $w_p \notin L$.

(c) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow BaD \\ B &\rightarrow dBbc \mid \lambda \\ D &\rightarrow dD \mid \lambda \end{aligned}$$

Problem 696 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, r\}$; $\Sigma = \{a, b\}$; $\Gamma = \{A, Z\}$; $F = \{q\}$; and the transition function δ is defined as follows:

$$\begin{aligned} & [q, b, \lambda, r, Z] \\ & [q, b, \lambda, r, ZA] \\ & [r, a, A, r, \lambda] \\ & [r, a, Z, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) List 5 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer: $ba, baa, babaa, bababaa, baabaabaa$

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: $(ba \cup baa)^*$

(c) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 303.

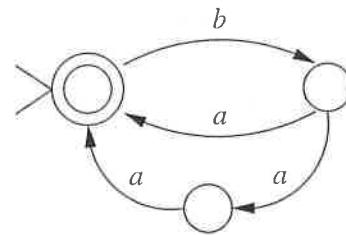


Figure 303:

(d) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S\}$, and P is:

$$S \rightarrow \lambda \mid SS \mid ba \mid baa$$

Problem 697 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$Q = \{q, t\}; \Sigma = \{a, b, c, d\}; \Gamma = \{A\}; F = \{t\};$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, q, A] \\ & [q, b, \lambda, t, \lambda] \\ & [t, c, A, t, \lambda] \\ & [t, d, A, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) List 5 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: The general template is:

$$a^n b (c \cup d)^n, n \geq 0$$

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSc \mid aSd \mid b$$

Problem 698 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$Q = \{q, t\}; \Sigma = \{a, b, c\}; \Gamma = \{A\}; F = \{t\};$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, t, A] \\ & [q, b, \lambda, q, \lambda] \\ & [t, a, A, t, \lambda] \\ & [t, c, \lambda, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) List 5 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: See part (b).

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$b^* a c^* a c^*$$

Problem 699 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$Q = \{q, t\}; \Sigma = \{a, b, c, d\}; \Gamma = \{A, B\}; F = \{t\};$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, q, A] \\ & [q, b, \lambda, t, B] \\ & [t, c, A, t, \lambda] \\ & [t, d, B, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) List 5 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: The general template is:

$$a^n b d c^n, n \geq 0$$

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSc \mid bd$$

Problem 700 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$Q = \{q, s, t\}; \Sigma = \{a, b, c, d\}; \Gamma = \{A\}; F = \{t\};$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, q, A] \\ & [q, \lambda, \lambda, s, \lambda] \\ & [s, c, \lambda, s, \lambda] \\ & [s, b, \lambda, s, \lambda] \\ & [s, \lambda, \lambda, t, \lambda] \\ & [t, d, A, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) List 5 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: The general template is:

$$a^n (b \cup c)^\ell d^n, \ell, n \geq 0$$

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Advice for Answer: L is not regular. Pumping is observed in a word:

$$w = a^n d^n$$

obtained from the general template by setting $\ell = 0$, where n is selected so as to exceed the pumping constant. Every word in L that does not contain any b 's or c 's must honor the template of w , for some n ; pumping, however, breaks the template.

Problem 701 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, t\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A, B\} \\ F &= \{t\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, q, A] \\ & [q, \lambda, \lambda, t, BB] \\ & [t, c, A, t, \lambda] \\ & [t, b, B, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) List 5 distinct strings that belong to the language L . If this is impossible, state it and explain why.

Advice for Answer: The general template for strings in L is:

$$a^n bb c^n, n \geq 0$$

(b) Write a complete formal definition of a context-free grammar that generates the language:

$$LL$$

If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AA \\ A &\rightarrow aAc \mid bb \end{aligned}$$

Problem 702 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, t\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A, B\} \\ F &= \{t\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, AA] \\ [q, \lambda, \lambda, t, B] \\ [t, b, A, t, \lambda] \\ [t, c, B, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) List 5 distinct strings that belong to the language L . If this is impossible, state it and explain why.

Advice for Answer: The general template for strings in L is:

$$a^n c b^{2n}, n \geq 0$$

(b) Write a complete formal definition of a context-free grammar that generates the language:

$$L^*$$

If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow \lambda \mid SS \mid A \\ A &\rightarrow aAbb \mid c \end{aligned}$$

Problem 703 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where:

$Q = \{q, s\}$; $\Sigma = \{a, b, c, d\}$; $\Gamma = \{A, B, K, D\}$; $F = \{s\}$; and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, KA] \\ [q, b, \lambda, q, KB] \\ [q, c, \lambda, q, KK] \\ [q, d, \lambda, q, KD] \\ [q, \lambda, \lambda, s, \lambda] \\ [s, a, A, s, \lambda] \\ [s, b, B, s, \lambda] \\ [s, c, K, s, \lambda] \\ [s, d, D, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) List 4 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer: λ, aac, bbc, ccc

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSac \mid bSbc \mid cScc \mid dSdc \mid \lambda$$

(c) Let L_1 be the set of all strings over the alphabet $\{a, b, c, d\}$ whose length is less than 6.

Write a complete formal definition of a context-free grammar that generates $L \cup L_1$. If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, T)$, where $\Sigma = \{a, b, c, d\}$, $V = \{T, S, X\}$, and the production set P is:

$$\begin{aligned} T &\rightarrow S \mid X \\ S &\rightarrow aSac \mid bSbc \mid cScc \mid dSdc \mid \lambda \\ X &\rightarrow YYYYY \\ Y &\rightarrow a \mid b \mid c \mid d \mid \lambda \end{aligned}$$

Problem 704 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where:

$Q = \{q, r, t\}$, $\Sigma = \{a, b, c\}$, $\Gamma = \{B\}$, $F = \{t\}$ and the transition function δ is defined as follows:

$$\begin{array}{ll} [q, a, \lambda, r, B] & [r, b, B, r, \lambda] \\ [q, a, \lambda, r, BB] & [r, c, \lambda, t, \lambda] \\ [q, a, \lambda, r, BBB] & [t, c, \lambda, t, \lambda] \end{array}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) List 6 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: L is given by the following regular expression.

$$a(b \cup bb \cup bbb)cc^*$$

- (b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, B, K\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aBcK \\ B &\rightarrow b \mid bb \mid bb \\ K &\rightarrow cK \mid \lambda \end{aligned}$$

- (c) Dangerous Professor has told her students to write a program that operates as follows:

INPUT: String w .

OUTPUT: **yes** if w is an element of the language accepted by the pushdown automaton M (defined at the beginning of this problem);

no otherwise.

Explain the algorithm that should be employed by this program, or state that it does not exist, and prove it.

Answer: Convert the regular expression given in the answer to part (a) into a finite automaton, convert this automaton into its deterministic equivalent, and simulate it.

Problem 705 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, s\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{A, B, K, X, Z\}$; $F = \{q\}$; and the transition function δ is defined as follows:

$$\begin{aligned} &[q, a, \lambda, s, XZA] \\ &[q, a, \lambda, s, XZZB] \\ &[q, a, \lambda, s, XZZZK] \\ &[s, a, A, s, \lambda] \\ &[s, b, B, s, \lambda] \\ &[s, c, K, s, \lambda] \\ &[s, a, Z, s, \lambda] \\ &[s, b, Z, s, \lambda] \\ &[s, c, Z, s, \lambda] \\ &[s, \lambda, X, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

- (a) List 4 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer: $\lambda, aaa, abcc, acbca$

- (b) Construct a state transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 304.

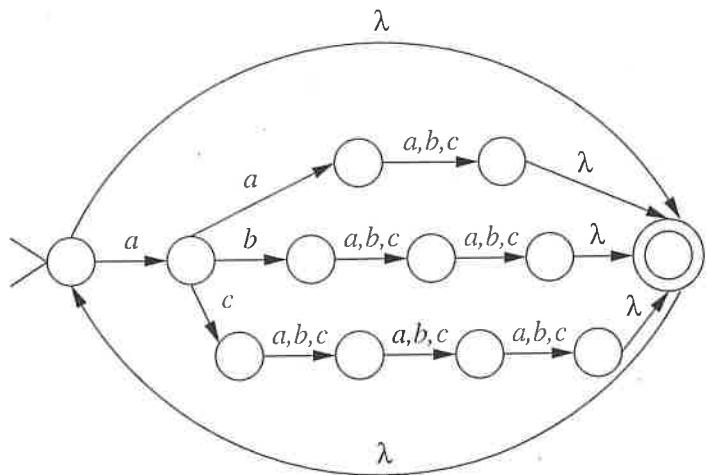


Figure 304:

- (c) Let L_1 be the set of all strings over the alphabet $\{a, b, c\}$ whose length is less than 6.

Construct a state transition graph of a finite automaton that accepts $L \cap L_1$. If such an automaton does not exist, prove it.

Answer: See Figure 305.

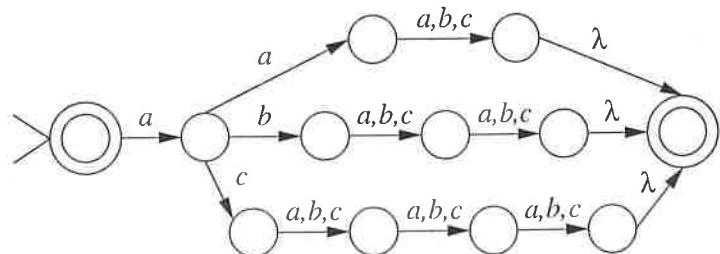


Figure 305:

Problem 706 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, r, s, t, x\}$; $\Sigma = \{a, b, c, d\}$; $\Gamma = \{B, D\}$; $F = \{t\}$; and the transition function δ is defined as follows:

$$\begin{aligned} &[q, a, \lambda, q, BB] \\ &[r, b, B, r, \lambda] \\ &[s, c, \lambda, x, D] \\ &[x, c, \lambda, s, \lambda] \\ &[t, d, D, t, \lambda] \\ &[q, \lambda, \lambda, r, \lambda] \\ &[r, \lambda, \lambda, s, \lambda] \\ &[s, \lambda, \lambda, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

Answer: See Figure 304.

- (a) List 4 distinct strings that belong to LL , but do not belong to L . If this is impossible, state it and explain why.

Advice for Answer: The template for L is:

$$a^n (bb)^n (cc)^k d^k \quad n, k \geq 0$$

- (b) Write a complete formal definition of a context-free grammar that generates the language:

$$LL$$

If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AA \\ A &\rightarrow BD \\ B &\rightarrow aBbb \mid \lambda \\ D &\rightarrow ccDd \mid \lambda \end{aligned}$$

Note: Compare this automaton with that given in Problem 706.

Problem 707 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, r, s, t, x\}$; $\Sigma = \{a, b, c, d\}$; $\Gamma = \{B, D\}$; $F = \{t\}$; and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, BB] \\ [r, b, B, r, \lambda] \\ [s, c, \lambda, x, D] \\ [x, c, \lambda, s, \lambda] \\ [t, d, D, t, \lambda] \\ [q, \lambda, \lambda, s, \lambda] \\ [s, \lambda, \lambda, t, \lambda] \\ [t, \lambda, \lambda, r, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

- (a) List 4 distinct strings that belong to L^* , but do not belong to L . If this is impossible, state it and explain why.

Advice for Answer: The template for L is:

$$a^n (cc)^k d^k (bb)^n \quad n, k \geq 0$$

- (b) Write a complete formal definition of a context-free grammar that generates the language:

$$L^*$$

If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow \lambda \mid SS \mid A \\ A &\rightarrow aAbb \mid B \\ B &\rightarrow ccBd \mid \lambda \end{aligned}$$

Note: Compare this automaton with that given in Problem 706.

Problem 708 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where: $Q = \{q, s\}$; $\Sigma = \{a, b, c, d\}$; $\Gamma = \{A, B, K, D\}$; $F = \{s\}$; and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, BKA] \\ [q, \lambda, \lambda, s, \lambda] \\ [q, d, \lambda, s, D] \\ [s, a, A, s, \lambda] \\ [s, b, B, s, \lambda] \\ [s, c, K, s, \lambda] \\ [s, d, D, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

- (a) List 4 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer:

$$\lambda, dd, aacb, addacb$$

- (b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSacb \mid \lambda \mid dd$$

- (c) Write a complete formal definition of a context-free grammar that generates L^* . If such a grammar does not exist, prove it.

Answer: $G = (V_1, \Sigma, P_1, T)$, where $\Sigma = \{a, b, c, d\}$, $V_1 = \{T, S\}$, and the production set P_1 is:

$$\begin{aligned} T &\rightarrow \lambda \mid TT \mid S \\ S &\rightarrow aSacb \mid \lambda \mid dd \end{aligned}$$

Problem 709 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, t\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{B, K, Z\} \\ F &= \{q\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, t, ZKK] \\ [q, a, \lambda, t, ZBB] \\ [t, b, B, t, \lambda] \\ [t, c, K, t, \lambda] \\ [t, \lambda, Z, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write 5 distinct strings that belong to L . If such strings do not exist, state it and explain why.

Answer:

$$\lambda, acc, abb, accabb, abbaabbabb$$

Observe that L is represented by the regular expression:

$$(acc \cup abb)^*$$

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S\}$, and P is:

$$S \rightarrow \lambda \mid SS \mid acc \mid abb$$

Problem 710 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, t\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{B, Z\} \\ F &= \{t\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, ZBB] \\ [q, \lambda, \lambda, t, \lambda] \\ [t, b, B, t, \lambda] \\ [t, c, Z, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write 5 distinct strings that belong to L . If such strings do not exist, state it and explain why.

Answer:

$$\lambda, abbc, aabbcbc, aaabbccbccbc, aaaabbccbccbccbc$$

Observe that the general template for strings of L is:

$$a^n(bbc)^n \text{ where } n \geq 0$$

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S\}$, and P is:

$$S \rightarrow aSbbc \mid \lambda$$

Problem 711 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, t\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{B, K, Z\} \\ F &= \{t\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, ZK] \\ [q, a, \lambda, q, ZB] \\ [q, \lambda, \lambda, t, \lambda] \\ [t, b, B, t, \lambda] \\ [t, c, K, t, \lambda] \\ [t, d, Z, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write 5 distinct strings that belong to L . If such strings do not exist, state it and explain why.

Answer:

$$\lambda, acd, abd, aacdcd, aaaacdbdbdc$$

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S\}$, and P is:

$$S \rightarrow aScd \mid aSbd \mid \lambda$$

Problem 712 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, s, t\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{B, K, Z\} \\ F &= \{q\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, a, \lambda, s, ZKK] \\ &[q, a, \lambda, t, ZBB] \\ &[t, b, B, t, \lambda] \\ &[t, d, Z, q, \lambda] \\ &[s, c, K, s, \lambda] \\ &[s, d, Z, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write 5 distinct strings that belong to L . If such strings do not exist, state it and explain why.

Advice for Answer: L is represented by the regular expression;

$$(accd \cup abbd)^*$$

Answer:

$$\lambda, accd, abbd, accdabbd, abbdaccdabbd$$

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S\}$, and P is:

$$S \rightarrow \lambda \mid SS \mid accd \mid abbd$$

(c) Is the automaton M deterministic? Explain your answer.

Answer: No. In state q , on input symbol a , independently from the stack content, the automaton has a non-deterministic choice between the following two transitions:

$$\begin{aligned} &[q, a, \lambda, s, ZKK] \\ &[q, a, \lambda, t, ZBB] \end{aligned}$$

(d) Is $L(M)$ decidable? Explain your answer.

Answer: Yes. As is explained in the answer to part (b), L is context-free, but every context-free language is decidable.

(e) Is $L(M)$ accepted by any deterministic finite automaton? Explain your answer.

Answer: Yes. As is explained in the answer to part (a), L is regular, but every regular language is accepted by some deterministic finite automaton.

(f) Is $L(M)$ accepted by any deterministic pushdown automaton? Explain your answer.

Answer: Yes. As is explained in the answer to part (e), L is accepted by some deterministic finite automaton, which is in turn algorithmically convertible into a deterministic push-down automaton.

Problem 713 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$

where: $Q = \{q, t\}; \Sigma = \{a, b, c, d\}; \Gamma = \{A\}; F = \{t\}$ and δ is defined by the following transition set:

$$\begin{aligned} &[q, a, \lambda, q, A] \\ &[q, d, \lambda, q, A] \\ &[q, \lambda, \lambda, t, \lambda] \\ &[t, b, A, t, \lambda] \\ &[t, c, A, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ASB \mid \lambda \\ A &\rightarrow a \mid d \\ B &\rightarrow b \mid c \end{aligned}$$

Problem 714 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$

where: $Q = \{q, t\}; \Sigma = \{a, b, c, d\}; \Gamma = \{A, B, E, M, N\}; F = \{t\}$ and δ is defined by the following transition set:

$$\begin{aligned} &[q, a, \lambda, q, NAME] \\ &[q, c, \lambda, q, NAME] \\ &[q, b, \lambda, t, B] \\ &[t, a, E, t, \lambda] \\ &[t, b, B, t, \lambda] \\ &[t, b, N, t, \lambda] \\ &[t, c, M, t, \lambda] \\ &[t, d, A, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ASB \mid bb \\ A &\rightarrow a \mid c \\ B &\rightarrow acdb \end{aligned}$$

Problem 715 Let L_1 be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where:

$$Q = \{q, t\}; \Sigma = \{a, b, c, d, e\}; \Gamma = \{D, E, K\}; F = \{t\};$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, DEK] \\ [q, b, \lambda, t, \lambda] \\ [t, c, K, t, \lambda] \\ [t, d, D, t, \lambda] \\ [t, e, E, t, \lambda] \end{aligned}$$

M accepts by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.

(a) List 3 distinct strings that belong to L_1 . If this is impossible, state it and explain why.

Answer:

$$b, abced, aabcedced$$

(b) Let L_2 be the language represented by the regular expression $(ab)^*$.

Write a complete formal definition of a context-free grammar that generates the language:

$$L_1 \cup L_2$$

If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, e\}$, $V = \{S, S_1, S_2\}$, and P is:

$$\begin{aligned} S &\rightarrow S_1 \mid S_2 \\ S_1 &\rightarrow aS_1ced \mid b \\ S_2 &\rightarrow abS_2 \mid \lambda \end{aligned}$$

Problem 716 Let L_1 be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where:

$$Q = \{q, s, t\}; \Sigma = \{a, b, c, d, e\}; \Gamma = \{A, D, E, K\}; F = \{s\};$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, b, \lambda, s, \lambda] \\ [q, b, \lambda, t, ADEK] \\ [t, a, A, q, \lambda] \\ [t, c, K, t, \lambda] \\ [t, d, D, t, \lambda] \\ [t, e, E, t, \lambda] \end{aligned}$$

M accepts by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where

$n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.

(a) List 3 distinct strings that belong to L_1 . If this is impossible, state it and explain why.

Advice for Answer: The language accepted by M is:

$$(bceda)^* b$$

(b) Let L_2 be the language represented by the regular expression $(a \cup b)^*$.

Construct a state-transition graph of a finite automaton that accepts the language:

$$L_1 L_2$$

If such an automaton does not exist, prove it.

Answer: See Figure 306.

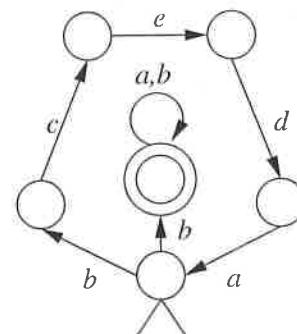


Figure 306:

Problem 717 Let L_1 be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where:

$$Q = \{q, t\}; \Sigma = \{a, b, c, d, e\}; \Gamma = \{A, D\}; F = \{t\};$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, A] \\ [q, b, \lambda, t, \lambda] \\ [q, c, \lambda, q, D] \\ [t, d, D, t, \lambda] \\ [t, e, A, t, \lambda] \end{aligned}$$

M accepts by final state and empty stack.

(a) List 5 distinct strings that belong to L_1 . If this is impossible, state it and explain why.

Answer:

$$b, abc, cbd, aabec, cabed$$

(b) Let L_2 be the language represented by the regular expression b^*a .

Write a complete formal definition of a context-free grammar that generates the language:

$$L_1 \cup L_2$$

If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b, c, d, e\}$, $V = \{S, S_1, S_2\}$, and P is:

$$\begin{aligned} S &\rightarrow S_1 \mid S_2 \\ S_1 &\rightarrow aS_1e \mid cS_1d \mid b \\ S_2 &\rightarrow bS_2 \mid a \end{aligned}$$

Problem 718 Let L_1 be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where:

$$Q = \{q, s, t\}; \Sigma = \{a, b, c, d, e\}; \Gamma = \{A, D\}; F = \{s\};$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, b, \lambda, s, \lambda] \\ [s, a, \lambda, t, A] \\ [s, c, \lambda, t, D] \\ [t, e, A, s, \lambda] \\ [t, d, D, s, \lambda] \end{aligned}$$

M accepts by final state and empty stack.

(a) List 5 distinct strings that belong to L_1 . If this is impossible, state it and explain why.

Advice for Answer: The language accepted by M is:

$$b (ae \cup cd)^*$$

(b) Let L_2 be the language represented by the regular expression $a b^*$.

Construct a state-transition graph of a finite automaton that accepts the language:

$$L_1 L_2$$

If such an automaton does not exist, prove it.

Answer: See Figure 307.

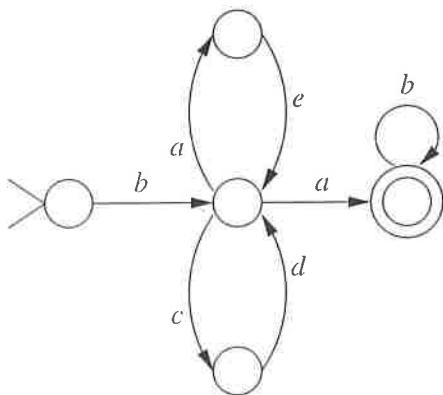


Figure 307:

Problem 719 Let L_1 be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where:

$$Q = \{q, s, t\}; \Sigma = \{a, b, c, d, e\}; \Gamma = \{A, D\}; F = \{t\};$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, A] \\ [q, \lambda, \lambda, s, \lambda] \\ [s, d, \lambda, s, D] \\ [s, b, \lambda, t, \lambda] \\ [t, c, D, t, \lambda] \\ [t, e, A, t, \lambda] \end{aligned}$$

M accepts by final state and empty stack.

(a) List 5 distinct strings that belong to L_1 . If this is impossible, state it and explain why.

Advice for Answer: The general template for string of L_1 is:

$$a^m d^\ell b c^\ell e^m \text{ for } m, \ell \geq 0$$

(b) Write a complete formal definition of a context-free grammar that generates the language:

$$L_1^*$$

If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, T)$, where

$\Sigma = \{a, b, c, d, e\}$, $V = \{T, S, B\}$, and the production set P is:

$$\begin{aligned} T &\rightarrow \lambda \mid TT \mid S \\ S &\rightarrow aSe \mid B \\ B &\rightarrow dBc \mid b \end{aligned}$$

Problem 720 Let L_1 be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where:

$$Q = \{q, s, t\}; \Sigma = \{a, b, c, d, e\}; \Gamma = \{A, D\}; F = \{t\};$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, s, AD] \\ [s, b, \lambda, s, \lambda] \\ [s, d, D, s, \lambda] \\ [s, \lambda, \lambda, t, \lambda] \\ [t, a, A, t, \lambda] \\ [t, d, A, t, \lambda] \\ [t, e, A, t, \lambda] \end{aligned}$$

M accepts by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.

(a) List 5 distinct strings that belong to L_1 . If this is impossible, state it and explain why.

Advice for Answer: The language L_1 , accepted by M is:

$$a b^* d b^* (a \cup d \cup e)$$

(b) Write a complete formal definition of a context-free grammar that generates the language:

$$L_1^*$$

If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, T)$, where
 $\Sigma = \{a, b, c, d, e\}$, $V = \{T, S, B, Z\}$,
and the production set P is:

$$\begin{aligned} T &\rightarrow \lambda \mid TT \mid S \\ S &\rightarrow aBdBZ \\ B &\rightarrow bB \mid \lambda \\ Z &\rightarrow a \mid d \mid e \end{aligned}$$

Problem 721 Let L_1 be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where:

$$Q = \{q, s, t\}; \Sigma = \{a, b, c, d, e\}; \Gamma = \{A, D\}; F = \{t\};$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, s, \lambda] \\ [q, b, \lambda, s, \lambda] \\ [s, c, \lambda, t, K] \\ [s, d, \lambda, t, D] \\ [t, b, D, t, \lambda] \\ [t, c, K, t, \lambda] \\ [t, e, \lambda, t, \lambda] \end{aligned}$$

M accepts by final state and empty stack.

(a) List 5 distinct strings that belong to L_1 . If this is impossible, state it and explain why.

Advice for Answer: The language L_1 , accepted by M is:

$$(a \cup b)(ce^* ce^* \cup de^* be^*)$$

(b) Construct a state-transition graph of a finite automaton that accepts the language L_1 . If such an automaton does not exist, prove it.

Advice for Answer: Convert the regular expression given in the answer to part (a).

Problem 722 Let L_1 be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where:

$$Q = \{q, r, s, t\}; \Sigma = \{a, b, c, d, e\}; \Gamma = \{A, D, E, K\}; F = \{r\};$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, \lambda, \lambda, r, \lambda] \\ [q, c, \lambda, s, DEK] \\ [s, b, \lambda, t, EAK] \\ [t, a, A, t, \lambda] \\ [t, d, D, q, \lambda] \\ [t, e, E, t, \lambda] \\ [t, c, K, t, \lambda] \end{aligned}$$

M accepts by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.

(a) List 5 distinct strings that belong to L_1 . If this is impossible, state it and explain why.

Advice for Answer: The language L_1 , accepted by M is:

$$(ebcaeced)^*$$

(b) Construct a state-transition graph of a finite automaton that accepts the language L_1 . If such an automaton does not exist, prove it.

Advice for Answer: Convert the regular expression given in the answer to part (a).

Problem 723 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, t\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{A\} \\ F &= \{t\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, AA] \\ [q, \lambda, \lambda, t, \lambda] \\ [t, b, A, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: By inspection of the automaton M , we conclude that the language L , accepted by M , it is the set of words whose general template is:

$$a^n b^{2n} \text{ where } n \geq 0$$

The required regular expression does not exist, since L is not a regular language.

To prove that L is not regular, assume the opposite—that L is regular.

Let k be the constant as in the Pumping Lemma for L . Consider a word $w \in L$, such that $w = a^m b^{2m}$, where $m > k$. Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < k < m$. Hence, $v = a^j$ for some j such that $0 < j < k$. By the pumping, for every $i \geq 0$, we have: $uv^i x \in L$. This means that all the words produced by the following template belong to L .

$$uv^i x = u v v^{i-1} x = u v a^{(i-1)j} x = a^{m+(i-1)j} b^{2m}$$

Consider the word:

$$w_2 = a^{m+j} b^{2m}$$

obtained by setting $i = 2$ in this template. Since $j > 0$ (as the pumping substring is never empty), it follows that:

$$2m < 2(m + j)$$

meaning that the number of b 's is less than twice the number of a 's, and word w_2 cannot belong to L . Since pumping produces strings that are not in L , we conclude that L cannot be regular, since it does not honor the Pumping Lemma.

- (b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b\}$, $V = \{S\}$, and P is:

$$S \rightarrow aSbb \mid \lambda$$

Problem 724 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, t\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, Z\} \\ F &= \{q\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, a, \lambda, t, ZA] \\ &[q, b, \lambda, t, ZAA] \\ &[q, c, \lambda, t, ZAAA] \\ &[t, d, A, t, \lambda] \\ &[t, d, Z, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

- (a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(add \cup bddd \cup cdddd)^*$$

- (b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b, c, d\}$, $V = \{S\}$, and P is:

$$S \rightarrow SS \mid \lambda \mid add \mid bddd \mid cdddd$$

Problem 725 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s, x\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, B, Z\} \\ F &= \{x\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, a, \lambda, x, \lambda] \\ &[q, a, \lambda, r, ZBA] \\ &[q, b, \lambda, r, ZBAA] \\ &[q, c, \lambda, r, ZBAAA] \\ &[r, a, A, r, \lambda] \\ &[r, d, B, r, \lambda] \\ &[r, \lambda, Z, s, \lambda] \\ &[s, \lambda, \lambda, q, \lambda] \\ &[s, c, \lambda, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

- (a) List 5 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: A regular expression for L is:

$$((aad \cup baad \cup caaad) c^*)^* a$$

- (b) Construct a state-transition graph of a finite automaton that accepts the language:

$$LL$$

If such an automaton does not exist, prove it.

Advice for Answer: See part (a).

Problem 726 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s, x\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, B, Z\} \\ F &= \{x\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, c, \lambda, r, ZBAAA] \\ &[q, c, \lambda, x, \lambda] \\ &[r, d, B, r, \lambda] \\ &[r, a, A, r, \lambda] \\ &[r, b, A, r, \lambda] \\ &[r, \lambda, Z, s, \lambda] \\ &[s, \lambda, \lambda, q, \lambda] \\ &[s, d, \lambda, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

- (a) List 5 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: A regular expression for L is:

$$(c(a \cup b)(a \cup b)(a \cup b)d d^*)^* c$$

- (b) Construct a state-transition graph of a finite automaton that accepts the language:

$$L^*$$

If such an automaton does not exist, prove it.

Advice for Answer: See part (a).

Problem 727 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, s\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, B\} \\ F &= \{s\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, c, \lambda, q, BAA] \\ [q, d, \lambda, s, \lambda] \\ [s, a, A, s, \lambda] \\ [s, b, B, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

- (a) Write 5 distinct strings that belong to L . If such strings do not exist, state it and explain why.

Answer:

$$\begin{aligned} d \\ cdaab \\ ccdaabaab \\ cccdaubaabaab \\ ccccdaabaabaabaab \end{aligned}$$

- (b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The general template for strings in L is:

$$c^n d(aab)^n, \text{ where } n \geq 0$$

which is generated by the grammar:

$G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$,
 $V = \{S\}$, and P is:

$$S \rightarrow cSaab \mid d$$

- (c) Write a complete formal definition of a context-free grammar that generates L^* (the Kleene star of L .) If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, T)$, where $\Sigma = \{a, b, c, d\}$,
 $V = \{T, S\}$, and P is:

$$\begin{aligned} T \rightarrow TT \mid \lambda \mid S \\ S \rightarrow cSaab \mid d \end{aligned}$$

Problem 728 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, s, t, v\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, B, Z\} \\ F &= \{q\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, b, \lambda, t, ZB] \\ [t, d, \lambda, s; AA] \\ [s, c, A, s, \lambda] \\ [s, \lambda, \lambda, v, \lambda] \\ [v, a, Z, q, \lambda] \\ [v, d, B, v, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

- (a) Write 4 distinct strings that belong to L . If such strings do not exist, state it and explain why.

Answer:

$$\begin{aligned} \lambda \\ bdccda \\ bdccda bdccda \\ bdccda bdccda bdccda \\ bdccda bdccda bdccda bdccda \end{aligned}$$

- (b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 308.

- (c) Write a complete formal definition of a regular context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The automaton given on Figure 308 is directly convertible into a regular grammar:

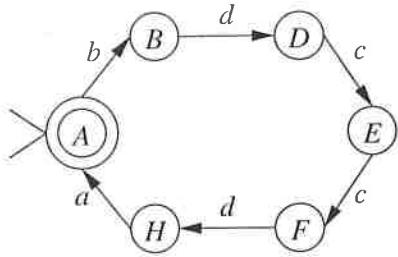


Figure 308:

$G = (V, \Sigma, P, A)$, where $\Sigma = \{a, b, c, d\}$, $V = \{A, B, D, E, F, H\}$, and P is:

$$\begin{aligned} A &\rightarrow bB \mid \lambda \\ B &\rightarrow dD \\ D &\rightarrow cE \\ E &\rightarrow cF \\ F &\rightarrow dH \\ H &\rightarrow aA \end{aligned}$$

Problem 729 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, t, r\} \\ \Sigma &= \{a, b, d\} \\ \Gamma &= \{A\} \\ F &= \{r\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, A] \\ [q, \lambda, \lambda, t, \lambda] \\ [t, d, \lambda, t, \lambda] \\ [t, \lambda, \lambda, r, \lambda] \\ [r, b, A, r, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: By inspection of the automaton M , we conclude that the language L , accepted by M , is the set of words whose general template is:

$$a^n d^m b^n \text{ where } m, n \geq 0$$

The required regular expression does not exist, since L is not a regular language.

To prove that L is not regular, assume the opposite—that L is regular.

Let k be the constant as in the Pumping Lemma for L . Consider a word $w \in L$, such that $w = a^n b^n$, where $n > k$, obtained from the general template by setting $m = 0$. Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < k < n$. Hence, $v = a^j$ for some j such that $0 < j < k$. By the pumping, for every $i \geq 0$, we have: $uv^i x \in L$. This means that all the words produced by the following template belong to L .

$$uv^i x = u v v^{i-1} x = u v a^{(i-1)j} x = a^{n+(i-1)j} b^n$$

Consider the word:

$$w_2 = a^{n+j} b^n$$

obtained by setting $i = 2$ in this template. Since $j > 0$ (as the pumping substring is never empty), it follows that:

$$n < n + j$$

meaning that the number of b 's is less than the number of a 's, and word w_2 cannot belong to L . Since pumping produces strings that are not in L , we conclude that L cannot be regular, since it does not honor the Pumping Lemma.

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b\}$, $V = \{S, A\}$, and P is:

$$\begin{aligned} S &\rightarrow aSb \mid A \\ A &\rightarrow dA \mid \lambda \end{aligned}$$

Problem 730 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, t, p, r\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, Z\} \\ F &= \{p\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, t, ZA] \\ [q, \lambda, \lambda, p, \lambda] \\ [t, d, A, t, \lambda] \\ [t, d, Z, q, \lambda] \\ [p, b, \lambda, r, ZAA] \\ [r, c, A, r, \lambda] \\ [r, c, Z, p, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

- (a) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer:

$$(add)^*(bccc)^*$$

This language is accepted by the automaton represented on Figure 309.

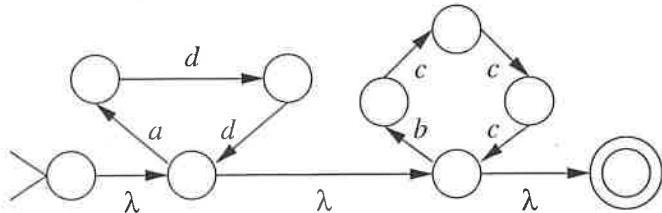


Figure 309:

- (b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow AA \mid \lambda \mid add \\ B &\rightarrow BB \mid \lambda \mid bccc \end{aligned}$$

Problem 731 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, t\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{A, B\} \\ F &= \{t\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, a, \lambda, q, A] \\ &[q, a, \lambda, t, B] \\ &[t, a, A, t, \lambda] \\ &[t, b, B, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

- (a) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: Such a finite automaton does not exist, since L is not regular.

The general template of the strings of L is:

$$a^{n+1}ba^n \text{ where } n \geq 0$$

To prove that L is not regular, assume the opposite—that L is regular.

Let k be the constant as in the Pumping Lemma for L . Consider a word $w \in L$, such that $w = a^{n+1}ba^n$, where $n > k$. Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < k < n$. Hence, the pumping part v is entirely within the a -segment, meaning that $v = a^j$ for some j such that $0 < j < k$. By the pumping, for every $i \geq 0$, we have: $uv^i x \in L$. This means that all the words produced by the following template belong to L .

$$uv^i x = uvv^{i-1}x = uva^{(i-1)j}x = a^{n+1+(i-1)j}ba^n$$

Consider the word:

$$w_2 = a^{n+1+j}ba^n$$

obtained by setting $i = 2$ in this template. Since $j > 0$ (as the length of the non-empty pumping part) we see that: $n + 1 < n + j$, meaning that the length of the leading a -segment of w_2 is by more than one greater than the length of the trailing a -segment. This means that $w_2 \notin L$, whence the contradiction.

- (b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S\}$, and P is:

$$S \rightarrow aSa \mid ab$$

Problem 732 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, t\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A, B\} \\ F &= \{t\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, a, \lambda, q, A] \\ &[q, a, \lambda, t, B] \\ &[t, b, A, t, \lambda] \\ &[t, c, B, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

Write a complete formal definition of a context-free grammar that generates the language:

$$LL$$

If such a grammar does not exist, prove it.

Answer: By inspection of the automaton, we conclude that:

$$L = \{a^{n+1}cb^n \mid n \geq 0\}$$

Hence, LL is generated by the grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AA \\ A &\rightarrow aAb \mid ac \end{aligned}$$

Problem 733 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, t\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A, B\} \\ F &= \{t\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, A] \\ [q, \lambda, \lambda, t, B] \\ [t, b, A, t, \lambda] \\ [t, c, B, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

Write a complete formal definition of a context-free grammar that generates the language:

$$L^*$$

If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow SS \mid \lambda \mid A \\ A &\rightarrow aAb \mid c \end{aligned}$$

Problem 734 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, B, Z\} \\ F &= \{q\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, c, \lambda, s, ZBAAA] \\ [s, \lambda, \lambda, r, \lambda] \\ [s, d, \lambda, s, \lambda] \\ [r, a, A, r, \lambda] \\ [r, b, B, r, \lambda] \\ [r, \lambda, Z, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on

the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 310.

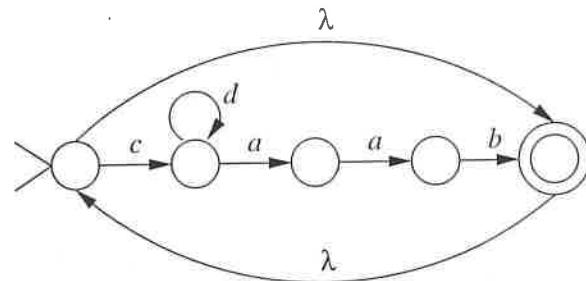


Figure 310:

Problem 735 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, B, Z\} \\ F &= \{r\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, c, \lambda, q, ZBAAA] \\ [q, \lambda, \lambda, r, \lambda] \\ [r, a, A, r, \lambda] \\ [r, b, B, r, \lambda] \\ [r, d, Z, r, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

Write a complete formal definition of a context-free grammar G that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S\}$, and P is:

$$S \rightarrow cSaabd \mid \lambda$$

Problem 736 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, Z\} \\ F &= \{q\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, \lambda, \lambda, r, ZAAA] \\ & [r, a, A, r, \lambda] \\ & [r, b, A, r, \lambda] \\ & [r, c, A, r, \lambda] \\ & [r, d, A, r, \lambda] \\ & [r, \lambda, Z, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 311.

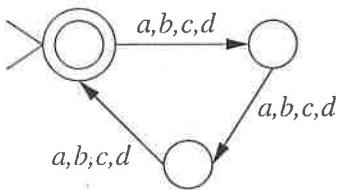


Figure 311;

Problem 737 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$, where

$$\begin{aligned} Q &= \{q, r\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A, Z\} \\ F &= \{q\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, \lambda, \lambda, r, Z] \\ & [q, \lambda, Z, q, \lambda] \\ & [r, a, Z, r, ZA] \\ & [r, c, A, r, \lambda] \\ & [r, b, Z, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the left-most symbol X_1 is pushed first, while the right-most symbol X_n is pushed last.)

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$((ac)^*b)^*$$

(b) Write a complete formal definition of a context-free grammar G that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow SS \mid \lambda \mid Ab \\ A &\rightarrow acA \mid \lambda \end{aligned}$$

Problem 738 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$, where $Q = \{q, r\}$, $\Sigma = \{a, b, c, d\}$, $\Gamma = \{A, B\}$, $F = \{r\}$, and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, q, A] \\ & [q, b, \lambda, q, B] \\ & [q, c, \lambda, q, A] \\ & [q, \lambda, r, \lambda] \\ & [r, d, A, r, \lambda] \\ & [r, b, B, r, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Write a complete formal definition of a context-free grammar G that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S\}$, and P is:

$$S \rightarrow aSd \mid cSd \mid bSb \mid \lambda$$

(b) Write a complete formal definition of a context-free grammar G_1 that generates L^* . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, T)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, T\}$, and P is:

$$\begin{aligned} T &\rightarrow TT \mid \lambda \mid S \\ S &\rightarrow aSd \mid cSd \mid bSb \mid \lambda \end{aligned}$$

Problem 739 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{A, B\} \\ F &= \{r\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, q, A] \\ & [q, b, \lambda, q, B] \\ & [q, \lambda, r, \lambda] \\ & [r, b, A, r, \lambda] \\ & [r, a, B, r, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost

symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

Write a complete formal definition of a context-free grammar that generates L^* (the Kleene star of L). If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, K)$, where $\Sigma = \{a, b\}$ is the set of terminals; $V = \{S, K\}$ is the set of variables; K is the start symbol, and the production set P is:

$$\begin{aligned} S &\rightarrow aSb \mid bSa \mid \lambda \\ K &\rightarrow KK \mid \lambda \mid S \end{aligned}$$

Problem 740 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{A\} \\ F &= \{r\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, A] \\ [r, b, A, r, \lambda] \\ [q, \lambda, \lambda, r, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write a complete formal definition of a context-free grammar G that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b\}$, $V = \{S\}$, and P is:

$$S \rightarrow aSb \mid \lambda$$

(b) Write a complete formal definition of a context-free grammar G_1 that generates \overline{L} (the complement of L). If such a grammar does not exist, prove it.

Answer: $G_1 = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow aSb \mid A \mid B \mid DbaD \\ A &\rightarrow aA \mid a \\ B &\rightarrow bB \mid b \\ D &\rightarrow DD \mid \lambda \mid a \mid b \end{aligned}$$

To verify the construction, observe that every string of \overline{L} contains some letters out of order or contains a surplus of one letter over the other.

Problem 741 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s, t\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{A\} \\ F &= \{t\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, r, A] \\ [r, a, A, s, \lambda] \\ [s, a, \lambda, q, A] \\ [t, b, A, t, \lambda] \\ [q, \lambda, \lambda, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S\}$, and P is:

$$S \rightarrow aaaSb \mid \lambda$$

Problem 742 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A\} \\ F &= \{q\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, \lambda, \lambda, r, A] \\ [r, a, A, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

Draw a state-transition graph of a finite automaton that accepts \overline{L} (the complement of L). If such an automaton does not exist, prove it.

Answer: Observe that L is represented by the regular expression a^* . Hence: $\overline{L} = a^*(b \cup c)(a \cup b \cup c)^*$, whence the automaton represented on Figure 312.

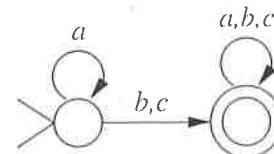


Figure 312:

Problem 743 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A, B\} \\ F &= \{s\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, \lambda] \\ [q, b, \lambda, q, B] \\ [q, c, \lambda, q, \lambda] \\ [q, \lambda, \lambda, r, \lambda] \\ [r, \lambda, B, s, \lambda] \\ [s, \lambda, B, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

Write a complete formal definition of a context-free grammar that generates \overline{L} (the complement of L). If such a grammar does not exist, prove it.

Answer: Observe that L contains exactly those strings over $\{a, b, c\}$ that contain at least one occurrence of the letter b . Hence, its complement is represented by the regular expression $(a \cup c)^*$, which corresponds to the grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$ is the set of terminals; $V = \{S\}$ is the set of variables; S is the start symbol, and the production set P is:

$$S \rightarrow aS \mid cS \mid \lambda$$

Problem 744 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A, B, D, Z\} \\ F &= \{q\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, \lambda, \lambda, r, ZABDA] \\ [r, a, A, r, \lambda] \\ [r, b, B, r, \lambda] \\ [r, c, D, r, \lambda] \\ [r, \lambda, Z, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

Write a regular expression that represents L . If such a regular expression does not exist, prove it.

Answer:

$$(acba)^*$$

Problem 745 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A, B, D, Z\} \\ F &= \{q\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, \lambda, \lambda, r, ZAB] \\ [q, \lambda, \lambda, r, ZDA] \\ [r, a, A, r, \lambda] \\ [r, b, B, r, \lambda] \\ [r, c, D, r, \lambda] \\ [r, \lambda, Z, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

Write a complete formal definition of a regular context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: The language L is represented by the regular expression:

$$(ba \cup ac)^*$$

which corresponds to the finite automaton given on Figure 313, which in turn is converted to the regular grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$ is the set of terminals; $V = \{S, A, B\}$ is the set of variables; S is the start symbol, and the production set P is:

$$\begin{aligned} S &\rightarrow bA \mid ab \\ A &\rightarrow aS \\ B &\rightarrow cS \end{aligned}$$

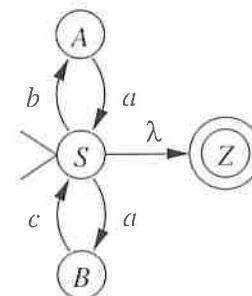


Figure 313:

Problem 746 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s, t\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{A\} \\ F &= \{t\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, r, A] \\ [r, a, A, q, \lambda] \\ [t, b, \lambda, s, A] \\ [s, b, A, t, \lambda] \\ [q, \lambda, \lambda, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 314.

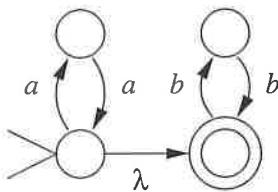


Figure 314:

Problem 747 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s, t\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{A, B\} \\ F &= \{t\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, r, A] \\ [q, b, \lambda, r, B] \\ [r, a, A, q, \lambda] \\ [r, a, B, q, \lambda] \\ [t, a, \lambda, s, A] \\ [s, a, A, t, \lambda] \\ [q, \lambda, \lambda, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 315.

Problem 748 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$, where

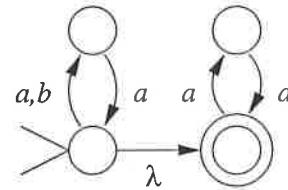


Figure 315:

$Q = \{q, r, s\}$, $\Sigma = \{a, b, d\}$, $\Gamma = \{A\}$, $F = \{s\}$, and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, A] \\ [r, d, \lambda, r, A] \\ [s, b, A, s, \lambda] \\ [q, \lambda, \lambda, r, \lambda] \\ [r, \lambda, \lambda, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Write a complete formal definition of a context-free grammar G that generates L . If such a grammar does not exist, prove it.

Answer: Note that:

$$L = \{a^m d^n b^{m+n} \mid m, n \geq 0\}$$

whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, d\}$, $V = \{S, A\}$, and P is:

$$\begin{aligned} S &\rightarrow aSb \mid A \\ A &\rightarrow dAb \mid \lambda \end{aligned}$$

(b) Write a complete formal definition of a *regular* context-free grammar G_1 that generates L . If such a grammar does not exist, prove it.

Answer: The language:

$$L = \{a^m d^n b^{m+n} \mid m, n \geq 0\}$$

is not regular, and there does not exist a regular grammar to generate it.

To prove this, assume the opposite, that L is regular. Let η be the constant as in the Pumping Lemma for L . Let $m > \eta$; then the word:

$$a^m b^m$$

belongs to L , as it is obtained from the general template by setting $n = 0$.

In any “pumping” decomposition such that:

$$a^m b^m = uvx$$

we have:

$$|uv| \leq \eta < m$$

Hence, the “pumping” substring v consists entirely of a 's, say $v = a^\ell$. Recall that $\ell > 0$, since the “pumping” substring cannot be empty. By the pumping, every word

of the form uv^ix , $i \geq 0$, belongs to L . However, such a word is of the form:

$$w_1 = a^{m+(i-1)\ell} b^m$$

Observe that the total number of a 's and d 's in this word is equal to $m + (i - 1)\ell$. Since $m + (i - 1)\ell > m$ whenever $i > 1$, word w_1 has more a 's and d 's than is appropriate for its number of b 's. Hence, $w_1 \notin L$, which is a contradiction.

Problem 749 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A, B\} \\ F &= \{q\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, r, AA] \\ [r, b, A, s, BBB] \\ [s, c, B, s, \lambda] \\ [s, b, A, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$(abccb)^*$$

(b) Draw a state-transition graph of a finite automaton M_1 that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 316.

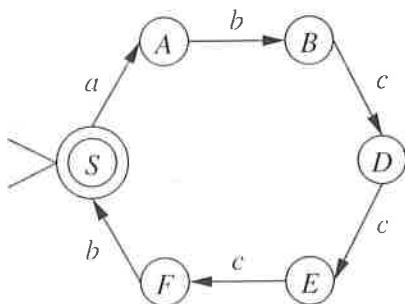


Figure 316:

(c) Write a complete formal definition of a context-free grammar G that generates L , such that G is a regular grammar. If such grammar does not exist, prove it.

Answer: The algorithmic conversion of the automaton obtained in the answer to part (b) gives the following

grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D, E, F\}$, and P is:

$$\begin{aligned} S &\rightarrow aA \mid \lambda \\ A &\rightarrow bB \\ B &\rightarrow cD \\ D &\rightarrow cE \\ E &\rightarrow cF \\ F &\rightarrow bS \end{aligned}$$

(d) Write a complete formal definition of a context-free grammar G_1 that generates L , such that G_1 is not a regular grammar. If such grammar does not exist, prove it.

Answer: The algorithmic conversion of the regular expression obtained in the answer to part (a) gives the following grammar: $G_1 = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S\}$, and P is:

$$S \rightarrow SS \mid \lambda \mid abccb$$

Problem 750 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{A\} \\ F &= \{r, s\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, \lambda] \\ [r, a, \lambda, r, \lambda] \\ [s, a, \lambda, r, \lambda] \\ [q, b, \lambda, s, AAA] \\ [s, b, A, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$a^* b b b b a^*$$

(b) Draw a state-transition graph of a finite automaton M_1 that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 317.

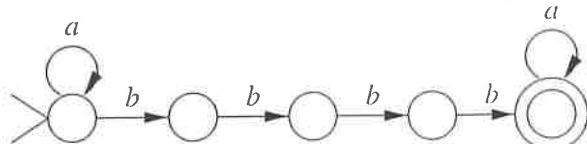


Figure 317:

Problem 751 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{A\} \\ F &= \{r, s\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, A] \\ [r, a, A, r, \lambda] \\ [s, a, \lambda, s, \lambda] \\ [q, \lambda, \lambda, s, \lambda] \\ [s, \lambda, \lambda, r, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S\}$, and P is:

$$S \rightarrow aS \mid \lambda$$

(b) Draw a state-transition graph of a finite automaton M_1 that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 318.

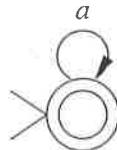


Figure 318:

Problem 752 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, r, s\}$; $\Sigma = \{a, b\}$; $\Gamma = \{A\}$; $F = \{q\}$; and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, s, A] \\ [r, a, A, q, \lambda] \\ [s, b, \lambda, s, \lambda] \\ [s, \lambda, \lambda, r, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) List 5 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer:

aaaa
aba
abbba abba
aa abba abba
aa aba aa abba aa

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: $(a b^* a)^*$

(c) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, B\}$, and P is:

$$\begin{aligned} S \rightarrow SS \mid \lambda \mid aBa \\ B \rightarrow bB \mid \lambda \end{aligned}$$

Problem 753 Let L_1 be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A\} \\ F &= \{r\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, A] \\ [q, c, \lambda, r, \lambda] \\ [r, b, A, r, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Write a complete formal definition of a context-free grammar G that generates L_1 . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b, c\}$, $V = \{S\}$, and the rule set P is:

$$S \rightarrow aSb \mid c$$

(b) Describe the algorithm that should be employed by a program that solves the following problem:

INPUT: An arbitrary string x over Σ .

QUESTION: Does x belong to L_1 ?

Explain your answer. If such algorithm does not exist, prove it.

Answer: This algorithm simulates the operation of the pushdown automaton M defined in part (a). Since M accepts when x belongs to L_1 and rejects when x does not belong to L_1 , our algorithm says yes when M accepts, and says no when M rejects.

Problem 754 Let L_4 be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A\} \\ F &= \{r\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, \lambda, \lambda, r, AA] \\ & [r, a, \lambda, r, \lambda] \\ & [r, b, \lambda, r, \lambda] \\ & [r, c, A, r, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Write a regular expression that defines L_4 . If such regular expression does not exist, prove it.

Answer:

$$(a \cup b)^* c (a \cup b)^* c (a \cup b)^*$$

(b) **Answer:** Draw a state-transition graph of a finite automaton M_1 that accepts L_4 . If such automaton does not exist, prove it.

Answer: See Figure 319.

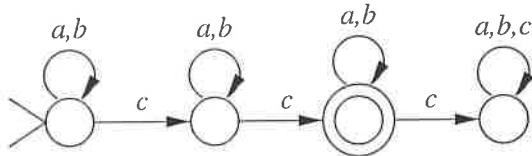


Figure 319:

(c) Draw a state-transition graph of a finite automaton M_2 that accepts $\overline{L_4}$ (the complement of L_4). If such automaton does not exist, prove it.

Answer: See Figure 320.

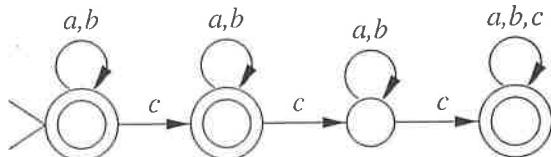


Figure 320:

Problem 755 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, s, t, v\}$; $\Sigma = \{a, b, c, d\}$; $\Gamma = \{A, B, Z\}$; $F = \{v\}$ and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, t, Z] \\ & [t, a, Z, v, \lambda] \\ & [t, d, \lambda, s, BA] \\ & [s, b, B, s, \lambda] \\ & [s, c, A, s, \lambda] \\ & [s, \lambda, Z, t, Z] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost

symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) List 5 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer:

$$\begin{aligned} & aa \\ & a dcba \\ & a dcba dcba \\ & a dcba dcba dcba \\ & a dcba dcba dcba dcba \end{aligned}$$

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: $a (dcba)^*$

(c) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A\}$, and P is:

$$\begin{aligned} S &\rightarrow aAa \\ A &\rightarrow dcba \mid \lambda \end{aligned}$$

Problem 756 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{A, Z\} \\ F &= \{q\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, r, Z] \\ & [q, a, \lambda, r, ZA] \\ & [q, a, \lambda, r, ZAA] \\ & [r, b, A, r, \lambda] \\ & [r, b, Z, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 321.

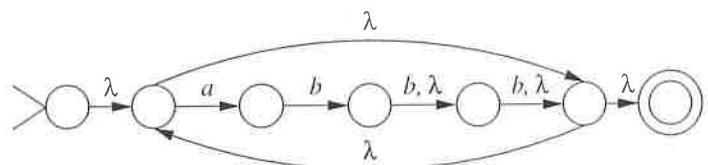


Figure 321:

(b) Does every pushdown automaton have an equivalent finite automaton? Explain your answer briefly.

Answer: No—some context-free languages are not regular. In fact, it is undecidable for an arbitrary pushdown automaton whether the language accepted by it is regular.

Problem 757 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A\} \\ F &= \{q\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, AA] \\ [q, b, A, r, \lambda] \\ [r, b, A, s, \lambda] \\ [s, c, A, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S\}$, and P is:

$$S \rightarrow \lambda$$

(b) Draw a state-transition graph of a finite automaton M that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 322.

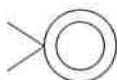


Figure 322:

Problem 758 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A\} \\ F &= \{s\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, AA] \\ [q, \lambda, \lambda, r, \lambda] \\ [r, b, \lambda, r, A] \\ [r, \lambda, \lambda, s, \lambda] \\ [s, c, A, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Draw a state-transition graph of a finite automaton M_1 that accepts L . If such automaton does not exist, prove it.

Answer: Such finite automaton does not exist, because:

$$L = \{a^m b^n c^{2m+n} \mid m, n \geq 0\}$$

which is not a regular language.

To prove that L is not regular, assume the opposite. Let k be the constant as in the Pumping Lemma. Let $m > k$ and $n = 0$; then $a^m c^{2m} \in L$. In the “pumping” decomposition: $a^m c^{2m} = uvx$, we have that $|uv| \leq k < m$, hence the “pumping” substring v consists entirely of a 's, say $v = a^j$. Recall that $j > 0$, since the “pumping” substring cannot be empty. By the pumping, every word of the form $uv^i x$, $i \geq 0$, belongs to L . However, such a word has $2m$ occurrences of c and $m + (i - 1)j$ occurrences of a , whereas it should have m occurrences of a for $2m$ occurrences of c . Since $m + (i - 1)j > m$ whenever $i > 1$, this is a contradiction.

(b) Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:

$$\Sigma = \{a, b, c\}, V = \{S, A\},$$

and the production set P is:

$$\begin{aligned} S \rightarrow aSc &\mid A \\ A \rightarrow bAc &\mid \lambda \end{aligned}$$

(c) Is L a recursively enumerable language? Explain your answer.

Answer: Yes—by the answer to part (b), L is context-free. Every context-free language is recursively enumerable.

Problem 759 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s, x, y\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, Z\} \\ F &= \{y\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, c, \lambda, x, Z] \\ [x, a, \lambda, r, A] \\ [r, a, \lambda, s, A] \\ [s, b, A, s, \lambda] \\ [s, d, Z, x, Z] \\ [x, c, Z, y, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$c(aabbd)^*c$$

(b) Is it possible to write a computer program (algorithm) that operates as follows:

INPUT: An arbitrary string ξ consisting of letters $\{a, b, c, d\}$.

OUTPUT: yes if $\xi \in L$ and no if $\xi \notin L$.

Explain your answer briefly.

Answer: Yes—such an algorithm simulates the pushdown automaton M , defined in part (a), which accepts L .

Problem 760 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A, B, D\} \\ F &= \{r\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, A] \\ [q, b, \lambda, q, B] \\ [q, c, \lambda, q, D] \\ [q, \lambda, \lambda, r, \lambda] \\ [r, a, A, r, \lambda] \\ [r, b, B, r, \lambda] \\ [r, c, D, r, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$ is the set of terminals; $V = \{S\}$ is the set of variables; S is the start symbol, and the production set P is:

$$S \rightarrow aSa \mid bSb \mid bSc \mid \lambda$$

Problem 761 Let L be the language accepted by the pushdown automaton

$$M = (Q, \Sigma, \Gamma, \delta, q_0, F)$$

where:

$$\begin{aligned} Q &= \{q_0, q_1, q_2\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A, B\} \\ F &= \{q_2\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} \delta(q_0, a, \epsilon) &= \{(q_0, A)\} \\ \delta(q_0, b, \epsilon) &= \{(q_0, B)\} \\ \delta(q_0, c, \epsilon) &= \{(q_1, \epsilon)\} \\ \delta(q_1, c, \epsilon) &= \{(q_2, \epsilon)\} \\ \delta(q_2, a, A) &= \{(q_2, \epsilon)\} \\ \delta(q_2, b, B) &= \{(q_2, \epsilon)\} \end{aligned}$$

Write a complete formal definition of a context-free grammar G that generates L . If such grammar G does not exist, explain why.

Answer: The language L accepted by the automaton M is:

$$L = \{wccw^R \mid w \in \{a, b\}^*\}$$

This language is generated by the grammar:

$G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c\}$ is the set of terminals;

$V = \{S\}$ is the set of variables;

S is the start symbol;

and the set of productions P is:

$$S \rightarrow aSa \mid bSb \mid cc$$

Problem 762 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A, B\} \\ F &= \{r\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, A] \\ [q, b, \lambda, q, B] \\ [q, c, \lambda, r, \lambda] \\ [r, a, A, r, \lambda] \\ [r, b, B, r, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$,

where $\Sigma = \{a, b, c\}$; $V = \{S\}$; and P is:

$$S \rightarrow aSa \mid bSb \mid c$$

(b) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: The language L is not accepted by any finite automaton, since it is not regular. Observe that L is the set of odd-length palindromes whose middle symbol is c , while all the other symbols are either a or b . To prove that L is not regular, assume the opposite: that L is regular.

Let k be the constant as in the Pumping Lemma for L . Consider a word $w \in L$, such that $w = a^m c a^m$, where $m > k$. Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < k < m$. Hence, $v = a^j$ for some $j > 0$. By the pumping, for every $i \geq 0$, we have: $uv^i x \in L$. However:

$$uv^i x = u v v^{i-1} x = u v a^{(i-1)j} x = a^{m+(i-1)j} c a^m$$

Since:

$$\beta = (i-1)j > 0 \text{ whenever } i \geq 2$$

we conclude that the word is in the form:

$$a^m a^\beta c a^m \text{ for some } \beta > 0$$

Observe that the $(m+1)$ st symbol from the right is c , while the $(m+1)$ st symbol from the left is a , meaning that the word is not a palindrome. Hence, the word cannot be in the language, whence the contradiction.

(c) Is L a recursive language? Explain your answer.

Answer: Yes— L is context-free, because it is accepted by the pushdown automaton M ; every context-free language is recursive.

(d) Is L a recursively enumerable language? Explain your answer.

Answer: Yes—by the answer given in part (c), L is recursive; every recursive language is recursively enumerable.

Problem 763 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{A\} \\ F &= \{r\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, a, \lambda, q, A] \\ &[q, a, \lambda, q, AA] \\ &[q, \lambda, \lambda, r, \lambda] \\ &[r, b, A, r, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Write 5 distinct strings that belong to L . If such strings do not exist, prove it.

(b) Write 5 distinct strings over alphabet Σ that do not belong to L . If such strings do not exist, prove it.

Answer:

$\in L$	$\notin L$
λ	a
ab	b
abb	ba
$aabb$	$aabb$
$aabbb$	aab

(c) Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, prove it.

Answer: The language accepted by M is:

$$L = \{a^m b^n \mid m \leq n \leq 2m\}$$

and it is generated by the grammar: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSb \mid aSbb \mid \lambda$$

(d) Is L countable? Explain your answer briefly.

Answer: Yes— L is a subset of Σ^* , which is (infinite and) countable.

Problem 764 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, s, t, v\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, Z\} \\ F &= \{v\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, d, \lambda, t, Z] \\ &[t, d, Z, v, \lambda] \\ &[t, a, \lambda, s, A] \\ &[s, b, A, t, \lambda] \\ &[s, c, A, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$d(ab \cup ac)^* d$$

Problem 765 **(a)** Let L_1 be the language accepted by the pushdown automaton:

$$M_1 = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s, t\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A\} \\ F &= \{r\} \end{aligned}$$

and the transition set δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, q, A] \\ & [q, \lambda, \lambda, r, \lambda] \\ & [r, b, A, s, \lambda] \\ & [s, c, \lambda, t, \lambda] \\ & [t, d, \lambda, r, \lambda] \end{aligned}$$

(Recall that M_1 is defined so as to accept by final state and empty stack.)

Write a complete formal definition of a context-free grammar G_1 that generates L_1 . If such grammar does not exist, prove it.

The language accepted by M_1 is:

$$L_1 = \{a^m(bcd)^m \mid m \geq 0\}$$

and it is generated by the grammar: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b, c, d\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSbcd \mid \lambda$$

(b) Let L_2 be the language accepted by the pushdown automaton:

$$M_2 = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s, t\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, B\} \\ F &= \{q\} \end{aligned}$$

and the transition set δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, r, A] \\ & [r, b, \lambda, s, B] \\ & [s, c, B, t, \lambda] \\ & [t, d, A, q, \lambda] \end{aligned}$$

(Recall that M_2 is defined so as to accept by final state and empty stack.)

Write a regular expression that defines L_2 . If such regular expression does not exist, prove it.

Answer:

$$(abcd)^*$$

Problem 766 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, s\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{A\} \\ F &= \{s\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, q, AAA] \\ & [s, b, A, s, \lambda] \\ & [q, \lambda, \lambda, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write a regular expression that defines L .

If such a regular expression does not exist, prove it.

Answer: The automaton M accepts the language:

$$L = \{a^m b^{3m} \mid m \geq 0\}$$

The required regular expression does not exist, since L is not a regular language.

To prove this, assume the opposite, that L is regular. Let η be the constant as in the Pumping Lemma for L . Let $m > \eta$; then $a^m b^{3m} \in L$. In any “pumping” decomposition such that $a^m b^{3m} = uvx$, we have: $|uv| \leq \eta < m$. Hence, the “pumping” substring v consists entirely of a 's, say $v = a^\ell$. Recall that $\ell > 0$, since the “pumping” substring cannot be empty. By the pumping, every word of the form $uv^i x$, $i \geq 0$, belongs to L . However, such a word is of the form:

$$w_1 = a^{m+(i-1)\ell} b^{3m}$$

Since $m + (i-1)\ell > m$ whenever $i > 1$, word w_1 has more a 's than is appropriate for its number of b 's. Hence, $w_1 \notin L$, which is a contradiction.

(b) Let \mathcal{T} be a set of strings over alphabet $\{a, b\}$, defined as follows:

String w is a member of \mathcal{T} if and only if w is not accepted by the pushdown automaton M .

What is the cardinality of \mathcal{T} ? Explain your answer.

Answer: \mathcal{T} is infinite and countable:

$$|\mathcal{T}| = |\overline{L}| = \aleph_0$$

To see that \mathcal{T} is countable, recall that every language is countable. To see that \mathcal{T} is infinite, observe, for instance, that it contains the infinite set a^* .

Problem 767 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, s\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, B, D\} \\ F &= \{q\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, c, \lambda, s, BADDA] \\ & [s, a, A, s, \lambda] \\ & [s, b, B, q, \lambda] \\ & [s, d, D, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

- (a) Draw a state-transition graph of a finite automaton M_1 that accepts L . If such an automaton does not exist, prove it.

Answer: The automaton M accepts the language represented by the regular expression:

$$(caddab)^*$$

whence the automaton given on Figure 323.

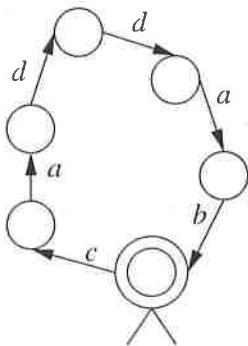


Figure 323:

- (b) Compare the cardinalities of L and \overline{L} (the complement of L), and determine which of them (if any) is greater. Explain your answer.

Answer: Both L and \overline{L} are infinite and countable:

$$|L| = |\overline{L}| = \aleph_0$$

To see that both languages are countable, recall that every language is countable. To see that they are both infinite, observe that each one contains a Kleene star of a non-empty set of non-empty strings. By the answer to part (a), L_1 itself is such a set. L_2 contains, for instance, the set a^* .

Problem 768 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, s\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, B, D\} \\ F &= \{s\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, a, \lambda, q, ABA] \\ &[q, b, \lambda, q, DD] \\ &[s, a, D, s, \lambda] \\ &[s, c, A, s, \lambda] \\ &[s, d, B, s, \lambda] \\ &[q, \lambda, \lambda, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

- (a) Write a complete formal definition of a context-free grammar G that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S\}$, and P is:

$$S \rightarrow aScdc \mid bSaa \mid \lambda$$

- (b) Let \mathcal{T} be a set of strings over alphabet $\{a, b, c, d\}$, defined as follows:

String w is a member of \mathcal{T} if and only if $w \in b^*a^*$ and w is accepted by the pushdown automaton M .

Is \mathcal{T} a context-free language? Prove your answer.

Answer: Yes. By construction:

$$\mathcal{T} = L \cap b^*a^*$$

L is context-free, as the language accepted by the pushdown automaton M . Language b^*a^* is certainly regular, since it is represented by a regular expression. The intersection of a regular language and a context-free language is always context-free—hence, \mathcal{T} is context-free.

Problem 769 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{A\} \\ F &= \{r\} \end{aligned}$$

and the transition set δ is defined as follows:

$$\begin{aligned} &[q, a, \lambda, q, A] \\ &[q, a, \lambda, q, AA] \\ &[q, a, \lambda, q, AAA] \\ &[q, \lambda, \lambda, r, \lambda] \\ &[r, b, A, r, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Advice for Answer:

$$L = \{a^m b^n \mid m \leq n \leq 3m\}$$

Problem 770 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{A\} \\ F &= \{r\} \end{aligned}$$

and the transition set δ is defined as follows:

$$\begin{aligned} &[q, a, \lambda, q, A] \\ &[q, a, \lambda, q, AA] \\ &[q, a, \lambda, q, AAA] \\ &[q, b, \lambda, r, \lambda] \\ &[r, b, A, r, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSb \mid aSbb \mid aSbbb \mid b$$

Problem 771 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A\} \\ F &= \{s\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, a, \lambda, q, AAA] \\ &[q, \lambda, \lambda, r, \lambda] \\ &[r, b, \lambda, r, AA] \\ &[r, \lambda, \lambda, s, \lambda] \\ &[s, c, A, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: Such a finite automaton does not exist, since:

$$L = \{a^n b^m c^{3n+2m} \mid m, n \geq 0\}$$

which is not a regular language.

To prove that L is not regular, assume the opposite. Let k be the constant as in the Pumping Lemma for L . Let m be arbitrary, and let $n > k$; then $a^n b^m c^{3n+2m} \in L$. In the “pumping” decomposition: $a^n b^m c^{3n+2m} = uvx$,

we have that $|uv| \leq k < n$, hence the “pumping” substring v consists entirely of a 's, say $v = a^j$. Recall that $j > 0$, since the “pumping” substring cannot be empty. By the pumping, every word of the form $uv^i x$, $i \geq 0$, belongs to L . However, such a word has m occurrences of b , $n + (i-1)j$ occurrences of a , and $3n + 2m$ occurrences of c . However, by definition of L , it should have $3(n + (i-1)j) + 2m$ occurrences of c for $n + (i-1)j$ occurrences of a and m occurrences of b . Since $3(n + (i-1)j) + 2m > 3n + 2m$ whenever $i > 1$, this is a contradiction.

(b) Is L context-free? Explain your answer briefly.

Answer: Yes— L is accepted by the pushdown automaton M , defined in part (a).

Problem 772 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A\} \\ F &= \{r\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, a, \lambda, q, A] \\ &[r, b, A, r, \lambda] \\ &[q, c, \lambda, r, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Write a regular expression that defines L .

If such regular expression does not exist, prove it.

Answer: This regular expression does not exist, since M accepts the language:

$$L = \{a^n c b^n \mid n \geq 0\}$$

which is not regular.

To prove this, assume the opposite, that L is regular. Let k be the constant as in the Pumping Lemma for L . Let $m > k$; then $a^m c b^m \in L$. In any “pumping” decomposition such that $a^m c b^m = uvx$, we have: $|uv| \leq k < m$. Hence, the “pumping” substring v consists entirely of a 's, say $v = a^\ell$. Recall that $\ell > 0$, since the “pumping” substring cannot be empty. By the pumping, every word of the form $uv^i x$, $i \geq 0$, belongs to L . However, such a word is of the form:

$$w_1 = a^{m+(i-1)\ell} c b^m$$

which means that w_1 has $m + (i-1)\ell$ occurrences of a and m occurrences of b . Whenever $i > 1$, it is true that $m + (i-1)\ell > m$. This means that w_1 has more a 's than b 's, and cannot belong to L , which is a contradiction.

(b) Let S be a class of languages over alphabet $\{a, b, c\}$, defined as follows:

Language L is a member of \mathcal{S} if and only if L is accepted by some pushdown automaton.

What is the cardinality of \mathcal{S} ? Explain your answer briefly.

Answer:

$$|\mathcal{S}| = \aleph_0$$

Class \mathcal{S} is infinite and countable. There are infinitely many pushdown automata, yet every pushdown automaton is a finite string, and the set of all finite strings over any alphabet is countable.

Problem 773 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s, t\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{A\} \\ F &= \{t\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, r, A] \\ [r, a, \lambda, q, \lambda] \\ [t, b, \lambda, s, A] \\ [s, b, A, t, \lambda] \\ [q, \lambda, \lambda, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(bb)^*$$

Problem 774 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A, B\} \\ F &= \{r\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, AA] \\ [r, b, A, r, \lambda] \\ [q, c, \lambda, s, B] \\ [s, c, B, r, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer: This regular expression does not exist, since M accepts the language:

$$L = \{a^n c c b^{2n} \mid n \geq 0\}$$

which is not regular.

To prove this, assume the opposite, that L is regular. Let k be the constant as in the Pumping Lemma for L . Let $m > k$; then $a^m c c b^{2m} \in L$. In any “pumping” decomposition such that $a^m c c b^{2m} = uvx$, we have: $|uv| \leq k < m$. Hence, the “pumping” substring v consists entirely of a 's, say $v = a^\ell$. Recall that $\ell > 0$, since the “pumping” substring cannot be empty. By the pumping, every word of the form $uv^i x$, $i \geq 0$, belongs to L . However, such a word is of the form:

$$w_1 = a^{m+(i-1)\ell} c c b^{2m}$$

which means that w_1 has $m + (i - 1)\ell$ occurrences of a and $2m$ occurrences of b . Whenever $i > 1$, it is true that $2(m + (i - 1)\ell) > 2m$. This means that w_1 has fewer b 's than twice the number of a 's, and cannot belong to L , which is a contradiction.

(b) Let \mathcal{T} be a set of strings over alphabet $\{a, b, c\}$, defined as follows:

String w is a member of \mathcal{T} if and only if w is not accepted by the pushdown automaton M .

What is the cardinality of \mathcal{T} ? Explain your answer briefly.

Answer:

$$|\mathcal{T}| = |\overline{L}| = \aleph_0$$

Set $\mathcal{T} = \overline{L}$ is infinite and countable. To see that \mathcal{T} is infinite, observe that it contains (for example) a^* . To see that \mathcal{T} is countable, recall that the entire set of strings $\{a, b, c\}^*$ is countable.

Problem 775 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{A, B\} \\ F &= \{q\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, r, A] \\ [r, a, \lambda, s, B] \\ [s, b, B, s, \lambda] \\ [s, b, A, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Draw a state-transition graph of a finite automaton M_1 that accepts L . If such automaton does not exist, prove it.

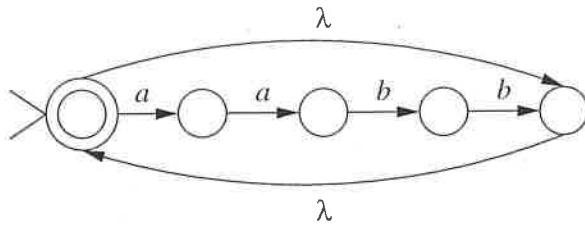


Figure 324:

Answer: See Figure 324.

- (b) Let \mathcal{S} be a class of languages over alphabet $\{a, b\}$, defined as follows:

Language L is a member of \mathcal{S} if and only if L is accepted by some finite automaton, but there does not exist a pushdown automaton that accepts L .

What is the cardinality of \mathcal{S} ? Explain your answer briefly.

Answer:

$$|\mathcal{S}| = 0, \text{ since } \mathcal{S} = \emptyset$$

Class \mathcal{S} is empty, since every language accepted by a finite automaton is also accepted by some pushdown automaton. In fact, this pushdown automaton is obtained by an algorithmic conversion of the original finite automaton.

- Problem 776** Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, i, F)$$

where:

$$\begin{aligned} Q &= \{i, q, r, s, t\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A, B, D\} \\ F &= \{t\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[i, \lambda, \lambda, q, D] \\ &[q, \lambda, D, t, \lambda] \\ &[q, a, \lambda, r, AA] \\ &[r, a, \lambda, s, BB] \\ &[s, b, B, s, \lambda] \\ &[s, c, A, s, \lambda] \\ &[s, c, D, q, D] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

- (a) Draw a state-transition graph of a finite automaton M_1 that accepts L . If such an automaton does not exist, prove it.

Answer: M accepts:

$$(aabccc)^*$$

whence the automaton represented of Figure 325.

- (b) Let \mathcal{T} be a set of strings over alphabet $\{a, b, c, d\}$, defined as follows:

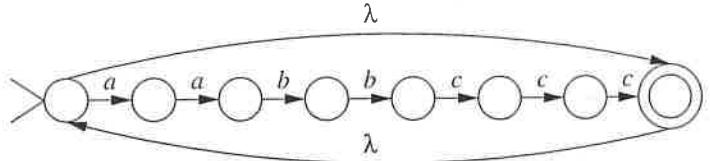


Figure 325:

String w is a member of \mathcal{T} if and only if w is accepted by the pushdown automaton M , and also $|w| > 6$.

What is the cardinality of \mathcal{T} ? Explain your answer briefly.

Answer:

$$|\mathcal{T}| = \aleph_0$$

Set \mathcal{T} is infinite and countable. To see that \mathcal{T} is infinite, observe that it contains all the strings of the infinite language L , except possibly only finitely many of them whose length does not exceed 6. (Question: Which?) To see that \mathcal{T} is countable, recall that the entire set of strings $\{a, b, c\}^*$ is countable.

- Problem 777** Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, i, F)$$

where:

$$\begin{aligned} Q &= \{i, q, r, s, t\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, B, D\} \\ F &= \{t\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[i, \lambda, \lambda, q, D] \\ &[q, d, D, t, \lambda] \\ &[q, c, \lambda, r, ABA] \\ &[r, d, \lambda, s, BAB] \\ &[s, a, A, s, \lambda] \\ &[s, b, B, s, \lambda] \\ &[s, c, D, q, D] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

- (a) Draw a state-transition graph of a finite automaton M_1 that accepts L . If such an automaton does not exist, prove it.

Answer: The automaton M accepts the language represented by the regular expression:

$$(cd\ bar{bababa}\ c)^*d$$

whence the automaton given on Figure 326.

- (b) Let \mathcal{T} be a set of strings over alphabet $\{a, b, c, d\}$, defined as follows:

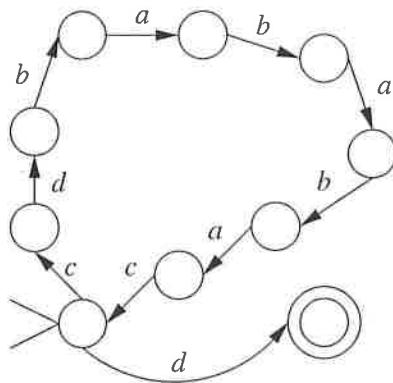


Figure 326:

String w is a member of \mathcal{T} if and only if w is accepted by the pushdown automaton M , and also $|w| < 4$.

What is the cardinality of \mathcal{T} ? Explain your answer briefly.

Answer:

$$|\mathcal{T}| = 1$$

By the answer to part (a), L contains only one string of length less than 4, precisely the string d .

Problem 778 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s, t\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A, B\} \\ F &= \{t\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, a, \lambda, q, A] \\ &[r, b, A, r, \lambda] \\ &[q, \lambda, \lambda, r, \lambda] \\ &[r, \lambda, \lambda, s, \lambda] \\ &[s, \lambda, \lambda, t, \lambda] \\ &[s, c, \lambda, s, B] \\ &[t, a, B, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, prove it.

Answer: M accepts the language:

$$L = \{a^m b^m c^k a^k \mid m, k \geq 0\}$$

whence the grammar: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b, c\}$, $V = \{S, L, R\}$,

and the production set P is:

$$\begin{aligned} S &\rightarrow LR \\ L &\rightarrow aLb \mid \lambda \\ R &\rightarrow cRa \mid \lambda \end{aligned}$$

(b) Let \mathcal{S} be a class of languages over alphabet $\{a, b, c\}$, defined as follows:

Language L is a member of \mathcal{S} if and only if L is accepted by some pushdown automaton, but there does not exist a context-free grammar that generates L .

What is the cardinality of \mathcal{S} ? Explain your answer briefly.

Answer:

$$|\mathcal{S}| = 0, \text{ since } \mathcal{S} = \emptyset$$

Class \mathcal{S} is empty, since every language accepted by a pushdown automaton is also generated by some context-free grammar. In fact, this context-free grammar is obtained by an algorithmic conversion of the original pushdown automaton.

Problem 779 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s, t, v\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, B, D\} \\ F &= \{v\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, a, \lambda, q, A] \\ &[r, b, A, r, \lambda] \\ &[s, c, \lambda, s, B] \\ &[t, a, B, t, \lambda] \\ &[v, d, \lambda, v, D] \\ &[v, d, D, v, \lambda] \\ &[q, \lambda, \lambda, r, \lambda] \\ &[r, \lambda, \lambda, s, \lambda] \\ &[s, \lambda, \lambda, t, \lambda] \\ &[t, \lambda, \lambda, v, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, prove it.

Answer: The template for words accepted by M is:

$$a^n b^n c^m a^m (dd)^k$$

whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and P comprises:

$$\begin{aligned} S &\rightarrow ABD \\ A &\rightarrow aAb \mid \lambda \\ B &\rightarrow cBa \mid \lambda \\ D &\rightarrow DD \mid \lambda \mid dd \end{aligned}$$

- (b) Let \mathcal{T} be a set of strings over alphabet $\{a, b, c\}$, defined as follows:

String w is a member of \mathcal{T} if and only if w is accepted by the pushdown automaton M and the length of w is greater than 6.

What is the cardinality of \mathcal{T} ? Explain your answer briefly.

Answer:

$$|\mathcal{T}| = \aleph_0$$

Set \mathcal{T} is infinite and countable. To see that \mathcal{T} is infinite, observe that it contains all the strings of the infinite language L , except possibly only finitely many of them whose length does not exceed 6. (Question: Which?) To see that \mathcal{T} is countable, recall that the entire set of strings $\{a, b, c\}^*$ is countable.

- Problem 780** Let L_3 be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, p, F)$$

where:

$$\begin{aligned} Q &= \{p, q, r, s\} \\ \Sigma &= \{0, 1\} \\ \Gamma &= \{A, T\} \\ F &= \{s\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [p, \lambda, \lambda, q, T] \\ [q, 1, T, s, \lambda] \\ [q, 1, \lambda, r, AAA] \\ [r, 0, A, r, \lambda] \\ [r, \lambda, T, q, T] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

- (a) Write a complete formal definition of a context-free grammar G that generates L_3 . If such grammar does not exist, prove it.

Answer: M accepts $(1000)^*1$, whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{0, 1\}$, $V = \{S, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow B1 \\ B &\rightarrow BB \mid \lambda \mid 1000 \end{aligned}$$

- (b) Write a complete formal definition of a context-free grammar G that generates L_3^* . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, A)$, where $\Sigma = \{0, 1\}$, $V = \{A, S, B\}$, and the production set P is:

$$\begin{aligned} A &\rightarrow AA \mid \lambda \mid S \\ S &\rightarrow B1 \\ B &\rightarrow BB \mid \lambda \mid 1000 \end{aligned}$$

- (c) Is $\overline{L_3}$ (the complement of L_3) a context-free language? Prove your answer.

Answer: Yes—the regular expression for L_3 is given in the answer to part (a); the complement of every regular language is regular, and thereby context-free.

- Problem 781** Let L_4 be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{A\} \\ F &= \{r, s\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, \lambda, \lambda, r, A] \\ [q, \lambda, \lambda, s, A] \\ [r, a, \lambda, r, \lambda] \\ [r, b, A, r, \lambda] \\ [s, a, A, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

- (a) Write a regular expression that defines L_4 . If such regular expression does not exist, prove it.

Answer:

$$a \cup a^*ba^*$$

- (b) Draw a state-transition graph of a finite automaton M that accepts L_4 . If such automaton does not exist, prove it.

Answer: See Figure 327.

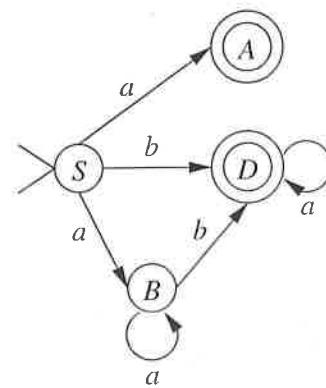


Figure 327:

- (c) Write a complete formal definition of a regular context-free grammar G that generates L_4 . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aA \mid bD \mid aB \\ A &\rightarrow \lambda \\ B &\rightarrow aB \mid bD \\ D &\rightarrow aD \mid \lambda \end{aligned}$$

Problem 782 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, s, t, v\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, Z\} \\ F &= \{v\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, d, \lambda, t, Z] \\ [t, a, \lambda, t, AA] \\ [t, \lambda, \lambda, s, \lambda] \\ [s, b, A, s, \lambda] \\ [s, c, Z, v, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer: Such regular expression does not exist, since:

$$L = \{da^n b^{2n} c \mid n \geq 0\}$$

which is not a regular language.

To prove that L is not regular, assume the opposite, that L is regular. Let k be the constant as in the Pumping Lemma for L . Let $m > k$; then $da^m b^{2m} c \in L$. In any “pumping” decomposition such that $da^m b^{2m} c = uvx$, we have: $|uv| \leq k < m$. Hence, the non-empty “pumping” substring v is contained entirely within the portion: da^m , and two cases are possible:

(1) v consists entirely of a 's, that is: $v = a^\ell$.

By the pumping, every word of the form $uv^i x$, $i \geq 0$, belongs to L . However, such a word is of the form:

$$w_1 = da^{m+(i-1)\ell} b^{2m} c$$

Since $m + (i-1)\ell > m$ whenever $i > 1$, word w_1 has fewer b 's than twice the number of a 's if $i > 1$. We conclude that $w_1 \notin L$, which is a contradiction.

(2) v contains the leading d , that is: $v = da^\ell$, where $\ell \geq 0$.

In this case, the pumping produces multiple occurrences of the letter d , which is impossible, since all strings of L have only one d .

(b) Is L context-free? Explain your answer briefly.

Answer: Yes--a pushdown automaton that accepts it is defined in part (a).

Problem 783 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A, Z\} \\ F &= \{q\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, r, Z] \\ [r, a, \lambda, s, A] \\ [r, b, \lambda, s, AAA] \\ [s, c, A, s, \lambda] \\ [s, a, Z, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Advice for Answer: The regular expression for L :

$$(a(ac \cup bccc)a)^*$$

(b) Does every non-deterministic pushdown automaton have an equivalent deterministic pushdown automaton? Explain your answer briefly.

Answer: No—for some context free languages there exists a non-deterministic pushdown automaton that accepts the language, but there cannot exist any equivalent deterministic pushdown automaton. For example, every pushdown automaton that accepts the set of palindromes over $\{a, b\}$ is non-deterministic.

Problem 784 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A, B\} \\ F &= \{r\} \end{aligned}$$

and the transition set δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, A] \\ [q, b, \lambda, q, BB] \\ [q, \lambda, \lambda, r, \lambda] \\ [r, c, A, r, \lambda] \\ [r, d, B, r, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:
 $\Sigma = \{a, b, c\}$, $V = \{S\}$, and P is:

$$S \rightarrow aSc \mid bSdd \mid \lambda$$

(b) Does there exist an algorithm that solves the following problem:

INPUT: An arbitrary pushdown automaton \mathcal{M} .

QUESTION: Is the language accepted by \mathcal{M} regular?

Explain your answer briefly.

Answer: No—it is undecidable for an arbitrary pushdown automaton whether the language accepted by it is regular.

Problem 785 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, s, t, v\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, Z\} \\ F &= \{v\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, c, \lambda, t, ZZZ] \\ &[t, a, \lambda, t, A] \\ &[t, \lambda, \lambda, s, \lambda] \\ &[s, b, A, s, \lambda] \\ &[s, d, Z, s, \lambda] \\ &[s, \lambda, \lambda, v, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer: Such regular expression does not exist, since:

$$L = \{ca^n b^n ddd \mid n \geq 0\}$$

which is not a regular language.

To prove that L is not regular, assume the opposite, that L is regular. Let k be the constant as in the Pumping Lemma for L . Let $m > k$; then $ca^m b^m ddd \in L$. In any “pumping” decomposition such that $ca^m b^m ddd = uvx$, we have: $|uv| \leq k < m$. Hence, the non-empty “pumping” substring v is contained entirely within the portion: ca^m , and two cases are possible:

(1) v consists entirely of a 's, that is: $v = a^\ell$.

By the pumping, every word of the form $uv^i x$, $i \geq 0$, belongs to L . However, such a word is of the form:

$$w_1 = ca^{m+(i-1)\ell} b^m ddd$$

Since $m + (i-1)\ell > m$ whenever $i > 1$, word w_1 has fewer b 's than a 's if $i > 1$. We conclude that $w_1 \notin L$, which is a contradiction.

(2) v contains the leading c , that is: $v = ca^\ell$, where $\ell \geq 0$.

In this case, the pumping produces multiple occurrences of the letter c , which is impossible, since all strings of L have only one c .

(b) Is L^* context-free? Explain your answer briefly.

Answer: Yes. L itself is context-free, since it is accepted by the pushdown automaton M . The class of context-free languages is closed under Kleene star—hence, L^* is also context-free.

Problem 786 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, s, t, v\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, Z\} \\ F &= \{v\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, c, \lambda, t, ZZZ] \\ &[t, a, \lambda, s, AA] \\ &[s, b, A, s, \lambda] \\ &[s, b, Z, t, Z] \\ &[s, d, Z, v, \lambda] \\ &[v, d, Z, v, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$c (abbb)^* abb ddd$$

(b) Is L^* regular? Explain your answer briefly.

Answer: Yes. L itself is regular, as witnessed by the regular expression constructed in the answer to part (a). The class of regular languages is closed under Kleene star—hence, L^* is also regular.

Problem 787 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, Z\} \\ F &= \{q\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, a, \lambda, r, Z] \\ &[r, d, \lambda, s, AA] \\ &[r, b, \lambda, s, AAA] \\ &[r, \lambda, \lambda, s, \lambda] \\ &[s, c, A, s, \lambda] \\ &[s, a, Z, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:
 $\Sigma = \{a, b, c, d\}$, $V = \{S, A\}$, and P is:

$$\begin{aligned} S &\rightarrow SS \mid \lambda \mid aAa \\ A &\rightarrow dc \mid bcc \mid \lambda \end{aligned}$$

(b) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 328.

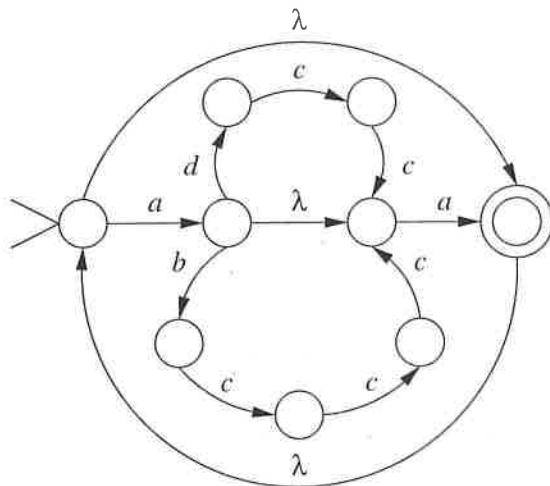


Figure 328:

(c) Describe an algorithm that solves the following problem:

INPUT: An arbitrary pushdown automaton M .

OUTPUT: A finite automaton M_1 that accepts the language accepted by M .

Explain your answer briefly. If such algorithm does not exist, explain why.

Answer: Such algorithm does not exist. Some context-free languages are not regular—in fact, it is undecidable for an arbitrary pushdown automaton whether the language accepted by it is regular.

Problem 788 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, B\} \\ F &= \{r\} \end{aligned}$$

and the transition set δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, AA] \\ [q, b, \lambda, q, BBBB] \\ [q, \lambda, \lambda, r, \lambda] \\ [r, c, A, r, \lambda] \\ [r, d, B, r, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b, c, d\}$, $V = \{S\}$,

and the production set P is:

$$S \rightarrow aScc \mid bSdd \mid \lambda$$

(b) Describe an algorithm that solves the following problem:

INPUT: An arbitrary pushdown automaton M .

OUTPUT: A pushdown automaton C that accepts the complement of the language accepted by M .

Explain your answer briefly. If such algorithm does not exist, explain why.

Answer: Such algorithm does not exist. Some context-free languages do not have a context-free complement—in fact, it is undecidable for an arbitrary pushdown automaton whether complement of the language accepted by it is context-free.

Problem 789 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, s, t, v\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, Z\} \\ F &= \{v\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, d, \lambda, t, ZZ] \\ [t, a, \lambda, t, AAA] \\ [t, \lambda, \lambda, s, \lambda] \\ [s, b, A, s, \lambda] \\ [s, \lambda, \lambda, v, \lambda] \\ [v, c, Z, v, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: Observe that:

$$L = \{da^k b^{3k} cc \mid k \geq 0\}$$

whence the grammar:

$G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$,

$V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow dAcc \\ A &\rightarrow aAbbb \mid \lambda \end{aligned}$$

(b) Is L decidable? Explain your answer.

Answer: Yes. Every context-free language is decidable.

Problem 790 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, s, t, v\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, Z\} \\ F &= \{q\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, d, \lambda, t, Z] \\ &[t, a, \lambda, s, AAA] \\ &[s, b, A, s, \lambda] \\ &[s, \lambda, \lambda, v, \lambda] \\ &[v, c, Z, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$(dabbbc)^*$$

(b) Is \bar{L} (the complement of L) recursively enumerable? Explain your answer.

Answer: Yes. The complement of a regular language is regular, so \bar{L} is regular. Every regular language is recursively enumerable (in fact recursive), so \bar{L} is recursively enumerable.

Problem 791 (a) Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, Z, V\} \\ F &= \{q\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, a, \lambda, r, Z] \\ &[q, b, \lambda, r, V] \\ &[r, a, \lambda, s, A] \\ &[r, b, \lambda, s, AA] \\ &[r, c, \lambda, s, AAA] \\ &[s, d, A, s, \lambda] \\ &[s, d, Z, q, \lambda] \\ &[s, c, V, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(a(ad \cup bdd \cup cddd)d \cup b(ad \cup bdd \cup cddd)c)^*$$

(b) Let \mathcal{C} be the class of languages that can be accepted by a pushdown automaton, and let \mathcal{N} be the class of languages that cannot be accepted by a pushdown automaton. State the cardinalities of \mathcal{C} and \mathcal{N} , and compare them.

Answer: Class \mathcal{C} is infinite and countable:

$$|\mathcal{C}| = \aleph_0$$

Class \mathcal{N} is infinite and uncountable; its cardinality is equal to the cardinality of the set of subsets of an infinite countable set:

$$|\mathcal{N}| = 2^{\aleph_0}$$

Hence:

$$|\mathcal{N}| > |\mathcal{C}|$$

Problem 792 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{A, Z, V\} \\ F &= \{q\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, a, \lambda, r, Z] \\ &[q, b, \lambda, r, V] \\ &[r, a, \lambda, s, A] \\ &[r, a, \lambda, s, AA] \\ &[r, a, \lambda, s, AAA] \\ &[s, b, A, s, \lambda] \\ &[s, b, Z, q, \lambda] \\ &[s, a, V, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 329.

(b) Is the complement of L context-free? Prove your answer.

Answer: Yes. L itself is regular—the finite automaton that accepts it is given in the answer to part (a). This guarantees that \bar{L} is also regular, since the complement of every regular language is regular. However, every regular language is also context-free—hence, \bar{L} is context-free.

Problem 793 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

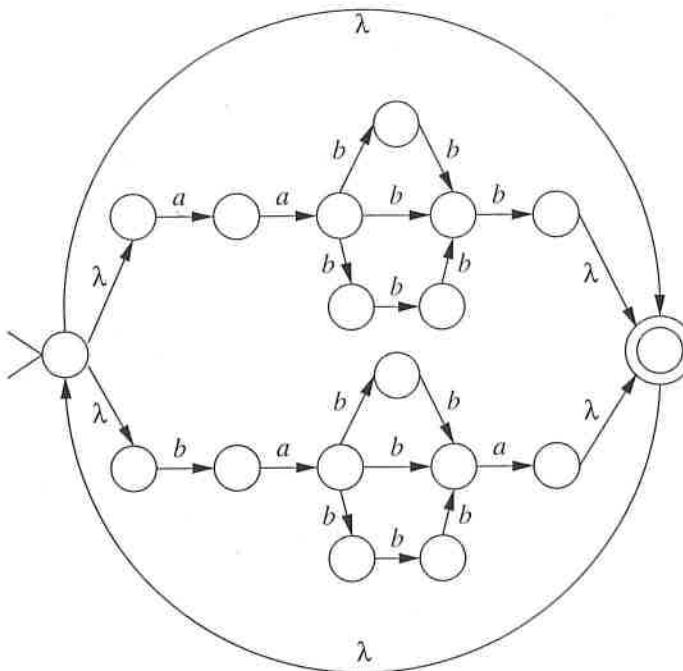


Figure 329:

where:

$$\begin{aligned} Q &= \{q, s, t, v\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, Z\} \\ F &= \{v\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, d, \lambda, t, ZZZ] \\ &[t, a, \lambda, t, AA] \\ &[t, \lambda, \lambda, s, \lambda] \\ &[s, b, A, s, \lambda] \\ &[s, \lambda, \lambda, v, \lambda] \\ &[v, c, Z, v, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)(a) Write 5 distinct strings that belong to L . If such strings do not exist, state it and explain why.**Answer:**

dccc
dabbccc
daabbbbccc
daaabbbbbbccc
daaaabbbbbbbccc

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.**Answer:** The general template for strings in L is:

$$da^n b^{2n} ccc, \text{ where } n \geq 0$$

which is generated by the grammar:
 $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A\}$, and P is:

$$\begin{aligned} S &\rightarrow dAcc \\ A &\rightarrow aAbb \mid \lambda \end{aligned}$$

(c) Is L decidable? Explain your answer.

Answer: Yes. In fact, there exists an algorithm to convert every push-down automaton to a Turing Machine that decides the same language.

(d) Is L context-free? Explain your answer.Answer: Yes—the context-free grammar for L is given in the answer to part (b).(e) Is \bar{L} (the complement of L) decidable? Explain your answer.Answer: Yes. The Turing Machine obtained by a conversion described in the answer to part (c) decides L and \bar{L} simultaneously.**Problem 794** Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, s, t, v\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{B, Z\} \\ F &= \{v\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[q, a, \lambda, t, ZZ] \\ &[t, b, \lambda, t, BB] \\ &[t, c, \lambda, s, \lambda] \\ &[s, c, B, s, \lambda] \\ &[s, \lambda, \lambda, v, \lambda] \\ &[v, d, Z, v, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)(a) Write 5 distinct strings that belong to L . If such strings do not exist, state it and explain why.**Answer:**

acdd
abcccd
abbccccdd
abbbccccccdd
abbbbccccccccdd

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.**Answer:** The general template for strings in L is:

$$ab^n c^{2n+1} dd, \text{ where } n \geq 0$$

which is generated by the grammar:

 $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$,
 $V = \{S, A\}$, and P is:

$$\begin{aligned} S &\rightarrow aAdd \\ A &\rightarrow bAcc \mid c \end{aligned}$$

(c) Is L recursively enumerable? Explain your answer.

Answer: Yes. As is explained in the answer to part (e), L is decidable, and every decidable language is recursively enumerable.

(d) Is L infinite? Explain your answer.

Answer: Yes. For every natural number n , string $ab^n c^{2n+1} dd$ belongs to L .

(e) Does there exist a Turing Machine that accepts L and halts on every input? Explain your answer.

Answer: Yes. In fact, there exists an algorithm to convert every push-down automaton to a Turing Machine that decides the same language.

Problem 795 Let L be the language accepted by the pushdown automaton

$$M = (Q, \Sigma, \Gamma, \delta, q_0, F)$$

where:

$$\begin{aligned} Q &= \{q_0, q_1, q_2, q_3\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A\} \\ F &= \{q_3\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} \delta(q_0, a, \lambda) &= \{(q_0, A)\} \\ \delta(q_0, c, \lambda) &= \{(q_1, \lambda)\} \\ \delta(q_1, c, \lambda) &= \{(q_2, \lambda)\} \\ \delta(q_2, c, \lambda) &= \{(q_3, \lambda)\} \\ \delta(q_3, b, A) &= \{(q_3, \lambda)\} \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Write 5 distinct strings that belong to L . If such strings do not exist, prove it.

(b) Write 5 distinct strings over alphabet Σ that do not belong to L . If such strings do not exist, prove it.

(c) Write a complete formal definition or a state-transition graph of a finite automaton M' that accepts L . If such automaton does not exist, prove it.

Advice for Answer:

$$L = \{a^n c c c b^n \mid n \geq 0\}$$

Problem 796 Let L be the language accepted by the pushdown automaton

$$M = (Q, \Sigma, \Gamma, \delta, q_0, F)$$

where:

$$\begin{aligned} Q &= \{q_0, q_1, q_2\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{A\} \\ F &= \{q_2\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} \delta(q_0, a, \lambda) &= \{(q_0, A)\} \\ \delta(q_0, b, A) &= \{(q_1, \lambda)\} \\ \delta(q_1, b, \lambda) &= \{(q_2, \lambda)\} \\ \delta(q_2, b, A) &= \{(q_1, \lambda)\} \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Write 5 distinct strings that belong to L . If such strings do not exist, prove it.

(b) Write 5 distinct strings over alphabet Σ that do not belong to L . If such strings do not exist, prove it.

(c) Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, prove it.

Advice for Answer:

$$L = \{a^n b^{2n} \mid n \geq 1\}$$

Problem 797 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q_0, F)$$

where:

$$\begin{aligned} Q &= \{q_0, q_1, q_2, q_3\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{A, a, b\} \\ F &= \{q_0, q_3\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} \delta(q_0, \varepsilon, \varepsilon) &= \{(q_1, A)\} \\ \delta(q_1, a, \varepsilon) &= \{(q_1, a)\} \\ \delta(q_1, b, a) &= \{(q_2, \varepsilon)\} \\ \delta(q_2, b, a) &= \{(q_2, \varepsilon)\} \\ \delta(q_2, \varepsilon, A) &= \{(q_3, \varepsilon)\} \end{aligned}$$

Write a regular expression that defines L . If such expression does not exist, prove it.

Advice for Answer:

$$L = \{a^n b^n \mid n \geq 0\}$$

Problem 798 Let L be the language accepted by the pushdown automaton

$$M = (Q, \Sigma, \Gamma, \delta, q_0, F)$$

where:

$$\begin{aligned} Q &= \{q_0, q_1\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{A\} \\ F &= \{q_0\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} \delta(q_0, a, \lambda) &= \{(q_1, A)\} \\ \delta(q_1, b, A) &= \{(q_0, \lambda)\} \end{aligned}$$

Write a complete formal definition or a state-transition graph of a finite automaton M that accepts L . If such automaton does not exist, prove it.

Advice for Answer: L is defined by the regular expression:

$$(ab)^*$$

Problem 799 (a) Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{A\} \\ F &= \{r\} \end{aligned}$$

and the transition set δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, A] \\ [q, \lambda, \lambda, r, \lambda] \\ [r, b, A, r, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, prove it.

Advice for Answer:

$$L = \{a^n b^n \mid n \geq 0\}$$

(b) Let L_1 be a language over alphabet $\{a, b\}$ defined as follows:

$$L_1 = \{a^m b^n \mid m \neq n, m \geq 0, n \geq 0\}$$

Write a complete formal definition of a context-free grammar G_1 that generates L_1 . If such grammar does not exist, prove it.

Advice for Answer:

$$L_1 = L_{<} \cup L_{>}$$

where:

$$\begin{aligned} L_{<} &= \{a^m b^n \mid m < n, m \geq 0, n \geq 0\} \\ &= \{a^m b^{m+p} \mid m \geq 0, p > 0\} \\ L_{>} &= \{a^m b^n \mid m > n, m \geq 0, n \geq 0\} \\ &= \{a^{m+p} b^m \mid m \geq 0, p > 0\} \end{aligned}$$

(c) Is \overline{L} recursive (decidable)? Explain your answer.

Answer: Yes— L is decidable because it is context-free, and \overline{L} is decidable because L is decidable.

Problem 800 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s, t\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{A\} \\ F &= \{t\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, r, A] \\ [r, a, A, s, \lambda] \\ [s, a, \lambda, q, A] \\ [t, b, A, t, \lambda] \\ [q, \lambda, \lambda, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Advice for Answer:

$$L = \{a^{3n} b^n \mid n \geq 0\}$$

Problem 801 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s, t\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{A\} \\ F &= \{t\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, r, A] \\ [r, a, \lambda, q, \lambda] \\ [t, b, \lambda, s, A] \\ [s, b, \lambda, t, \lambda] \\ [q, \lambda, \lambda, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

λ

6 Turing Machines & General Algorithms

Problem 802 Consider the Turing machine $M = (Q, \Sigma, \Gamma, \delta, q, F)$ such that: $Q = \{q, p, v, z, x\}$; $\Sigma = \{0, 1\}$; $\Gamma = \{B, 0, 1, N\}$; $F = \{x\}$; and δ is defined by the following transition set:

$$\begin{array}{lll} [q, 0, p, N, R] & [p, 0, p, 0, R] & [v, 1, v, 1, L] \\ [q, 1, q, 1, R] & [p, 1, p, 1, R] & [v, 0, x, 0, R] \\ [q, B, q, B, R] & [p, B, v, B, L] & [v, N, z, 0, R] \end{array}$$

(M has an one-way infinite tape (infinite to the right only.) B is the designated blank symbol. M accepts by final state.)

(a) List all pairs of state and symbol such that M halts if M is in that state with that symbol under the head. Prove your answer. (Employ the format suggested below, and fill in the required rows of the table.)

Answer:

state	symbol
q	N
p	N
v	B
z	0
z	1
z	B
z	N
x	0
x	1
x	B
x	N

Proof: Listed are exactly those pairs (s, α) such that δ does not contain a tuple $[s, \alpha, , ,]$.

(b) Assume that M is in the configuration

$$11N11011p$$

Write the next configuration of M , and prove your answer. If the next configuration does not exist, state it and prove your answer.

Answer:

$$11N1101v1$$

since tuple $[p, B, v, B, L]$ is executed.

(c) Assume that M is in the configuration

$$11N11v011$$

Write the next configuration of M , and prove your answer. If the next configuration does not exist, state it and prove your answer.

Answer:

$$11N110x11$$

since tuple $[v, 0, x, 0, R]$ is executed.

(d) Assume that M is in the configuration

$$11N1101010p11$$

Does M reject in this configuration? Prove your answer.

Answer: No. M cannot reject because it does not halt in this configuration, but it executes tuple $[p, 1, p, 1, R]$.

Problem 803 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$\begin{aligned} Q &= \{q, r, s, t, v\}; \\ \Sigma &= \{a, b, c\}; \end{aligned}$$

$\Gamma = \{B, a, b, c\}$;
and δ is defined by the following transition set:

$$\begin{aligned} &[q, a, r, b, R] \\ &[q, b, r, a, R] \\ &[q, c, t, c, R] \\ &[t, a, t, a, R] \\ &[t, b, t, b, R] \\ &[t, B, s, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

(a) Does M halt on input $abba$? If your answer is “yes”, write the configuration in which M halts. If your answer is “no”, write the configuration of M after it makes exactly 9 moves.

Answer: Yes; M halts in the configuration:

$$brbba$$

(b) Does M halt on input $abcbbcb$? If your answer is “yes”, write the configuration in which M halts. If your answer is “no”, write the configuration of M after it makes exactly 9 moves.

Answer: Yes; M halts in the configuration:

$$brbcbcb$$

(c) Does M halt on input cba ? If your answer is “yes”, write the configuration in which M halts. If your answer is “no”, write the configuration of M after it makes exactly 9 moves.

Answer: Yes; M halts in the configuration:

$$cbabBs$$

Problem 804 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$\begin{aligned} Q &= \{q, r, s, t, v\}; \\ \Sigma &= \{a, b\}; \\ \Gamma &= \{B, a, b, \Psi\}; \\ F &= \{t\}; \end{aligned}$$

and δ is defined by the following transition set:

$$\begin{aligned} &[q, a, q, a, R] \\ &[q, b, q, \Psi, R] \\ &[q, B, r, B, L] \end{aligned}$$

$$\begin{aligned} &[r, a, r, a, L] \\ &[r, \Psi, s, \Psi, L] \end{aligned}$$

$$\begin{aligned} &[s, a, s, a, L] \\ &[s, \Psi, t, \Psi, L] \end{aligned}$$

$$\begin{aligned} &[t, a, t, a, L] \\ &[t, \Psi, v, \Psi, R] \end{aligned}$$

$$\begin{aligned} &[v, a, v, a, R] \\ &[v, \Psi, v, \Psi, R] \\ &[v, B, v, B, R] \end{aligned}$$

(B is the designated blank symbol. M has an one-way infinite tape and accepts by final state.)

(a) Write a regular expression that defines the set of strings on which M diverges. If such regular expression does not exist, prove it.

Answer:

$$a^*ba^*ba^*b(a \cup b)^*$$

(b) Write a regular expression that defines the set of strings on which M halts and accepts. If such regular expression does not exist, prove it.

Answer:

$$\emptyset$$

(c) Write a regular expression that defines the set of strings on which M halts and rejects. If such regular expression does not exist, prove it.

Answer:

$$\emptyset$$

(d) Write a regular expression that defines the set of strings on which M terminates abnormally (attempts to move the head to the left of the leftmost cell.) If such regular expression does not exist, prove it.

Answer:

$$a^*(b \cup \lambda)a^*(b \cup \lambda)a^*$$

Problem 805 Consider the Turing machine

$M_8 = (Q, \Sigma, \Gamma, \delta, q, F)$ such that: $Q = \{q, p, t, s, x\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, a, b, c, A, D, Z\}$; $F = \{x\}$; and δ is defined by the following transition set:

$[q, a, p, A, R]$	$[p, a, p, a, R]$	$[t, c, s, c, L]$
$[q, b, p, P, R]$	$[p, b, p, b, R]$	$[s, a, s, a, L]$
$[q, c, p, K, R]$	$[p, c, p, c, R]$	$[s, b, s, b, L]$
	$[p, B, t, B, L]$	$[s, c, s, c, L]$
		$[s, A, x, A, R]$

(M has an one-way infinite tape (infinite to the right only). B is the designated blank symbol. M_8 accepts by final state.)

Let L_{8A} be the set of strings which M_8 accepts.

Let L_{8R} be the set of strings which M_8 rejects.

Let $L_{8\infty}$ be the set of strings on which M_8 diverges.

(a) Write a regular expression that defines L_{8A} . If such a regular expression does not exist, prove it.

Answer:

$$a(a \cup b \cup c)^*$$

(b) Write a regular expression that defines L_{8R} . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)(a \cup b \cup c)^*(a \cup b) \cup \\ (b \cup c)(a \cup b \cup c)^*c \cup \\ \lambda \cup a \cup b \cup c$$

(c) Write a regular expression that defines $L_{8\infty}$. If such a regular expression does not exist, prove it.

Answer:

$$\emptyset$$

(d) Write a regular expression that defines the language (if any) which is decided by M_8 . If such a regular expression does not exist, prove it.

Answer: M_8 halts on every input, and thus it decides the language which it accepts:

$$a(a \cup b \cup c)^*c$$

Problem 806 Consider the Turing machine $M = (Q, \Sigma, \Gamma, \delta, q)$ such that: $Q = \{q, p, s, x\}$; $\Sigma = \{a, b, c, d\}$; $\Gamma = \{B, a, b, c, d, A, D, Z\}$; and δ is defined by the following transition set:

$[q, a, p, A, R]$	$[p, a, p, a, R]$	$[s, a, x, a, R]$
$[q, b, p, D, R]$	$[p, b, p, b, R]$	$[s, b, p, b, R]$
$[q, c, p, D, R]$	$[p, c, s, Z, L]$	$[s, c, p, c, R]$
$[q, d, p, D, R]$	$[p, d, p, d, R]$	$[s, d, p, d, R]$
$[q, B, q, B, R]$	$[p, Z, p, Z, R]$	$[s, A, x, A, R]$
		$[p, B, p, B, R]$

(M has an one-way infinite tape (infinite to the right only). B is the designated blank symbol.)

Let L be the set of strings on which M halts.

Let L_∞ be the set of strings on which M diverges.

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Advice for Answer: M halts on those strings that contain as substring ac or cc , but also on those strings in which c is the second symbol. Observe that M promptly escapes on empty string and all strings of length equal to 1. Otherwise, it turns the first symbol to a sentinel and looks for c . Upon finding c , M turns c into a sentinel Z , and inspects a symbol to the left of this c —call this symbol α . If α is a (or a sentinel A), M promptly halts in state x . If α is another input letter (unconverted to sentinel), M proceeds to the right looking for the next c in state p . (Observe that it skips correctly over the sentinel Z just written.) Otherwise, M halts in state s having no eligible transition in s , because α is the sentinel D of the first symbol or the sentinel Z of another c . If M arrives at the end of input in state p without halting, then M escapes.

Answer:

$$(a \cup b \cup c \cup d)^* (a \cup c) \quad c \quad (a \cup b \cup c \cup d)^* \\ (a \cup b \cup c \cup d) \quad c \quad (a \cup b \cup c \cup d)^*$$

(b) Draw a state-transition graph of a finite-state automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 330.

(c) List four distinct strings that belong to L_∞ . If this is impossible, prove it.

Answer: λ, a, b, c

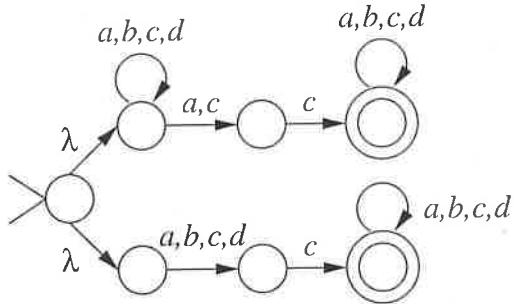


Figure 330:

(d) Is L_∞ decidable? Prove your answer.

Answer: Yes. L_∞ is decidable because all regular languages are decidable, and L_∞ is regular because the complement of every regular language is regular, and L_∞ is the complement of L , which is regular since it is defined by the regular expression given in the answer to part (a).

(e) Write a regular expression that defines the language which M decides. If such a regular expression does not exist, prove it. (M is the machine defined at the beginning of this problem.)

Answer: This regular expression does not exist, because M does not decide any language, since M diverges on strings in L_∞ , and L_∞ is not empty, as follows from the answer given in part (c).

Problem 807 Consider the Turing machine $M = (Q, \Sigma, \Gamma, \delta, q, F)$ such that: $Q = \{q, p, t, x\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, a, b, c, A, E, K\}$; $F = \{x\}$; and δ is defined by the following transition set:

$[q, a, p, A, R]$	$[p, a, p, a, R]$	$[t, a, q, a, R]$
$[q, b, p, E, R]$	$[p, b, p, b, R]$	$[t, b, q, b, R]$
$[q, c, p, K, R]$	$[p, c, p, c, R]$	$[t, c, q, c, R]$
$[q, B, q, B, R]$	$[p, B, t, B, L]$	$[t, K, x, K, R]$

(M has an one-way infinite tape (infinite to the right only.) B is the designated blank symbol. M accepts by final state.)

Let L_A be the set of strings which M accepts.

Let L_R be the set of strings which M rejects.

Let L_∞ be the set of strings on which M diverges.

(a) Write a regular expression that defines L_A . If such a regular expression does not exist, prove it.

Advice for Answer: M escapes on empty string promptly. Otherwise, M turns the first symbol into sentinel and skips to the end of the input. Then, M looks at the last symbol and escapes if it is an input letter (unconverted to sentinel), which says that M diverges on all strings longer than 1 (and halts only on strings of length equal to 1.) If this single letter is c , then M enters its final state x . Otherwise M halts in (a non-final state) t .

Answer: c

(b) Write a regular expression that defines L_R . If such a regular expression does not exist, prove it.

Answer: $a \cup b$

(c) Write a regular expression that defines L_∞ . If such a regular expression does not exist, prove it.

Answer:

$\lambda \cup (a \cup b \cup c) (a \cup b \cup c) (a \cup b \cup c)^*$

(e) Explain how to construct an algorithm that solves the following problem:

INPUT: Pushdown automaton P_1 that accepts a language L_1 , and a Turing Machine T_2 that accepts a language L_2 ;

OUTPUT: **yes** if $L_1 \cap L_2 = \emptyset$;

no otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist. If it existed, we could fix L_1 to be any non-empty (context free) language, and this algorithm would then decide the set of Turing Machines whose languages have a nontrivial property:

has non-empty intersection with L_1 .

This property is nontrivial because it assumes different values for two regular (and thereby recursively enumerable) languages: it is true for Σ^* (since $L_1 \cap \Sigma^* = L_1 \neq \emptyset$) but it is false for \emptyset (since $L_1 \cap \emptyset = \emptyset$.) By Rice's Theorem, such an algorithm is impossible.

Problem 808 Consider the Turing machine $M = (Q, \Sigma, \Gamma, \delta, q, F)$ such that: $Q = \{q, p, t, s, x\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, a, b, c, A, E, K\}$; $F = \{x\}$; and δ is defined by the following transition set:

$[q, a, t, A, R]$	$[p, a, t, a, R]$	$[t, a, p, a, R]$
$[q, b, p, E, R]$	$[p, b, p, b, R]$	$[t, b, t, b, R]$
$[q, c, p, K, R]$	$[p, c, p, c, R]$	$[t, c, t, c, R]$
$[q, B, q, B, R]$	$[p, B, p, B, R]$	$[t, B, s, B, L]$
$[s, a, s, a, L]$		
$[s, b, s, b, L]$		
$[s, c, s, c, L]$		
$[s, E, x, E, R]$		

(M has an one-way infinite tape (infinite to the right only.) B is the designated blank symbol. M accepts by final state.)

Let L_A be the set of strings which M accepts.

Let L_R be the set of strings which M rejects.

Let L_∞ be the set of strings on which M diverges.

(a) Write a regular expression that defines L_∞ . If such a regular expression does not exist, prove it.

Advice for Answer: M scans the input to count a 's by toggling between states p (even) and t (odd.) M diverges if it finds an even number of a 's. If M finds an odd

number of a 's, then it returns to inspect the sentinel of the first symbol. If the first symbol was b then M accepts, otherwise it rejects.

Answer:

$$(a(b \cup c)^* a \cup b \cup c)^*$$

(b) Write a regular expression that defines L_A . If such a regular expression does not exist, prove it.

Answer:

$$b(a(b \cup c)^* a \cup b \cup c)^* a(b \cup c)^*$$

(c) Write a regular expression that defines L_R . If such a regular expression does not exist, prove it.

Answer:

$$\begin{aligned} &c(a(b \cup c)^* a \cup b \cup c)^* a(b \cup c)^* \\ &\quad \cup \\ &a(a(b \cup c)^* a \cup b \cup c)^* \end{aligned}$$

(d) Explain how to construct an algorithm that solves the following problem:

INPUT: Pushdown automaton P_1 that accepts a language L_1 , a Turing Machine T_2 that decides a language L_2 , and a string w ;

OUTPUT: yes if $w \in L_1 \cap L_2$;
no otherwise.

If this algorithm does not exist, prove it.

Answer: Simulate P_1 and T_2 on input w —observe that both machines must halt. Return yes if both P_1 and T_2 return yes, and return no in all other cases.

Problem 809 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$Q = \{q, r, s, t, v, x\};$$

$$\Sigma = \{a, b, c\};$$

$$\Gamma = \{B, a, b, c\};$$

$$F = \{t\};$$

and δ is defined by the following transition set:

$$\begin{aligned} &[q, a, q, a, R] \\ &[q, b, q, b, R] \\ &[q, c, v, b, R] \\ &[q, B, r, B, L] \end{aligned}$$

$$\begin{aligned} &[r, a, s, B, L] \\ &[r, b, s, B, L] \\ &[r, c, s, B, L] \end{aligned}$$

$$\begin{aligned} &[s, a, x, a, L] \\ &[s, b, t, b, L] \\ &[s, c, x, c, L] \end{aligned}$$

$$\begin{aligned} &[v, B, v, B, R] \\ &[v, a, v, B, R] \\ &[v, b, v, B, R] \\ &[v, c, v, B, R] \end{aligned}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) M accepts by final state. B is the designated blank symbol.)

(a) Write a regular expression that defines the set of strings on which M halts and accepts. If such a regular expression does not exist, explain why.

Answer:

$$(a \cup b)^*(a \cup b) b (a \cup b)$$

(b) Write a regular expression that defines the set of strings on which M halts and rejects. If such a regular expression does not exist, explain why.

Answer:

$$(a \cup b)^*(a \cup b) a (a \cup b)$$

(c) Write a regular expression that defines the set of strings on which M diverges. If such a regular expression does not exist, explain why.

Answer:

$$(a \cup b \cup c)^* c (a \cup b \cup c)^*$$

(d) Write a regular expression that defines the set of strings on which M terminates abnormally (attempts to move the head to the left of the leftmost cell.) If such a regular expression does not exist, explain why.

Answer:

$$(a \cup b \cup \lambda)(a \cup b \cup \lambda)$$

Note that M diverges whenever the input string contains at least one c . Otherwise, M attempts to step over the left end if the length of the input string is less than 3. Finally, if M gets to halt, then it accepts if the next-to-last symbol is b , and rejects if the next-to-last symbol is a .

Problem 810 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$
 such that:

$$Q = \{q, p, x, t\}; \Sigma = \{a, b, c\}; \Gamma = \{B, a, b, c\};$$

and δ is defined by the following transition set:

$$\begin{aligned} &[q, a, p, a, R] \\ &[q, b, p, b, R] \\ &[q, c, p, c, R] \end{aligned}$$

$$\begin{aligned} &[t, a, p, a, R] \\ &[t, b, p, b, R] \\ &[t, c, p, c, R] \end{aligned}$$

$$\begin{aligned} &[p, a, t, a, R] \\ &[p, b, t, b, R] \\ &[p, c, t, c, R] \\ &[p, B, x, B, L] \end{aligned}$$

$$\begin{aligned} &[x, b, x, b, R] \\ &[x, c, x, c, R] \\ &[x, B, x, B, R] \end{aligned}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.)
 B is the designated blank symbol.)

Let L_h be the set of strings on which M halts. Let L_d be the set of strings on which M diverges.

(a) Write a regular expression that defines L_h . If such a regular expression does not exist, explain why.

Answer: $((a \cup b \cup c)(a \cup b \cup c))^* (a \cup \lambda)$

(To verify the answer, observe that M halts whenever the length of the input string is even. When the input length is odd, M looks at the last input symbol—if the last symbol is b or c , M diverges; otherwise it halts.)

(b) Write a regular expression that defines L_d . If such a regular expression does not exist, explain why.

Answer: $((a \cup b \cup c)(a \cup b \cup c))^* (b \cup c)$

(c) Is L_d a recursive (decidable) language? Explain your answer.

Answer: Yes— L_d is a regular language, as is demonstrated by the regular expression given in the answer to part (b). Every regular language is recursive.

Problem 811 Consider the Turing machine:

$M = (Q, \Sigma, \Gamma, \delta, q, F)$ such that: $Q = \{q, p, r, t\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, a, b, c\}$; $F = \{t\}$; and δ is defined by the following transition set:

$[q, a, q, a, R]$
 $[q, b, p, b, R]$
 $[q, c, r, c, R]$

$[p, a, p, a, R]$
 $[p, b, p, b, R]$
 $[p, c, r, c, R]$
 $[p, B, t, B, L]$

$[r, a, r, a, R]$
 $[r, b, r, b, R]$
 $[r, c, r, c, R]$
 $[r, B, r, B, R]$

$[t, a, t, a, R]$
 $[t, b, r, b, R]$

$[q, a, r, a, L]$
 $[q, b, q, b, R]$
 $[q, c, q, c, R]$

$[r, b, t, b, R]$
 $[r, c, s, c, R]$

$[s, B, s, B, R]$
 $[s, a, s, a, R]$
 $[s, b, s, b, R]$
 $[s, c, s, c, R]$

$[t, B, v, B, L]$
 $[t, a, t, a, R]$
 $[t, b, t, b, R]$
 $[t, c, t, c, R]$

$[v, a, x, a, L]$
 $[v, b, y, b, R]$
 $[v, c, y, c, R]$

$[x, c, z, c, R]$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.)

M accepts by final state. B is the designated blank symbol.)

Let L_a be the set of strings accepted by M .

Let L_r be the set of strings rejected by M .

Let L_d be the set of strings on which M diverges.

(a) Write a regular expression that defines L_a . If such a regular expression does not exist, explain why.

Answer: $a^*b (a \cup b)^* a$

(To verify the answer, observe that M halts and rejects unless it finds b or c in the input. If c is found in the input, M diverges. If the input does not contain any c 's but contains at least one b , M looks at the last letter—if the last letter is b , M diverges; if the last letter is a , M halts at the first blank to the left of the input.)

(b) Write a regular expression that defines L_r . If such a regular expression does not exist, explain why.

Answer: a^*

(c) Write a regular expression that defines L_d . If such a regular expression does not exist, explain why.

Answer: $(a \cup b)^* c (a \cup b \cup c)^* \cup (a \cup b)^* b$

Problem 812 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$Q = \{q, r, s, t, v, x, y, z\};$$

$$\Sigma = \{a, b, c\};$$

$$\Gamma = \{B, a, b, c\};$$

$$F = \{z\};$$

and δ is defined by the following transition set:

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.)

M accepts by final state. B is the designated blank symbol.)

Let L_a be the set of strings accepted by M .

Let L_r be the set of strings rejected by M .

Let L_d be the set of strings on which M diverges.

(a) Write a regular expression that defines L_a . If such a regular expression does not exist, explain why.

Answer: $a^*b (a \cup b)^* a$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) M accepts by final state. B is the designated blank symbol.)

(a) Write a regular expression that defines the set of strings on which M halts and accepts. If such a regular expression does not exist, explain why.

Answer: Observe that M attempts to step out of the leftmost cell if the input string begins with a . If no a is found, M rejects. Otherwise, M looks at the symbol to the left of the first a . If this symbol is c , M diverges. If this symbol is b , M looks at the end of the input

string—if the input ends with ca , M accepts; otherwise it rejects.

Hence, the set of strings on which M halts and accepts is represented by the following regular expression.

$$(b \cup c)^*ba(a \cup b \cup c)^*ca$$

(b) Write a regular expression that defines the set of strings on which M diverges. If such a regular expression does not exist, explain why.

Answer:

$$(b \cup c)^*ca(a \cup b \cup c)^*$$

(c) Write a regular expression that defines the set of strings on which M terminates abnormally (attempts to move the head to the left of the leftmost cell.) If such a regular expression does not exist, explain why.

Answer:

$$a(a \cup b \cup c)^*$$

Problem 813 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$Q = \{q, r, s, t, v\};$$

$$\Sigma = \{a, b, c\};$$

$$\Gamma = \{B, a, b, c\};$$

and δ is defined by the following transition set:

$$\begin{aligned} &[q, a, q, a, R] \\ &[q, b, q, b, R] \\ &[q, c, q, c, R] \\ &[q, B, r, B, L] \end{aligned}$$

$$\begin{aligned} &[r, a, v, a, L] \\ &[r, b, s, b, R] \\ &[r, c, t, c, R] \end{aligned}$$

$$\begin{aligned} &[v, a, v, a, L] \\ &[v, b, v, b, L] \\ &[v, c, v, c, L] \end{aligned}$$

$$[s, B, s, B, R]$$

(B is the designated blank symbol. M has an one-way infinite tape.)

(a) Write a regular expression that defines the set of strings on which M diverges (does not halt).

If such regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)^*b$$

(b) Write a regular expression that defines the set of strings on which M halts. If such regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)^*c$$

(c) Write a regular expression that defines the set of strings on which M terminates abnormally (attempts to move the head to the left of the leftmost cell.) If such regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)^*a \cup \lambda$$

Problem 814 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$Q = \{q, r, s, t, v\};$$

$$\Sigma = \{a, b, c\};$$

$$\Gamma = \{B, a, b, c\};$$

and δ is defined by the following transition set:

$$\begin{aligned} &[q, a, q, a, R] \\ &[q, b, q, b, R] \\ &[q, c, s, c, R] \\ &[q, B, r, B, L] \end{aligned}$$

$$\begin{aligned} &[r, a, s, a, R] \\ &[r, b, t, b, L] \\ &[t, b, v, b, L] \end{aligned}$$

$$\begin{aligned} &[v, a, v, a, L] \\ &[v, b, v, b, L] \\ &[v, c, v, c, L] \end{aligned}$$

$$\begin{aligned} &[s, a, s, a, R] \\ &[s, b, s, b, R] \\ &[s, c, s, c, R] \\ &[s, B, s, B, R] \end{aligned}$$

(B is the designated blank symbol. M has an one-way infinite tape.)

(a) Write a regular expression that defines the set of strings on which M diverges (does not halt).

If such regular expression does not exist, prove it.

Answer:

$$(a \cup b)^*c(a \cup b \cup c)^* \cup (a \cup b \cup c)^*a$$

(b) Write a regular expression that defines the set of strings on which M halts.

If such regular expression does not exist, prove it.

Answer:

$$(a \cup b)^*ab$$

(c) Write a regular expression that defines the set of strings on which M terminates abnormally (attempts to move the head to the left of the leftmost cell.)

If such regular expression does not exist, prove it.

Answer:

$$\lambda \cup b \cup (a \cup b)^*bb$$

Problem 815 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$Q = \{q, r, s, t, v, x\};$$

$$\Sigma = \{a, b, c\};$$

$$\Gamma = \{B, a, b, c, \Psi, \Delta\};$$

$$F = \{x\};$$

and δ is defined by the following transition set:

$$[q, a, q, \Delta, R]$$

$$[q, b, q, \Psi, R]$$

$$[q, c, v, \Psi, R]$$

$$[q, B, r, B, L]$$

$$[r, \Delta, t, B, R]$$

$$[r, \Psi, s, \Psi, L]$$

$$[s, \Delta, s, B, L]$$

$$[s, \Psi, s, B, L]$$

$$[t, B, t, B, R]$$

$$[t, \Psi, t, \Psi, R]$$

$$[v, B, x, B, R]$$

such that:

$$Q = \{q, q_1, q_2, q_3, r, t, v\};$$

$$\Sigma = \{a, b, c\};$$

$$\Gamma = \{B, a, b, c\};$$

$$F = \{t\};$$

and δ is defined by the following transition set:

$$[q, a, q_1, a, R]$$

$$[q, b, q_1, b, R]$$

$$[q, c, v, c, R]$$

$$[q_1, a, q_2, a, R]$$

$$[q_1, b, q_2, b, R]$$

$$[q_1, c, v, c, R]$$

$$[q_2, a, q_3, a, R]$$

$$[q_2, b, q_3, b, R]$$

$$[q_2, c, v, c, R]$$

$$[q_3, B, r, B, L]$$

$$[r, a, r, a, L]$$

$$[r, b, t, b, R]$$

(where B is the designated blank symbol. M has an one-way infinite tape and accepts by final state.)

(a) Write a regular expression that defines the set of strings on which M halts and accepts. If such regular expression does not exist, prove it.

Answer:

$$(a \cup b)^* c$$

(b) Write a regular expression that defines the set of strings on which M halts and rejects. If such regular expression does not exist, prove it.

Answer:

$$(a \cup b)^* c(a \cup b \cup c)(a \cup b \cup c)^*$$

(c) Write a regular expression that defines the set of strings on which M diverges. If such regular expression does not exist, prove it.

Answer:

$$(a \cup b)^* a$$

(d) Write a regular expression that defines the set of strings on which M terminates abnormally (attempts to move the head to the left of the leftmost cell.) If such regular expression does not exist, prove it.

Answer:

$$\lambda \cup (a \cup b)^* b$$

Problem 816 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) M accepts by final state. B is the designated blank symbol.)

(a) Write a regular expression that defines the set of strings on which M halts and accepts. If such a regular expression does not exist, prove it.

Answer: Observe that M halts in a non-accepting state on all strings whose length is less than 3, as well as on all strings that contain at least one c . Next, if the length of the input string is at least equal to 3 and it contains no c 's, then M halts if at least one b is found in the input—otherwise, it hits the left end. Finally, M never diverges.

Hence, M accepts the set of strings over $\{a, b\}$ of length not less than 3 that contain at least one b :

$$\begin{aligned} & b(a \cup b)(a \cup b)(a \cup b)^* \\ & \quad \cup \\ & ab(a \cup b)(a \cup b)^* \\ & \quad \cup \\ & aaa^*b(a \cup b)^* \end{aligned}$$

(b) Write a regular expression that defines the set of strings on which M halts and rejects. If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b)^* c(a \cup b \cup c)^* \cup (a \cup b \cup \lambda)(a \cup b \cup \lambda)^*$$

(c) Write a regular expression that defines the set of strings on which M diverges. If such a regular expression does not exist, prove it.

Answer: \emptyset

(d) Write a regular expression that defines the set of strings on which M terminates abnormally (attempts to move the head to the left of the leftmost cell.) If such a regular expression does not exist, prove it.

Answer: $aaaa^*$

Problem 817 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$Q = \{q, r, s, t, v, x\}; \Sigma = \{a, b, c\};$$

$$\Gamma = \{B, a, b, c\}; F = \{t\};$$

and δ is defined by the following transition set:

$$[q, a, q, a, R]$$

$$[q, b, q, b, R]$$

$$[q, c, q, c, R]$$

$$[q, B, r, B, L]$$

$$[r, a, s, a, L]$$

$$[r, b, s, b, L]$$

$$[r, c, s, c, L]$$

$$[s, a, t, a, R]$$

$$[s, b, v, b, R]$$

$$[s, c, x, c, R]$$

$$[x, a, x, B, R]$$

$$[x, b, x, B, R]$$

$$[x, c, x, B, R]$$

$$[x, B, x, B, R]$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) M accepts by final state. B is the designated blank symbol.)

(a) Write a regular expression that defines the set of strings on which M halts and accepts. If such a regular expression does not exist, prove it.

Answer: The operation of M can be summarized as follows. M crashes into the left-side wall just when the input string has fewer than two symbols. Otherwise, M sets its course of action according to the penultimate (next to last) symbol. If the penultimate symbol is a , M accepts:

$$(a \cup b \cup c)^* a (a \cup b \cup c)$$

(b) Write a regular expression that defines the set of strings on which M halts and rejects. If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)^* b (a \cup b \cup c)$$

(c) Write a regular expression that defines the set of strings on which M diverges. If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)^* c (a \cup b \cup c)$$

(d) Write a regular expression that defines the set of strings on which M terminates abnormally (attempts to move the head to the left of the leftmost cell.) If such a regular expression does not exist, prove it.

Answer:

$$\lambda \cup a \cup b \cup c$$

Problem 818 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$Q = \{q, r, s, t, v\}; \Sigma = \{a, b, c\}; \Gamma = \{B, a, b, c\};$$

and δ is defined by the following transition set:

$$[q, a, q, a, R]$$

$$[q, b, q, b, R]$$

$$[q, c, q, c, R]$$

$$[q, B, t, B, L]$$

$$[t, a, r, a, L]$$

$$[t, b, v, b, R]$$

$$[t, c, v, c, R]$$

$$[r, a, v, a, R]$$

$$[r, b, v, b, R]$$

$$[r, c, s, c, R]$$

$$[s, a, s, a, R]$$

$$[s, b, s, b, R]$$

$$[s, c, s, c, R]$$

$$[s, B, s, B, R]$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.)

Let L be the set of strings on which the Turing machine M diverges.

Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)^* ca$$

Problem 819 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$Q = \{q, r, t\}; \Sigma = \{a, b, c\}; \Gamma = \{B, a, b, c\};$$

and δ is defined by the following transition set:

$$[q, a, q, a, R]$$

$$[q, b, t, b, R]$$

$$[q, c, q, c, R]$$

$$[q, B, q, B, R]$$

$$[t, a, r, a, R]$$

$$[t, b, q, b, R]$$

$$[t, c, r, c, R]$$

$$[t, B, q, B, R]$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.)

Let L be the set of strings on which the Turing machine M halts.

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: By inspection of the code for M , we conclude that it accepts the language represented by the following regular expression:

$$(a \cup bb \cup c)^* b (a \cup c) (a \cup b \cup c)^*$$

whence the required grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AbBD \\ B &\rightarrow a \mid c \\ A &\rightarrow \lambda \mid AA \mid a \mid bb \mid c \\ D &\rightarrow \lambda \mid AA \mid a \mid b \mid c \end{aligned}$$

Problem 820 Consider the Turing machine $M = (Q, \Sigma, \Gamma, \delta, q)$ such that: $Q = \{q, x, y, z, t, v, w, r\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, a, b, c, Z\}$; and δ is defined by the following transition set:

$$\begin{array}{lll} [q, a, x, Z, R] & [r, a, r, a, R] \\ [q, b, y, Z, R] & [r, b, r, b, R] \\ [q, c, z, Z, R] & [r, c, r, c, R] \\ & [r, B, r, B, R] \\ \\ [x, a, x, a, R] & [y, a, y, a, R] & [z, a, z, a, R] \\ [x, b, x, b, R] & [y, b, y, b, R] & [z, b, z, b, R] \\ [x, c, x, c, R] & [y, c, y, c, R] & [z, c, z, c, R] \\ [x, B, t, B, L] & [y, B, v, B, L] & [z, B, w, B, L] \\ \\ [t, b, r, b, R] & [v, a, r, a, R] & [w, a, r, a, R] \\ [t, c, r, c, R] & [v, c, r, c, R] & [w, b, r, b, R] \end{array}$$

(M has an one-way infinite tape (infinite to the right only.) B is the designated blank symbol.)

Let L be the set of strings on which M halts.

Let L_D be the set of strings on which M diverges.

(a) Write a regular expression that defines L_D . If such a regular expression does not exist, prove it.

Answer:

$$\begin{aligned} a (a \cup b \cup c)^* (b \cup c) \\ \cup \\ b (a \cup b \cup c)^* (a \cup c) \\ \cup \\ c (a \cup b \cup c)^* (a \cup b) \end{aligned}$$

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$\begin{aligned} \lambda \cup (a \cup b \cup c) \\ \cup \\ a (a \cup b \cup c)^* a \\ \cup \\ b (a \cup b \cup c)^* b \\ \cup \\ c (a \cup b \cup c)^* c \end{aligned}$$

(c) Write a regular expression that defines the language which the Turing Machine M decides. If such a regular expression does not exist, prove it.

Answer: This regular expression does not exist, because M does not decide any language: it diverges on some inputs strings (exactly those in L_D .)

(d) Is L decidable? Prove your answer.

Answer: Yes. L is regular (as is shown in the answer to part (b)) and all regular languages are decidable.

(e) Is L_D decidable? Prove your answer.

Answer: Yes. L_D is regular (as is shown in the answer to part (a)) and all regular languages are decidable.

Problem 821 Consider the Turing machine

$M = (Q, \Sigma, \Gamma, \delta, q)$ such that: $Q = \{q, r, x, s, t, f, g\}$; $\Sigma = \{a, b, c, d\}$; $\Gamma = \{B, a, b, c, d, A, D\}$; $F = \{f\}$; and δ is defined by the following transition set:

$$\begin{array}{lll} [q, a, x, A, R] & [r, a, r, a, R] & [x, a, x, a, R] \\ [q, b, x, A, R] & [r, b, r, b, R] & [x, b, x, b, R] \\ [q, c, x, D, R] & [r, c, r, c, R] & [x, c, x, c, R] \\ [q, d, x, D, R] & [r, d, r, d, R] & [x, d, x, d, R] \\ [q, B, r, B, R] & [r, B, r, B, R] & [x, B, s, B, L] \\ \\ [s, c, t, c, L] & [t, a, t, a, L] & [t, A, f, A, R] \\ [s, A, r, A, R] & [t, b, t, b, L] & [t, D, g, D, R] \\ [s, D, r, D, R] & [t, c, t, c, L] & \\ & [t, d, t, d, L] & \end{array}$$

(M has an one-way infinite tape (infinite to the right only.) B is the designated blank symbol. M accepts by final state.)

Let L_A be the set of strings which M accepts.

Let L_R be the set of strings which M rejects.

Let L_D be the set of strings on which M diverges.

(a) Write a regular expression that defines L_A . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b) (a \cup b \cup c \cup d)^* c$$

(b) Write a regular expression that defines L_R . If such a regular expression does not exist, prove it.

Answer:

$$\begin{aligned} (a \cup b \cup c \cup d) (a \cup b \cup c \cup d)^* (a \cup b \cup d) \\ \cup \\ (c \cup d) (a \cup b \cup c \cup d)^* c \end{aligned}$$

(c) Write a regular expression that defines L_D . If such a regular expression does not exist, prove it.

Answer:

$$\lambda \cup a \cup b \cup c \cup d$$

- (d) Explain how to construct an algorithm that solves the following problem:

INPUT: Turing Machine T that accepts a language L_T .QUESTION: Is $L_T = L_R$?

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist. If it existed, it would decide the set of Turing machines whose languages have a nontrivial property "equal to L_R ", which is impossible by Rice's theorem. The property is nontrivial, since L_R has it, while $\overline{L_R}$, for instance, does not have it.

Problem 822 Consider the Turing machine $M = (Q, \Sigma, \Gamma, \delta, q)$ such that: $Q = \{q, x, s, f, g\}$; $\Sigma = \{a, b, c, d\}$; $\Gamma = \{B, a, b, c, d, T, D\}$; $F = \{f\}$; and δ is defined by the following transition set:

$[q, a, q, a, R]$	$[x, a, x, a, R]$	$[s, a, s, a, L]$
$[q, b, q, b, R]$	$[x, b, x, b, R]$	$[s, b, s, b, L]$
$[q, c, q, c, R]$	$[x, c, x, c, R]$	$[s, c, s, c, L]$
$[q, d, x, D, R]$	$[x, d, x, T, R]$	$[s, d, s, d, L]$
$[q, B, q, B, R]$	$[x, B, s, B, L]$	$[s, T, f, T, R]$
		$[s, D, g, D, R]$

(M has an one-way infinite tape (infinite to the right only.) B is the designated blank symbol. M accepts by final state.)

Let L_A be the set of strings which M accepts.Let L_R be the set of strings which M rejects.Let L_D be the set of strings on which M diverges.

- (a) Write a regular expression that defines L_A . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)^* d (a \cup b \cup c)^* d (a \cup b \cup c \cup d)^*$$

- (b) Write a regular expression that defines L_R . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)^* d (a \cup b \cup c)^*$$

- (c) Write a regular expression that defines L_D . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)^*$$

- (d) Explain how to construct an algorithm that solves the following problem:

INPUT: String $w_0 \in \Sigma^*$ QUESTION: Is $w_0 \in L_D$?

If this algorithm does not exist, prove it.

Answer: Convert the regular expression for L_D , given in the answer to part (c), into a finite automaton, convert this automaton into a deterministic one, and decide as it does.

Problem 823 Consider the Turing machine $M = (Q, \Sigma, \Gamma, \delta, s, F)$ such that: $Q = \{s, p, x\}$; $\Sigma = \{0, 1\}$; $\Gamma = \{B, 0, 1\}$; $F = \{s\}$; and δ is defined by the following transition set:

$[s, 0, s, 0, R]$	$[p, 0, p, 0, R]$	$[x, 0, x, 0, R]$
$[s, 1, p, 1, R]$	$[p, 1, x, 1, R]$	$[x, 1, s, 1, R]$
$[s, B, s, B, R]$		

(M has an one-way infinite tape (infinite to the right only.) B is the designated blank symbol. M accepts by final state.)

Let L be the set of strings which M accepts.Let L_R be the set of strings which M rejects.Let L_∞ be the set of strings on which M diverges.

- (a) List four distinct strings that belong to L_∞ . If this is impossible, state it and explain why it is so.

Advice for Answer: M cannot halt in state s , so it does not accept any strings. If the number of 1's is divisible by 3, M diverges, otherwise it halts in one of the two non-final states.

Answer:

$$\lambda, 0, 001110, 010101$$

- (b) List four distinct strings that belong to L_R . If this is impossible, state it and explain why it is so.

Answer:

$$1, 01, 10, 11$$

- (c) List four distinct strings that belong to L . If this is impossible, state it and explain why it is so.

Answer: Impossible— L is empty and there are no strings to list.

- (d) Write a regular expression that defines L_∞ . If such a regular expression does not exist, prove it.

Answer:

$$(0 \cup 10^* 10^* 1)^*$$

- (e) Write a regular expression that defines L_R . If such a regular expression does not exist, prove it.

Answer:

$$(0 \cup 10^* 10^* 1)^* (1 \cup 10^* 1) 0^*$$

- (f) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$\emptyset$$

- (g) Define the language (if any) which M decides. Prove your answer. (M is the machine defined at the beginning of this problem.)

Answer: M does not decide any language, since it diverges on some input strings—exactly those in L_∞ .

(h) Explain how to construct an algorithm that solves the following problem:

INPUT: A Turing machine T that accepts a language L_T .

OUTPUT: yes if $L \subseteq L_T$;

no otherwise.

(L is the language defined at the beginning of this problem.)

If this algorithm does not exist, prove it.

Answer: This algorithm always returns answer yes, since $L = \emptyset$ (as is stated in the answer to part (f)), and thus L is a subset of every set (including L_T .)

Problem 824 Consider the following Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$Q = \{q, r, s, p, v, t, x, y\};$$

$$\Sigma = \{0, 1\};$$

$$\Gamma = \{B, 0, 1\};$$

$$F = \{t\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, 0, r, 0, R] \\ & [q, 1, s, 1, R] \\ & [r, 1, r, 1, R] \\ & [r, B, x, B, L] \\ & [s, 0, s, 0, R] \\ & [s, B, y, B, L] \\ & [x, 1, t, 1, L] \\ & [y, 0, t, 0, L] \\ & [t, 0, v, 0, R] \\ & [v, 0, v, B, R] \\ & [v, 1, v, B, R] \\ & [v, B, v, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

Let L be the set of string on which M diverges.

Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$10^*00 \cup 01$$

Problem 825 Consider the Turing machine $M = (Q, \Sigma, \Gamma, \delta, q, F)$ such that: $Q = \{q, p, v, x, z\}$; $\Sigma = \{a, b, c, d\}$; $\Gamma = \{a, b, c, d, B, M, Q\}$; $F = \{z\}$; and δ is defined by the following transition set:

$$\begin{array}{lll} [q, a, p, M, R] & [p, a, p, a, R] & [v, a, x, a, R] \\ [q, b, p, M, R] & [p, b, v, b, L] & [v, b, x, b, R] \\ [q, c, p, Q, R] & [p, c, p, c, R] & [v, c, z, c, R] \\ [q, d, p, M, R] & [p, d, p, d, R] & [v, d, x, d, R] \\ [q, B, q, B, R] & [p, B, p, B, R] & [v, Q, z, Q, R] \end{array}$$

(M has an one-way infinite tape (infinite to the right only.) B is the designated blank symbol. M accepts by final state.)

Let L_A be the set of string which M accepts.

Let L_R be the set of string which M rejects.

Let L_D be the set of string on which M diverges.

(a) Write a regular expression that defines L_D . If such a regular expression does not exist, prove it.

Answer:

$$\lambda \cup (a \cup b \cup c \cup d)(a \cup c \cup d)^*$$

(b) Write a regular expression that defines L_A . If such a regular expression does not exist, prove it.

Answer:

$$(b \cup \lambda)(a \cup c \cup d)^* cb (a \cup b \cup c \cup d)^*$$

(c) Write a regular expression that defines L_R . If such a regular expression does not exist, prove it.

Answer:

$$(b \cup (a \cup c \cup d)^* (a \cup d)) b (a \cup b \cup c \cup d)^*$$

Problem 826 Consider the Turing machine

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

$$\text{such that: } \Sigma = \{a, b, c\};$$

$$\Gamma = \{B, a, b, c, A\}; Q = \{q, p, t, v, x, y, s, z, m, e\};$$

$F = \{m\}$; and δ consists of the following transition set:

$$\begin{array}{lll} [q, a, p, A, R] & [t, a, t, a, R] & [s, a, s, a, L] \\ [q, b, e, A, R] & [t, b, t, b, R] & [s, b, z, b, L] \\ [q, c, p, A, R] & [t, c, t, c, R] & [s, c, s, c, L] \\ [q, B, e, B, R] & [t, B, v, B, L] & [z, a, m, a, L] \\ [p, a, t, A, R] & [v, a, x, a, L] & [z, b, z, b, L] \\ [p, b, e, A, R] & [x, c, y, c, L] & [z, c, z, c, L] \\ [p, c, t, A, R] & [y, b, s, b, L] & [e, B, e, B, R] \\ [p, B, e, B, R] & & \end{array}$$

Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) M accepts by final state. B is the designated blank symbol.

Let L be the set of strings that the Turing machine M accepts. Let L^∞ be the set of strings on which the Turing machine M diverges.

(a) List four distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: See the answer to part (b).

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup c)(a \cup c)(a \cup b \cup c)^* a (b \cup c)^* b (a \cup c)^* bca$$

(c) List four distinct strings that belong to L^∞ . If this is impossible, state it and explain why.

Advice for Answer: See the answer to part (d).

(d) Write a regular expression that defines L^∞ . If such a regular expression does not exist, prove it.

Answer:

$$\lambda \cup a \cup b \cup c \cup ab \cup cb$$

(e) Is L recursively enumerable? Explain your answer.**Answer:** Yes, by definition, since L is accepted by Turing Machine M .

Problem 827 Consider the Turing machine $M = (Q, \Sigma, \Gamma, \delta, q)$ such that: $\Sigma = \{0, 1\}$; $\Gamma = \{B, 0, 1, N\}$; $Q = \{q, p, t, v, x, y, s, m\}$; and δ consists of the following transition set:

$[q, 0, p, Z, R]$	$[t, 0, t, 0, R]$	$[s, 0, s, 0, R]$
$[q, 1, p, N, R]$	$[t, 1, t, 1, R]$	$[s, 1, s, 1, R]$
$[q, B, m, B, R]$	$[t, B, v, B, L]$	$[s, B, s, B, R]$
$[p, 0, t, 0, R]$	$[x, 0, x, 0, L]$	$[y, 0, y, 0, L]$
$[p, 1, t, 1, R]$	$[x, 1, x, 1, L]$	$[y, 1, y, 1, L]$
$[p, B, m, B, R]$	$[x, Z, s, 0, R]$	$[y, N, s, 1, R]$
$[v, 0, x, 0, L]$		
$[v, 1, y, 1, L]$		

Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.

Let L be the set of strings on which the Turing machine M halts. Let L^∞ be the set of strings on which the Turing machine M diverges.

(a) List four distinct strings that belong to L^∞ . If this is impossible, state it and explain why.

Advice for Answer: L^∞ is defined by the following regular expression.

$$0(0 \cup 1)^*0 \cup 1(0 \cup 1)^*1$$

(b) Draw a state-transition graph of a finite automaton M that accepts the language L^∞ . If such an automaton does not exist, state it and prove your answer.

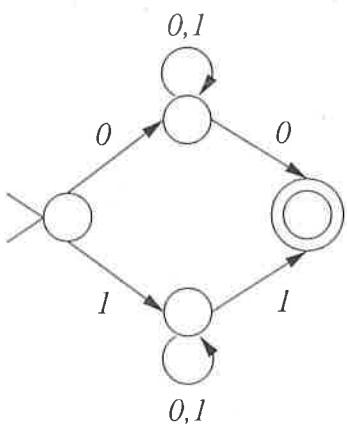
Answer: See Figure 331.

Figure 331:

(c) Explain how to construct an algorithm that solves the following problem:

INPUT: A string $w \in \Sigma^*$.**OUTPUT:** yes if $w \in L^\infty$; no otherwise.

If this algorithm does not exist, prove it.

Answer: Simulate the finite automaton given in the answer to part (b) and accept exactly when it accepts.(d) Is L^∞ decidable? Explain your answer.**Answer:** Yes—the algorithm given in the answer to part (c) decides L .**Problem 828** Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$Q = \{q, p, s, t, v, w, x\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, a, b, c\}$; and δ is defined by the following transition set:

$[q, a, q, a, R]$
$[q, b, p, b, R]$
$[q, c, q, c, R]$

$[p, a, p, a, R]$
$[p, b, s, b, R]$
$[p, c, p, c, R]$

$[s, a, s, a, R]$
$[s, b, t, b, L]$
$[s, c, s, c, R]$

$[t, a, v, a, L]$
$[t, b, v, b, L]$
$[t, c, x, c, L]$
$[t, B, v, B, L]$

$[x, a, x, a, R]$
$[x, b, x, b, R]$
$[x, c, w, c, R]$

$[w, a, w, a, R]$
$[w, b, w, b, R]$
$[w, c, w, c, R]$
$[w, B, w, B, R]$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.)

Let L be the set of strings on which the Turing machine M diverges.

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup c)^*b(a \cup c)^*b(a \cup c)^*cb(a \cup b \cup c)^*$$

(b) Describe the construction of a Turing Machine T which halts on every input and also accepts exactly the

strings that represent Turing Machines that accept L . In short:

$$(L(\tau) = L) \implies (T(\tau) \downarrow \text{ and accept})$$

$$\text{otherwise} \implies (T(\tau) \downarrow \text{ and reject})$$

If such a Turing machine T does not exist, prove it.

Answer: This Turing machine does not exist. If it existed, it would decide the set of Turing machines whose languages have the nontrivial property:

is equal to L

To see that the property is nontrivial, recall that L has it, but \emptyset does not. By Rice's Theorem, such a machine is impossible.

Problem 829 Consider the following Turing machine: $M = (Q, \Sigma, \Gamma, \delta, q)$ such that:

$Q = \{q, p, s, t, v\}$; $\Sigma = \{0, 1\}$; $\Gamma = \{1, 0, B\}$; and δ is defined by the following transition set:

$[q, 0, q, 0, R]$	$[t, 0, v, 0, L]$
$[q, 1, p, 1, R]$	$[t, 1, s, 1, R]$
$[q, B, q, B, R]$	$[v, 0, v, 0, L]$
$[p, 0, p, 0, R]$	$[v, 1, s, 1, R]$
$[p, 1, t, 1, L]$	
$[p, B, p, B, R]$	$[s, 0, s, 0, R]$
	$[s, 1, s, 1, R]$
	$[s, B, s, B, R]$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right only.) B is the designated blank symbol.)

Let L be the set of string on which M halts.

(a) List 6 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer: Impossible, because M does not halt on any input string.

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

\emptyset

Advice for Answer: Straightforwardly, M diverges unless it finds the second occurrence of 1. If it finds the second occurrence of 1, then it skips over all the 0's (if any) to the left of this second occurrence of 1 until it reaches (as it must) the first occurrence of 1, at which point it escapes to diverge.

(c) Dangerous Professor has told her students to write a program that operates as follows:

INPUT: String w over $\{0, 1\}$.

OUTPUT: yes if w is an element of the set of exactly those strings on which the Turing Machine M (defined at the beginning of this problem) diverges;
no otherwise.

Explain the algorithm that should be employed by this program, or state that it does not exist and prove it.

Answer: Return yes.

Problem 830 Consider the Turing machine:

$M = (Q, \Sigma, \Gamma, \delta, q)$ such that:

$Q = \{q, p, t, r, s\}$; $\Sigma = \{a, b\}$; $\Gamma = \{B, a, b, 0, 1, 2, \Psi\}$; and δ is defined by the following transition set:

$$\begin{aligned} &[q, a, p, 1, R] \\ &[q, b, p, 1, R] \end{aligned}$$

$$\begin{aligned} &[p, a, t, 2, R] \\ &[p, b, t, 2, R] \\ &[p, B, r, B, L] \end{aligned}$$

$$\begin{aligned} &[t, a, q, 0, R] \\ &[t, b, q, 0, R] \\ &[t, B, r, B, L] \end{aligned}$$

$$\begin{aligned} &[r, 1, s, 1, R] \\ &[s, a, s, a, R] \\ &[s, b, s, b, R] \\ &[s, B, s, B, R] \end{aligned}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.)

Let L be the set of strings on which the Turing machine M diverges.

(a) List 4 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: M diverges exactly when the length of its input string gives the remainder of 1 after a division by 3.

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$((a \cup b)(a \cup b)(a \cup b))^* (a \cup b)$$

Problem 831 Consider the Turing machine:

$M = (Q, \Sigma, \Gamma, \delta, q)$ such that: $Q = \{q, p, t, r, s\}$;

$\Sigma = \{a, b\}$; $\Gamma = \{B, a, b, 0, 1, 2, \Psi\}$; and δ is defined by

the following transition set:

$$\begin{aligned} & [q, a, p, 1, R] \\ & [q, b, p, 1, R] \\ & [q, B, s, B, R] \end{aligned}$$

$$\begin{aligned} & [p, a, t, 2, R] \\ & [p, b, t, 2, R] \\ & [p, B, r, B, L] \end{aligned}$$

$$\begin{aligned} & [t, a, t, \Psi, R] \\ & [t, b, t, \Psi, R] \\ & [t, B, r, B, L] \end{aligned}$$

$$\begin{aligned} & [r, 1, s, 1, R] \\ & [r, \Psi, s, \Psi, R] \end{aligned}$$

$$\begin{aligned} & [s, a, s, a, R] \\ & [s, b, s, b, R] \\ & [s, B, s, B, R] \end{aligned}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.)

Let L be the set of strings on which the Turing machine M halts.

(a) List 4 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: M halts exactly when the length of its input string is equal to 2.

(b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 332.



Figure 332:

Problem 832 Consider the Turing machine:

$M = (Q, \Sigma, \Gamma, \delta, q, F)$ such that: $Q = \{q, p, t, r, s, x, y\}$; $\Sigma = \{a, b\}$; $\Gamma = \{B, a, b, 0, 1, 2, \Psi\}$; $F = \{y\}$; and δ is

defined by the following transition set:

$$\begin{aligned} & [q, a, p, 1, R] \\ & [q, b, q, 0, R] \\ & [q, B, s, B, R] \end{aligned}$$

$$\begin{aligned} & [p, a, t, 2, R] \\ & [p, b, p, 1, R] \end{aligned}$$

$$\begin{aligned} & [t, a, t, \Psi, R] \\ & [t, b, t, \Psi, R] \\ & [t, B, r, B, L] \end{aligned}$$

$$\begin{aligned} & [r, \Psi, x, B, L] \\ & [x, 2, y, B, L] \end{aligned}$$

$$\begin{aligned} & [s, a, s, a, R] \\ & [s, b, s, b, R] \\ & [s, B, s, B, R] \end{aligned}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) M accepts by final state. B is the designated blank symbol.)

Let L be the set of strings that the Turing machine M accepts.

(a) List 4 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: M accepts exactly those strings whose second occurrence of a is followed by exactly one letter. Precisely, these are the strings defined by the regular expression:

$$b^* a b^* a (a \cup b)$$

(b) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 333.

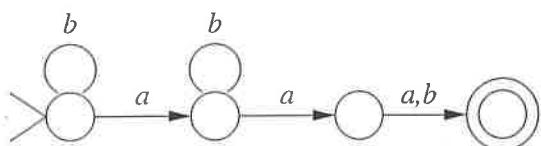


Figure 333:

Problem 833 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$Q = \{q, r, s, t, v\}; \quad \Sigma = \{a, b, c\}; \quad \Gamma = \{B, a, b, c\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, q, a, R] \\ & [q, b, q, b, R] \\ & [q, c, q, c, R] \\ & [q, B, t, B, L] \end{aligned}$$

$$\begin{aligned} & [t, a, r, a, L] \\ & [t, b, v, b, R] \\ & [t, c, v, c, R] \end{aligned}$$

$$\begin{aligned} & [r, a, v, a, R] \\ & [r, b, s, b, R] \\ & [r, c, v, c, R] \end{aligned}$$

$$\begin{aligned} & [s, a, s, a, R] \\ & [s, b, s, b, R] \\ & [s, c, s, c, R] \\ & [s, B, s, B, R] \end{aligned}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.)

Let L be the set of strings on which the Turing machine M diverges.

Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)^* ba$$

Problem 834 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$Q = \{q, s, t, v\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, a, b, c\}$; and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, q, a, R] \\ & [q, b, q, b, R] \\ & [q, c, q, c, R] \\ & [q, B, t, B, L] \end{aligned}$$

$$\begin{aligned} & [t, a, s, a, R] \\ & [t, b, t, b, L] \\ & [t, c, v, c, R] \end{aligned}$$

$$\begin{aligned} & [s, a, s, a, R] \\ & [s, b, s, b, R] \\ & [s, c, s, c, R] \\ & [s, B, s, B, R] \end{aligned}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.)

Let L be the set of input strings on which the Turing machine M diverges.

(a) List 5 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: See part (b).

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)^* a b^*$$

Problem 835 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$Q = \{q, r, s, t, v\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, a, b, c\}$; and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, q, a, R] \\ & [q, b, q, b, R] \\ & [q, c, q, c, R] \\ & [q, B, v, B, L] \end{aligned}$$

$$\begin{aligned} & [v, a, s, a, R] \\ & [v, b, r, b, L] \\ & [v, c, r, c, L] \end{aligned}$$

$$\begin{aligned} & [r, a, t, a, R] \\ & [r, b, s, b, R] \\ & [r, c, s, c, R] \end{aligned}$$

$$\begin{aligned} & [t, b, t, b, R] \\ & [t, c, t, c, R] \\ & [t, B, t, B, R] \end{aligned}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.)

Let L be the set of input strings on which the Turing machine M diverges.

(a) List 5 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: See part (b).

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)^* a (b \cup c)$$

Problem 836 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$Q = \{q, t\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, a, b, c\}$; and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, q, a, R] \\ & [q, b, t, b, R] \\ & [q, c, q, c, R] \end{aligned}$$

$$\begin{aligned} & [t, a, q, a, R] \\ & [t, b, t, b, R] \\ & [t, c, q, c, R] \\ & [t, B, t, B, R] \end{aligned}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.)

Let L be the set of strings on which the Turing machine M halts.

(a) List 5 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: L is given by the regular expression:

$$(a \cup b \cup c)^* (a \cup c)$$

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow \lambda \mid AA \mid a \mid b \mid c \\ B &\rightarrow a \mid c \end{aligned}$$

Problem 837 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$Q = \{q, t\}; \quad \Sigma = \{a, b, c\}; \quad \Gamma = \{B, a, b, c\};$$

and δ is defined by the following transition set:

$$\begin{aligned} [q, a, q, a, R] \\ [q, b, t, b, R] \\ [q, c, q, c, R] \\ [q, B, q, B, R] \end{aligned}$$

$$\begin{aligned} [t, a, q, a, R] \\ [t, b, t, b, R] \\ [t, c, q, c, R] \end{aligned}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.)

Let L be the set of strings on which the Turing machine M halts.

(a) List 5 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: L is given by the regular expression:

$$(a \cup b \cup c)^* b$$

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow \lambda \mid AA \mid a \mid b \mid c \\ B &\rightarrow b \end{aligned}$$

Problem 838 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$Q = \{q, r, t\}; \quad \Sigma = \{a, b, c\}; \quad \Gamma = \{B, a, b, c\};$$

and δ is defined by the following transition set:

$$\begin{aligned} [q, a, q, a, R] \\ [q, b, q, b, R] \\ [q, c, t, c, R] \\ [q, B, q, B, R] \end{aligned}$$

$$\begin{aligned} [t, a, r, a, R] \\ [t, b, q, b, R] \\ [t, c, r, c, R] \\ [t, B, q, B, R] \end{aligned}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.)

Let L be the set of strings on which the Turing machine M halts.

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$ is the set of terminals; $V = \{S, A, D\}$ is the set of variables; S is the start symbol, and the production set P is:

$$\begin{aligned} S &\rightarrow DcAD \\ A &\rightarrow a \mid c \\ D &\rightarrow \lambda \mid DD \mid a \mid b \mid c \end{aligned}$$

Problem 839 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$Q = \{q, s, v\};$$

$$\Sigma = \{a, b, c\};$$

$$\Gamma = \{B, a, b, c\};$$

and δ is defined by the following transition set:

$$\begin{aligned} [q, a, q, a, R] \\ [q, b, q, b, R] \\ [q, c, s, b, R] \end{aligned}$$

$$\begin{aligned} [s, a, s, a, R] \\ [s, b, s, b, R] \\ [s, c, v, b, R] \end{aligned}$$

$$\begin{aligned} [v, B, v, B, R] \\ [v, a, v, B, R] \\ [v, b, v, B, R] \\ [v, c, v, B, R] \end{aligned}$$

(B is the designated blank symbol.)

Write a regular expression that defines the set of input strings on which M halts. If such a regular expression does not exist, explain why.

Answer:

$$(a \cup b)^*(c \cup \lambda)(a \cup b)^*$$

Problem 840 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$Q = \{q, s, t, v, x\};$$

$$\Sigma = \{a, b, c\};$$

$$\Gamma = \{B, a, b, c\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, q, a, R] \\ & [q, b, q, b, R] \\ & [q, c, q, c, R] \\ & [q, B, s, B, L] \end{aligned}$$

$$\begin{aligned} & [s, a, s, B, L] \\ & [s, b, t, B, L] \\ & [s, c, x, b, R] \\ & [t, c, v, B, R] \end{aligned}$$

$$\begin{aligned} & [v, B, v, B, R] \\ & [v, a, v, B, R] \\ & [v, b, v, B, R] \\ & [v, c, v, B, R] \end{aligned}$$

M has an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.

Write a regular expression that defines the set of input strings on which M diverges. If such a regular expression does not exist, explain why.

Answer:

$$(a \cup b \cup c)^* c b a^*$$

Problem 841 Let L be the language accepted (by final state) by the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$Q = \{q, r, s, t, x, y\};$$

$$\Sigma = \{a, b\};$$

$$\Gamma = \{B, a, b\};$$

$$F = \{t\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, r, a, R] \\ & [q, b, s, b, R] \\ & [r, a, r, a, R] \\ & [r, b, r, b, R] \\ & [r, B, x, B, L] \\ & [s, a, s, a, R] \\ & [s, b, s, b, R] \\ & [s, B, y, B, L] \\ & [x, a, t, a, R] \\ & [y, b, t, b, R] \end{aligned}$$

(where B is the designated blank symbol.)

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$a(a \cup b)^* a \cup a \cup b(a \cup b)^* b \cup b$$

(b) Is L recursively enumerable? Explain your answer briefly.

Answer: Yes—the Turing machine M defined in part (a) accepts it.

Problem 842 Consider a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q_0)$$

such that:

$$\begin{aligned} Q &= \{q_0, q_1, q_2\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{a, b, c, B\} \end{aligned}$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q_0, a, q_1, a, R] \\ & [q_0, b, q_0, b, R] \\ & [q_0, c, q_0, c, R] \\ & [q_0, B, q_0, B, R] \\ & [q_1, a, q_2, a, R] \\ & [q_1, b, q_1, b, R] \\ & [q_1, c, q_1, c, R] \\ & [q_1, B, q_1, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

(a) Let L be the set of those strings over Σ on which the Turing machine M does not halt. Draw a state transition graph of a deterministic finite automaton M_1 that accepts L . If such finite automaton M_1 does not exist, prove it.

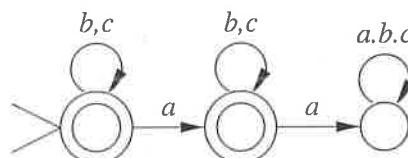
Answer: See Figure 334.

Figure 334:

(b) Is L a recursive language? Explain your answer briefly.

Answer: Yes—by the answer to part (a), L is regular. Every regular language is recursive, since a Turing machine can decide it by simulating a finite automaton that accepts it.

Problem 843 Consider a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$\begin{aligned} Q &= \{q, r, s, t, v\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{a, b, c, d, B\} \\ F &= \{v\} \end{aligned}$$

and δ is defined by the following transition set:

$$\begin{aligned} [q, a, q, a, R] \\ [q, b, q, b, R] \\ [q, c, q, c, R] \\ [q, d, r, d, R] \\ [q, B, q, B, R] \\ \\ [r, a, r, a, R] \\ [r, b, r, b, R] \\ [r, c, r, c, R] \\ [r, d, r, d, R] \\ [r, B, s, B, L] \\ \\ [s, a, t, a, L] \\ [t, b, v, b, R] \end{aligned}$$

(where B is the designated blank symbol.)

Let L_1 be the set of strings which M accepts, and let L_2 be the set of strings on which M does not halt.

(a) Write a regular expression that defines L_1 . If such regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)^* d (a \cup b \cup c \cup d)^* ba$$

(b) Write a regular expression that defines L_2 . If such regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)^*$$

(c) Is L_2 a recursive language? Explain your answer.

Answer: Yes—by the answer to part (b), L_2 is regular. Every regular language is recursive.

Problem 844 Let L be the language accepted (by final state) by the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$\begin{aligned} Q &= \{q, r, s, t, v, x\}; \\ \Sigma &= \{a, b\}; \\ \Gamma &= \{B, a, b\}; \\ F &= \{x\}; \end{aligned}$$

and δ is defined by the following transition set:

$$\begin{aligned} [q, a, r, a, R] \\ [q, b, r, b, R] \\ [r, a, s, a, R] \\ [r, b, s, b, R] \\ \\ [s, a, s, a, R] \\ [s, b, s, b, R] \\ [s, B, t, B, L] \\ \\ [t, a, v, a, L] \\ [t, b, v, b, L] \\ \\ [v, b, x, b, R] \end{aligned}$$

(where B is the designated blank symbol.)

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$(a \cup b)^* b (a \cup b)$$

(b) Is L regular? Explain your answer briefly.

Answer: Yes—the regular expression for L is given in the answer to part (a).

Problem 845 Consider a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$\begin{aligned} Q &= \{q, r\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{a, b, c, B\} \end{aligned}$$

and δ is defined by the following transition set:

$$\begin{aligned} [q, a, r, a, R] \\ [q, b, r, b, R] \\ [q, c, q, c, R] \\ [q, B, q, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

Let L be the set of those strings over Σ on which the Turing machine M does not halt.

(a) Draw a state transition graph of a deterministic finite automaton M_1 over alphabet Σ that accepts L . If such finite automaton does not exist, prove it.

Answer: See Figure 335.

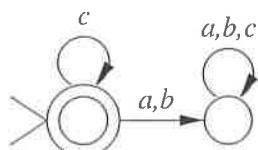


Figure 335:

- (b) Is it possible to write a computer program (algorithm) that operates as follows:

INPUT: An arbitrary string τ consisting of letters $\{a, b, c\}$.

OUTPUT: yes if $\tau \in L$ and no if $\tau \notin L$.

Explain your answer briefly.

Answer: Yes—such an algorithm simulates the finite automaton constructed in the answer to part (a).

Problem 846 Consider a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$\begin{aligned} Q &= \{q, r, s\} \\ \Sigma &= \{0, 1\} \\ \Gamma &= \{0, 1, B, \alpha, \beta\} \end{aligned}$$

and δ is defined by the following transition set:

$$\begin{aligned} [q, 0, q, \alpha, R] \\ [q, 1, r, \beta, R] \\ [r, 0, q, \beta, R] \\ [r, 1, s, \alpha, R] \\ [q, B, q, B, R] \\ [r, B, q, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

- (a) Write 8 distinct strings on which M halts. If such strings do not exist, explain why.

- (b) Write 8 distinct strings over alphabet $\{0, 1\}$ on which M does not halt. If such strings do not exist, explain why.

Answer:

$\in L$	$\notin L$
11	0
011	00
110011	101010
0010101100	001001
110	000
11111	1000
001100	0100
010101101011100	λ

- (c) Write a regular expression that defines the set of strings on which M halts. If such regular expression does not exist, explain why.

Answer:

$$(0 \cup 1)^* 11(0 \cup 1)^*$$

- (d) Let L be the set of strings on which M halts. Is L a recursive language? Explain your answer.

Answer: L is recursive because it is regular. All regular languages are recursive.

Problem 847 Consider a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$\begin{aligned} Q &= \{q, r, s\} \\ \Sigma &= \{0, 1\} \\ \Gamma &= \{0, 1, B, \alpha, \beta\} \end{aligned}$$

and δ is defined by the following transition set:

$$\begin{aligned} [q, 0, r, \alpha, R] \\ [q, 1, q, \beta, R] \\ [r, 0, s, \alpha, R] \\ [r, 1, r, \beta, R] \\ [q, B, q, B, R] \\ [r, B, r, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

- (a) Write 8 distinct strings on which M halts. If such strings do not exist, explain why.

Answer:

$$00, 001, 100, 010, 000, 1010, 110011, 1101011101$$

- (b) Write 8 distinct strings over alphabet $\{0, 1\}$ on which M does not halt. If such strings do not exist, explain why.

Answer:

$$\lambda, 0, 1, 01, 10, 11, 011, 101$$

- (c) Write a regular expression that defines the set of strings on which M halts. If such regular expression does not exist, explain why.

Answer:

$$1^* 01^* 0 (0 \cup 1)^*$$

- (d) Does M halt on input 11011101110? If your answer is “yes”, write the configuration in which M halts. If your answer is “no”, write the configuration of M after it performs 10 computational steps.

Answer: Yes, M halts on input 11011101110, in the configuration:

$$\beta\beta\alpha\beta\beta\beta\alpha s1110$$

Problem 848 Let L be the set of strings over alphabet $\{a, b, c\}$ that have even length and contain exactly one a .

- (a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

- (b) Does there exist a Turing machine that accepts L and halts on every input? Explain your answer briefly.

Problem 849 Let L be the language accepted (by final state) by the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$\begin{aligned} Q &= \{q, r, s\}; \\ \Sigma &= \{a, b, c\}; \\ \Gamma &= \{B, a, b, c\}; \end{aligned}$$

$$F = \{s\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, r, a, L] \\ & [q, b, q, b, R] \\ & [q, c, q, c, R] \\ & [q, B, q, B, R] \\ & [r, a, r, a, L] \\ & [r, b, r, b, L] \\ & [r, c, s, c, R] \end{aligned}$$

(where B is the designated blank symbol.)

Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$(b \cup c)^* c b^* a (a \cup b \cup c)^*$$

Problem 850 Consider a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q_0)$$

such that:

$$\begin{aligned} Q &= \{q_0, q_1\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{a, b, c, B\} \end{aligned}$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q_0, a, q_1, a, R] \\ & [q_0, b, q_0, b, R] \\ & [q_0, c, q_0, c, R] \\ & [q_0, B, q_0, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

Draw a state transition graph of a deterministic finite automaton M_1 such that M_1 accepts the set of those strings over Σ on which the Turing machine M does not halt. If such finite automaton M_1 does not exist, prove it.

Answer: See Figure 336.

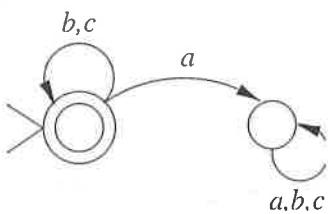


Figure 336:

Problem 851 Let L be the language accepted by the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q_0, F)$$

such that:

$$\begin{aligned} Q &= \{q_0, q_1, q_2, q_3, q_f\}; \\ \Sigma &= \{a, b\}; \end{aligned}$$

$$\Gamma = \{B, a, b\};$$

$$F = \{q_f\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q_0, a, q_1, a, R], \\ & [q_0, b, q_1, b, R], \\ & [q_1, b, q_2, b, R], \\ & [q_2, a, q_2, a, R], \\ & [q_2, b, q_3, b, R], \\ & [q_2, B, q_3, B, R], \\ & [q_3, b, q_3, b, R], \\ & [q_3, B, q_f, B, R] \end{aligned}$$

(a) Write 5 distinct strings that belong to L . If such strings do not exist, prove it.

(b) Write 5 distinct strings over alphabet Σ that do not belong to L . If such strings do not exist, prove it.

(c) Write a complete formal definition or a state-transition graph of a finite automaton M' that accepts L . If such automaton does not exist, prove it.

(d) Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, prove it.

Advice for Answer: L is defined by the regular expression:

$$(a \cup b)ba^*b^*$$

Problem 852 Let L be the language accepted by the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, q_a)$$

such that:

$$Q = \{q, q_1, q_2, q_a\};$$

$$\Sigma = \{a, b\};$$

$$\Gamma = \{B, a, b\};$$

q is the initial state;

q_a is the single accepting state;

and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, q_1, a, R], \\ & [q, b, q_1, b, R], \\ & [q_1, b, q_2, b, R], \\ & [q_2, a, q_2, a, R], \\ & [q_2, b, q_2, b, R], \\ & [q_2, B, q_a, B, R] \end{aligned}$$

Write a complete formal definition or a state-transition graph of a finite automaton M that accepts L . If such automaton does not exist, prove it.

Advice for Answer: L is defined by the regular expression:

$$(a \cup b)b(a \cup b)^*$$

Problem 853 Let L be the language accepted by the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q_0, q_a)$$

such that:

$$Q = \{q_0, q_1, q_2, q_3, q_a\};$$

$$\Sigma = \{a, b\};$$

$$\Gamma = \{B, a, b\};$$

q_0 is the initial state;

q_a is the single accepting state;

and δ is defined by the following transition set:

$$[q_0, a, q_1, B, R],$$

$$[q_0, b, q_1, B, R],$$

$$[q_1, a, q_2, B, R],$$

$$[q_1, b, q_2, B, R],$$

$$[q_2, a, q_3, B, R],$$

$$[q_2, b, q_3, B, R],$$

$$[q_3, a, q_3, B, R],$$

$$[q_3, b, q_3, B, R],$$

$$[q_3, B, q_a, B, R]$$

Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, prove it.

Advice for Answer: L is defined by the regular expression:

$$(a \cup b)(a \cup b)(a \cup b)(a \cup b)^*$$

Problem 854 Let L be the language accepted by Turing machine M :

$$M = (Q, \Sigma, \Gamma, \delta, q_0, F)$$

such that:

$$Q = \{q, r, s, t, v, w\}$$

$$\Sigma = \{0, 1\}$$

$$\Gamma = \{0, 1, B, C, D\}$$

$$F = \{w\}$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, 0, r, C, R] \\ & [q, 1, s, D, R] \\ & [r, 1, t, C, R] \\ & [t, 1, w, C, R] \\ & [s, 0, v, D, R] \\ & [v, 0, w, D, R] \\ & [w, 0, w, B, R] \\ & [w, 1, w, B, R] \end{aligned}$$

(a) Write 10 distinct strings that belong to L . If such strings do not exist, explain why.

(b) Write 10 distinct strings over alphabet $\{0, 1\}$ that do not belong to L . If such strings do not exist, explain why.

Advice for Answer: L is defined by the regular expression:

$$(011 \cup 100)(0 \cup 1)^*$$

Problem 855 Let L be the language accepted by the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q_0, F)$$

such that:

$$Q = \{q, r, s, t\};$$

$$\Sigma = \{a, b, c\};$$

$$\Gamma = \{B, a, b, c\};$$

$$F = \{t\};$$

and δ is defined by the following transition set:

$$[q, b, r, b, R],$$

$$[r, c, s, c, R],$$

$$[s, a, q, a, R],$$

$$[q, B, t, B, R]$$

Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(bca)^*$$

Problem 856 Let L be the language accepted by the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q_0, F)$$

such that:

$$Q = \{q_0, q_1, q_2, q_3, q_f\};$$

$$\Sigma = \{a, b, c\};$$

$$\Gamma = \{B, a, b, c\};$$

$$F = \{q_f\};$$

and δ is defined by the following transition set:

$$[q_0, a, q_1, a, R],$$

$$[q_0, b, q_2, b, R],$$

$$[q_0, c, q_3, c, R],$$

$$[q_1, b, q_1, b, R],$$

$$[q_2, c, q_2, c, R],$$

$$[q_3, a, q_3, a, R],$$

$$[q_1, B, q_f, B, R],$$

$$[q_2, B, q_f, B, R],$$

$$[q_3, B, q_f, B, R]$$

Write a complete formal definition or a state-transition graph of a finite automaton M' that accepts L . If such automaton does not exist, prove it.

Advice for Answer: L is defined by the regular expression:

$$ab^* \cup bc^* \cup ca^*$$

Problem 857 Consider a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q_0)$$

such that:

$$Q = \{q_0, q_1, q_2\}$$

$$\Sigma = \{a, b, c\}$$

$$\Gamma = \{a, b, c, B\}$$

and δ is defined by the following transition set:

$[q_0, a, q_1, a, R]$
 $[q_0, b, q_0, b, R]$
 $[q_0, c, q_0, c, R]$
 $[q_0, B, q_0, B, R]$
 $[q_1, a, q_2, a, R]$
 $[q_1, b, q_1, b, R]$
 $[q_1, c, q_1, c, R]$
 $[q_1, B, q_1, B, R]$

(where B is the designated blank symbol.)

(a) Let L be the set of those strings over Σ on which the Turing machine M halts. Draw a state transition graph of a finite automaton M_1 that accepts L . If such finite automaton M_1 does not exist, prove it.

Advice for Answer: The regular expression for L :

$$(b \cup c)^* a (b \cup c)^* a (a \cup b \cup c)^*$$

(b) Is \bar{L} (the complement of L) a recursive language? Explain your answer briefly.

Answer: Yes— \bar{L} is regular, which is demonstrated by the regular expression constructed in part (a); \bar{L} is regular because L is regular and the class of regular languages is closed under complement; every regular language is recursive as the decision procedure consists of a simulation of the finite automaton that accepts the language.

Problem 858 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$Q = \{q, r, s, t, x, y, z, p\};$$

$$\Sigma = \{a, b\};$$

$$\Gamma = \{B, a, b\};$$

$$F = \{t\};$$

and δ is defined by the following transition set:

$[q, a, r, a, R]$
 $[q, b, s, b, R]$
 $[r, a, p, a, R]$
 $[r, b, r, b, R]$
 $[r, B, x, B, L]$
 $[s, a, s, a, R]$
 $[s, b, p, b, R]$
 $[s, B, y, B, L]$
 $[x, a, t, a, R]$
 $[x, b, z, B, R]$
 $[y, b, t, b, R]$
 $[y, a, z, B, R]$
 $[z, B, z, B, R]$

(where B is the designated blank symbol.) Let L be the set of strings over Σ which M rejects.

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$\lambda \cup ab^*a(a \cup b)^* \cup ba^*b(a \cup b)^*$$

(b) Is \bar{L} (the complement of L) decidable? Explain your answer briefly.

Answer: Yes. L itself is regular, as witnessed by the regular expression constructed in the answer to part (a). The class of regular languages is closed under complement—hence, \bar{L} is also regular. Every regular language is decidable.

Problem 859 Consider a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$Q = \{q, r, s, t\}$$

$$\Sigma = \{a, b, c\}$$

$$\Gamma = \{a, b, c, B\}$$

and δ is defined by the following transition set:

$[q, a, r, a, R]$
 $[q, b, q, b, R]$
 $[q, c, q, c, R]$
 $[q, B, q, B, R]$
 $[r, a, r, a, R]$
 $[r, b, s, b, R]$
 $[r, c, r, c, R]$
 $[r, B, r, B, R]$
 $[s, a, s, a, R]$
 $[s, b, s, b, R]$
 $[s, c, t, c, R]$
 $[s, B, s, B, R]$

(where B is the designated blank symbol.)

Let L be the set of those strings over Σ on which the Turing machine M halts.

(a) Draw a state transition graph of a finite automaton M_1 that accepts L . If such finite automaton M_1 does not exist, prove it.

Answer: See Figure 337.

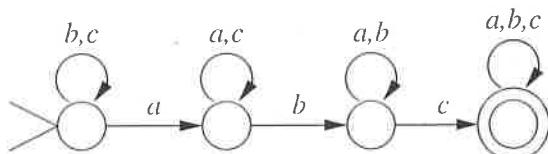


Figure 337:

(b) Is \bar{L} (the complement of L) decidable? Explain your answer briefly.

Answer: Yes. L itself is regular, as witnessed by the finite automaton constructed in the answer to part (a). The class of regular languages is closed under complement—hence, \bar{L} is also regular. Every regular language is decidable.

Problem 860 Consider a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$Q = \{q, s, t, x, v\};$$

$$\Sigma = \{a, b, c\};$$

$$\Gamma = \{B, a, b, c\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, s, a, R] \\ & [q, b, q, b, R] \\ & [q, c, q, c, R] \\ & [s, a, s, a, R] \\ & [s, b, s, b, R] \\ & [s, c, t, c, R] \\ & [t, a, t, a, R] \\ & [t, b, t, b, R] \\ & [t, c, t, c, R] \\ & [t, B, x, B, L] \\ & [x, b, v, B, R] \\ & [v, B, v, B, R] \end{aligned}$$

Let L be the set of strings on which M diverges.

- (a) Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: Note that L is defined by the following regular expression:

$$(b \cup c)^* a (a \cup b)^* c (a \cup b \cup c)^* b$$

whence the grammar: $G = (V, \Sigma, P, S)$, where

$$\Sigma = \{a, b, c\}, V = \{S, A, B, D\},$$

and the production set P is:

$$\begin{aligned} S &\rightarrow BaAcDb \\ A &\rightarrow aA \mid bA \mid \lambda \\ B &\rightarrow bB \mid cB \mid \lambda \\ D &\rightarrow aD \mid bD \mid cD \mid \lambda \end{aligned}$$

- (b) Is \overline{L} (the complement of L) context-free? Explain your answer briefly.

Answer: Yes. L itself is regular, as witnessed by the regular expression constructed in the answer to part (a). The class of regular languages is closed under complement—hence, \overline{L} is also regular. Every regular language is context-free.

Problem 861 Let L be the language accepted (by final state) by the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$Q = \{q, r, s, t, x, y\};$$

$$\Sigma = \{0, 1\};$$

$$\Gamma = \{B, 0, 1\};$$

$$F = \{t\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, 0, r, 0, R] \\ & [q, 1, s, 1, R] \\ & [r, 1, r, 1, R] \\ & [r, B, x, B, L] \\ & [s, 0, s, 0, R] \\ & [s, B, y, B, L] \\ & [x, 1, t, B, R] \\ & [y, 0, t, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

- (a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$10^*0 \cup 01^*1$$

- (b) Dangerous Professor has told her students to write a program that operates as follows:

INPUT: A pair of strings x, y over $\{0, 1\}$, such that $y \in L$.

OUTPUT: **yes** if x is a finite automaton that accepts y , and **no** otherwise.

Explain the algorithm that should be employed by this program, or prove that it does not exist.

Answer: Simulate the operation of the finite automaton x on the input string y . If x accepts, output **yes**; if x is not a legal finite automaton, or if it rejects y , output **no**.

Problem 862 (a) Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$Q = \{q, r, s\};$$

$$\Sigma = \{a, b\};$$

$$\Gamma = \{B, a, b\};$$

$$F = \{q\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, r, a, R] \\ & [q, b, s, b, R] \\ & [r, a, r, a, R] \\ & [s, b, r, b, R] \\ & [r, B, q, B, R] \\ & [q, B, q, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

Let L be the set of strings over Σ on which M does not halt. Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$a^* \cup bba^*$$

- (b) State the cardinality of the language L defined in part (a) and the cardinality of its complement \overline{L} , and compare the two cardinalities.

Answer: Languages L and \bar{L} have equal cardinalities; both of them are infinite and countable:

$$|L| = |\bar{L}| = \aleph_0$$

The two languages are countable, since every language is countable. To see that L is infinite, observe that it contains a Kleene star of a non-empty language $\{a\}$. Analogously, to see that \bar{L} is infinite, observe that it also contains a set as numerous as a Kleene star of a non-empty language, say: $bbbb^*$

Problem 863 (a) Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$Q = \{q, r, s, t\};$$

$$\Sigma = \{a, b, c\};$$

$$\Gamma = \{B, a, b, c\};$$

$$F = \{q\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, r, a, R] \\ & [q, b, s, b, R] \\ & [q, c, t, b, R] \\ & [r, a, r, a, R] \\ & [s, a, r, b, R] \\ & [t, B, q, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

Let L be the set of strings over Σ which are rejected by M . Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$\begin{aligned} & b \\ & \cup \\ & \quad c(a \cup b \cup c)^* \\ & \cup \\ & \quad aa^*(b \cup c)(a \cup b \cup c)^* \\ & \cup \\ & \quad baa^*(b \cup c)(a \cup b \cup c)^* \\ & \cup \\ & \quad b(b \cup c)(a \cup b \cup c)^* \end{aligned}$$

(b) State the cardinality of the language L defined in part (a) and the cardinality of L^* , and compare the two cardinalities.

Answer: Languages L and L^* have equal cardinalities; both of them are infinite and countable:

$$|L| = |L^*| = \aleph_0$$

The two languages are countable, since every language is countable. Furthermore, each of them contains (or has a surjection to) a Kleene star of a non-empty set: L , for instance, contains $c(a \cup b \cup c)^*$, while L^* is the Kleene star of L itself.

Problem 864 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, p, F)$$

such that: $Q = \{p, q, s, t, e\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, X, Y, Z, a, b, c\}$; $F = \{t\}$. and δ is defined by the following transition set:

$$\begin{aligned} & [p, a, q, X, R] \\ & [p, b, q, Y, R] \\ & [p, c, q, Z, R] \end{aligned}$$

$$\begin{aligned} & [q, a, q, X, R] \\ & [q, b, q, Y, R] \\ & [q, c, q, Z, R] \\ & [q, B, s, B, L] \end{aligned}$$

$$\begin{aligned} & [s, X, t, B, R] \\ & [s, Y, s, B, R] \\ & [s, Z, e, B, R] \end{aligned}$$

$$[e, B, e, B, R]$$

(where B is the designated blank symbol.)

Let L_1 be the set of strings rejected by M , and let L_2 be the set of strings on which M diverges.

(a) Write a regular expression that defines L_1 . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)^*b \cup \lambda$$

(b) Write a regular expression that defines L_2 . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)^*c$$

(c) Which (if any) of the two languages: L_1 and L_2 are recursive (i.e., decidable)? Explain your answer.

Answer: Both L_1 and L_2 are recursive. By the answer to part (a), they are both regular, and all regular languages are recursive.

Problem 865 Let L be the language accepted (by final state) by the Turing Machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F),$$
 such that:

$$Q = \{q, r, s\}; \Sigma = \{0, 1\}; \Gamma = \{B, 0, 1\}; F = \{s\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, 0, q, 0, R] \\ & [q, 1, r, 1, R] \\ & [r, 0, q, 0, R] \\ & [r, 1, s, 1, R] \end{aligned}$$

(where B is the designated blank symbol.)

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(0 \cup 1)^*11(0 \cup 1)^*$$

(b) Explain how to construct an algorithm that solves the following problem:

INPUT: A string x over $\{0, 1\}$.

OUTPUT: **yes** if the Turing Machine M does not accept x ;

no otherwise.

If this algorithm does not exist, prove it.

Answer: The required algorithm is obtained as a sequence of the following algorithmic conversions.

- convert the regular expression given in the answer to part (a) into an equivalent finite automaton F —the result is represented on Figure 338;
- convert the finite automaton F into an equivalent deterministic finite automaton F_1 ;
- convert the deterministic finite automaton F_1 into an equivalent Turing Machine T , which accepts by final state and halts on every input.

On input x , our algorithm returns **yes** if T rejects x , and returns **no** if T accepts x .

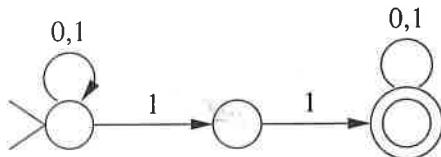


Figure 338:

Problem 866 Let L be the language accepted (by final state) by the Turing Machine:

$M = (Q, \Sigma, \Gamma, \delta, q_f, F)$, such that:

$Q = \{q, r, s, t, v, x, y\}$; $\Sigma = \{0, 1\}$; $\Gamma = \{B, 0, 1\}$; $F = \{t\}$; and δ is defined by the following transition set:

$[q, 0, r, 0, R]$
 $[q, 1, q, 1, R]$
 $[r, 0, s, 0, R]$
 $[r, 1, q, 1, R]$
 $[s, 0, t, 0, R]$
 $[s, 1, q, 1, R]$

(where B is the designated blank symbol.)

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$(0 \cup 1)^* 000 (0 \cup 1)^*$$

(b) Explain how to construct an algorithm that solves the following problem:

INPUT: A string x over $\{0, 1\}$.

OUTPUT: **yes** if x is a string such that the Turing Machine M halts on input x in a final state;

no otherwise.

If this algorithm does not exist, prove it.

Answer: The required algorithm is obtained as a sequence of the following algorithmic conversions.

- convert the regular expression given in the answer to part (a) into an equivalent finite automaton F —the result is represented on Figure 339;
- convert the finite automaton F into an equivalent deterministic finite automaton F_1 ;
- convert the deterministic finite automaton F_1 into an equivalent Turing Machine T , which accepts by final state and halts on every input.

On input x , our algorithm returns **yes** if T accepts x , and returns **no** if T rejects x .

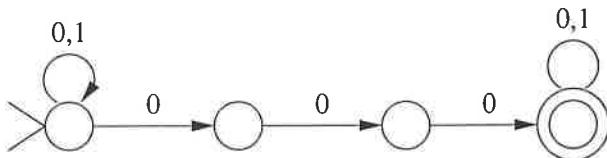


Figure 339:

Problem 867 Consider the Turing Machine: $M = (Q, \Sigma, \Gamma, \delta, q_f)$ such that:
 $Q = \{q, r, s, t, v, x, y\}$; $\Sigma = \{0, 1\}$; $\Gamma = \{B, 0, 1\}$; and δ is defined by the following transition set:

$[q, 0, y, 0, R]$
 $[q, 1, x, 1, R]$
 $[x, 0, x, 0, R]$
 $[x, 1, x, 1, R]$
 $[x, B, r, B, L]$
 $[r, 1, s, 1, L]$
 $[s, 0, t, 0, L]$
 $[t, 0, v, 0, R]$
 $[v, 0, v, 0, R]$
 $[v, 1, v, 1, R]$
 $[v, B, v, B, R]$

(where B is the designated blank symbol.)

Let L be the set of strings on which M diverges.

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$1 (0 \cup 1)^* 001$$

(b) Explain how to construct an algorithm that solves the following problem:

INPUT: A string x over $\{0, 1\}$.

OUTPUT: **yes** if x is a string such that the Turing Machine M decides the language L ;

no otherwise.

If this algorithm does not exist, prove it.

Answer: This is impossible by Rice's Theorem. Precisely, the property “*is equal to L*” is a non-trivial property of recursively enumerable languages—language L has this property while \emptyset does not have it. Hence, it is impossible to have an algorithm to recognize Turing Machines that accept a language with this particular property.

Problem 868 Consider the Turing Machine:

$M = (Q, \Sigma, \Gamma, \delta, q)$ such that:

$Q = \{q, r, s, t, v\}; \Sigma = \{0, 1\}; \Gamma = \{B, 0, 1\}$;

and δ is defined by the following transition set:

$[q, 0, q, 0, R]$
 $[q, 1, q, 1, R]$
 $[q, B, r, B, L]$
 $[r, 0, r, 0, L]$
 $[r, 1, s, 1, L]$
 $[s, 0, t, 0, L]$
 $[t, 0, v, 0, R]$
 $[v, 0, v, 0, R]$
 $[v, 1, v, 1, R]$
 $[v, B, v, B, R]$

(where B is the designated blank symbol.)

Let L be the set of string on which M diverges.

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$(0 \cup 1)^* 0010^*$$

(b) Explain how to construct an algorithm that solves the following problem:

INPUT: A string x over $\{0, 1\}$.

OUTPUT: **yes** if x is a Turing Machine that halts exactly when the machine M diverges;
no otherwise.

If this algorithm does not exist, prove it.

Answer: This is impossible by Rice's Theorem, since this algorithm would recognize Turing Machines that accept the language L , defined in the answer to part (a). Precisely, the property “*is equal to L*” is a non-trivial property of recursively enumerable languages—language L has this property while \emptyset does not have it. Hence, it is impossible to have an algorithm to recognize Turing Machines that accept a language with this particular property.

Problem 869 Consider the Turing Machine:

$M = (Q, \Sigma, \Gamma, \delta, q)$ such that:

$Q = \{q, r, s, t, v, x\}; \Sigma = \{0, 1\}; \Gamma = \{B, 0, 1\}$;

and δ is defined by the following transition set:

$[q, 0, q, 0, R]$
 $[q, 1, q, 1, R]$
 $[q, B, r, B, L]$
 $[r, 0, s, 0, L]$
 $[r, 1, t, 1, R]$
 $[s, 0, x, 0, L]$
 $[s, 1, v, 1, R]$
 $[x, 0, v, 0, R]$
 $[x, 1, t, 1, R]$
 $[v, 0, v, 0, R]$
 $[v, 1, v, 1, R]$
 $[v, B, v, B, R]$

(where B is the designated blank symbol.)

Let L be the set of string on which M halts.

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$(0 \cup 1)^* (100 \cup 1)$$

(b) Explain how to construct an algorithm that solves the following problem:

INPUT: A string x over $\{0, 1\}$.

OUTPUT: **yes** if x is a finite automaton that accepts the set of strings accepted by the Turing Machine M by halting;

no otherwise.

If this algorithm does not exist, prove it.

Answer: The required algorithm is obtained as a sequence of the following algorithmic conversions and constructions.

- convert the regular expression given in the answer to part (a) into an equivalent finite automaton F ;
- convert the finite automaton F into an equivalent deterministic finite automaton F_1 ;
- convert the input finite automaton x into an equivalent deterministic finite automaton y ;
- construct a finite automaton D that accepts the language:

$$\left(L(F_1) \cap \overline{L(y)} \right) \cup \left(L(y) \cap \overline{L(F_1)} \right)$$

- convert the finite automaton D into an equivalent deterministic finite automaton D_1 ; let n be the number of states of this automaton.

On input x , our algorithm returns **yes** if D_1 does not accept any strings of length less than n , and returns **no** otherwise.

Problem 870 Consider the Turing Machine:
 $M = (Q, \Sigma, \Gamma, \delta, q)$ such that:
 $Q = \{q, r, t, x, y\}; \Sigma = \{0, 1\}; \Gamma = \{B, 0, 1\}$;
and δ is defined by the following transition set:

$$\begin{aligned} & [q, 0, q, 0, R] \\ & [q, 1, x, 1, L] \\ & [q, B, q, B, R] \\ & [x, 0, y, 0, L] \\ & [x, 1, r, 1, R] \\ & [y, 0, t, 0, R] \\ & [y, 1, r, 1, R] \\ & [r, 0, r, 0, R] \\ & [r, 1, r, 1, R] \\ & [r, B, r, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

Let L be the set of strings on which M halts.

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$0^* 001 (0 \cup 1)^*$$

(b) Explain how to construct an algorithm that solves the following problem:

INPUT: A string x over $\{0, 1\}$.

OUTPUT: yes if x is a finite automaton that accepts exactly those strings on which the Turing Machine M diverges;
no otherwise.

If this algorithm does not exist, prove it.

Answer: The required algorithm is obtained as a sequence of the following algorithmic conversions and constructions.

- convert the regular expression given in the answer to part (a) into an equivalent finite automaton F ;
- convert the finite automaton F into an equivalent deterministic finite automaton F_1 ;
- from the finite automaton F_1 , construct a deterministic finite automaton F_2 that accepts $\overline{L(F_1)}$;
- convert the input finite automaton x into an equivalent deterministic finite automaton y ;
- construct a finite automaton D that accepts the language:

$$(L(F_2) \cap \overline{L(y)}) \cup (L(y) \cap \overline{L(F_2)})$$

- convert the finite automaton D into an equivalent deterministic finite automaton D_1 ; let n be the number of states of this automaton.

On input x , our algorithm returns yes if D_1 does not accept any strings of length less than n , and returns no otherwise.

Problem 871 (a) Write a complete formal definition of a Turing machine M_1 over input alphabet $\{a, b\}$ such that M_1 halts on every input. If such a machine does not exist, explain why.

Answer:

$$M_1 = (Q, \Sigma, \Gamma, \delta, q_0)$$

where: $Q = \{q_0\}; \Sigma = \{a, b\}; \Gamma = \{B, a, b\}; \delta = \emptyset$.

(b) Write a complete formal definition of a Turing machine M_2 over input alphabet $\{a, b\}$ such that M_2 does not halt on any input. If such a machine does not exist, explain why.

Answer:

$$M_2 = (Q, \Sigma, \Gamma, \delta, q_0)$$

where: $Q = \{q_0\}; \Sigma = \{a, b\}; \Gamma = \{B, a, b\}$; and δ contains 3 transitions:

$$[q_0, a, q_0, a, R], [q_0, b, q_0, b, R], [q_0, B, q_0, B, R]$$

(c) Write a complete formal definition of a Turing machine M_3 over input alphabet $\{a, b\}$ such that M_3 halts on every input and rejects Σ^* . If such a machine does not exist, explain why.

Answer:

$$M_3 = (Q, \Sigma, \Gamma, \delta, q_0, F)$$

where:

$$Q = \{q_0\}; \Sigma = \{a, b\}; \Gamma = \{B, a, b\}; \delta = \emptyset; F = \emptyset$$

(d) Write a complete formal definition of a Turing machine M_4 over input alphabet $\{a, b\}$ such that M_4 does not halt on any input and accepts Σ^* . If such a machine does not exist, explain why.

Answer: Impossible. A machine cannot accept unless it halts.

Problem 872 Write a complete formal definition of a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q_0)$$

over input alphabet $\{0, 1\}$, such that M halts on every input, after making exactly 5 moves. If such machine does not exist, explain why.

Answer: $Q = \{q_0, q_1, q_2, q_3, q_4, q_5\}; \Sigma = \{0, 1\}; \Gamma = \{B, 0, 1\}$; and δ is defined by the following transition set:

$$\begin{aligned} & [q_0, 0, q_1, 0, R], [q_0, 1, q_1, 1, R], [q_0, B, q_1, B, R], \\ & [q_1, 0, q_2, 0, R], [q_1, 1, q_2, 1, R], [q_1, B, q_2, B, R], \\ & [q_2, 0, q_3, 0, R], [q_2, 1, q_3, 1, R], [q_2, B, q_3, B, R], \\ & [q_3, 0, q_4, 0, R], [q_3, 1, q_4, 1, R], [q_3, B, q_4, B, R], \\ & [q_4, 0, q_5, 0, R], [q_4, 1, q_5, 1, R], [q_4, B, q_5, B, R]. \end{aligned}$$

Problem 873 Write a complete formal definition of a Turing machine M with input alphabet $\{a, b\}$ such that M accepts every input string $w \in \{a, b\}^*$, making no more than 5 moves. If such M does not exist, explain why.

Problem 874 Write a complete formal definition of a Turing machine M with input alphabet $\{a, b\}$ such that M does not halt on any input string $w \in \{a, b\}^*$. If such M does not exist, explain why.

Problem 875 Let L be the language defined by the regular expression:

$$b^*$$

(a) Write a complete formal definition of a Turing machine M_1 over input alphabet $\{a, b\}$ such that M_1 accepts L by halting. If such a machine does not exist, explain why.

Answer:

$$M_1 = (Q, \Sigma, \Gamma, \delta, q_0)$$

such that:

$$\begin{aligned} Q &= \{q_0, q_\infty\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{a, b, B\} \end{aligned}$$

and δ is defined by the following transition set:

$$\begin{aligned} &[q_0, a, q_\infty, a, R] \\ &[q_0, b, q_0, b, R] \\ &[q_\infty, a, q_\infty, B, R] \\ &[q_\infty, b, q_\infty, B, R] \\ &[q_\infty, B, q_\infty, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

(b) Write a complete formal definition of a Turing machine M_1 over input alphabet $\{a, b\}$ such that M_1 halts on every input and accepts L by final state. If such a machine does not exist, explain why.

Answer:

$$M_1 = (Q, \Sigma, \Gamma, \delta, q_0, F)$$

such that:

$$\begin{aligned} Q &= \{q_0, r, f\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{a, b, B\} \\ F &= \{f\} \end{aligned}$$

and δ is defined by the following transition set:

$$\begin{aligned} &[q_0, a, r, a, R] \\ &[q_0, b, q_0, b, R] \\ &[q_0, B, f, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

(c) Write a complete formal definition of a Turing machine M_1 over input alphabet $\{a, b\}$ such that M_1 accepts $\overline{L} = \Sigma^* \setminus L$ by halting. If such a machine does not exist, explain why.

Answer: Observe that \overline{L} is the set of strings that contain at least one a .

$$M_1 = (Q, \Sigma, \Gamma, \delta, q_0)$$

such that:

$$\begin{aligned} Q &= \{q_0, q_1\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{a, b, B\} \end{aligned}$$

and δ is defined by the following transition set:

$$\begin{aligned} &[q_0, a, q_1, a, R] \\ &[q_0, b, q_0, b, R] \\ &[q_0, B, q_0, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

Problem 876 Write a complete formal definition of a Turing machine M with input alphabet $\{a, b\}$ such that M halts on every input string $w \in \{a, b\}^*$, while its tape, after halting, contains someplace symbol A and is otherwise blank. If such M does not exist, explain why.

Answer: One of the shortest possible solutions is:

$$M = (Q, \Sigma, \Gamma, \delta, q_0)$$

where:

$$\begin{aligned} Q &= \{q_0, q_1\}; \Sigma = \{a, b\}; \Gamma = \{B, A, a, b\}; \\ \text{and } \delta \text{ is defined by the following transition set:} \\ &[q_0, a, q_0, B, R], [q_0, b, q_0, B, R], [q_0, B, q_1, A, R]. \end{aligned}$$

M starts by erasing its input string, if any, and then halts in state q_1 , having written A .

Problem 877 Write a complete formal definition of a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q_0)$$

over input alphabet $\{0, 1\}$, such that M halts on blank-tape input, and does not halt on any other input. If such machine does not exist, explain why.

Answer: Assume that the tape head is scanning the leftmost symbol of the input string in the initial state. $Q = \{q_0, q_1\}$; $\Sigma = \{0, 1\}$; $\Gamma = \{B, 0, 1\}$; and δ is defined by the following transition set:

$$\begin{aligned} &[q_0, 0, q_1, 0, R], [q_0, 1, q_1, 1, R], \\ &[q_1, 0, q_1, 0, R], [q_1, 1, q_1, 1, R], [q_1, B, q_1, B, R]. \end{aligned}$$

Problem 878 Write a complete formal definition of a Turing machine M with input alphabet $\{a, b\}$ such that M appends the five-symbol string HELLO after the rightmost symbol of its input and then halts. If such machine does not exist, explain why.

Answer:

$$M = (Q, \Sigma, \Gamma, \delta, q, q_a, q_r)$$

where: $Q = \{q, q_H, q_E, q_L, q_{LL}, q_O, q_a, q_r\}$;
 $\Sigma = \{a, b\}$;

$$\Gamma = \{B, a, b, H, E, L, O\}$$

and δ is defined by the following transition set:

$$\begin{aligned} &[q, a, q, a, R] \\ &[q, b, q, b, R] \\ &[q, B, q_H, H, R] \end{aligned}$$

$[q_H, B, q_E, E, R]$
 $[q_E, B, q_L, L, R]$
 $[q_L, B, q_{LL}, L, R]$
 $[q_{LL}, B, q_O, O, R]$
 $[q_O, B, q_a, B, R]$

Problem 879 Let L be the language defined by the regular expression

$$(ab)^*$$

Write a complete formal definition of a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q_0, F)$$

such that M accepts L . If such machine does not exist, explain why.

Answer: $Q = \{q_0, q_1, q_e\}$; $\Sigma = \{a, b\}$; $\Gamma = \{B, a, b\}$; $F = \{q_0\}$; and δ is defined by the following transition set:

$[q_0, a, q_1, a, R]$, $[q_0, b, q_e, b, R]$,
 $[q_1, b, q_0, b, R]$, $[q_1, a, q_e, a, R]$.

Problem 880 Let L be the language defined by the regular expression:

$$a^*(b \cup c)a^*b$$

Write a complete formal definition of a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q_0, F)$$

such that M on input w operates as follows:

- If $w \in L$, then M halts and accepts.
- If $w \notin L$, then M halts and rejects.

If such a Turing machine M does not exist, prove it.

Answer:

$$M = (Q, \Sigma, \Gamma, \delta, q_0, F)$$

such that:

$$\begin{aligned} Q &= \{q_0, r, s, f\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{a, b, c, B\} \\ F &= \{f\} \end{aligned}$$

and δ is defined by the following transition set:

$[q_0, a, q_0, B, R]$
 $[q_0, b, r, B, R]$
 $[q_0, c, r, B, R]$
 $[r, a, r, B, R]$
 $[r, b, s, B, R]$
 $[s, B, f, B, R]$

(where B is the designated blank symbol.)

Problem 881 Write a complete formal definition of a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q_0, F)$$

such that M on input w operates as follows:

If w is a representation of a Turing machine that accepts a regular language, then M halts and accepts.

If w is not a representation of a Turing machine that accepts a regular language, then M halts and rejects.

If such a Turing machine M does not exist, prove it.

Problem 882 Let L be the language accepted by the finite automaton represented of Figure 340.

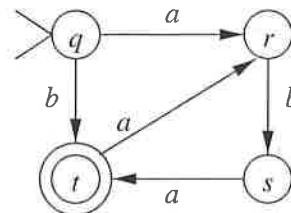


Figure 340:

Write a complete formal definition of a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q_0, F)$$

such that M on input w operates as follows:

If $w \in L$, then M halts and accepts.

If $w \notin L$, then M halts and rejects.

If such a Turing machine M does not exist, prove it.

Answer: The conversion of a finite automaton into a Turing Machine is algorithmic:

$$\begin{aligned} M &= (Q, \Sigma, \Gamma, \delta, q_0, F) \\ Q &= \{q_0, r, s, t, v\} \\ \Sigma &= \{a, b\}, \Gamma = \{a, b, B\} \\ F &= \{v\} \end{aligned}$$

and δ is defined by the following transition set:

$[q_0, a, r, a, R]$
 $[q_0, b, t, b, R]$
 $[r, b, s, b, R]$
 $[s, a, t, a, R]$
 $[t, a, r, a, R]$
 $[t, B, v, B, R]$

(where B is the designated blank symbol.)

Problem 883 Let L be the language accepted by the finite automaton represented of Figure 340.

Write a complete formal definition of a Turing machine:

$$M = (Q, \Sigma, F, \delta, q_0, F)$$

such that M on input w operates as follows:

If w is a representation of a Turing machine that accepts L , then M halts and accepts.

If w is not a representation of a Turing machine that accepts L , then M halts and rejects.

If such a Turing machine M does not exist, prove it.

Problem 884 Let L be the language accepted by the finite automaton represented by the state diagram on Figure 341.

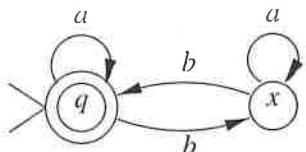


Figure 341:

Write a complete formal definition of a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q_0, F)$$

such that M accepts L . If such machine does not exist, explain why.

Answer: $Q = \{q_0, x, f\}$; $\Sigma = \{a, b\}$; $\Gamma = \{B, a, b\}$; $F = \{f\}$; and δ is defined by the following transition set:

$$\begin{aligned} & [q_0, a, q_0, a, R], [q_0, b, x, b, R], [q_0, B, f, B, R], \\ & [x, a, x, a, R], [x, b, q_0, b, R]. \end{aligned}$$

Problem 885 Let $L(M)$ be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s, t\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A, B\} \\ F &= \{t\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, r, A] \\ & [r, b, \lambda, s, B] \\ & [s, c, B, s, \lambda] \\ & [s, c, A, q, \lambda] \\ & [q, \lambda, \lambda, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Draw a state-transition graph of a finite automaton M_1 that accepts $L(M)$. If such automaton does not exist, prove it.

Answer: See Figure 342.

(b) Write a complete formal definition of a Turing machine M_2 such that M_2 halts on every input and accepts

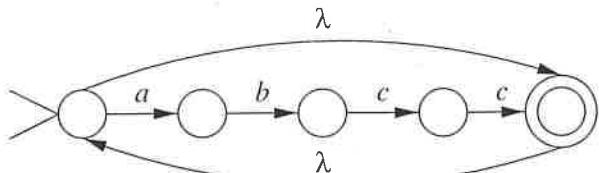


Figure 342:

exactly those strings that are accepted by the automaton M . In short:

$$(\tau \in L(M)) \rightarrow (M_2(\tau) \searrow \text{and accepts})$$

and also:

$$(\tau \notin L(M)) \rightarrow (M_2(\tau) \searrow \text{and rejects})$$

If such Turing machine does not exist, prove it.

Answer:

$$M_2 = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$\begin{aligned} Q &= \{q, r, s, t, v\} \\ \Sigma &= \{a, b, c, d\}, \Gamma = \{a, b, c, d, B\} \\ F &= \{v\} \end{aligned}$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, r, a, R] \\ & [r, b, s, b, R] \\ & [s, c, t, c, R] \\ & [t, c, q, c, R] \\ & [q, B, v, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

Problem 886 Let L be the language accepted by the finite automaton represented on Figure 229. Does there exist a Turing Machine M_T that operates as follows:

On input w , machine M_T halts and accepts if $w \in L$, and halts and rejects if $w \notin L$.

Explain your answer briefly.

Answer: Yes—on input w , the machine M_T simulates the finite automaton M , and decides as M decides.

Problem 887 Consider the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$Q = \{q, t\}; \Sigma = \{a, b\}; \Gamma = \{A, Z\}; F = \{q\};$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, t, ZAAA] \\ & [t, b, A, t, \lambda] \\ & [t, \lambda, Z, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

Let $L(M)$ be the language accepted by M .

- (a) List five different strings that belong to $L(M)$, or explain why this is impossible.

Answer: Observe that $L(M)$ is represented by the regular expression $(abb)^*$, whence the answer:

$$\lambda, abbb, abbbabbb, abbbabbbbabbb, abbbabbbbabbbabbb$$

- (b) Write a complete formal definition of a Turing machine T such that T accepts input ξ if ξ is a word that belongs to $L(M)$, and T rejects input ξ if ξ is not a word that belongs to $L(M)$, for all $\xi \in \Sigma^*$. In short:

$$(\xi \in L(M)) \rightarrow (T(\xi) \searrow \text{and accepts})$$

and also:

$$(\xi \notin L(M)) \rightarrow (T(\xi) \searrow \text{and rejects})$$

If such a Turing machine does not exist, prove it.

Answer: This Turing Machine is obtained by an algorithmic conversion of the deterministic finite automaton, constructed in the answer to part (c).

$$T = (Q, \Sigma, \Gamma, \delta, q, F)$$

where: $Q = \{q, r, s, t, x\}$;

$\Sigma = \{a, b\}$;

$\Gamma = \{B, a, b\}$;

$F = \{q\}$;

and δ is defined by the following transition set:

$$\begin{array}{l} [q, a, r, a, R] \\ [q, b, x, b, R] \end{array}$$

$$\begin{array}{l} [r, a, x, a, R] \\ [r, b, s, b, R] \end{array}$$

$$\begin{array}{l} [s, a, x, a, R] \\ [s, b, t, b, R] \end{array}$$

$$\begin{array}{l} [t, a, x, a, R] \\ [t, b, q, b, R] \end{array}$$

$$\begin{array}{l} [x, a, x, a, R] \\ [x, b, x, b, R] \end{array}$$

- (c) Write a complete formal definition (or draw a state transition graph) of a finite automaton F such that F accepts $L(M)$:

$$L(M) = L(F)$$

If such a finite automaton does not exist, prove it.

Answer: See Figure 343.

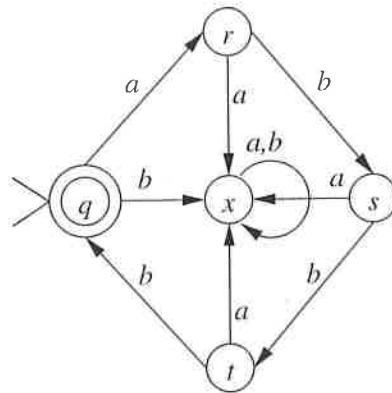


Figure 343:

Problem 888 Write a complete formal definition of a Turing machine M_1 such that M_1 accepts τ if τ is a Turing machine that accepts $L(G)$, and M_1 rejects τ if τ is not a Turing machine that accepts $L(G)$, for all $\tau \in \Sigma^*$. In short:

$$(L(\tau) = L(G)) \rightarrow (M_1(\tau) \searrow \text{and accepts})$$

and also:

$$(L(\tau) \neq L(G)) \rightarrow (M_1(\tau) \searrow \text{and rejects})$$

If such a Turing machine M_1 does not exist, prove it.

Answer: Such a Turing machine does not exist. The property “is equal to $L(G)$ ” is non-trivial. Hence, there does not exist a procedure to recognize the Turing machines that accept languages equal to $L(G)$.

Problem 889 Let L be the language accepted by the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$Q = \{q, r, s\}; \Sigma = \{a, b, c\}$;

$\Gamma = \{B, a, b, c\}; F = \{s\}$;

and δ is defined by the following transition set:

$$\begin{array}{l} [q, a, q, a, R] \\ [q, b, r, b, R] \\ [q, c, q, c, R] \end{array}$$

$$\begin{array}{l} [r, a, q, a, R] \\ [r, b, s, b, R] \\ [r, c, q, c, R] \end{array}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) M accepts by final state. B is the designated blank symbol.)

- (a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)^* bb (a \cup b \cup c)^*$$

(b) Write a complete formal definition of a Turing machine M_1 that accepts L by halting. If such a Turing machine does not exist, prove it.

Answer: Employ the conversion algorithm to obtain $M_1 = (Q_1, \Sigma, \Gamma, \delta_1, q)$ from M . M_1 is constructed as follows.

$$Q_1 = Q \cup \{z\} = \{q, r, s, z\}$$

δ_1 is obtained by augmenting δ with transitions that take M_1 out of every halting but non-accepting configuration of M to an infinite escape to the right, via the new state z . Hence, δ_1 comprises the following transition set.

$$\begin{aligned} & [q, a, q, a, R] \\ & [q, b, r, b, R] \\ & [q, c, q, c, R] \\ & [q, B, z, B, R] \end{aligned}$$

$$\begin{aligned} & [r, a, q, a, R] \\ & [r, b, s, b, R] \\ & [r, c, q, c, R] \\ & [r, B, z, B, R] \end{aligned}$$

$$\begin{aligned} & [z, a, z, a, R] \\ & [z, b, z, b, R] \\ & [z, c, z, c, R] \\ & [z, B, z, B, R] \end{aligned}$$

(c) Is L a decidable language? Explain your answer.

Answer: Yes—the analysis performed in the answer to part (a) shows that L is regular, but every regular language is decidable.

Problem 890 Let L be the language of strings over alphabet $\{a, b\}$ that contain at least three occurrences of letter a .

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

(b) Describe a Turing machine M that decides the following problem:

INPUT: A representation of a Turing machine M .

QUESTION: Is $L(M) = L$?

If such Turing machine does not exist, prove it.

Advice for Answer: The property:

is defined by the regular expression:

$$b^*ab^*ab^*a(a \cup b)^*$$

is non-trivial.

Problem 891 Consider a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$\begin{aligned} Q &= \{q_0, q_1\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{a, b, B\} \end{aligned}$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q_0, a, q_1, B, R] \\ & [q_0, b, q_1, B, R] \\ & [q_1, a, q_0, B, R] \\ & [q_1, b, q_0, B, R] \\ & [q_1, B, q_1, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

(a) Write a complete formal definition of a Turing machine M_1 such that M_1 halts on input η if and only if M does not halt on input η , for all $\eta \in \Sigma^*$. In short:

$$(M(\eta) \searrow) \rightarrow (M_1(\eta) \nearrow)$$

and also:

$$(M(\eta) \nearrow) \rightarrow (M_1(\eta) \searrow)$$

If such Turing machine M_1 does not exist, prove it.

Answer: Observe that M halts on strings of even length. Hence, M_1 halts on strings of odd length.

$$M_1 = (Q, \Sigma, \Gamma, \delta_1, q)$$

such that:

$$\begin{aligned} Q &= \{q_0, q_1\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{a, b, B\} \end{aligned}$$

and δ_1 is defined by the following transition set:

$$\begin{aligned} & [q_0, a, q_1, B, R] \\ & [q_0, b, q_1, B, R] \\ & [q_1, a, q_0, B, R] \\ & [q_1, b, q_0, B, R] \\ & [q_0, B, q_0, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

(b) Is the language accepted by M recursive? Explain your answer.

(c) Is the language accepted by M recursively enumerable? Explain your answer.

Problem 892 Consider a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$\begin{aligned} Q &= \{q_0, q_1\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{a, b, c, B\} \end{aligned}$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q_0, a, q_0, a, R] \\ & [q_0, b, q_0, b, R] \\ & [q_0, c, q_1, c, R] \\ & [q_0, B, q_0, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

(a) Write a complete formal definition of a Turing machine M_1 such that M_1 accepts η if M halts on η , and

M_1 rejects η if M does not halt on η , for all $\eta \in \Sigma^*$. In short:

$$(M(\eta) \searrow) \rightarrow (M_1(\eta) \searrow \text{ and accepts})$$

and also:

$$(M(\eta) \nearrow) \rightarrow (M_1(\eta) \searrow \text{ and rejects})$$

If such Turing machine M_1 does not exist, prove it.

Answer: Observe that M halts on those input strings that contain c .

$$M_1 = (Q, \Sigma, \Gamma, \delta_1, q, F)$$

such that:

$$\begin{aligned} Q &= \{q_0, q_1\} \\ \Sigma &= \{a, b, c\}, \quad \Gamma = \{a, b, c, B\} \\ F &= \{q_1\} \end{aligned}$$

and δ_1 is defined by the following transition set:

$$\begin{aligned} [q_0, a, q_0, a, R] \\ [q_0, b, q_0, b, R] \\ [q_0, c, q_1, c, R] \end{aligned}$$

(where B is the designated blank symbol.)

(b) Is the language accepted by M recursive? Explain your answer.

(c) Is the language accepted by M recursively enumerable? Explain your answer.

Problem 893 Let L_1 be a language over alphabet $\{a, b, c, d, f\}$ defined as follows:

$$L_1 = \{d^k a^{2i+j} b^i c^j f^k \mid i, j, k \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar G that generates L_1 . If such grammar does not exist, prove it.

(b) Write a complete formal definition of a Turing machine M_1 such that M_1 accepts η if η represents a Turing machine that accepts L_1 , and M_1 rejects η if η does not represent a Turing machine that accepts L_1 , for all $\eta \in \Sigma^*$. In short:

$$(L(\eta) = L_1) \rightarrow (M_1(\eta) \searrow \text{ and accepts})$$

and also:

$$(\neg(L(\eta) = L_1)) \rightarrow (M_1(\eta) \searrow \text{ and rejects})$$

If such Turing machine M_1 does not exist, prove it.

Problem 894 Let:

$$L = \{(R(M), n) \mid M \text{ halts on blank tape after } \leq n \text{ steps}\}$$

where $R(M)$ is a representation of Turing machine M and n is a natural number. Describe a Turing machine M' that accepts L . If such M' does not exist, explain why.

Answer: M' simulates M on a blank-tape input and counts the simulated steps. If M halts before n steps are counted, M' accepts.

Problem 895 Let L be a non-recursive language over the English alphabet $\{a, b, c, \dots, x, y, z\}$, accepted by a Turing machine M . Describe a Turing machine M' such that M' writes *error* on its tape and halts if and only if its input string does not belong to L . If such M' does not exist, explain why.

Answer: M' does not exist, lest it would accept \overline{L} . This would mean that \overline{L} is recursively enumerable. However, since L is not recursive, its complement \overline{L} cannot be recursively enumerable.

Problem 896 Let L be a non-recursive language, accepted by a Turing machine M , and let k be a natural number. Describe a Turing machine M' , such that on input w , M' writes *error* on its tape and halts if and only if M does not accept w within the first k computation steps. If such M' does not exist, explain why.

Answer: M' simulates M on input w , and counts the simulated steps. If and when the number of steps reaches k , M' does as follows. If M has not (yet) accepted w or if M has rejected w then M' writes *error* and halts. If M accepts w before the number of steps reaches k then M' diverges.

Problem 897 Let L be the language accepted by the pushdown automaton:

$$M_p = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r, s, t, v\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A, B\} \\ F &= \{q\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, r, A] \\ [r, a, \lambda, s, BB] \\ [s, b, B, t, \lambda] \\ [t, c, B, v, \lambda] \\ [v, b, A, q, \lambda] \end{aligned}$$

(Recall that M_p is defined so as to accept by final state and empty stack.)

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$(aabcb)^*$$

(b) Write a complete formal definition of a Turing machine:

$$M_T = (Q_T, \Sigma, \Gamma_T, \delta_T, q, F_T)$$

such that M_T on input w operates as follows:

If w belongs to the set of strings accepted by M_p , then M_T halts and accepts.

If w does not belong to the set of strings accepted by M_p , then M_T halts and rejects.

Answer: $Q_T = \{q, r, s, t, v, f\}$; $\Sigma = \{a, b, c\}$; $\Gamma_T = \{B, a, b, c\}$; $F_T = \{f\}$; and δ_T is defined by the following transition set:

$$\begin{aligned} & [q, a, r, B, R] \\ & [r, a, s, B, R] \\ & [s, b, t, B, R] \\ & [t, c, v, B, R] \\ & [v, b, q, B, R] \\ & [q, B, f, B, R] \end{aligned}$$

Problem 898 Let L be the set of all strings over alphabet $\{a, b, c\}$ that do not contain bcc as a substring.

(a) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 344.

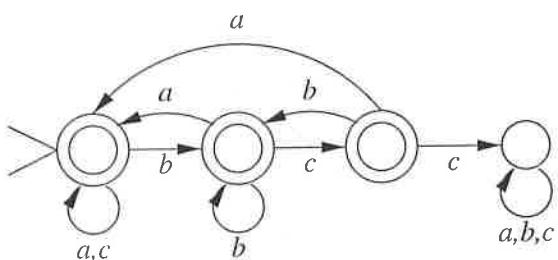


Figure 344:

(b) Write a complete formal definition of a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q_0, F)$$

such that M on input w operates as follows:

If w is a representation of a Turing machine that accepts L , then M halts and accepts.

If w is not a representation of a Turing machine that accepts L , then M halts and rejects.

If such Turing machine does not exist, prove it.

Answer: This Turing machine does not exist. If M existed, it would decide the following language:

$$\{M \mid L(M) = L\}$$

The property is equal to L is non-trivial. Note that L satisfies this property, whereas, say, a^* does not satisfy this property. By Rice's theorem, the set of representations of Turing machines that accept the languages that satisfy any non-trivial property is undecidable.

Problem 899 Let:

$$L = \{a^i b^j c^k \mid j = 2i \text{ and } i = 4k, i, j, k \geq 0\}$$

(a) Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, prove it.

Answer: Such grammar does not exist, because:

$$L = \{a^{4k} b^{8k} c^k \mid k \geq 0\}$$

which is not a context-free language.

To prove that L is not context-free, assume the opposite—that the Pumping Lemma holds for the language.

Let ν be the constant as in the Pumping Lemma for L . Let $w = a^{4m} b^{8m} c^m$, such that $m > \nu$. Let $w = uvxyz$ be a decomposition of w , where vy is the pumping part.

Observe that every substring of w that contains all the three letters (a, b, c) must contain the entire run of b 's of length $8m$, plus some a 's before this run of b 's and some c 's after it. Hence, every substring of w that contains all the three letters has length at least $8m + 2$. By the Lemma, it must be that $|vxy| < \nu < m < 8m + 2$. Thus, the pumping part vy is too short to contain all the three letters—it lacks at least one of them, say letter ξ , for some $\xi \in \{a, b, c\}$.

After the pumping is admitted, any word of the form $uv^i xy^i z$, for $i > 1$, is claimed to be in L . However, since the other two letters are pumped, while ξ is not pumped, such a word has a surplus of the other two letters, relative to ξ , thereby violating the pattern $a^{4k} b^{8k} c^k$ —a contradiction.

(b) Write a complete formal definition of a Turing machine M such that M halts on every input and accepts exactly those strings which are Turing machines that accept L . In short:

$$(L(\tau) = L) \rightarrow (M(\tau) \downarrow \text{and accepts})$$

and also:

$$(L(\tau) \neq L) \rightarrow (M(\tau) \downarrow \text{and rejects})$$

If such Turing machine does not exist, prove it.

Answer: Such Turing machine does not exist. The property "is equal to L " is non-trivial. By Rice's Theorem, there is no algorithm to decide which Turing machines accept a language that has a given non-trivial property.

Problem 900 Consider a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$Q = \{q, r_0, r_1, r_2, s\};$$

$$\Sigma = \{a, b\};$$

$$\Gamma = \{B, a, b\};$$

$$F = \{s\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, q, a, R] \\ & [q, b, q, b, R] \\ & [q, B, r_0, B, L] \\ & [r_0, b, r_1, b, L] \\ & [r_1, b, r_2, b, L] \\ & [r_2, a, s, b, R] \end{aligned}$$

Write a complete formal definition of a Turing machine M_1 such that M_1 halts on τ if M rejects τ , and M_1 does not halt on τ if M accepts τ , for all $\tau \in \Sigma^*$. In short:

$$(M(\tau) \text{ accepts}) \rightarrow (M_1(\tau) \nearrow)$$

and also:

$$(M(\tau) \text{ rejects}) \rightarrow (M_1(\tau) \searrow)$$

If such a Turing machine M_1 does not exist, prove it.

Answer: To obtain M_1 , we modify M so as to let M_1 diverge from all accepting configurations of M . These happen to be all the configurations whose state is s .

$$M_1 = (Q, \Sigma, \Gamma, \delta_1, q)$$

where:

$$Q = \{q, r_0, r_1, r_2, s\};$$

$$\Sigma = \{a, b\};$$

$$\Gamma = \{B, a, b\};$$

and δ_1 is defined by the following transition set:

$$\begin{aligned} & [q, a, q, a, R] \\ & [q, b, q, b, R] \\ & [q, B, r_0, B, L] \\ & [r_0, b, r_1, b, L] \\ & [r_1, b, r_2, b, L] \\ & [r_2, a, s, b, R] \\ & [s, a, s, a, R] \\ & [s, b, s, b, R] \\ & [s, B, s, B, R] \end{aligned}$$

Problem 901 Consider a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$Q = \{q, s, s_1, s_2, t\};$$

$$\Sigma = \{a, b, c\};$$

$$\Gamma = \{B, a, b, c\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, q, a, R] \\ & [q, b, q, b, R] \\ & [q, c, q, c, R] \\ & [q, B, s, B, L] \\ & [s, a, s_1, B, L] \\ & [s_1, b, s_2, B, L] \\ & [s_2, c, t, B, R] \\ & [t, B, t, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

Write a complete formal definition of a Turing machine M_1 such that M_1 rejects τ if τ is a Turing machine which accepts the same language as the Turing machine M , and M_1 accepts τ if τ is a Turing machine which does not accept the same language as the Turing machine M . In short:

$$(L(\tau) = L(M)) \rightarrow (M_1(\tau) \text{ rejects})$$

and also:

$$(L(\tau) \neq L(M)) \rightarrow (M_1(\tau) \text{ accepts})$$

If such a Turing machine M_1 does not exist, prove it.

Answer: The Turing machine M_1 does not exist, since it would decide if an arbitrary Turing machine τ accepts a language equal to $L(M)$. The property “is equal to $L(M)$ ” is non-trivial—hence, by Rice’s Theorem, it is undecidable whether it holds for the language accepted by an arbitrary Turing machine.

Problem 902 Consider a Turing machine

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$Q = \{q, r_0, r_1, r_2, s\};$$

$$\Sigma = \{a, b\};$$

$$\Gamma = \{B, a, b\};$$

$$F = \{s\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, q, a, R] \\ & [q, b, q, b, R] \\ & [q, B, r_0, B, L] \\ & [r_0, b, r_1, b, L] \\ & [r_1, b, r_2, b, L] \\ & [r_2, a, s, b, R] \end{aligned}$$

Draw a state transition graph of a finite automaton M_1 such that M_1 accepts τ if M accepts τ , and M_1 rejects τ if M rejects τ , for all $\tau \in \Sigma^*$. In short:

$$(M(\tau) \text{ accepts}) \rightarrow (M_1(\tau) \text{ accepts})$$

and also:

$$(M(\tau) \text{ rejects}) \rightarrow (M_1(\tau) \text{ rejects})$$

If such a finite automaton M_1 does not exist, prove it.

Answer: The machine M halts on every input—hence, the construction is possible. See Figure 345.

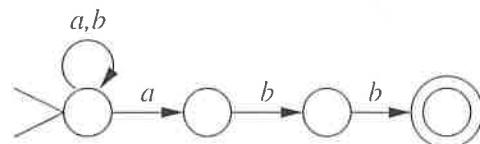


Figure 345:

Problem 903 Consider a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$Q = \{q, r_0, r_1, r_2, s\};$$

$$\Sigma = \{a, b, c\};$$

$$\Gamma = \{B, a, b, c\};$$

$$F = \{s\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, q, a, R] \\ & [q, b, q, b, R] \\ & [q, c, q, c, R] \\ & [q, B, r_0, B, L] \\ & [r_0, a, r_1, a, L] \\ & [r_1, b, r_2, b, L] \\ & [r_2, c, s, c, R] \end{aligned}$$

Draw a state transition graph of a finite automaton M_1 such that M_1 accepts τ if M accepts τ , and M_1 rejects τ if M rejects τ , for all $\tau \in \Sigma^*$. In short:

$$(M(\tau) \text{ accepts}) \rightarrow (M_1(\tau) \text{ accepts})$$

and also:

$$(M(\tau) \text{ rejects}) \rightarrow (M_1(\tau) \text{ rejects})$$

If such finite automaton M_1 does not exist, prove it.

Answer: See Figure 346.

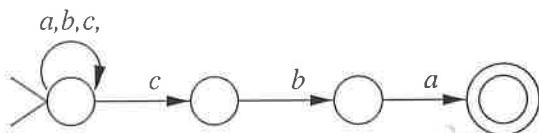


Figure 346:

Problem 904 Consider a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$Q = \{q, s, s_1, s_2, t\};$$

$$\Sigma = \{a, b, c\};$$

$$\Gamma = \{B, a, b, c\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, q, a, R] \\ & [q, b, q, b, R] \\ & [q, c, q, c, R] \\ & [q, B, s, B, L] \\ & [s, a, s_1, B, L] \\ & [s_1, b, s_2, B, L] \\ & [s_2, c, t, B, R] \\ & [t, B, t, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

Draw a state transition graph of a finite automaton M_1 such that M_1 rejects τ if τ is a string on which the Turing machine M halts, and M_1 accepts τ if τ is a string on which the Turing machine M does not halt, for all $\tau \in \Sigma^*$. In short:

$$(M(\tau) \setminus_s) \rightarrow (M_1(\tau) \text{ rejects})$$

and also:

$$(M(\tau) \nearrow) \rightarrow (M_1(\tau) \text{ accepts})$$

If such a finite automaton M_1 does not exist, prove it.

Answer: See Figure 346.

Problem 905 Consider a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$Q = \{q, s, s_1, s_2, t\};$$

$$\Sigma = \{a, b, c\};$$

$$\Gamma = \{B, a, b, c\};$$

and δ is defined by the following transition set (as in Problem 904):

$$\begin{aligned} & [q, a, q, a, R] \\ & [q, b, q, b, R] \\ & [q, c, q, c, R] \\ & [q, B, s, B, L] \\ & [s, a, s_1, B, L] \\ & [s_1, b, s_2, B, L] \\ & [s_2, c, t, B, R] \\ & [t, B, t, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

Write a complete formal definition of a Turing machine M_1 such that M_1 rejects τ if τ is a Turing machine which accepts the complement of the language accepted by the Turing machine M , and M_1 accepts τ if τ is a Turing machine which does not accept the complement of the language accepted by the Turing machine M . In short:

$$(L(\tau) = \overline{L(M)}) \rightarrow (M_1(\tau) \text{ rejects})$$

and also:

$$(L(\tau) \neq \overline{L(M)}) \rightarrow (M_1(\tau) \text{ accepts})$$

If such a Turing machine M_1 does not exist, prove it.

Answer: The Turing machine M_1 does not exist, since it would decide if an arbitrary Turing machine τ accepts a language equal to $\overline{L(M)}$. The property “is equal to the complement of $L(M)$ ” is non-trivial—hence, by Rice’s Theorem, it is undecidable whether it holds for the language accepted by an arbitrary Turing machine.

Problem 906 Consider a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$Q = \{q, r_0, r_1, r_2, s\};$$

$$\Sigma = \{a, b\};$$

$$\Gamma = \{B, a, b\};$$

$$F = \{q\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, b, r_1, b, R] \\ & [r_1, b, r_2, b, R] \\ & [r_2, b, s, b, R] \\ & [s, a, s, a, R] \\ & [s, b, s, b, R] \\ & [s, B, s, B, R] \end{aligned}$$

Write a complete formal definition of a Turing machine M_1 such that M_1 halts on τ exactly when M does not halt on τ . In short:

$$(M(\tau) \searrow) \rightarrow (M_1(\tau) \nearrow)$$

and also:

$$(M(\tau) \nearrow) \rightarrow (M_1(\tau) \searrow)$$

If such a Turing machine M_1 does not exist, prove it.

Answer: Observe that M diverges on all input strings of the form:

$$bbb(a \cup b)^*$$

whence the construction for M_1 :

$$M_1 = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$Q = \{q, r_0, r_1, r_2, s, e\};$$

$$\Sigma = \{a, b\};$$

$$\Gamma = \{B, a, b\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, b, r_1, b, R] \\ & [q, a, e, a, R] \\ & [q, B, e, B, R] \\ & [r_1, b, r_2, b, R] \\ & [r_1, a, e, a, R] \\ & [r_1, B, e, B, R] \\ & [r_2, b, s, b, R] \\ & [r_2, a, e, b, R] \\ & [r_2, B, e, B, R] \\ & [e, a, e, a, R] \\ & [e, b, e, b, R] \\ & [e, B, e, B, R] \end{aligned}$$

Problem 907 Consider a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$Q = \{q, s\};$$

$$\Sigma = \{a, b, c\};$$

$$\Gamma = \{B, a, b, c\};$$

$$F = \{q\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, s, a, R] \\ & [q, b, s, b, R] \\ & [q, c, q, c, R] \\ & [s, a, s, a, R] \\ & [s, b, s, b, R] \\ & [s, c, s, c, R] \\ & [s, B, s, B, R] \end{aligned}$$

Draw a state transition graph of a finite automaton M_1 such that M_1 accepts τ if M halts on τ , and M_1 rejects τ if M does not halt on τ , for all $\tau \in \Sigma^*$. In short:

$$(M(\tau) \searrow) \rightarrow (M_1(\tau) \text{ accepts})$$

and also:

$$(M(\tau) \nearrow) \rightarrow (M_1(\tau) \text{ rejects})$$

If such finite automaton M_1 does not exist, prove it.

Advice for Answer: M diverges on the set of input strings defined by the regular expression:

$$c^*$$

Problem 908 Consider a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$Q = \{q, s, t\};$$

$$\Sigma = \{a, b\};$$

$$\Gamma = \{B, a, b\};$$

$$F = \{s\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, s, a, R] \\ & [q, b, q, b, R] \\ & [s, a, t, a, R] \\ & [s, b, t, b, R] \\ & [t, a, t, a, R] \\ & [t, b, t, b, R] \\ & [t, B, t, B, R] \end{aligned}$$

Write a complete formal definition of a Turing machine M_1 such that M_1 halts on every input and accepts exactly those strings on which M does not halt. In short:

$$(M(\tau) \nearrow) \rightarrow (M_1(\tau) \text{ accepts})$$

and also:

$$(\neg(M(\tau) \nearrow)) \rightarrow (M_1(\tau) \text{ rejects})$$

If such Turing machine M_1 does not exist, prove it.

Advice for Answer: M diverges on the set of input strings defined by the regular expression:

$$b^*a(a \cup b)(a \cup b)^*$$

Problem 909 Consider a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$Q = \{q, s, t\};$$

$$\Sigma = \{a, b, c\};$$

$$\Gamma = \{B, a, b, c\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, s, a, R] \\ & [q, b, q, b, R] \\ & [q, c, q, c, R] \\ & [s, a, s, a, R] \\ & [s, b, s, b, R] \\ & [s, c, t, c, R] \\ & [t, a, t, a, R] \\ & [t, b, t, b, R] \\ & [t, c, t, c, R] \\ & [t, B, t, B, R] \end{aligned}$$

- (a) Write a complete formal definition of a finite automaton M' , such that M' accepts exactly those strings on which M diverges. If such finite automaton does not exist, prove it.

Advice for Answer: Language accepted by M' :

$$(b \cup c)^* a (a \cup b)^* c (a \cup b \cup c)^*$$

- Problem 910** (a) Write a complete formal definition of a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q_0, \{q_f\})$$

with input alphabet $\{a, b\}$, such that M computes function $f : \Sigma^* \rightarrow \Sigma^*$, defined as follows:

$$f(w) = \begin{cases} a & \text{if the length of } w \text{ is even} \\ b & \text{otherwise} \end{cases}$$

If such machine does not exist, explain why.

Answer: (NOTE: This solution assumes the conventions stated in Chapter 12 of Sudkamp's book.)

$$Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_f\};$$

$$\Sigma = \{a, b\};$$

$$\Gamma = \{a, b, B\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q_0, B, q_1, B, R], [q_1, B, q_f, a, L], \\ & [q_1, a, q_3, a, R], [q_1, b, q_3, a, R], \\ & [q_3, a, q_2, B, R], [q_3, b, q_2, B, R], \\ & [q_2, a, q_3, B, R], [q_2, b, q_3, B, R], \\ & [q_2, B, q_4, B, L], [q_4, B, q_4, B, L], [q_4, a, q_f, a, L], \\ & [q_3, B, q_5, B, L], [q_5, B, q_5, B, L], [q_5, a, q_f, b, L], \end{aligned}$$

- (b) Describe briefly the general idea of your solution of part (a), and explain briefly the overall structure of the Turing machine which you constructed in your answer to part (a).

Answer: Step through the input, erasing all but the first symbol (if any), alternating between two states as the parity of the input-length toggles. On the way back, depending on the parity, write a or b .

- Problem 911** List 4 different non-trivial properties of recursively enumerable languages over alphabet $\Sigma = \{0, 1\}$. For each of these 4 properties, show that it is non-trivial. If such 4 properties do not exist, prove it.

- Problem 912** Let L_2 be the language accepted (by final state) by the Turing machine M , defined as follows:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$Q = \{q, r, s, t, v, x\};$$

$$\Sigma = \{a, b, c\};$$

$$\Gamma = \{B, a, b, c\};$$

$$F = \{x\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, q, a, R] \\ & [q, b, q, b, R] \\ & [q, c, q, c, R] \\ & [q, B, r, B, L] \end{aligned}$$

$$\begin{aligned} & [r, c, s, c, L] \\ & [s, c, t, c, L] \\ & [t, b, v, b, L] \\ & [v, a, x, a, R] \end{aligned}$$

(where B is the designated blank symbol.)

- (a) Write a complete formal definition of a context-free grammar G that generates L_2 . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c\}$, $V = \{S, A\}$, and the rule set P is:

$$\begin{aligned} S &\rightarrow Aabcc \\ A &\rightarrow AA \mid a \mid b \mid c \mid \lambda \end{aligned}$$

- (b) State a trivial property of language L_2 that a^* does not have. Explain why this property is trivial, and show that L_2 indeed has it, while a^* does not. If such property does not exist, prove it.

Answer: Such property does not exist. By definition, a property is trivial if all recursively enumerable languages satisfy it, or if there are no recursively enumerable languages that satisfy it. Hence, every trivial property either holds for both of a^* and L_2 or for none of them.

- (c) State a non-trivial property of language L_2 that a^* does not have. Explain why this property is non-trivial, and show that L_2 indeed has it, while a^* does not. If such property does not exist, prove it.

Answer: The property:

$$\text{every string ends with substring } abcc$$

is non-trivial, because there exists a recursively enumerable language, L_2 , that has it, while there exists another recursively enumerable language, a^* , that does not have it. To confirm this, inspect the grammar G , constructed in part(a), which witnesses that every string in L_2 ends with substring $abcc$. At the same time, not a single string of a^* contains any c 's.

- Problem 913** Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$Q = \{q, r\}$$

$$\Sigma = \{a, b, c\}$$

$$\Gamma = \{A, B\}$$

$$F = \{r\}$$

and the transition set δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, q, A] \\ & [q, b, \lambda, q, B] \\ & [q, \lambda, \lambda, r, \lambda] \\ & [r, a, A, r, \lambda] \\ & [r, b, B, r, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

- (a) Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, explain why.

Answer: L is the language of even-length palindromes containing a and b . $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S\}$, and P is:

$$S \rightarrow aSa \mid bSb \mid \lambda$$

- (b) State three different non-trivial properties that language L has. For each of these properties, show that L indeed has it, and explain why it is non-trivial. If such properties do not exist, explain why.

Answer: There are infinitely many correct answers. Some simple ones are:

1. is context-free.

The property is non-trivial since there are languages that are not context-free. A context-free grammar for L is given in part (a).

2. is infinite.

The property is non-trivial since there are languages that are not infinite, for example $\{a, aa, aaa\}$. To verify that L is infinite, observe that its grammar admits derivations of any length, where each step in any derivation adds more terminals to the sentential form.

3. all strings are of even length.

The property is non-trivial since there are languages that contain strings of odd length, for example a^* . To verify that every string in L has even length, observe that each step in any derivation adds more terminals to the sentential form.

Problem 914 Let L_6 be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, p, t\}$; $\Sigma = \{a, b, c, g, h, f\}$; $\Gamma = \{A, D, I, N, R, Y\}$; $F = \{t\}$ and δ is defined by the following transition set:

$$\begin{array}{lll} [q, f, \lambda, q, RAINY] & [t, a, A, t, \lambda] & [t, g, Y, t, \lambda] \\ [p, a, \lambda, p, DAY] & [t, b, R, t, \lambda] & [t, h, N, t, \lambda] \\ [q, \lambda, \lambda, p, \lambda] & [t, c, D, t, \lambda] & [t, f, I, t, \lambda] \\ [p, \lambda, \lambda, t, \lambda] & & \end{array}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on

the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

- (a) Write a complete formal definition of a context-free grammar that generates L_6 . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, g, h, f\}$, $V = \{S\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow fSghfab \mid A \\ A &\rightarrow aAgac \mid \lambda \end{aligned}$$

Let \mathcal{R} be a property of recursively enumerable languages, which is true for exactly those recursively enumerable languages whose every element string has a length greater than 6:

$$\mathcal{R}(X) \iff (\forall w \in X) (|w| > 6)$$

- (b) Is $\mathcal{R}(\emptyset)$ true? Prove your answer.

Answer: Yes.

$$(\forall w \in \emptyset)(P(w))$$

is true for every predicate $P(w)$, including $|w| > 6$, since $w \in \emptyset$ is always false.

- (c) Is $\mathcal{R}(L_6)$ true? Prove your answer.

Answer: No. By inspection of the code, we see that $\lambda \in L_6$, but $|\lambda| = 0 < 6$. Likewise, $agac \in L_6$, but $|agac| = 4 < 6$.

- (d) Is property \mathcal{R} trivial? Prove your answer.

Answer: No. We just proved that it is true for \emptyset and false for L_6 .

Problem 915 Let L_6 be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, p, s, t\}$; $\Sigma = \{a, b, c, g, h, f\}$;

$\Gamma = \{E, M, O, P, T, Y\}$; $F = \{s\}$ and δ is defined by the following transition set:

$$\begin{array}{lll} [q, g, \lambda, q, EMPTY] & [t, a, O, t, \lambda] & [s, a, O, s, \lambda] \\ [p, a, \lambda, p, POT] & [t, b, T, t, \lambda] & [s, b, T, s, \lambda] \\ [q, \lambda, \lambda, t, \lambda] & [t, c, E, t, \lambda] & [s, c, E, s, \lambda] \\ [t, \lambda, \lambda, p, \lambda] & [t, g, Y, t, \lambda] & [s, g, Y, s, \lambda] \\ [p, \lambda, \lambda, s, \lambda] & [t, h, M, t, \lambda] & [s, h, M, s, \lambda] \\ & [t, f, P, t, \lambda] & [s, f, P, s, \lambda] \end{array}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

- (a) Write a complete formal definition of a context-free grammar that generates L_6 . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, g, h, f\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow gAgbfhc \mid \lambda \\ B &\rightarrow aBba \mid \lambda \end{aligned}$$

Let \mathcal{R} be a property of recursively enumerable languages, which is true for exactly those recursively enumerable languages that contain string $acac$ as element:

$$\mathcal{R}(X) \iff (acac \in X)$$

(b) Is $\mathcal{R}(\emptyset)$ true? Prove your answer.

Answer: No. $\mathcal{R}(\emptyset)$ is false since $acac \notin \emptyset$, because \emptyset has no elements.

(c) Is $\mathcal{R}(L_6)$ true? Prove your answer.

Answer: No. $\mathcal{R}(L_6)$ is false since $acac \notin L_6$: by inspection of the code we confirm that no string in L_6 may contain a unless it also contains b and f .

(d) Is property \mathcal{R} nontrivial? Prove your answer.

Answer: Yes. It is false for \emptyset and L_6 (parts (b) and (c)) but true for $(a \cup c)^*$ since $acac \in (a \cup c)^*$.

Problem 916 Let L be the language accepted by the pushdown automaton $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, p, v\}$, $\Sigma = \{a, b, c, d\}$, $\Gamma = \{T, E, S\}$, $F = \{v\}$ and the transition function δ is defined as follows:

$$\begin{array}{ll} [q, a, \lambda, p, TEST] & [v, a, \lambda, v, \lambda] \\ [p, a, \lambda, p, \lambda] & [v, b, T, v, \lambda] \\ [p, \lambda, \lambda, v, \lambda] & [v, c, E, v, \lambda] \\ & [v, d, S, v, \lambda] \end{array}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write 6 distinct strings that belong to L . If such strings do not exist, state it and prove it.

Advice for Answer: See answer to part (b).

(b) Write a regular expression that defines L . If such a regular expression does not exist, state it and prove it.

Answer:

$$a a^* b a^* d a^* c a^* b a^*$$

(c) Is L decidable? Explain your answer.

Answer: Yes. L is context free, and all context free languages are decidable.

(d) For a recursively enumerable language G , let the property $P_1(G)$ be true if and only if $L \cap G$ is infinite.

Is P_1 a non-trivial property of recursively enumerable languages? Explain your answer.

Answer: Yes. P_1 is non-trivial because it is true for say L , since $L \cap L = L$ is infinite, but false for say \emptyset , since $L \cap \emptyset = \emptyset$ is finite.

(e) State the value of $P_1(\emptyset)$.

Answer: False.

(f) State the value of $P_1(a^*)$.

Answer: False.

(g) Explain how to construct an algorithm that solves the following problem:

INPUT: String w over Σ and a pushdown automaton D ;

OUTPUT: yes if w represents a Turing Machine that accepts a set of exactly those strings that are accepted by D ;

no otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist. If it existed, it would decide the set of Turing Machines whose languages have the non-trivial property

is equal to $L(D)$

which is impossible by Rice's Theorem. This property is non-trivial because it is true for $L(D)$ and false for any other language, say for $\overline{L(D)}$.

Problem 917 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where:

$Q = \{q, s, t, v\}$, $\Sigma = \{a, b, c, d, e, f\}$,

$\Gamma = \{A, B, K, D, E\}$, $F = \{s\}$ and the transition function δ is defined as follows:

$$\begin{array}{ll} [q, a, \lambda, q, A] & [s, a, A, s, \lambda] \\ [q, a, \lambda, q, E] & [s, b, B, s, \lambda] \\ [q, b, \lambda, q, B] & [s, c, K, s, \lambda] \\ [q, b, \lambda, q, K] & [s, d, D, t, \lambda] \\ [q, d, \lambda, q, D] & [s, e, E, s, \lambda] \\ [q, \lambda, \lambda, s, \lambda] & [t, f, \lambda, v, \lambda] \\ & [v, f, \lambda, s, \lambda] \end{array}$$

(Recall that M is defined so as to accept by final state and empty stack.

(a) List 6 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer:

$$aa, ac, abbe, abddffca, babaccab, dadbcdffedff$$

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, e, f\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSa \mid aSc \mid bSb \mid bSc \mid dSdf \mid \lambda$$

(c) State one trivial property of the language L , such that a^*b^* does not have this property. Explain carefully

why this property is trivial, and prove that L indeed has it, while a^*b^* does not. If such a property does not exist, state it, and explain why it is so.

Answer: Such a property does not exist—if L had this property but a^*b^* did not have it, then the property would by definition be non-trivial.

(d) State one non-trivial property of the language L , such that a^*b^* does not have this property. Explain carefully why this property is non-trivial, and prove that L indeed has it, while a^*b^* does not. If such a property does not exist, state it, and explain why it is so.

Answer: One of the infinitely many properties which answer to these requirements is:

contains at least one word which contains c .

The property is non-trivial, since at least one language, in this case L , has this property while at least one language, in this case a^*b^* , does not have this property. To see that the property is true for L , inspect the grammar constructed in the part (b). To see that it is false for a^*b^* , observe that all strings of the latter contain no letters other than a and b .

Problem 918 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where:

$Q = \{q, s, t, v\}$, $\Sigma = \{a, b, c, d, e, f\}$,
 $\Gamma = \{A, B, K, D, E\}$, $F = \{s\}$ and the transition function δ is defined as follows:

$[q, a, \lambda, q, A]$	$[s, a, A, s, \lambda]$
$[q, a, \lambda, q, E]$	$[s, b, B, s, \lambda]$
$[q, b, \lambda, q, B]$	$[s, c, K, s, \lambda]$
$[q, b, \lambda, q, K]$	$[s, d, D, t, \lambda]$
$[q, d, \lambda, q, D]$	$[s, e, E, s, \lambda]$
$[q, \lambda, \lambda, s, \lambda]$	$[t, f, \lambda, v, \lambda]$
	$[v, f, \lambda, s, \lambda]$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) List 6 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer:

$aa, ac, abbe, abddffca, babaccab, dadbcdffedff$

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, e, f\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSa \mid aSe \mid bSb \mid bSc \mid dSd \mid \lambda$$

(c) State one trivial property of the language L , such that a^*b^* does not have this property. Explain carefully why this property is trivial, and prove that L indeed has it, while a^*b^* does not. If such a property does not exist, state it, and explain why it is so.

Answer: Such a property does not exist—if L had this property but a^*b^* did not have it, then the property would by definition be non-trivial.

(d) State one non-trivial property of the language L , such that a^*b^* does not have this property. Explain carefully why this property is non-trivial, and prove that L indeed has it, while a^*b^* does not. If such a property does not exist, state it, and explain why it is so.

Answer: One of the infinitely many properties which answer to these requirements is:

contains at least one word which contains c .

The property is non-trivial, since at least one language, in this case L , has this property while at least one language, in this case a^*b^* , does not have this property. To see that the property is true for L , inspect the grammar constructed in the part (b). To see that it is false for a^*b^* , observe that all strings of the latter contain no letters other than a and b .

Problem 919 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, t\}$; $\Sigma = \{a, b, c, d\}$; $\Gamma = \{A\}$; $F = \{t\}$; and the transition function δ is defined as follows:

$[q, c, \lambda, t, A]$
$[q, d, \lambda, q, AA]$
$[q, b, \lambda, q, AAA]$
$[t, a, A, t, \lambda]$

(M accepts by final state and empty stack.)

(a) List 5 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: The machine pushes two A 's for every d seen in the input, and pushes three A 's for every b seen in the input. Once the machine sees a c , it pushes one A and transfers to state t . In t , every A is redeemed by an a . See the answer to part (b).

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, $\Sigma = \{a, b, c, d\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow dSaa \mid bSaaa \mid ca$$

(c) Find a trivial property of recursively enumerable languages which is true for the language L but false for the language $(a \cup b \cup c \cup d)^*$. In your answer, state the property, show that it is trivial, show that it is true for L , and show that it is false for $(a \cup b \cup c \cup d)^*$. If such a property does not exist, state it and explain why it does not exist.

Answer: This property does not exist. By definition, a property of recursively enumerable languages is trivial if it has the same value (either true or false) for every recursively enumerable language. Hence, every trivial

property that is true for the language L must also be true for $(a \cup b \cup c \cup d)^*$.

(d) Find a nontrivial property of recursively enumerable languages which is true for the language L but false for the language a^* . In your answer, state the property, show that it is nontrivial, show that it is true for L , and show that it is false for a^* . If such a property does not exist, state it and explain why it does not exist.

Answer: There are infinitely many such properties, and one is: "does not contain any string which begins with a ". This property is nontrivial by definition, since it is true for \emptyset (because \emptyset does not contain any strings) and is false for a^* (because strings of a^* begin with no letter other than a .) To see that this property is true for L , inspect the grammar given in the answer to part (b): the right-hand side of every rule (with S on the left-hand side) begins with a symbol different from a .

Problem 920 L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, t\}$; $\Sigma = \{a, b, c, d\}$; $\Gamma = \{A\}$; $F = \{t\}$; and the transition function δ is defined as follows:

$$\begin{aligned} & [q, b, \lambda, t, AAA] \\ & [t, a, \lambda, t, \lambda] \\ & [t, d, \lambda, t, \lambda] \\ & [t, c, A, t, \lambda] \end{aligned}$$

(M accepts by final state and empty stack.)

(a) List 5 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: The machine sees one b and pushes three A 's, transferring into state t . In t , any number of a 's and d 's are admitted any time, but each A is redeemed by one c . See the answer to part (b).

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: $b(a \cup d)^*c(a \cup d)^*c(a \cup d)^*c(a \cup d)^*$

(c) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 347.

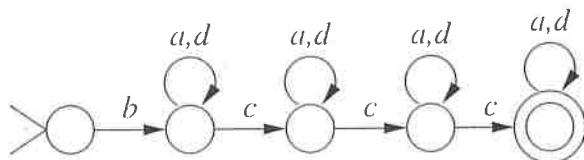


Figure 347:

(d) Describe the construction of a Turing Machine T which halts on every input and also accepts exactly the set of all those strings that do not belong to the language L . In short:

$$(\tau \notin L) \implies (T(\tau) \downarrow \text{ and accept})$$

$$(\tau \in L) \implies (T(\tau) \downarrow \text{ and reject})$$

If such a Turing machine T does not exist, prove it.

Answer: Convert algorithmically the automaton given in the answer to part (c) to a deterministic automaton M_1 , convert M_1 algorithmically to an automaton M_2 that accepts the complement $L_2 = \overline{L(M_1)}$, convert M_2 algorithmically to a Turing Machine T that decides L_2 .

Problem 921 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, s, t, v\} \\ \Sigma &= \{a, b, c, d, e\} \\ \Gamma &= \{B, K, Z\} \\ F &= \{s\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, q, A] \\ & [q, b, \lambda, q, B] \\ & [q, d, \lambda, q, Z] \\ & [q, \lambda, \lambda, s, \lambda] \\ & [s, c, B, s, \lambda] \\ & [s, e, A, s, \lambda] \\ & [s, d, Z, t, \lambda] \\ & [t, d, \lambda, v, \lambda] \\ & [v, d, \lambda, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) List 6 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer:

$$ae, abaece, dddd, adaeddde, \lambda, aaee$$

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, e\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSe \mid bSc \mid dSdd \mid \lambda$$

(c) State one non-trivial property of the language L , such that a^* does not have this property. Explain carefully why this property is non-trivial, and prove that L indeed has it, while a^* does not.

Answer: The property is:

contains at least one string which contains b

To see that L has this property observe that $bc \in L$ (by inspection of the grammar constructed in the answer to part (a).) Since no string of a^* contains any symbol other than a , a^* straightforwardly does not have this property. By definition, since one language(L) has the property while another (a^*) does not have it, the property is non-trivial.

(d) State one trivial property of the language L , such that a^* does not have this property. Explain carefully why this property is trivial, and prove that L indeed has it, while a^* does not.

Answer: This is impossible, by definition. Every trivial property assumes the same value for every recursively enumerable language, and thus cannot be true for L while being false for a^* .

Problem 922 (a) Let L_1 be a language over alphabet $\{a, b, c\}$ defined as follows:

$$L_1 = \{ba^i b^j c^k a \mid i + k = j\}$$

Write a complete formal definition of a context-free grammar G_1 that generates L_1 . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow bAa \\ A &\rightarrow BD \\ B &\rightarrow aBb \mid \lambda \\ D &\rightarrow bDc \mid \lambda \end{aligned}$$

(b) Let L_2 be a language over alphabet $\{a, b, c\}$ defined as follows:

$$L_2 = \{a^i b^j c^k \mid i = j \text{ or } j = k \text{ or } i = k\}$$

Write a complete formal definition of a context-free grammar G_2 that generates L_2 . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B, D, A_{ij}, B_{jk}, K, I, J\}$, and P is:

$$\begin{aligned} S &\rightarrow A \mid B \mid D \\ A &\rightarrow A_{ij}K \\ B &\rightarrow I B_{jk} \\ D &\rightarrow aDc \mid J \\ A_{ij} &\rightarrow aA_{ij}b \mid \lambda \\ B_{jk} &\rightarrow bB_{jk}c \mid \lambda \\ K &\rightarrow cK \mid \lambda \\ I &\rightarrow aI \mid \lambda \\ J &\rightarrow bJ \mid \lambda \end{aligned}$$

(c) State a non-trivial property of language L_1 that L_2 does not have. Explain why this property is non-trivial, and show that L_1 indeed has it, while L_2 does not.

Answer: all strings begin with b .

The property is non-trivial because there exist languages, for example our L_2 , where some strings do not begin with b . For example: $abc \in L_2$.

Problem 923 (a) Let L_1 be the set of all strings over $\{a, b\}$ that do not begin with the substring aaa . Construct a state-transition graph of a finite automaton that accepts L_1 . If such automaton does not exist, prove it.

Answer: See Figure 348.

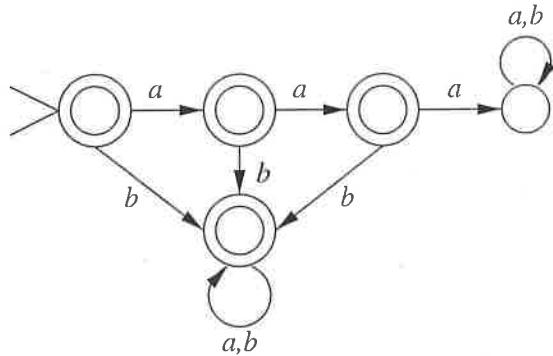


Figure 348:

(b) Let L_2 be the set of all strings over $\{a, b\}$ that contain an even number of substrings ba . Construct a state-transition graph of a finite automaton that accepts L_2 . If such automaton does not exist, prove it.

Answer: See Figure 349.

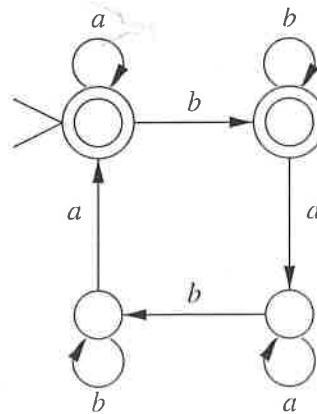


Figure 349:

(c) State a non-trivial property of languages that is true for both L_1 and L_2 . Explain why this property is non-trivial, and show that L_1 and L_2 indeed have it.

Answer: is regular.

The property is non-trivial because there are languages that are not regular. L_1 and L_2 are accepted by finite automata given in parts (a) and (b), respectively.

Problem 924 (a) Let L_1 be the language defined by the regular expression:

$$a(b \cup c)^* aa(bc \cup cd)^*$$

Write a complete formal definition of a context-free grammar that generates L_1 . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aAaaB \\ A &\rightarrow AA \mid \lambda \mid b \mid c \\ B &\rightarrow BB \mid \lambda \mid bc \mid cd \end{aligned}$$

(b) Let L_2 be a language over alphabet $\{a, b, c, d\}$, defined as follows:

$$L_2 = \{a^{2n+1}b^{k+1}c^{3k}d^n \mid k, n \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L_2 . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{a, b, c, d\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aaSd \mid aA \\ A &\rightarrow bAccc \mid b \end{aligned}$$

(c) State a trivial property of language L_1 that L_2 does not have. Explain why this property is trivial, and show that L_1 indeed has it, while L_2 does not. If such property does not exist, prove it.

Answer: Such property does not exist. By definition, a property is trivial if all recursively enumerable languages satisfy it, or if there are no recursively enumerable languages that satisfy it. L_1 and L_2 are two recursively enumerable languages—every trivial property holds for either both of them or none.

Problem 925 (a) Let L_1 be the language defined by the regular expression:

$$a(b \cup c)^* \cup (ab)^*(cd)^*$$

Draw a state-transition graph of a finite automaton that accepts L_1 . If such automaton does not exist, prove it.

Answer: See Figure 350.

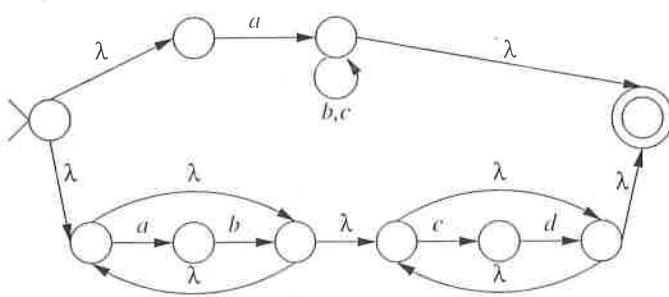


Figure 350:

(b) Let L_2 be the set of all strings over alphabet $\{a, b, c, d\}$ that contain exactly two d 's.

Draw a state-transition graph of a finite automaton that accepts L_2 . If such automaton does not exist, prove it.

Answer: See Figure 351.

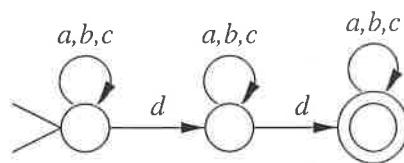


Figure 351:

(c) State a non-trivial property of language L_1 that L_2 does not have. Explain why this property is non-trivial, and show that L_1 indeed has it, while L_2 does not. If such property does not exist, prove it.

Answer: Language L_1 satisfies the property:

no string contains ad as a substring.

This is proved by inspection of the regular expression for L_1 . Those strings from L_1 that are of the form $a(b \cup c)^*$ cannot contain ad because they do not contain d . In those strings from L_1 that are of the form $(ab)^*(cd)^*$ every d follows a c , hence a d cannot follow an a . Language L_2 does not satisfy this property, since some strings that belong to L_2 , say dad , contain ad as a substring. By definition, this property is non-trivial as one recursively enumerable language, precisely L_1 , satisfies it, whereas another recursively enumerable language, precisely L_2 , does not satisfy it.

Problem 926 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{A\} \\ F &= \{r\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, AAA] \\ [q, c, \lambda, r, \lambda] \\ [r, b, A, r, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) Write a complete formal definition of a context-free grammar G that generates L . If such grammar does not exist, prove it.

Answer: The language L accepted by the automaton M is:

$$L = \{a^n cb^{3n} \mid n \geq 0\}$$

This language is generated by the grammar:

$G = (V, \Sigma, P, S)$, where:

$\Sigma = \{a, b, c\}$ is the set of terminals;

$V = \{S\}$ is the set of variables;

S is the start symbol;

and the set of productions P is:

$$S \rightarrow aSbbb \mid c$$

- (b) State precisely three non-trivial properties of the language L such that $a^*b^*c^*$ does not have any of these properties. Explain why these properties are non-trivial, and show that L indeed has them, while $a^*b^*c^*$ does not.

Answer:

1. is not regular.

The property is non-trivial since there are languages that are regular, and those that are not. By the Pumping Lemma, it is easily shown that L is not regular, whereas $a^*b^*c^*$ certainly is regular, since it has a regular expression.

2. all strings contain exactly one c .

The property is non-trivial since there are languages whose strings may contain varying number of occurrences of letter c . By inspection of the grammar for L , we see that all of its strings contain exactly one c . In contrast, for any $i \geq 0$, there exists a string in $a^*b^*c^*$ that contains exactly i occurrences of letter c .

3. does not contain the empty string.

The property is non-trivial since there are languages that contain the empty string, and those that do not. To see that L does not contain λ , recall that all of its strings contain at least one c . In contrast, by inspection of the regular expression $a^*b^*c^*$, we see that it derives λ .

Problem 927 Let L_2 be the language accepted (by final state) by the Turing machine M , defined as follows:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$Q = \{q, r, s, t, v\};$$

$$\Sigma = \{a, b, c\};$$

$$\Gamma = \{B, A, Z\};$$

$$F = \{t\};$$

and δ is defined by the following transition set:

$$\begin{array}{l} [q, a, q, a, R] \\ [q, b, r, b, R] \\ [q, c, q, c, R] \end{array}$$

$$\begin{array}{l} [r, a, r, a, R] \\ [r, b, r, b, R] \\ [r, c, r, c, R] \\ [r, B, s, B, L] \end{array}$$

$$\begin{array}{l} [s, a, t, B, L] \\ [t, c, v, B, L] \end{array}$$

(where B is the designated blank symbol.)

- (a) Write a regular expression that defines L_2 . If such regular expression does not exist, prove it.

Answer: M accepts those strings that contain at least one b and end with a but not with ca , whence the regular expression:

$$(a \cup b \cup c)^* b (a \cup b \cup c)^* aa \cup (a \cup b \cup c)^* ba$$

- (b) Write a complete formal definition of a context-free grammar G that generates L_2 . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b, c\}$, $V = \{S, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow BbBaa \mid Bba \\ B &\rightarrow BB \mid \lambda \mid a \mid b \mid c \end{aligned}$$

- (c) State a non-trivial property of language L_2 that $(abc)^*$ does not have. Explain why this property is non-trivial, and show that L_2 indeed has it, while $(abc)^*$ does not. If such property does not exist, prove it.

Answer: One such property is:

every string ends with a .

L_2 has this property by the analysis done in the answer to part (a). $(abc)^*$ does not have this property because every non-empty element of $(abc)^*$ ends with c . The property is non-trivial because there exists a language that has it, e.g., L_2 , and there exists a language that does not have it, e.g., $(abc)^*$.

Problem 928 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$Q = \{q, t, p, r\}$$

$$\Sigma = \{a, b, c, d\}$$

$$\Gamma = \{A, Z\}$$

$$F = \{p\}$$

and the transition function δ is defined as follows:

$$\begin{array}{l} [q, a, \lambda, t, ZA] \\ [q, b, \lambda, p, \lambda] \\ [t, c, Z, q, \lambda] \\ [t, d, A, t, \lambda] \\ [p, a, \lambda, r, ZAA] \\ [r, d, Z, p, \lambda] \\ [r, c, A, r, \lambda] \end{array}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

- (a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(adc)^* b (accd)^*$$

- (b) State precisely three non-trivial properties of the language L such that $a^*b^*c^*d^*$ does not have any of these properties. Explain why these properties are non-trivial, and show that L indeed has them, while $a^*b^*c^*d^*$ does not.

Answer: By inspection of the two regular expressions, we conclude that L_1 has the following properties, while $a^*b^*c^*d^*$ does not have them. These properties are non-trivial since some recursively enumerable languages have them (such as L), while some others (such as $a^*b^*c^*d^*$) do not have them.

1. every string contains at least one b ;
2. no string ends with c ;
3. no string contains aa as a substring.

Problem 929 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and P is:

$$\begin{aligned} S &\rightarrow AD \mid DB \\ A &\rightarrow Ab \mid d \mid a \\ B &\rightarrow cB \mid cb \\ D &\rightarrow DD \mid \lambda \mid ca \end{aligned}$$

- (a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(d \cup a) b^* (ca)^* \cup (ca)^* c^* cb$$

- (b) State a non-trivial property of the language L that does not hold for the language $(a \cup b \cup c \cup d)^*$. If such a property does not exist, explain why.

Answer: Among the infinitely many such properties, here are a few that are easily observed:

- has an infinite complement;
- does not contain the empty string;
- does not contain any strings that end with c ;
- does not contain any strings that contain aa as a substring;
- has an empty intersection with c^* .

- (c) State a non-trivial property of the language $(a \cup b \cup c \cup d)^*$ that does not hold for the language L . If such a property does not exist, explain why.

Answer: Among the infinitely many such properties, here are a few that are easily observed:

- has an empty complement;
- contains the empty string;
- contains every string that ends with c ;

- some of its strings contain aa as a substring;
- contains c^* as a subset.

- (d) State a trivial property of the language L that does not hold for the language $(a \cup b \cup c \cup d)^*$. If such a property does not exist, explain why.

Answer: Impossible—by definition, a property is trivial if every recursively enumerable language has it or if none has it. Hence, every trivial property that holds true for L also holds true for every other recursively enumerable language, including $(a \cup b \cup c \cup d)^*$.

Problem 930 Dangerous Professor has told her students to write a program that operates as follows:

INPUT: A string x over $\{0, 1\}$.

OUTPUT: **yes** if x is a Turing machine that accepts the language L defined in Problem 789; **no** otherwise.

Explain the algorithm that should be employed by this program, or prove that it does not exist.

Answer: Property “is equal to L ” is a non-trivial property (since L has it, but \emptyset does not). If this algorithm existed, it would decide the set of Turing machines whose languages have this property. By Rice’s Theorem, this is impossible.

Problem 931 Dangerous Professor has told her students to write a Turing Machine T such that:

INPUT: An arbitrary string x over $\{a, b\}$.

OUTPUT: T operates as follows:

$$\begin{aligned} x \in L &\rightarrow (T(x) \nearrow) \\ x \notin L &\rightarrow (T(x) \searrow) \end{aligned}$$

where L is the language defined in Problem 790.

If such a Turing machine does not exist, prove it.

Explain the algorithm that should be employed by this Turing Machine, or prove that it does not exist.

Answer: The construction proceeds as a sequence of five algorithmic conversions, as follows.

1. convert the regular expression obtained in the answer to Problem 790 into a finite automaton, say F ;
2. convert the finite automaton F to a deterministic finite automaton F_1 ;
3. convert the finite automaton F_1 into another deterministic finite automaton F_2 , which accepts the complement \overline{L} ;
4. convert the deterministic finite automaton F_2 into a Turing Machine M_1 that decides \overline{L} ;
5. convert the Turing Machine M_1 into a Turing Machine M_2 that accepts \overline{L} by halting; M_2 is the result.

Problem 932 Dangerous Professor has told her students to write a program that operates as follows:

INPUT: A pair of strings x, y over $\{0, 1\}$;

OUTPUT: **yes** if x is a string accepted by the Turing Machine M defined in Problem 861, and y is a string not accepted by the same Turing Machine M ;
no otherwise.

Explain the algorithm that should be employed by this program, or prove that it does not exist.

Answer: The construction proceeds as a sequence of algorithmic conversions and simulations, as follows.

1. convert the regular expression obtained in the answer to Problem 861 into a finite automaton, say F ;
2. convert the finite automaton F to a deterministic finite automaton F_1 ;
3. convert the finite automaton F_1 into another deterministic finite automaton F_2 , which accepts the complement \overline{L} ;
4. convert the deterministic finite automaton F_1 into a Turing Machine M_1 that decides L ;
5. convert the deterministic finite automaton F_1 into a Turing Machine M_1 that decides \overline{L} ;
6. convert the Turing Machine M_1 into a Turing Machine M_2 that accepts \overline{L} by halting;
7. simulate $M_1(x)$;
8. simulate $M_2(y)$;
9. the result is conjunction of answers returned by M_1 and M_2 in the previous two steps.

Problem 933 Dangerous Professor has told her students to write a program that operates as follows:

INPUT: A pair of strings x, y over $\{0, 1\}$.

OUTPUT: **yes** if x is a Turing machine that accepts y ;
no otherwise.

Explain the algorithm that should be employed by this program, or prove that it does not exist.

Answer: Impossible. If this machine existed, then we could fix argument y to be equal to, say, string 0. The program would then implement the following algorithm.

INPUT: String x over $\{0, 1\}$.

OUTPUT: **yes** if x is a Turing machine whose language contains 0 as a substring;
no otherwise.

However, the property “contains string 0” is non-trivial (since Σ^* has it, but \emptyset does not). If this algorithm existed, it would decide the set of Turing machines whose languages have this property. By Rice’s Theorem, this is impossible.

Problem 934 Let L be the language accepted by the pushdown automaton:

$M = (Q, \Sigma, \Gamma, \delta, s, F)$ where: $Q = \{s, x\}$;

$\Sigma = \{a, b, c, g\}; \Gamma = \{B, E, R, T, U\}; F = \{x\}$; and δ is defined by the following transition set:

$[s, c, \lambda, s, BUTTER]$	$[x, a, U, x, \lambda]$	$[x, g, R, x, \lambda]$
$[s, g, \lambda, s, BEER]$	$[x, b, B, x, \lambda]$	$[x, g, T, x, \lambda]$
$[s, a, \lambda, x, \lambda]$	$[x, c, E, x, \lambda]$	
$[s, b, \lambda, x, T]$		

(Recall that M is defined so as to accept by final state and empty stack.) Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) List four distinct strings that belong to L . If this is impossible, state it and explain why it is so.

Answer:

$a, bg, gagccb, bgggccb$

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, g\}$, $V = \{S\}$, and P is:

$$S \rightarrow cSgaggab \mid gSgccb \mid a \mid bg$$

Let Q be a property of recursively enumerable languages, whose value for any language Y is defined as follows:

$$Q(Y) \iff (Y \cap L \neq \emptyset)$$

(where L is defined at the beginning of this problem.)

(c) Is $Q(\emptyset)$ true? Prove your answer.

Answer: No.

$$Q(\emptyset) \iff (\emptyset \cap L \neq \emptyset)$$

but

$$\emptyset \cap L = \emptyset$$

for every set L , hence $Q(\emptyset)$ is false.

(d) Is $Q(\Sigma^*)$ true? (Σ is defined at the beginning of this problem.) Prove your answer.

Answer: Yes.

$$Q(\Sigma^*) \iff (\Sigma^* \cap L \neq \emptyset)$$

which is true since

$$\Sigma^* \cap L = L$$

and L is not empty (as is shown in the answers to parts (a) and (b).)

(e) Is $Q(a^*)$ true? Prove your answer.

Answer: Yes.

$$Q(a^*) \iff (a^* \cap L \neq \emptyset)$$

Since $a \in a^*$ but also $a \in L$ (as is stated in the answer to part (a)), indeed $a^* \cap L$ is not empty.

(f) Is property Q nontrivial? Prove your answer.

Answer: Yes. As is shown in the answers to parts (a) and (b), $Q(\emptyset)$ is false, but $Q(\Sigma^*)$ is true. Hence, by definition, Q is nontrivial.

Problem 935 Let L be the language accepted by the pushdown automaton:

$M = (Q, \Sigma, \Gamma, \delta, s, F)$ where: $Q = \{s, t, x\}$; $\Sigma = \{a, b, c, g\}$; $\Gamma = \{A, C, E, L, R, T\}$; $F = \{x\}$; and δ is defined by the following transitions set:

$[s, a, \lambda, s, TREACLE]$	$[x, a, A, x, \lambda]$	$[x, c, C, x, \lambda]$
$[t, c, \lambda, t, TART]$	$[x, a, L, x, \lambda]$	$[x, g, T, x, \lambda]$
$[s, b, \lambda, t, \lambda]$	$[x, b, E, x, \lambda]$	
$[t, g, \lambda, x, R]$	$[x, b, R, x, \lambda]$	

(Recall that M is defined so as to accept by final state and empty stack.) Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) List four distinct strings that belong to L . If this is impossible, state it and explain why it is so.

Answer:

$bgb, bcggbag, bccggbagbag, abgbacabbg$

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, g\}$, $V = \{S, W\}$, and P is:

$$\begin{aligned} S &\rightarrow aSbacabbg \mid bW \\ W &\rightarrow cWgbag \mid gb \end{aligned}$$

(c) Explain how to construct an algorithm that solves the following problem:

INPUT: Finite automaton F that accepts a language L_F , and string w ;

OUTPUT: yes if $w \in L_F \cup L$, where L is the language defined at the beginning of this problem;

no otherwise.

If this algorithm does not exist, prove it.

Answer: Simulate $L_F(w)$ and accept if L_F accepts. Otherwise, simulate $M(w)$ and decide exactly as M decides.

(d) Explain how to construct an algorithm that solves the following problem:

INPUT: Turing machine T that decides a language L_T , and string w ;

OUTPUT: yes if $w \in L_T \setminus L$, where L is the language defined at the beginning of this problem; no otherwise.

If this algorithm does not exist, prove it.

Answer: Simulate $T(w)$ (which must terminate since T decides) and reject if T rejects. Otherwise, simulate $M(w)$ and decide the opposite of what M decides.

Problem 936 Consider the Turing machine

$M_8 = (Q, \Sigma, \Gamma, \delta, q, F)$ such that: $Q = \{q, p, t, s, v, x, y\}$; $\Sigma = \{0, 1\}$; $\Gamma = \{B, 0, 1, E, N\}$; $F = \{s\}$; and δ is defined by the following transition set:

$[q, 0, p, E, R]$	$[p, 0, t, E, R]$	$[t, 0, x, 0, L]$
$[q, 1, p, N, R]$	$[p, 1, t, N, R]$	$[t, 1, x, 1, L]$
$[q, B, q, B, R]$	$[p, B, p, B, R]$	$[t, B, x, B, R]$
$[x, N, v, N, L]$	$[v, E, s, E, R]$	
$[x, E, y, E, L]$	$[y, N, s, N, R]$	

(M has an one-way infinite tape (infinite to the right only). B is the designated blank symbol. M_8 accepts by final state.)

Let L_{8A} be the set of strings which M_8 accepts.

Let L_{8R} be the set of strings which M_8 rejects.

Let $L_{8\infty}$ be the set of strings on which M_8 diverges.

(a) Write a regular expression that defines L_{8A} . If such a regular expression does not exist, prove it.

Answer:

$$(01 \cup 10)(0 \cup 1)^*$$

(b) Write a regular expression that defines L_{8R} . If such a regular expression does not exist, prove it.

Answer:

$$(00 \cup 11)(0 \cup 1)^*$$

(c) Write a regular expression that defines $L_{8\infty}$. If such a regular expression does not exist, prove it.

Answer:

$$\lambda \cup 0 \cup 1$$

(d) Explain how to construct an algorithm that solves the following problem:

INPUT: A Turing machine T that decides a language $L(T)$, and a finite set S of input strings;

OUTPUT: yes if $S \subseteq L(T)$; no otherwise.

If this algorithm does not exist, prove it.

Answer: For each element $w \in S$, run $T(w)$. If $T(w)$ returns "yes" for all $w \in S$ then return "yes", otherwise return "no". T will halt and return answer for all $w \in S$, since T decides (and must halt on every input).

Problem 937 Consider the Turing machine $M_7 = (Q, \Sigma, \Gamma, \delta, q)$ such that: $Q = \{q, p, t\}$; $\Sigma = \{0, 1\}$; $\Gamma = \{B, 0, 1\}$; and δ is defined by the following transition set:

$$\begin{array}{ll} [q, 0, q, 0, R] & [p, 0, p, 0, R] \\ [q, 1, p, 1, R] & [p, 1, t, 1, R] \\ [q, B, q, B, R] & [p, B, p, B, R] \end{array}$$

(M_7 has an one-way infinite tape (infinite to the right only.) B is the designated blank symbol.)

Let L_7 be the set of strings on which M_7 halts.

Let $L_{7\infty}$ be the set of strings on which M_7 diverges.

(a) Write a regular expression that defines L_7 . If such a regular expression does not exist, prove it.

Answer:

$$0^* 10^* 1 (0 \cup 1)^*$$

(b) Write a regular expression that defines $L_{7\infty}$. If such a regular expression does not exist, prove it.

Answer:

$$0^* \cup 0^* 10^*$$

(c) Explain how to construct an algorithm that solves the following problem:

INPUT: Turing Machine T_1 that accepts a language $L(T_1)$;

OUTPUT: Turing Machine T_2 that decides the language $L(T_1)$;

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist: some recursively enumerable languages are not decidable—for a given $L(T_1)$, T_2 need not exist.

(d) Explain how to construct an algorithm that solves the following problem:

INPUT: Two Turing Machines: T_1 that decides a language $L(T_1)$ and T_2 that decides a language $L(T_2)$, and an input string w ;

OUTPUT: yes is $w \in L(T_1) \cap L(T_2)$;
no otherwise.

If this algorithm does not exist, prove it.

Answer: Run $T_1(w)$ and $T_2(w)$ and answer “yes” if both T_1 and T_2 answer “yes”; otherwise answer “no”. T_1 and T_2 will answer, because they decide (so each one must halt on every input, including w).

Problem 938 Consider the Turing machine

$M_7 = (Q, \Sigma, \Gamma, \delta, q)$ such that: $Q = \{q, p\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, a, b, c\}$; and δ is defined by the following transition set:

$$\begin{array}{ll} [q, a, p, a, R] & [p, a, p, a, R] \\ [q, b, q, b, R] & [p, b, p, b, R] \\ [q, c, q, c, R] & [p, c, p, c, R] \\ & [p, B, p, B, R] \end{array}$$

(M_7 has an one-way infinite tape (infinite to the right only.) B is the designated blank symbol.)

Let L_7 be the set of strings on which M_7 halts.

Let $L_{7\infty}$ be the set of strings on which M_7 diverges.

(a) Write a regular expression that defines L_7 . If such a regular expression does not exist, prove it.

Answer:

$$(b \cup c)^*$$

(b) Write a regular expression that defines $L_{7\infty}$. If such a regular expression does not exist, prove it.

Answer:

$$(b \cup c)^* a (a \cup b \cup c)^*$$

(c) Write a regular expression that defines the language (if any) which is decided by M_7 . If such a regular expression does not exist, prove it.

Answer: This regular expression does not exist, since M does not decide any language, because it diverges on exactly those strings that belong to $L_{7\infty}$.

(d) Explain how to construct an algorithm that solves the following problem:

INPUT: A Turing machine T and an input string w ;

OUTPUT: yes is T halts on input w ;
no otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist, because it is exactly the M_H , the non-existent Turing Machine that solves the Halting Problem.

Problem 939 Consider the Turing machine

$M = (Q, \Sigma, \Gamma, \delta, q, F)$ such that: $Q = \{q, p, s, x\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, a, b, c\}$; $F = \{x\}$; and δ is defined by the following transition set:

$$\begin{array}{lll} [q, a, p, a, R] & [p, a, s, a, R] & [s, a, q, a, R] \\ [q, b, p, b, R] & [p, b, s, b, R] & [s, b, q, b, R] \\ [q, c, p, c, R] & [p, c, s, c, R] & [s, c, q, c, R] \\ [q, B, q, B, R] & [p, B, x, c, R] & \end{array}$$

(M has an one-way infinite tape (infinite to the right only.) B is the designated blank symbol.) M accepts by final state.)

Let L_A be the set of strings which M accepts.

Let L_R be the set of strings which M rejects.

Let L_∞ be the set of strings on which M diverges.

(a) Write a regular expression that defines L_A . If such a regular expression does not exist, prove it.

Advice for Answer: M counts modulo 3.

Answer:

$$((a \cup b \cup c)(a \cup b \cup c)(a \cup b \cup c))^* (a \cup b \cup c)$$

(b) Write a regular expression that defines L_R . If such a regular expression does not exist, prove it.

Answer:

$$((a \cup b \cup c)(a \cup b \cup c)(a \cup b \cup c))^* (a \cup b \cup c)(a \cup b \cup c)$$

- (c) Write a regular expression that defines L_∞ . If such a regular expression does not exist, prove it.

Answer:

$$((a \cup b \cup c)(a \cup b \cup c)(a \cup b \cup c))^*$$

- (d) Explain how to construct an algorithm that solves the following problem:

INPUT: Turing Machine T_1 that accepts a language L_1 ;

OUTPUT: Regular expression that defines L_1 .

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist: some recursively enumerable languages are not regular. Furthermore, there is no algorithm to decide whether L_1 is regular, since the property of regularity is nontrivial—true for \emptyset and false for $\{a^n b^n\}$.

Problem 940 Consider the Turing machine $M = (Q, \Sigma, \Gamma, \delta, q, F)$ such that: $Q = \{q, p, t, s, v, x\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, a, b, c, A, E, K\}$; $F = \{x\}$; and δ is defined by the following transition set:

$[q, a, p, A, R]$	$[p, a, p, a, R]$	$[t, A, x, a, R]$
$[q, b, p, E, R]$	$[p, b, p, b, R]$	$[t, E, q, c, R]$
$[q, c, p, K, R]$	$[p, c, p, c, R]$	$[t, K, x, K, R]$
$[q, B, q, B, R]$	$[p, B, t, B, L]$	

(M has an one-way infinite tape (infinite to the right only.) B is the designated blank symbol. M accepts by final state.)

Let L_A be the set of strings which M accepts.

Let L_R be the set of strings which M rejects.

Let L_∞ be the set of strings on which M diverges.

- (a) Write a regular expression that defines L_A . If such a regular expression does not exist, prove it.

Answer:

$$a \cup c$$

- (b) Write a regular expression that defines L_R . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c)(a \cup b \cup c)(a \cup b \cup c)^*$$

- (c) Write a regular expression that defines L_∞ . If such a regular expression does not exist, prove it.

Answer:

$$\lambda \cup b$$

- (d) Explain how to construct an algorithm that solves the following problem:

INPUT: Turing Machine T_1 that accepts a language L_1 ;

OUTPUT: yes if L_1 is regular;

no otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist. If it existed, it would decide the set of Turing Machines whose languages have the nontrivial property "is regular", which in turn is impossible by Rice's Theorem. The property of regularity is nontrivial since it is true for Σ^* but false for the set of palindromes over Σ .

Problem 941 Consider the Turing machine $M = (Q, \Sigma, \Gamma, \delta, q, F)$ such that: $Q = \{q, p, t, s, v, x\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, a, b, c, A, E, K\}$; $F = \{x\}$; and δ is defined by the following transition set:

$[q, a, p, a, R]$	$[p, a, t, A, R]$	$[t, a, t, a, R]$
$[q, b, p, b, R]$	$[p, b, t, E, R]$	$[t, b, t, b, R]$
$[q, c, p, c, R]$	$[p, c, t, K, R]$	$[t, c, t, c, R]$
$[q, B, q, B, R]$	$[p, B, p, B, R]$	$[t, B, s, B, L]$

$[s, a, v, a, L]$	$[v, a, v, a, L]$
	$[v, b, v, b, L]$
	$[v, c, v, c, L]$
	$[v, K, x, K, R]$

(M has an one-way infinite tape (infinite to the right only.) B is the designated blank symbol. M accepts by final state.)

Let L_A be the set of strings which M accepts.

Let L_R be the set of strings which M rejects.

Let L_∞ be the set of strings on which M diverges.

- (a) Write a regular expression that defines L_A . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup b \cup c) c (a \cup b \cup c)^* a$$

- (b) Write a regular expression that defines L_∞ . If such a regular expression does not exist, prove it.

Answer:

$$\lambda \cup a \cup b \cup c$$

- (c) Explain how to construct an algorithm that solves the following problem:

INPUT: Turing Machine T_1 that decides a language L_1 ;

OUTPUT: Turing Machine T_2 that decides the language $\overline{L_1}$;

If this algorithm does not exist, prove it.

Answer: T_2 is identical to T_1 , but the decision of T_2 is the negation of the decision of T_1 .

- (d) Explain how to construct an algorithm that solves the following problem:

INPUT: Turing Machine T_3 that accepts a language L_3 ;

OUTPUT: Turing Machine T_4 that accepts the language $\overline{L_3}$;

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist: the complement $\overline{L_3}$ of a recursively enumerable language L_3 need not be recursively enumerable—hence, T_4 need not exist.

Furthermore, there is no algorithm to decide whether \overline{L} is recursively enumerable, since the property of having a recursively enumerable complement is nontrivial—true for \emptyset and false for L_H .

Problem 942 Explain how to construct an algorithm that solves the following problem:

INPUT: Pushdown automaton P_1 that accepts a language L_1 and a Turing Machine T_2 that accepts a language L_2 ;

OUTPUT: Turing Machine T_3 that accepts the language $L_1 \cup L_2$;

If this algorithm does not exist, prove it.

Answer: T_3 simulates P_1 and T_2 simultaneously, and accepts if and only if at least one of the simulated machines accepts.

Problem 943 Let L be the language accepted by the pushdown automaton $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, p\}$, $\Sigma = \{a, b, c, d, e, f\}$, $\Gamma = \{A, E, M, X\}$, $F = \{p\}$ and the transition function δ is defined as follows:

$$\begin{aligned} & [q, f, \lambda, p, EXAM] \\ & [p, a, A, p, \lambda] \\ & [p, b, E, p, \lambda] \\ & [p, c, M, p, \lambda] \\ & [p, d, X, p, \lambda] \\ & [p, e, E, p, \lambda] \\ & [p, f, \lambda, p, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) List 6 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: Regular expression for L :

$$f f^* c f^* a f^* d f^* (b \cup e) f^*$$

(b) Draw a state transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 352.

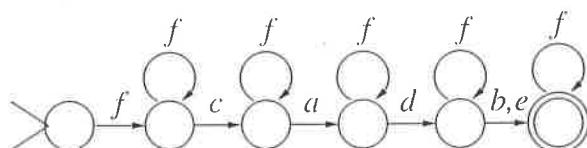


Figure 352:

(c) Is L decidable? Prove your answer.

Answer: Yes. By the answer to part (b), L is regular, and every regular language is decidable.

(d) Is \overline{L} (the complement of L) recursively enumerable? Prove your answer.

Answer: Yes. As the complement of a regular language, \overline{L} is regular, and thus (decidable) and recursively enumerable.

(e) State the cardinality of L . If L is finite, state the exact number; if L is infinite, specify whether it is countable or not countable.

Answer: L is infinite and countable.

Problem 944 Let L be the language accepted by the pushdown automaton $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, p\}$, $\Sigma = \{a, b, c, d, e, f\}$, $\Gamma = \{A, E, M, X\}$, $F = \{p\}$ and the transition function δ is defined as follows:

$$\begin{aligned} & [q, f, \lambda, q, EX] \\ & [q, e, \lambda, q, AM] \\ & [p, a, A, p, \lambda] \\ & [p, b, E, p, \lambda] \\ & [p, c, M, p, \lambda] \\ & [p, d, X, p, \lambda] \\ & [q, \lambda, \lambda, p, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) List 6 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer:

$$\lambda, fdb, eca, fecadb, fefcadbca, fffldbdbdb$$

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, e, f\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow fSdb \mid eSca \mid \lambda$$

(c) State one non-trivial property of recursively enumerable languages that is true for L and true for \emptyset . Explain carefully why this property is non-trivial, and prove that it is true for L and true for \emptyset . If such a property does not exist, state it, and explain why it is so.

Answer:

is context free

The property is non-trivial, since some recursively enumerable languages are context-free (such as \emptyset) and some are not context-free (such as the set

$\{a^n b^n c^n \mid n \geq 0\}.$) L is context-free, as is witnessed by the grammar constructed in the answer to part (b). (\emptyset is context-free because it is finite.)

(d) State one non-trivial property of recursively enumerable languages that is true for L but false for \emptyset . Explain carefully why this property is non-trivial, and prove that it is true for L and false for \emptyset . If such a property does not exist, state it, and explain why it is so.

Answer:

is not empty

The property is non-trivial, since some recursively enumerable languages are not empty (such as L) and some are empty (such as \emptyset). L is not empty, as is witnessed by the strings listed in the answer to part (a).

(e) State one trivial property of recursively enumerable languages that is false for L but true for \emptyset . Explain carefully why this property is trivial, and prove that it is false for L and true for \emptyset . If such a property does not exist, state it, and explain why it is so.

Answer: This property does not exist. By definition, a trivial property assumes the same value for every recursively enumerable language, and if it is true for L it cannot be false for \emptyset .

Problem 945 Let L_1 be the language accepted by the pushdown automaton $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where $\Sigma = \{a, b, c, d, e\}$; $\Gamma = \{A, B\}$; $Q = \{q, t, s\}$; $F = \{s\}$; and the transition function δ is defined as follows:

$$\begin{aligned} & [q, b, \lambda, q, BBB] \\ & [q, \lambda, \lambda, t, \lambda] \\ & [t, d, \lambda, t, AA] \\ & [t, e, \lambda, s, \lambda] \\ & [s, c, A, s, \lambda] \\ & [s, a, B, s, \lambda] \end{aligned}$$

(M is defined so as to accept by final state and empty stack.)

(a) List four distinct strings that belong to L_1 . If this is impossible, state it and explain why.

Advice for Answer: The general template for strings of this language is:

$$b^n d^k e c^{2k} a^{3n}$$

(b) Write a complete formal definition of a context-free grammar that generates L_1 . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, e\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow bSaaa \mid A \\ A &\rightarrow dAcc \mid c \end{aligned}$$

(c) Explain how to construct an algorithm that solves the following problem:

INPUT: A deterministic Turing Machine M which accepts a language $L(M)$.

OUTPUT: yes if $L(M) \cap L_1 = \emptyset$; no otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist. If it existed, it would decide the set of Turing machine whose languages satisfy the property:

has an empty intersection with L_1 .

This property is nontrivial, since it is true say for empty set (because $\emptyset \cap L_1 = \emptyset$), but false for say Σ^* (because $\Sigma^* \cap L_1 = L_1 \neq \emptyset$). By Rice's Theorem a Turing machine cannot exist that decides the set of Turing machines whose languages satisfy any nontrivial property.

(d) State a nontrivial property of recursively enumerable languages that is true for L_1 but false for a^* . Explain why the property is nontrivial and show that it indeed is true for L_1 but false for a^* . If such a property does not exist, state it and explain your answer.

Answer: "Does not contain the empty string." This property is true for L_1 , since every string in L_1 contains one e , which is proved by inspection of M . This property is false for a^* , because a^* contains the empty string, by definition of Kleene star. Since it is true for one recursively enumerable language (L_1) but false for another (a^*), the property is by definition non-trivial.

Problem 946 Let L_1 be the language accepted by the pushdown automaton $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where $\Sigma = \{a, b, c, d, e\}$; $\Gamma = \{A, B\}$; $Q = \{q, t, v, x, s\}$; $F = \{s\}$; and the transition function δ is defined as follows:

$$\begin{aligned} & [q, b, \lambda, q, BB] \\ & [v, d, \lambda, v, AAA] \\ & [t, c, B, t, \lambda] \\ & [x, a, A, x, \lambda] \\ & [g, \lambda, \lambda, t, \lambda] \\ & [t, \lambda, \lambda, v, \lambda] \\ & [v, \lambda, \lambda, x, \lambda] \\ & [x, e, \lambda, s, \lambda] \end{aligned}$$

(M is defined so as to accept by final state and empty stack.)

(a) List four distinct strings that belong to L_1 . If this is impossible, state it and explain why.

Advice for Answer: The general template for strings of this language is:

$$b^n c^{2n} d^k a^{3k} e$$

(b) Write a complete formal definition of a context-free grammar that generates L_1 . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d, e\}$, $V = \{S, A, B\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow ABe \\ A &\rightarrow bAcc \mid \lambda \\ B &\rightarrow dBaaa \mid \lambda \end{aligned}$$

(c) Explain how to construct an algorithm that solves the following problem:

INPUT: A deterministic finite automaton M which accepts a language $L(M)$, and a string $w \in \Sigma^*$.

OUTPUT: yes if $w \in L(M) \cap L_1$; no otherwise.

If this algorithm does not exist, prove it.

Answer: Recall that the intersection of a context free language with a regular language is context free. Apply the known algorithm to construct a pushdown automaton M_X that accepts the intersection of the context free language L_1 (accepted by the original pushdown automaton) and the regular language $L(M)$ (accepted by the argument finite automaton M). Simulate M_X on w and return exactly what M_X returns.

(d) State a trivial property of recursively enumerable languages that is true for L_1 but false for $(aa)^*$. Explain why the property is trivial and show that it indeed is true for L_1 but false for $(aa)^*$. If such a property does not exist, state it and explain your answer.

Answer: Such a property does not exist. If a property of recursively enumerable languages was true for L_1 but false for $(aa)^*$ then such a property would not be trivial, because by definition a trivial property has the same value for all recursively enumerable languages.

Problem 947 Consider the Turing machine

$M = (Q, \Sigma, \Gamma, \delta, q)$ such that: $\Sigma = \{0, 1\}$;

$\Gamma = \{B, 0, 1, Z, N\}$; $Q = \{q, p, t, v, m, x, y\}$; and δ is defined by the following transition set:

$[q, 0, t, Z, R]$	$[t, 0, t, 0, R]$	$[p, 0, p, 0, R]$
$[q, 1, t, N, R]$	$[t, 1, t, 1, R]$	$[p, 1, p, 1, R]$
$[q, B, p, B, R]$	$[t, B, v, B, L]$	$[p, B, p, B, R]$
$[v, 0, x, 0, L]$	$[x, 0, x, 0, L]$	$[y, 0, y, 0, L]$
$[v, 1, y, 1, L]$	$[x, 1, x, 1, L]$	$[y, 1, y, 1, L]$
$[v, Z, p, 0, R]$	$[x, Z, p, 0, R]$	$[y, Z, m, 0, R]$
$[v, N, p, 1, R]$	$[x, N, m, 1, R]$	$[y, N, p, 1, R]$

Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.

Let L be the set of strings on which the Turing machine M halts.

(a) List 5 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: The machine rewrites the first input symbol (or diverges if the input is empty), skips over the entire input and then returns to the left, diverging if no symbols are found after the first one. Otherwise it remembers the last input symbol in the state and compares it with the first symbol, halting exactly when they are different. See the answer to part (b).

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: $0(0 \cup 1)^*1 \cup 1(0 \cup 1)^*0$

(c) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{0, 1\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow 0A1 \mid 1A0 \\ A &\rightarrow \lambda \mid AA \mid 0 \mid 1 \end{aligned}$$

(d) Describe the construction of a Turing Machine T which halts on every input and also accepts the strings that represent Turing Machines that accept only strings that do not belong to L . In short:

$$(L(\tau) \subseteq \bar{L}) \implies (T(\tau) \downarrow \text{and accept})$$

$$(L(\tau) \not\subseteq \bar{L}) \implies (T(\tau) \downarrow \text{and reject})$$

If such a Turing machine T does not exist, prove it.

Answer: This machine does not exist. If it existed, it would decide the set of Turing Machines whose languages have the property: "is a subset of the complement of L ". This property is nontrivial, since \emptyset has it, but $(0 \cup 1)^*$ does not have it. By Rice's Theorem, it is impossible to decide the set of Turning Machines whose languages have any nontrivial property.

Problem 948 Consider the Turing machine

$M = (Q, \Sigma, \Gamma, \delta, q, F)$ such that: $\Sigma = \{0, 1\}$;

$\Gamma = \{B, 0, 1, \Psi\}$; $Q = \{q, p, t, v, m, x, y\}$; $F = \{m\}$; and δ is defined by the following transition set:

$[q, 0, t, \Psi, R]$	$[t, 0, t, 0, R]$	$[p, 0, p, 0, R]$
$[q, 1, t, \Psi, R]$	$[t, 1, t, 1, R]$	$[p, 1, p, 1, R]$
$[q, B, p, B, R]$	$[t, B, v, B, L]$	$[p, B, p, B, R]$
$[v, 0, x, 0, L]$	$[x, 0, x, 0, L]$	$[y, 1, y, 1, L]$
$[v, 1, y, 1, L]$	$[x, \Psi, m, 0, R]$	$[y, \Psi, m, 1, R]$

Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) M accepts by final state. B is the designated blank symbol.

Let L be the set of strings which the Turing machine M accepts.

(a) List 5 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: The machine marks the first input symbol (or diverges if the input is empty), skips over the entire input and then returns to the left, but only if at least one more symbol is found after the first. It moves to the left as long as it sees only 1's or only 0's, and halts in m if it so arrives at the first symbol. See the answer to part (b).

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: $(0 \cup 1)(0^*0 \cup 1^*)$

- (c) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 353.

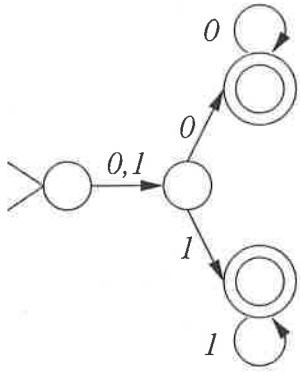


Figure 353:

- (d) Describe the construction of a Turing Machine T which diverges exactly on the set of those strings which the Turing machine M accepts. In short:

$$(M(\tau) \searrow \text{and accept}) \iff (T(\tau) \nearrow)$$

If such a Turing machine T does not exist, prove it.

Answer: Convert algorithmically the automaton given in the answer to part (c) to a deterministic automaton, simulate this automaton and diverge exactly when it accepts.

Problem 949 Consider the Turing machine $M = (Q, \Sigma, \Gamma, \delta, q, F)$ such that: $Q = \{q, p, s, t\}$; $\Sigma = \{0, 1\}$; $\Gamma = \{B, 0, 1\}$; $F = \{t\}$; and δ is defined by the following transition set:

$$\begin{array}{lll} [q, 0, p, 0, R] & [p, 0, s, 0, R] & [s, 0, q, 0, R] \\ [q, 1, p, 1, R] & [p, 1, s, 1, R] & [s, 1, q, 1, R] \\ [q, B, t, B, R] & & [s, B, s, B, R] \end{array}$$

(M has an one-way infinite tape (infinite to the right only). B is the designated blank symbol.) M accepts by final state.)

Let L_A be the set of strings which M accepts.

Let L_R be the set of strings which M rejects.

Let L_∞ be the set of strings on which M diverges.

- (a) Write a regular expression that defines L_A . If such a regular expression does not exist, prove it.

Answer:

$$((0 \cup 1)(0 \cup 1)(0 \cup 1))^*$$

- (b) Write a regular expression that defines L_R . If such a regular expression does not exist, prove it.

Answer:

$$((0 \cup 1)(0 \cup 1)(0 \cup 1))^*(0 \cup 1)$$

- (c) Write a regular expression that defines L_∞ . If such a regular expression does not exist, prove it.

Answer:

$$((0 \cup 1)(0 \cup 1)(0 \cup 1))^*(0 \cup 1)(0 \cup 1)$$

- (d) Explain how to construct an algorithm that solves the following problem:

INPUT: Turing Machine T_1 that decides a language $L(T_1)$;

OUTPUT: Regular expression that defines $L(T_1)$.

If this algorithm does not exist, prove it.

Answer: Impossible—there exist decidable languages that are not regular.

Problem 950 Consider the Turing machine $M = (Q, \Sigma, \Gamma, \delta, q, F)$ such that: $Q = \{q, p, t, x\}$; $\Sigma = \{0, 1\}$; $\Gamma = \{B, 0, 1, A, E\}$; $F = \{x\}$; and δ is defined by the following transition set:

$$\begin{array}{lll} [q, 0, p, A, R] & [p, 0, p, 0, R] & [t, A, x, 0, R] \\ [q, 1, p, E, R] & [p, 1, p, 1, R] & [t, E, q, 1, R] \\ [q, B, q, B, R] & [p, B, t, B, L] & \end{array}$$

(M has an one-way infinite tape (infinite to the right only). B is the designated blank symbol. M accepts by final state.)

Let L_A be the set of strings which M accepts.

Let L_R be the set of strings which M rejects.

Let L_∞ be the set of strings on which M diverges.

- (a) Write a regular expression that defines L_∞ . If such a regular expression does not exist, prove it.

Answer:

$$\lambda \cup 1$$

- (b) Write a regular expression that defines L_A . If such a regular expression does not exist, prove it.

Answer:

$$0$$

- (c) Write a regular expression that defines L_R . If such a regular expression does not exist, prove it.

Answer:

$$(0 \cup 1)(0 \cup 1)(0 \cup 1)^*$$

- (d) Explain how to construct an algorithm that solves the following problem:

INPUT: Turing Machine T_1 that accepts a language $L(T_1)$;

OUTPUT: yes if $L(T_1)$ is finite;
no otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist. If it existed, it would decide the set of Turing Machines whose languages have the non-trivial property

is finite

which is impossible by Rice's Theorem. This property is non-trivial because it is true for \emptyset and false for 0^* .

Problem 951 Consider the Turing machine $M = (Q, \Sigma, \Gamma, \delta, q, F)$ such that: $Q = \{q, p, t, s, v, x\}$; $\Sigma = \{0, 1\}$; $\Gamma = \{B, 0, 1, A, E\}$; $F = \{x\}$; and δ is defined by the following transition set:

$[q, 0, p, 0, R]$	$[p, 0, t, A, R]$	$[t, 0, t, 0, R]$
$[q, 1, p, 1, R]$	$[p, 1, t, E, R]$	$[t, 1, t, 1, R]$
$[q, B, q, B, R]$	$[p, B, p, B, R]$	$[t, B, s, B, L]$
$[s, 0, v, 0, L]$	$[v, 0, v, 0, L]$	
	$[v, 1, v, 1, L]$	
	$[v, E, x, E, R]$	

(M has an one-way infinite tape (infinite to the right only.) B is the designated blank symbol. M accepts by final state.)

Let L_A be the set of strings which M accepts.

Let L_R be the set of strings which M rejects.

Let L_∞ be the set of strings on which M diverges.

(a) Write a regular expression that defines L_∞ . If such a regular expression does not exist, prove it.

Answer:

$$\lambda \cup 0 \cup 1$$

(b) Write a regular expression that defines L_A . If such a regular expression does not exist, prove it.

Answer:

$$(0 \cup 1) \ 1 \ (0 \cup 1)^*$$

(c) Write six distinct strings that belong to L_R . If such strings do not exist, prove it.

Advice for Answer: All strings of length equal to two, as well as those whose length is greater than two that have 0 at the second position or 1 at the last position.

(d) Explain how to construct an algorithm that solves the following problem:

INPUT: Turing Machine T_1 that accepts a language $L(T_1)$;

OUTPUT: Pushdown automaton P_2 that accepts the language $L(T_1)$;

If this algorithm does not exist, prove it.

Answer: Impossible—there exist recursively enumerable languages that are not context free.

(e) Explain how to construct an algorithm that solves the following problem:

INPUT: Turing Machine T_3 that decides a language $L(T_3)$;

OUTPUT: Turing Machine T_4 that decides the language $\overline{L}(T_3)$;

If this algorithm does not exist, prove it.

Answer: T_4 is identical to T_3 , except that the decision of T_4 is the negation of the decision of T_3 .

Problem 952 Consider the Turing machine $M = (Q, \Sigma, \Gamma, \delta, s, F)$ such that: $Q = \{s, p, x, v, z, w\}$; $\Sigma = \{a, b, c, g\}$; $\Gamma = \{B, a, b, c, g, A, K, D, G\}$; $F = \{w\}$; and δ consists of the following transition set:

$[s, a, p, A, R]$	$[p, a, x, D, R]$	$[x, a, x, a, R]$
$[s, b, s, b, R]$	$[p, b, p, b, R]$	$[x, b, x, b, R]$
$[s, c, p, K, R]$	$[p, c, x, G, R]$	$[x, c, x, c, R]$
$[s, g, s, g, R]$	$[p, g, p, g, R]$	$[x, g, x, g, R]$
$[s, B, s, B, R]$	$[p, B, p, B, R]$	$[x, B, v, B, L]$
$[v, a, v, a, L]$	$[z, a, z, a, L]$	
$[v, b, v, b, L]$	$[z, b, z, b, L]$	
$[v, c, v, c, L]$	$[z, c, z, c, L]$	
$[v, g, v, g, L]$	$[z, g, z, g, L]$	
$[v, D, z, D, L]$	$[z, K, w, K, R]$	

(M has an one-way infinite tape (infinite to the right only.) B is the designated blank symbol. M accepts by final state.)

Let L be the set of strings which M accepts.

Let L_R be the set of strings which M rejects.

Let L_∞ be the set of strings on which M diverges.

(a) List four distinct strings that belong to L_∞ . If this is impossible, state it and explain why it is so.

Advice for Answer: M diverges unless it finds at least two occurrences of symbols belonging to $\{a, c\}$. If such two symbols are found, then M accepts if the leftmost two such symbols are c and a in that order, from left to right. Otherwise, M rejects.

Answer:

$$\lambda, b, g, a$$

(b) List four distinct strings that belong to L . If this is impossible, state it and explain why it is so.

Answer:

$$ca,caa,bcab,gca$$

(c) List four distinct strings that belong to L_R . If this is impossible, state it and explain why it is so.

Answer:

$$aa, ac, cc, baa$$

(d) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(b \cup g)^* c (b \cup g)^* a (a \cup b \cup c \cup g)^*$$

(e) Write a regular expression that defines L_∞ . If such a regular expression does not exist, prove it.

Answer:

$$(b \cup g)^* \cup (b \cup g)^* (a \cup c) (b \cup g)^*$$

(f) Write a regular expression that defines L_R . If such a regular expression does not exist, prove it.

Answer:

$$(b \cup g)^* a (b \cup g)^* a (a \cup b \cup c \cup g)^*$$

$$\quad \quad \quad \cup$$

$$(b \cup g)^* a (b \cup g)^* c (a \cup b \cup c \cup g)^*$$

$$\quad \quad \quad \cup$$

$$(b \cup g)^* c (b \cup g)^* c (a \cup b \cup c \cup g)^*$$

- (g) List four distinct strings that belong to $(a \cup b \cup c \cup g)^*$ such that the machine M (defined at the beginning of this problem) must write symbol G on its tape if invoked on any of these four strings (as input.) If this is impossible, state it and explain why it is so.

Answer:

$$ac, ccaa, bac, gcca$$

Problem 953 Consider the Turing machine $M = (Q, \Sigma, \Gamma, \delta, q)$ such that: $Q = \{q, p, s, t, v\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, a, b, c, T\}$; and δ is defined by the following transition set:

$[q, a, p, T, R]$	$[p, a, p, a, R]$	$[v, a, v, a, R]$
$[q, b, p, T, R]$	$[p, b, p, b, R]$	$[v, b, v, b, R]$
$[q, c, p, T, R]$	$[p, c, p, c, R]$	$[v, c, v, c, R]$
	$[p, B, s, B, L]$	$[v, B, v, B, R]$
$[s, a, t, a, L]$		
$[s, b, s, b, L]$	$[t, c, v, c, R]$	

(M has an one-way infinite tape (infinite to the right only.) B is the designated blank symbol.)

Let L_D be the set of strings on which M diverges.

- (a) Write 6 distinct strings that belong to L_D . If such strings do not exist, state it and prove it.

Advice for Answer: See answer to part (b).

- (b) Write a regular expression that defines L_D . If such a regular expression does not exist, state it and prove it.

Answer:

$$(a \cup b \cup c) (a \cup b \cup c)^* ca b^*$$

- (c) Explain how to construct an algorithm that solves the following problem:

INPUT: String w over Σ .

OUTPUT: **yes** if w represents a Turing Machine that accepts exactly those strings on which the Turing Machine M (defined at the beginning of this problem) diverges;

no otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist. If it existed, it would decide the set of Turing Machines whose languages have the non-trivial property

is equal to L_D

which is impossible by Rice's Theorem. This property is non-trivial because it is true for L_D and false for any other language, say for $\overline{L_D}$.

- (d) Explain how to construct an algorithm that solves the following problem:

INPUT: String w over Σ .

OUTPUT: **yes** if w is a string which belongs to the set of exactly those strings on which the Turing Machine M (defined at the beginning of this problem) halts;

no otherwise.

If this algorithm does not exist, prove it.

Answer: We need an algorithm to decide $\overline{L_D}$. Therefore, convert the regular expression for L_D , given in the answer to part (b), into a finite automaton, convert this automaton into a deterministic one, simulate this latter automaton, and return the opposite of whatever it decides.

Problem 954 Consider the Turing machine $M = (Q, \Sigma, \Gamma, \delta, q, F)$ such that: $Q = \{q, p, s, t, x, y\}$; $\Sigma = \{a, b, c, d\}$; $\Gamma = \{B, a, b, c, d, E, T\}$; $F = \{x\}$; and δ is defined by the following transition set:

$[q, a, p, E, R]$	$[p, a, p, a, R]$	$[s, a, y, a, L]$
$[q, b, p, T, R]$	$[p, b, p, b, R]$	$[s, b, t, b, L]$
$[q, c, p, T, R]$	$[p, c, p, c, R]$	$[s, c, t, c, L]$
$[q, d, p, E, R]$	$[p, d, p, d, R]$	$[s, d, y, d, L]$
	$[p, B, s, B, L]$	

$[t, a, t, a, L]$	$[y, a, y, a, L]$	$[t, T, x, T, R]$
$[t, b, t, b, L]$	$[y, b, y, b, L]$	$[y, E, x, E, R]$
$[t, c, t, c, L]$	$[y, c, y, c, L]$	
$[t, d, t, d, L]$	$[y, d, y, d, L]$	

(M has an one-way infinite tape (infinite to the right only.) B is the designated blank symbol. M accepts by final state.)

Let L be the set of strings which M accepts.

(a) Write 6 distinct strings that belong to L . If such strings do not exist, state it and prove it.

Advice for Answer: See answer to part (b).

(b) Draw a state transition graph of a finite automaton that accepts L . If such an automaton does not exist, state it and prove it.

Answer: See Figure 354.

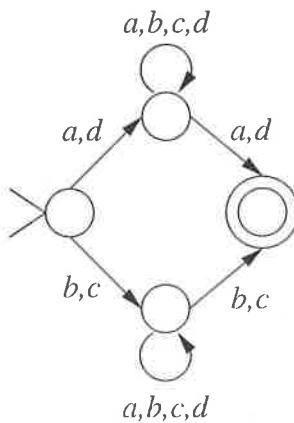


Figure 354:

(c) Explain how to construct an algorithm that solves the following problem:

INPUT: String w over Σ .

OUTPUT: yes if w is a string which belongs to the set of exactly those strings which the Turing Machine M (defined at the beginning of this problem) does not accept; no otherwise.

If this algorithm does not exist, prove it.

Answer: We need an algorithm to decide \overline{L} . Therefore, convert the automaton for L , given in the answer to part (b), into a deterministic one, simulate this latter automaton, and return the opposite of whatever it decides.

(d) Explain how to construct an algorithm that solves the following problem:

INPUT: String w over Σ .

OUTPUT: yes if w represents a Turing Machine that accepts exactly those strings which the Turing Machine M (defined at the beginning of this problem) accepts; no otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist. If it existed, it would decide the set of Turing Machines whose languages have the non-trivial property

is equal to L

which is impossible by Rice's Theorem. This property is non-trivial because it is true for L and false for any other language, say for \overline{L} .

Problem 955 Consider the Turing machine

$M = (Q, \Sigma, \Gamma, \delta, q, F)$ such that: $Q = \{q, p, v, z, x\}$; $\Sigma = \{0, 1\}$; $\Gamma = \{B, 0, 1\}$; $F = \{x\}$; and δ is defined by the following transition set:

$[q, 0, q, 0, R]$	$[p, 1, q, 1, R]$	$[v, 0, z, 0, R]$
$[q, 1, p, 1, R]$	$[p, 0, p, 0, R]$	$[v, 1, x, 1, R]$
$[q, B, q, B, R]$	$[p, B, v, B, L]$	

(M has an one-way infinite tape (infinite to the right only.) B is the designated blank symbol. M accepts by final state.)

Let L_A be the set of string which M accepts.

Let L_R be the set of string which M rejects.

Let L_D be the set of string on which M diverges.

(a) List 6 distinct strings that belong to L_A . If this is impossible, state it and explain why.

Advice for Answer: M skips over the input to the right, flipping states to remember the parity of the number of 1's seen. If this number is even, M diverges. If the number of 1's is odd, M accepts or rejects according to the rightmost input symbol.

See the answer to part (d).

(b) List 6 distinct strings that belong to L_R . If this is impossible, state it and explain why.

Advice for Answer: See the answer to part (e).

(c) List 6 distinct strings that belong to L_D . If this is impossible, state it and explain why.

Advice for Answer: See the answer to part (f).

(d) Write a regular expression that defines L_A . If such a regular expression does not exist, prove it.

Answer:

$$0^* (10^*)^* 0^*$$

(e) Write a regular expression that defines L_R . If such a regular expression does not exist, prove it.

Answer:

$$0^* (10^*)^* 0^* 10^*$$

- (f) Write a regular expression that defines L_D . If such a regular expression does not exist, prove it.

Answer:

$$0^* (10^*)^* 0^*$$

- (g) Explain how to construct an algorithm that solves the following problem:

INPUT: String w over $\{0, 1\}$.

OUTPUT: **yes** if w is a string such that the Turing Machine M (defined at the beginning of this problem) rejects w ;

no otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm decides whether the given input string belongs to the language L_R , whose regular expression is given in the answer to part (e). Therefore, convert this regular expression into a finite automaton, convert this automaton into a deterministic finite automaton, simulate this deterministic automaton and decide as it decides.

Problem 956 Consider the Turing machine $M = (Q, \Sigma, \Gamma, \delta, q, F)$ such that: $Q = \{q, p, v, z, x\}$; $\Sigma = \{0, 1\}$; $\Gamma = \{B, 0, 1, N\}$; $F = \{x\}$; and δ is defined by the following transition set:

$$\begin{array}{lll} [q, 0, p, N, R] & [p, 0, p, 0, R] & [v, 1, v, 1, L] \\ [q, 1, q, 1, R] & [p, 1, p, 1, R] & [v, 0, x, 0, R] \\ [q, B, q, B, R] & [p, B, v, B, L] & [v, N, z, 0, R] \end{array}$$

(M has an one-way infinite tape (infinite to the right only.) B is the designated blank symbol. M accepts by final state.)

Let L_A be the set of string which M accepts.

Let L_R be the set of string which M rejects.

Let L_D be the set of string on which M diverges.

- (a) List 6 distinct strings that belong to L_A . If this is impossible, state it and explain why.

Advice for Answer: M moves to the right, looking for a 0. If M does not find a 0, it diverges. Otherwise, M rewrites the first 0 into N , skips over the input, and on the way back looks for yet another 0 (not rewritten.) M accepts if it finds this second 0 and rejects otherwise.

See the answer to part (d).

- (b) List 6 distinct strings that belong to L_R . If this is impossible, state it and explain why.

Advice for Answer: See the answer to part (e).

- (c) List 6 distinct strings that belong to L_D . If this is impossible, state it and explain why.

Advice for Answer: See the answer to part (f).

- (d) Write a regular expression that defines L_A . If such a regular expression does not exist, prove it.

Answer:

$$1^* 0 (0 \cup 1)^* 01^*$$

- (e) Write a regular expression that defines L_R . If such a regular expression does not exist, prove it.

Answer:

$$1^* 01^*$$

- (f) Write a regular expression that defines L_D . If such a regular expression does not exist, prove it.

Answer:

$$1^*$$

- (g) Explain how to construct an algorithm that solves the following problem:

INPUT: String w over $\{0, 1\}$.

OUTPUT: **yes** if w represents a Turing Machine that accepts exactly those strings which the Turing Machine M (defined at the beginning of this problem) accepts; **no** otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist. If it existed, it would decide the set of those Turing Machines whose languages have the non-trivial property: "is equal to L_A ". By Rice's Theorem, this is impossible. This property is non-trivial because it is true for L_A but false for L_D .

Problem 957 Consider the Turing machine

$M = (Q, \Sigma, \Gamma, \delta, q, F)$ such that: $Q = \{q, r, s, p, v, t, z, x, y\}$; $\Sigma = \{0, 1\}$; $\Gamma = \{B, 0, 1\}$; $F = \{x\}$; and δ is defined by the following transition set:

$$\begin{array}{lll} [q, 1, r, 1, R] & [p, 0, p, 0, R] & [y, 0, y, 0, R] \\ [r, 0, s, 0, R] & [p, 1, p, 1, R] & [y, 1, y, 1, R] \\ [s, 1, t, 1, R] & [p, B, v, B, L] & [y, B, y, B, R] \\ [t, 0, p, 0, R] & [v, 0, z, 0, L] & [z, 0, x, 0, L] \\ [t, 1, p, 1, R] & [v, 1, x, 1, L] & [z, 1, y, 1, L] \end{array}$$

(M has an one-way infinite tape (infinite to the right only.) B is the designated blank symbol. M accepts by final state.)

Let L_A be the set of string which M accepts.

Let L_R be the set of string which M rejects.

Let L_D be the set of string on which M diverges.

- (a) List 6 distinct strings that belong to L_A . If this is impossible, state it and explain why.

Advice for Answer: M halts and rejects immediately unless the input has at least four symbols and begins with the substring 101. If M finds such four symbols, it skips over the input and inspects the rightmost symbols to decide whether to accept or diverge.

See the answer to part (d).

- (b) List 6 distinct strings that belong to L_R . If this is impossible, state it and explain why.

Advice for Answer: See the answer to part (e).

- (c) List 6 distinct strings that belong to L_D . If this is impossible, state it and explain why.

Advice for Answer: See the answer to part (f).

(d) Write a regular expression that defines L_A . If such a regular expression does not exist, prove it.

Answer:

$$101(0 \cup 1)^*(1 \cup 00)$$

(e) Write a regular expression that defines L_R . If such a regular expression does not exist, prove it.

Answer:

$$(0 \cup 1)(0 \cup 1)(0 \cup 1) \cup (0 \cup 11 \cup 100)(0 \cup 1)^*$$

(f) Write a regular expression that defines L_D . If such a regular expression does not exist, prove it.

Answer:

$$101(0 \cup 1)^* 10 \cup 1010$$

(g) Explain how to construct an algorithm that solves the following problem:

INPUT: String w over $\{0, 1\}$.

OUTPUT: **yes** if w represents a Turing Machine that accepts exactly those strings on which the Turing Machine M (defined at the beginning of this problem) halts; **no** otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist. If it existed, it would decide the set of those Turing Machines whose languages have the non-trivial property: "is equal to $L_A \cup L_R$ ". By Rice's Theorem, this is impossible. This property is non-trivial because it is true for $L_A \cup L_R$ but false for L_D .

Problem 958 Consider the Turing machine $M = (Q, \Sigma, \Gamma, \delta, q)$ where: $\Sigma = \{0, 1\}$; $\Gamma = \{1, 0, B\}$; $Q = \{q, p, s, t, v\}$; and δ is defined by the following transition set:

$[q, 0, q, 0, R]$	$[t, 0, v, 0, L]$
$[q, 1, p, 1, R]$	$[t, 1, s, 1, R]$
$[q, B, q, B, R]$	$[v, 0, v, 0, L]$
$[p, 0, p, 0, R]$	$[s, 0, s, 0, R]$
$[p, 1, t, 1, L]$	$[s, 1, s, 1, R]$
$[p, B, p, B, R]$	$[s, B, s, B, R]$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right only.) B is the designated blank symbol.)

Let L be the set of string on which M halts.

(a) List 6 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: See the answer to part (b).

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Advice for Answer: M diverges unless it finds at least two occurrences of 1. If and when M finds the second occurrence of 1, it inspects the symbol immediately to the left of it—if this symbol is 1, then M diverges. Otherwise, M moves to the left skipping over zeros, and must halt when it finds the first occurrence of 1.

Answer:

$$0^* 10^* 01(0 \cup 1)^*$$

(c) Dangerous Professor has told her students to write a program that operates as follows:

INPUT: String w over $\{0, 1\}$.

OUTPUT: **yes** if w is an element of the set of exactly those strings on which the Turing Machine M (defined at the beginning of this problem) halts; **no** otherwise.

Explain the algorithm that should be employed by this program, or state that it does not exist and prove it.

Answer: Apply the known algorithm to convert the regular expression obtained in the answer to part (b) into a finite automaton, then apply the known algorithm to make this automaton deterministic, then run this deterministic automaton and decide as it does.

Problem 959 Consider the Turing machine $M = (Q, \Sigma, \Gamma, \delta, q)$ such that: $\Sigma = \{0, 1\}$; $\Gamma = \{B, 0, 1\}$; $Q = \{q, r, s, p, v, t, z, x, y\}$; and δ is defined by the following transition set:

$[q, 0, r, 0, R]$	$[v, 1, z, 0, L]$
$[r, 1, s, 1, R]$	$[v, 0, y, 0, L]$
$[s, 1, t, 1, R]$	$[z, 0, y, 1, L]$
$[t, 0, p, 0, R]$	$[z, 1, x, 1, L]$
$[t, 1, p, 1, R]$	$[y, 0, y, 0, R]$
$[p, 0, p, 0, R]$	$[y, 1, y, 1, R]$
$[p, 1, p, 1, R]$	$[y, B, y, B, R]$
$[p, B, v, B, L]$	

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right only.) B is the designated blank symbol.)

Let L be the set of strings on which M diverges.

(a) List 6 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: See the answer to part (b).

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Advice for Answer: M halts immediately unless it finds 011 as the prefix of the input string. Next, M requires at least one more symbol to follow the prefix 011, and proceeds to look at the end of the input string. If the input string ends with 0, then M diverges. If the input string ends with 1, then M inspects the penultimate symbol and diverges exactly when it finds there a 0.

Answer:

$$011(0 \cup 1)^*(0 \cup 01)$$

(c) Dangerous Professor has told her students to write a program that operates as follows:

INPUT: String w over $\{0, 1\}$.

OUTPUT: yes if w is a string that represents a Turing Machine which accepts exactly those strings that belong to the set L (defined at the beginning of this problem); no otherwise.

Explain the algorithm that should be employed by this program, or state that it does not exist, and prove it.

Answer: Impossible. If this algorithm existed, then it would decide the set of those Turing Machines w that accept languages that have a property

$$L(w) = L$$

which is a non-trivial property because L has it but 1^* does not. By Rice's Theorem, such an algorithm does not exist.

Problem 960 Consider the Turing machine

$M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $\Sigma = \{0, 1\}$; $\Gamma = \{B, 0, 1\}$; $Q = \{q, s, x, t, p, y, z, v, m\}$; $F = \{m\}$; and δ is defined by the following transition set:

$[q, 0, q, 0, R]$	$[y, 1, z, 1, L]$
$[q, 1, s, 1, R]$	$[y, 0, x, 0, R]$
$[q, B, x, B, R]$	
	$[z, 1, v, 1, R]$
$[s, 0, q, 0, R]$	$[z, 0, m, 0, R]$
$[s, 1, t, 1, L]$	
$[s, B, x, B, R]$	$[x, 0, x, 0, R]$
	$[x, 1, x, 1, R]$
$[t, 0, q, 0, R]$	$[x, B, x, B, R]$
$[t, 1, p, 1, R]$	
$[t, B, x, B, R]$	$[p, 0, p, 0, R]$
	$[p, 1, p, 1, R]$
	$[p, B, y, B, L]$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right only.) M accepts by final state. B is the designated blank symbol.)

Let L_1 be the set of string which M accepts.

Let L_2 be the set of string which M rejects.

(a) Write a regular expression that defines L_1 . If such a regular expression does not exist, prove it.

Advice for Answer: M diverges unless it finds a substring 11. If and when it finds the substring 11, it inspects the end of its input string. If the input string ends with 0, then M diverges. If the input string ends with 01, then M accepts. If the input string ends with 11, then M rejects.

Answer:

$$(0 \cup 1)^* 11 (0 \cup 1)^* 01$$

(b) Write a regular expression that defines L_2 . If such a regular expression does not exist, prove it.

Answer:

$$(0 \cup 1)^* 11$$

Question: Find (at least) one tuple in the transition function of M such that M never executes that tuple, and prove that the tuple indeed never gets executed.

(c) Dangerous Professor has told her students to write a program that operates as follows:

INPUT: String w over $\{0, 1\}$.

OUTPUT: yes if w is a string such that the Turing Machine represented by w halts exactly when the machine M (defined at the beginning of this problem) accepts; no otherwise.

Explain the algorithm that should be employed by this program, or state that it does not exist and prove it.

Answer: Impossible. If this algorithm existed, then it would decide the set of those Turing Machines w that accept languages that have a property

$$L(w) = L_1$$

which is a non-trivial property because L_1 has it but 1^* does not. By Rice's Theorem, such an algorithm does not exist.

Problem 961 Consider the following Turing machine: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ such that: $\Sigma = \{0, 1\}$;

$Q = \{q, s, x, t, p, y, z, v, m\}$; $\Gamma = \{B, 0, 1\}$; $F = \{m\}$; and δ is defined by the following transition set:

$[q, 0, q, 0, R]$	$[y, 1, z, 1, L]$
$[q, 1, s, 1, R]$	$[y, 0, x, 0, R]$
$[q, B, x, B, R]$	
	$[z, 1, x, 1, R]$
$[s, 0, q, 0, R]$	$[z, 0, v, 0, R]$
$[s, 1, t, 1, L]$	
$[s, B, x, B, R]$	$[v, 1, x, 1, R]$
	$[v, 0, m, 0, R]$
$[t, 0, q, 0, R]$	
$[t, 1, p, 1, R]$	$[x, 0, x, 0, R]$
$[t, B, x, B, R]$	$[x, 1, x, 1, R]$
	$[x, B, x, B, R]$
$[p, 0, p, 0, R]$	
$[p, 1, p, 1, R]$	$[p, B, y, B, L]$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right only.) M accepts by final state. B is the designated blank symbol.)

Let L be the set of string which M accepts.

(a) List 6 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer: Impossible, because M does not accept any input string.

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer: \emptyset

Advice for Answer: Straightforwardly, M diverges unless it finds two consecutive occurrences of 1. If it finds two consecutive occurrences of 1, then it returns to the left to examine the first of them (in state t) which then leads to skipping over the entire input (moving to the right in state p) and inspection of the last input symbol (in state y .) If the last symbol is 0, M escapes to diverge. If the last symbol is 1, M inspects the next-to-last symbol (in state z .) If the next-to-last symbol is 1, it escapes to diverge. If the next-to-last symbol is 0, it returns to inspect the last symbol again (in state v .) Since the last symbol (in this case) is always 1, it escapes to diverge.

(c) Dangerous Professor has told her students to write a program that operates as follows:

INPUT: String w over $\{0, 1\}$.

OUTPUT: yes if w is a string such that the Turing Machine represented by w halts exactly when the machine M (defined at the beginning of this problem) accepts;

no otherwise.

Explain the algorithm that should be employed by this program, or state that it does not exist and prove it.

Answer: This algorithm does not exist. If it existed, it would decide the set of Turing Machines whose languages have the following property:

is empty.

This property is non-trivial because (by construction) the language $L(M)$ has it, while the language $\overline{L(M)}$ does not have it. Hence, by Rice's Theorem the described algorithm is impossible.

Problem 962 Let L be the set of strings over alphabet $\{a, b, c\}$ whose length is not divisible by 3.

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$((a \cup b \cup c)(a \cup b \cup c)(a \cup b \cup c))^*(a \cup b \cup c)(a \cup b \cup c \cup \lambda)$$

(b) Is L^* decidable? Explain your answer briefly.

Answer: Yes- L^* is regular because L is regular and the class of regular languages is closed under Kleene star; every regular language is decidable.

Problem 963 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, p)$$

such that: $Q = \{p, q, s, t, c\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, X, Y, Z, a, b, c\}$.

and δ is defined by the following transition set:

$$\begin{aligned} & [p, a, q, X, R] \\ & [p, b, q, Y, R] \\ & [p, c, q, Z, R] \end{aligned}$$

$$\begin{aligned} & [q, a, q, a, R] \\ & [q, b, q, b, R] \\ & [q, c, q, c, R] \\ & [q, B, s, B, L] \end{aligned}$$

$$\begin{aligned} & [s, X, t, B, R] \\ & [s, Y, t, B, R] \\ & [s, Z, t, B, R] \\ & [s, a, e, a, R] \\ & [s, b, e, b, R] \\ & [s, c, e, c, R] \end{aligned}$$

$$[e, B, e, B, R]$$

(where B is the designated blank symbol.)

Let L be the set of strings accepted by M (by halting.)

(a) Draw a state-transition graph of a finite automaton M' that accepts L . If such an automaton does not exist, prove it.

Answer: Observe that L is given by the regular expression:

$$a \cup b \cup c \cup \lambda$$

whence the automaton given on Figure 355.

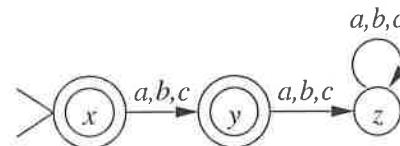


Figure 355:

(b) Write a complete formal definition of a Turing machine M_1 that accepts the language L and halts on every input. In short:

$$(\tau \in L) \implies (M_1(\tau) \downarrow \text{ and accept})$$

and also:

$$(\tau \notin L) \implies (M_1(\tau) \downarrow \text{ and reject})$$

Your construction should be readable as well as accurate; you may comment it. If such a Turing machine does not exist, prove it.

Answer: We employ the algorithmic conversion to obtain M_1 from the deterministic finite automaton constructed in the answer to part (a).

$$M_1 = (Q, \Sigma, \Gamma, \delta, x, F)$$

where: $Q = \{x, y, z\}$; $\Sigma = \{a, b, c\}$;

$\Gamma = \{B, a, b, c\}$; $F = \{x, y\}$.

and δ is defined by the following transition set:

$[x, a, y, B, R]$
 $[x, b, y, B, R]$
 $[x, c, y, B, R]$

$[y, a, z, B, R]$
 $[y, b, z, B, R]$
 $[y, c, z, B, R]$

$[z, a, z, B, R]$
 $[z, b, z, B, R]$
 $[z, c, z, B, R]$

(where B is the designated blank symbol.)

(c) Write a complete formal definition of a Turing machine M_2 that halts on every input and recognizes Turing machines that accept L . In short:

$$(L(\eta) = L) \implies (M_2(\eta) \downarrow \text{ and accept})$$

and also:

$$(L(\eta) \neq L) \implies (M_2(\eta) \downarrow \text{ and reject})$$

Your construction should be readable as well as accurate; you may comment it. If such a Turing machine does not exist, prove it.

Answer: Such Turing machine does not exist, since it would decide if an arbitrary Turing machine accepts a language equal to L . The property “is equal to L ” is non-trivial—hence, by Rice’s Theorem it is undecidable whether it holds for the language accepted by an arbitrary Turing machine.

Problem 964 Let L be the language accepted by the Turing machine: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ such that:

$Q = \{q, r, s\}$; $\Sigma = \{a, b\}$; $\Gamma = \{B, a, b\}$; $F = \{s\}$;

and δ is defined by the following transition set:

$[q, a, r, a, L]$
 $[q, b, q, b, R]$
 $[q, B, q, B, R]$
 $[r, b, s, b, R]$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.)

M accepts by final state. B is the designated blank symbol.)

(a) Write a regular expression that defines L . If such a regular expression does not exist, explain why.

Answer: $b^*ba(a \cup b)^*$

(Observe that M diverges unless it finds an a in the input. Once the first a is found, M halts after visiting the symbol to the left of the first a . Therefore, there has to be at least one symbol before the leftmost a .)

Recall that L is the language defined in part (a).

(b) Explain the construction of an algorithm that operates as follows.

INPUT: A pair of strings x, y over Σ .

OUTPUT: **yes** if x is a Turing machine that accepts L , and y is a string that belongs to L ;
no otherwise.

If such an algorithm does not exist, state it and explain why.

Answer: This algorithm does not exist. If it existed, it would decide whether the Turing machine x accepts a language that has the nontrivial property of being equal to L . By Rice’s theorem, this is impossible.

(c) Explain the construction of an algorithm that operates as follows.

INPUT: A pair of strings x, y over Σ .

OUTPUT: **yes** if x is a string not accepted by any Turing machine that accepts L and y is a string that belongs to L ;
no otherwise.

If such an algorithm does not exist, state it and explain why.

Answer: The algorithm says **yes** exactly when:

$$x \notin L \wedge y \in L$$

(and **no** otherwise.)

To accomplish this, the algorithm should simulate a (deterministic) finite automaton A that accepts L . A nondeterministic finite automaton that accepts L is constructed by an algorithmic conversion of the regular expression obtained in the answer to part (a); this automaton is converted to a deterministic one by an application of the conversion algorithm. The algorithm should return **yes** exactly when A rejects x but accepts y .

(d) Is L recursively enumerable? Explain your answer briefly.

Answer: Yes—every language accepted by a (nondeterministic) Turing machine is recursively enumerable, by definition.

Problem 965 Consider the Turing machine:

$M = (Q, \Sigma, \Gamma, \delta, q, F)$ such that:

$Q = \{q, r, s, t, x, y, z\}$; $\Sigma = \{a, b\}$; $\Gamma = \{B, a, b\}$; $F =$

$\{t\}$; and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, r, a, R] \\ & [q, b, s, b, R] \end{aligned}$$

$$\begin{aligned} & [r, a, r, a, R] \\ & [r, b, r, b, R] \\ & [r, B, x, B, L] \end{aligned}$$

$$\begin{aligned} & [s, a, s, a, R] \\ & [s, b, s, b, R] \\ & [s, B, y, B, L] \end{aligned}$$

$$\begin{aligned} & [x, a, t, a, R] \\ & [x, b, z, B, R] \end{aligned}$$

$$\begin{aligned} & [y, b, t, b, R] \\ & [y, a, z, B, R] \end{aligned}$$

$$[z, B, z, B, R]$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only).)

M accepts by final state. B is the designated blank symbol.)

Let L_d be the set of strings on which M diverges.

(a) Write a regular expression that defines L_d . If such a regular expression does not exist, explain why.

Answer: $a(a \cup b)^*b \cup b(a \cup b)^*a$

(b) Does there exist a Turing Machine M' that accepts L_d by halting? Explain your answer.

Answer: Yes—the machine M'' constructed in the answer to part (c) accepts L_d by final state, but can be converted algorithmically to another machine, namely M' , that accepts by halting. To this end, introduce a new state e for M'' , such that M'' in e moves to the right on every tape symbol; and add to M'' a tuple $[s, a, e, a, R]$ whenever M'' does not have a transition from s on a and s is not final in M'' .

(c) Does there exist a Turing Machine M'' that accepts L_d and halts on every input? Explain your answer.

Answer: Yes. To construct this Turing machine, start with a nondeterministic finite automaton that accepts L_d , which is constructed by an algorithmic conversion of the regular expression obtained in the answer to part (c); this automaton is converted to a deterministic one by an application of the conversion algorithm. The deterministic finite automaton is straightforwardly converted into a deterministic Turing Machine—the simulating Turing Machine M'' behaves exactly as the simulated finite automaton, moving right-only read-only on the tape; by construction, M'' halts on every input.

(d) Is L_d recursive? Explain your answer briefly.

Answer: Yes. L_d is regular, as witnessed by the regular expression constructed in the answer to part (a). Every regular language is recursive.

Problem 966 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B\}$, and P is:

$$S \rightarrow aSb \mid cSb \mid \lambda$$

(a) List 5 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer:

$$\lambda, ab, cb, acbb, aabb$$

(b) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: This automaton does not exist, since L is not regular.

Observe that L comprises exactly those strings that can be written in the form: $\xi^n b^n$ where every ξ is either an a or a c .

To prove that L is not regular, assume the opposite, that L is regular. Let k be the constant as in the Pumping Lemma for L . Consider a word $w \in L$, such that $w = a^m b^m$, where $m > k$.

Let $w = uvx$ be a decomposition of w , where v is the pumping part. By the Lemma, it must be that $|uv| < k < m$. Hence, the pumping part v is entirely within the a -segment, meaning that $v = a^j$ for some j such that $0 < j \leq k$. After pumping once, we obtain a word: $w_p = a^{m+j} b^m$. However, $m + j > m$, meaning that w_p has more a 's than b 's, which in turn implies that $w_p \notin L$. Recall that G is the grammar defined in part (a).

(c) Explain the construction of an algorithm that operates as follows.

INPUT: A deterministic Turing Machine T .

OUTPUT: **yes** if T accepts the language generated by G ; **no** otherwise.

If such an algorithm does not exist, state it and explain why.

Answer: This algorithm does not exist. If it existed, it would decide whether the Turing machine T accepts a language that has the nontrivial property of being equal to the language generated by G . By Rice's theorem, this is impossible.

(d) Explain the construction of an algorithm that operates as follows.

INPUT: A deterministic Turing Machine T .

OUTPUT: **yes** if T accepts a context-free language; **no** otherwise.

If such an algorithm does not exist, state it and explain why.

Answer: This algorithm does not exist. If it existed, it would decide whether the Turing machine T accepts a language that has the nontrivial property of being context-free. By Rice's theorem, this is impossible.

Problem 967 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, r\}$; $\Sigma = \{a, b\}$; $\Gamma = \{A, Z\}$; $F = \{r\}$; and the transition function δ is defined as follows:

$$\begin{aligned} & [q, b, \lambda, r, Z] \\ & [q, b, \lambda, q, A] \\ & [r, a, A, r, \lambda] \\ & [r, \lambda, Z, r, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

- (a) List 5 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer: $b, bba, bbbaa, bbbbaaa, bbbbbaaaa$

(The general template is: $b^{n+1}a^n$, $n \geq 0$.)

- (b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S\}$, and P is:

$$S \rightarrow bSa \mid b$$

Recall that M is the pushdown automaton defined in part (a).

- (c) Explain the construction of an algorithm that operates as follows.

INPUT: A string x over Σ .

OUTPUT: yes if x is an element of the language accepted by M ;

no otherwise.

If such an algorithm does not exist, state it and explain why.

Answer: The algorithm should simulate the pushdown automaton M , and accept exactly when M does. A pushdown automaton is straightforwardly simulated by a Turing Machine—the simulating Turing machine T behaves exactly as the simulated pushdown automaton M , moving right-only read-only on the input tape, and simulating the stack on another tape. Since T is nondeterministic (because M is nondeterministic), our algorithm should simulate T according to the simulation algorithm by M_u —the universal deterministic Turing Machine.

- (d) Explain the construction of an algorithm that operates as follows.

INPUT: A string x over Σ .

OUTPUT: yes if x is a representation of a Turing Machine that accepts the language accepted by M ;

no otherwise.

If such an algorithm does not exist, state it and explain why.

Answer: This algorithm does not exist. If it existed, it would decide whether the Turing machine T accepts a language that has the nontrivial property of being equal to the language accepted by M . By Rice's theorem, this is impossible.

Problem 968 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow aS \mid bA \mid cS \mid a \mid c \mid \lambda \\ A &\rightarrow aA \mid bB \mid cA \\ B &\rightarrow aB \mid bS \mid cB \mid b \end{aligned}$$

- (a) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: This grammar is regular, hence the construction of the automaton is algorithmic. See Figure 124.

- (b) Explain the construction of an algorithm that operates as follows.

INPUT: A regular context-free grammar G .

OUTPUT: A deterministic finite automaton M that accepts the language generated by G .

If such an algorithm does not exist, state it and explain why.

Answer: A regular context-free grammar $G = (V, \Sigma, P, S)$ is algorithmically convertible to an equivalent finite automaton $M' = (V \cup \{Z\}, \Sigma, \delta, S, \{Z\})$, where $Z \notin V$. For every rule of G of the form $A \rightarrow aB$, where $A, B \in V$, $a \in \Sigma$, δ contains a tuple: $[A, a, B]$. For every rule of G of the form $A \rightarrow a$, where $A \in V$, $a \in (\Sigma \cup \{\lambda\})$, δ contains a tuple: $[A, a, Z]$. If M' is not deterministic, it is converted into a deterministic automaton, by the algorithm for deterministic conversion.

- (c) Explain the construction of an algorithm that operates as follows.

INPUT: A deterministic Turing Machine T .

OUTPUT: A deterministic finite automaton M that accepts the language accepted by T .

If such an algorithm does not exist, state it and explain why.

Answer: This algorithm does not exist. If it existed, it would decide whether the Turing machine T accepts a language that has the nontrivial property of being regular. By Rice's theorem, this is impossible.

Problem 969 Let L be the set of all strings over alphabet $\{a, b, c\}$ that do not end with a .

- (a) Write a regular expression that represents the language L . If such a regular expression does not exist, prove it.

Answer:

$$\lambda \cup (a \cup b \cup c)^* (b \cup c)$$

- (b) Construct a state-transition graph of a deterministic finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: A straightforward way is to start with a finite automaton that accepts \overline{L} , which is given on Figure 356. Since this is an extremely small automaton, it

is promptly convertible into a deterministic one, which is in turn directly “complemented.” The answer is given on Figure 357.

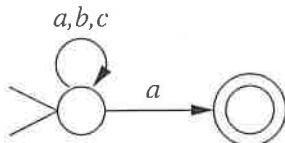


Figure 356:

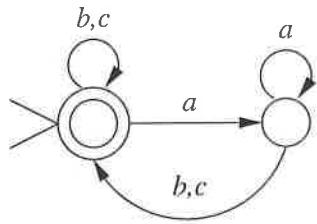


Figure 357:

(c) Explain how to construct an algorithm that solves the following problem:

INPUT: A string x and a regular expression e over the alphabet $\{0, 1\}$.

OUTPUT: **yes** if x is a Turing Machine that accepts the language defined by e ;
no otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist. If it existed, then we could fix the expression e , so that, for instance, $e = \emptyset$. Then, the algorithm would recognize those Turing Machines that accept languages with the non-trivial property:

“is empty”

This property is evidently non-trivial, since the language \emptyset has it, while the language a^* does not have it. By Rice’s theorem, there is no algorithm to recognize Turing Machines whose languages have a given non-trivial property.

Problem 970 (a) Let L be the language accepted (by final state) by the Turing Machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$\begin{aligned} Q &= \{q, r, s, t, v, x, y\}; \\ \Sigma &= \{0, 1\}; \\ \Gamma &= \{B, 0, 1\}; \\ F &= \{t\}; \end{aligned}$$

and δ is defined by the following transition set:

$$\begin{aligned} &[q, 0, r, 0, R] \\ &[q, 1, s, 1, R] \\ &[r, 0, r, 0, R] \\ &[r, 1, r, 1, R] \\ &[r, B, r, B, R] \\ &[s, 1, v, 1, R] \\ &[v, 0, v, 0, R] \\ &[v, 1, v, 1, R] \\ &[v, B, x, B, L] \\ &[x, 1, t, B, L] \\ &[t, 0, y, 0, R] \\ &[y, B, y, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$11(0 \cup 1)^* 11 \cup 11 \cup 111$$

(b) Explain how to construct an algorithm that solves the following problem:

INPUT: A string x over $\{0, 1\}$.

OUTPUT: **yes** if x is a string that belongs to LL , where L is the language defined in part (a);
no otherwise.

If this algorithm does not exist, prove it.

Answer: First, apply the known algorithm to convert the regular expression $11(0 \cup 1)^* 11 \cup 11 \cup 111$, which represents L , into a finite automaton, say F , that accepts L . Next, apply the known algorithm to convert F into a finite automaton, say H , that accepts LL . Finally, apply the known algorithm to convert H into a deterministic finite automaton, say D . Our algorithm should simulate the automaton D on x , returning **yes** if D accepts x and returning **no** if D rejects x .

(c) Explain how to construct an algorithm that solves the following problem:

INPUT: A string y over $\{0, 1\}$.

OUTPUT: **yes** if y is a string that contains an even number of 0’s and belongs to L , the language defined in part (a);
no otherwise.

If this algorithm does not exist, prove it.

Answer: Let L_1 denote the set of all strings over $\{0, 1\}$ that contain an even number of 0’s. Observe that our algorithm is required to decide the language $L \cap L_1$.

First, apply the known algorithm to convert the regular expression $11(0 \cup 1)^* 11 \cup 11 \cup 111$, which represents L , into a finite automaton, say F , that accepts L . Analogously, apply the known algorithm to convert the regular expression $(1^* 0 1^* 0 1^*)^* 1^*$, which represents L_1 , into a finite automaton, say F_1 , that accepts L_1 . Next, apply the known algorithm to construct a finite automaton, say T , which accepts the intersection

$L \cap L_1 = L(F) \cap L(F_1)$. Finally, apply the known algorithm to convert T into a deterministic finite automaton, say D . Our algorithm should simulate the automaton D on y , returning **yes** if D accepts y and returning **no** if D rejects y .

Problem 971 (a) Consider the following Turing Machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$Q = \{q, r, s, v, t, x, y\};$$

$$\Sigma = \{0, 1\};$$

$$\Gamma = \{B, 0, 1\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, 0, r, 0, R] \\ & [r, 1, s, 1, R] \\ & [s, 1, v, 1, R] \\ & [v, 0, v, 0, R] \\ & [v, 1, v, 1, R] \\ & [v, B, x, B, L] \\ & [x, 1, t, B, L] \\ & [t, 0, y, 0, R] \\ & [y, B, y, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

Let L be the set of string on which M diverges.

Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$011(0 \cup 1)^* 01$$

(b) Explain how to construct an algorithm that solves the following problem:

INPUT: A string x over $\{0, 1\}$.

OUTPUT: **yes** if x is a Turing Machine that accepts the language \bar{L} , the complement of the language defined in part (a);

no otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist. If it existed, it would recognize those Turing Machines that accept languages with the non-trivial property:

“is equal to \bar{L} ”

This property is evidently non-trivial, since the language \bar{L} has it, while the language L does not have it. By Rice’s theorem, there is no algorithm to recognize Turing Machines whose languages have a given non-trivial property.

(c) Explain how to construct an algorithm that solves the following problem:

INPUT: A string y over $\{0, 1\}$.

OUTPUT: **yes** if y is a Turing Machine that accepts the language L^* , the Kleene star of the language defined in

part (a);

no otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist. If it existed, it would recognize those Turing Machines that accept languages with the non-trivial property:

“is equal to L^* ”

This property is evidently non-trivial, since the language L^* has it, while the language \emptyset does not have it. By Rice’s theorem, there is no algorithm to recognize Turing Machines whose languages have a given non-trivial property.

Problem 972 (a) Let L be the language accepted (by final state) by the Turing Machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$Q = \{q, r, s, t, v, x, y\};$$

$$\Sigma = \{0, 1\};$$

$$\Gamma = \{B, 0, 1\};$$

$$F = \{t\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, 0, r, 0, R] \\ & [r, 1, s, 1, R] \\ & [s, 1, v, 1, R] \\ & [v, 0, v, 0, R] \\ & [v, 1, v, 1, R] \\ & [v, B, x, B, L] \\ & [x, 0, t, B, L] \\ & [t, 1, y, 1, R] \\ & [y, B, y, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$011(0 \cup 1)^* 00$$

(b) Explain how to construct an algorithm that solves the following problem:

INPUT: A string x over $\{0, 1\}$.

OUTPUT: **yes** if x is a Turing Machine that accepts at least one string that belongs to the language L defined in part (a);

no otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist. If it existed, it would recognize those Turing Machines that accept languages with the non-trivial property:

“contains at least one string that belongs to $011(0 \cup 1)^* 00$ ”

This property is evidently non-trivial, since the language $011(0 \cup 1)^*00$ has it, while the language \emptyset does not have it. By Rice's theorem, there is no algorithm to recognize Turing Machines whose languages have a given non-trivial property.

(c) Explain how to construct an algorithm that solves the following problem:

INPUT: A string y over $\{0, 1\}$.

OUTPUT: yes if y is a string that belongs to \bar{L} , the complement of the language L defined in part (a); no otherwise.

If this algorithm does not exist, prove it.

Answer: Apply the known algorithm to convert the regular expression $011(0 \cup 1)^*00$ into a finite automaton, say F , and then apply the known algorithm to convert F to a deterministic finite automaton F_1 ; next, apply the known algorithm to convert the automaton F_1 into a Turing Machine, say D , which decides the language $L = 011(0 \cup 1)^*00$. Our algorithm should answer yes exactly when D answers no; and should answer no exactly when D answers yes.

Problem 973 (a) Consider the following Turing Machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$Q = \{q, r, s, v, t, x, y\};$$

$$\Sigma = \{0, 1\};$$

$$\Gamma = \{B, 0, 1\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, 0, r, 0, R] \\ & [q, 1, s, 1, R] \\ & [r, 0, r, 0, R] \\ & [r, 1, r, 1, R] \\ & [r, B, r, B, R] \\ & [s, 1, v, 1, R] \\ & [v, 0, v, 0, R] \\ & [v, 1, v, 1, R] \\ & [v, B, x, B, L] \\ & [x, 1, t, B, L] \\ & [t, 0, y, 0, R] \\ & [y, B, y, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

Let L be the set of strings on which M diverges.

Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$0(0 \cup 1)^* \cup 11(0 \cup 1)^*01$$

(b) Explain how to construct an algorithm that solves the following problem:

INPUT: A string x over $\{0, 1\}$.

OUTPUT: yes if the Turing Machine M defined in part (a) diverges on input x ; no otherwise.

If this algorithm does not exist, prove it.

Answer: Apply the known algorithm to convert the regular expression obtained in the answer to part (a) into a finite automaton, say F , and then apply the known algorithm to convert F to a deterministic finite automaton F_1 ; next, apply the known algorithm to convert the automaton F_1 into a Turing Machine, say D , which decides the language L . This Turing machine D implements our algorithm.

(c) Explain how to construct an algorithm that solves the following problem:

INPUT: A string y over $\{0, 1\}$.

OUTPUT: yes if y is a Turing Machine that accepts the language L defined in part (a); no otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist. If it existed, it would recognize those Turing Machines that accept languages with the non-trivial property:

"is equal to L "

This property is evidently non-trivial, since the language L has it, while the language \emptyset does not have it. By Rice's theorem, there is no algorithm to recognize Turing Machines whose languages have a given non-trivial property.

Problem 974 (a) Let L be the language accepted (by final state) by the Turing Machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$Q = \{q, r, s, t, v, x, y\};$$

$$\Sigma = \{0, 1\};$$

$$\Gamma = \{B, 0, 1, Z\};$$

$$F = \{t\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, 0, r, Z, R] \\ & [r, 1, s, Z, R] \\ & [s, 0, v, Z, R] \\ & [v, 0, v, 0, R] \\ & [v, 1, v, 1, R] \\ & [v, B, x, B, L] \\ & [x, 0, y, B, L] \\ & [y, Z, t, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$0100$$

(Exactly one string.)

(b) Is \overline{L} , the complement of the language L defined in part (a), decidable? Prove your answer.

Answer: Yes. L is finite and thereby evidently regular; the complement of every regular language is regular—hence, \overline{L} is also regular; every regular language is decidable— \overline{L} is also decidable.

(c) Explain how to construct an algorithm that solves the following problem:

INPUT: A string x over $\{0,1\}$.

OUTPUT: **yes** if x belongs to \overline{L} , the complement of the language L defined in part (a); **no** otherwise.

If this algorithm does not exist, prove it.

Answer: Perform the following algorithmic steps:

1. convert the regular expression obtained in the answer to part (a) into an equivalent finite automaton A ;
2. convert A into an equivalent deterministic finite automaton D ;
3. convert D into a deterministic finite automaton D' that accepts $\overline{L(D)} = \overline{L}$; on input x decide as D' does.

(d) Explain how to construct an algorithm that solves the following problem:

INPUT: A string x over $\{0,1\}$.

OUTPUT: **yes** if x is a Turing Machine that accepts \overline{L} , the complement of the language L defined in part (a); **no** otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist. If it existed, it would decide whether the language accepted by the input Turing machine x has the nontrivial property of being “equal to \overline{L} ”. However, by Rice’s Theorem, there is no algorithm to recognize those Turing machines that accept a language with any non-trivial property.

Problem 975 (a) Let L be the language accepted (by final state) by the Turing Machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that: $Q = \{q, r, s, t, v, x, y\}$;
 $\Sigma = \{0,1\}$; $\Gamma = \{B, 0, 1, Z\}$; $F = \{t\}$;

and δ is defined by the following transition set:

$[q, 0, r, Z, R]$
$[q, 1, r, Z, R]$
$[r, 0, s, Z, R]$
$[r, 1, s, Z, R]$
$[s, 0, v, Z, R]$
$[s, 1, v, Z, R]$
$[v, 0, v, 0, R]$
$[v, 1, v, 1, R]$
$[v, B, x, B, L]$
$[x, 1, y, B, L]$
$[y, B, t, B, R]$

(where B is the designated blank symbol.)

Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

\emptyset

(b) Let L' be the set of strings on which the Turing Machine M defined in part (a) diverges. Is L' decidable? Prove your answer.

Answer: Yes— L' is equal to \emptyset and evidently regular; every regular language is decidable.

(c) Explain how to construct an algorithm that solves the following problem:

INPUT: A string x over $\{0,1\}$.

OUTPUT: **yes** if x belongs to L' , the language defined in part (b); **no** otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm should always return **no**, since L' is empty.

(d) Explain how to construct an algorithm that solves the following problem:

INPUT: A string x over $\{0,1\}$.

OUTPUT: **yes** if x is a Turing Machine such that $L(x)$ contains no more than two strings accepted by M (the Turing machine defined in part (a)); **no** otherwise.

If this algorithm does not exist, prove it.

Answer: The algorithm should return **yes** if x is a syntactically correct Turing Machine and **no** otherwise. The syntactic correctness can be verified by simulating the pushdown automaton that is equivalent to the context-free grammar that defines all texts of Turing Machines.

To see that every recursively enumerable language satisfies the property “contains no more than two strings accepted by M ”, recall that M accepts an empty language, and the property is equivalent to: “has an intersection of size no more than 2 with an empty set.” However, every set has a zero-size intersection with the empty set—the property is trivial and satisfied by every recursively enumerable language.

Problem 976 Consider the Turing machine: $M = (Q, \Sigma, \Gamma, \delta, q)$ such that: $Q = \{q, p, s, t\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, a, b, c\}$; and δ is defined by the following transition set:

$[q, a, p, a, R]$
 $[q, b, q, b, R]$
 $[q, c, q, c, R]$

$[p, a, p, a, R]$
 $[p, B, t, B, L]$

$[t, a, t, a, L]$
 $[t, c, s, c, R]$

$[s, a, s, a, R]$
 $[s, b, s, b, R]$
 $[s, c, s, c, R]$
 $[s, B, s, B, R]$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.)

Let L be the set of strings on which the Turing machine M diverges.

(a) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 358.

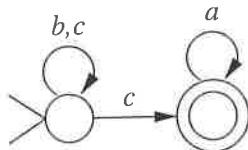


Figure 358:

(M always halts at the end of the input unless it has seen a . If it sees an a , then it also halts, unless the entire input on the right-hand side of the first a also consists of a 's alone. If this is the case, the machine tests the last symbol before these a 's. (Only if this symbol is c , M diverges.)

(b) Describe the construction of a Turing Machine T which halts on every input and also accepts exactly those strings on which the Turing Machine M diverges. In short:

$$(M(\tau) \nearrow) \implies (T(\tau) \searrow \text{and accept})$$

$$(M(\tau) \searrow) \implies (T(\tau) \searrow \text{and reject})$$

If such a Turing machine T does not exist, prove it.

Answer: Construction:

- convert the automaton obtained in the answer to part (a) into a deterministic one;

- simulate this deterministic finite automaton and return exactly what it returns.

Problem 977 Consider the Turing machine: $M = (Q, \Sigma, \Gamma, \delta, q)$ such that: $Q = \{q, p, r, s, t, v\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, a, b, c\}$; and δ is defined by the following transition set:

$[q, a, p, a, R]$
 $[q, b, q, b, R]$
 $[q, c, q, c, R]$
 $[q, B, t, B, L]$

$[p, a, s, a, L]$
 $[p, b, s, b, L]$
 $[p, c, s, c, L]$
 $[p, B, s, B, L]$

$[t, c, r, c, L]$
 $[r, c, s, c, R]$
 $[r, b, v, b, R]$

$[s, a, s, a, R]$
 $[s, b, s, b, R]$
 $[s, c, s, c, R]$
 $[s, B, s, B, R]$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.)

Let L be the set of strings on which the Turing machine M halts.

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(b \cup c)^* b (c \cup \lambda)$$

(By inspection of the first, second, and last blocks of the code, one concludes that M diverges whenever the input string contains a . Otherwise, M visits the end of the input and stays there unless the rightmost input symbol is c . If the rightmost input symbol is c , the machine moves, but diverges only if the next-to-last symbol is also c .)

(b) Describe the construction of a Turing Machine T which halts on every input and also accepts exactly those strings that represent Turing Machines that accept the complement of the language accepted by M . In short:

$$(L(\tau) = \overline{L(M)}) \implies (T(\tau) \searrow \text{and accept})$$

$$(L(\tau) \neq \overline{L(M)}) \implies (T(\tau) \searrow \text{and reject})$$

If such a Turing machine T does not exist, prove it.

Answer: This is impossible. To prove it, observe that the property " $= \overline{L(M)}$ " is nontrivial, since $\overline{L(M)}$ has this property, but Σ^* does not have it. If T existed, it would decide the set of all those Turing Machines whose languages have the nontrivial property " $= \overline{L(M)}$ ", which is impossible, by Rice's Theorem.

Problem 978 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that: $Q = \{q, p, r, s, t, v\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, a, b, c\}$; $F = \{t\}$; and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, p, a, R] \\ & [q, b, q, b, R] \\ & [q, c, q, c, R] \\ & [q, B, t, B, L] \end{aligned}$$

$$\begin{aligned} & [p, a, s, a, L] \\ & [p, b, s, b, L] \\ & [p, c, s, c, L] \\ & [p, B, s, B, L] \end{aligned}$$

$$[t, c, r, c, L]$$

$$\begin{aligned} & [r, c, s, c, R] \\ & [r, b, v, b, R] \end{aligned}$$

$$\begin{aligned} & [s, a, s, a, R] \\ & [s, b, s, b, R] \\ & [s, c, s, c, R] \\ & [s, B, s, B, R] \end{aligned}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.)

Let L be the set of strings on which the Turing machine M (halts and) rejects.

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(b \cup c)^*bc$$

(By inspection of the first, second, and last blocks of the code, one concludes that M diverges whenever the input string contains a . Otherwise, M visits the end of the input and stays there in the final state t unless the rightmost input symbol is c . If the rightmost input symbol is c , the machine exits its single final state t , but halts in the non-final state v if the next-to-last symbol is b .)

(b) Describe the construction of a Turing Machine T which halts on every input and also accepts exactly those strings which M rejects. In short:

$$(M(\tau) \searrow \text{and reject}) \implies (T(\tau) \searrow \text{and accept})$$

$$(\neg(M(\tau) \searrow \text{and reject})) \implies (T(\tau) \searrow \text{and reject})$$

If such a Turing machine T does not exist, prove it.

Answer: Construction:

- convert the regular expression obtained in the answer to part (a) into a (nondeterministic) finite automaton;

- convert this automaton into a deterministic one;
- simulate this deterministic finite automaton and return exactly what it returns.

Problem 979 Consider the Turing machine:

$(M = (Q, \Sigma, \Gamma, \delta, q, F))$ such that:

$Q = \{q, r, s, t, x, y, w\}$; $\Sigma = \{0, 1\}$; $\Gamma = \{B, 0, 1\}$; $F = \{t\}$; and δ is defined by the transition set:

$$\begin{aligned} & [q, 0, r, 0, R] \\ & [q, 1, s, 1, R] \\ & [q, B, q, B, R] \\ & [r, 0, y, 0, R] \\ & [r, 1, r, 1, R] \\ & [r, B, x, B, L] \\ & [s, 0, s, 0, R] \\ & [s, 1, y, 1, R] \\ & [s, B, w, B, L] \\ & [x, 1, t, 1, L] \\ & [w, 0, t, 0, L] \\ & [y, 0, y, 0, R] \\ & [y, 1, y, 1, R] \\ & [y, B, y, B, R] \end{aligned}$$

(B is the designated blank symbol.)

(a) Write a regular expression that defines the set of strings that M accepts. If such a regular expression does not exist, prove it.

Answer: $011^* \cup 100^*$

(b) Write a regular expression that defines the set of strings that M rejects. If such a regular expression does not exist, prove it.

Answer: $0 \cup 1$

(c) Write a regular expression that defines the set of strings on which M diverges. If such a regular expression does not exist, prove it.

Answer: $\lambda \cup (00 \cup 11 \cup 011^*0 \cup 100^*1)(0 \cup 1)^*$

(d) Dangerous Professor has told her students to write a program that operates as follows:

INPUT: an arbitrary string x over $\{0, 1\}$;

OUTPUT: yes if x is a Turing Machine that accepts at least one string rejected by M ; no otherwise.

Explain the algorithm that should be employed by this program, or prove that it does not exist.

Answer: This algorithm does not exist. If it existed, it would decide whether its argument Turing machine accepts a language with the non-trivial property of having non-empty intersection with $(011^*0 \cup 100^*1)(0 \cup 1)^*$

By Rice's Theorem, this is impossible.

Problem 980 Consider the Turing machine:

$(M = (Q, \Sigma, \Gamma, \delta, q))$ such that:

$Q = \{q, r, s, y, w\}$; $\Sigma = \{0, 1\}$; $\Gamma = \{B, 0, 1\}$;
and δ is defined by the following transition set:

$[q, 0, q, 0, R]$
$[q, 1, s, 1, R]$
$[q, B, q, B, R]$
$[s, 0, r, 0, R]$
$[s, 1, y, 1, R]$
$[s, B, w, B, L]$
$[r, 0, y, 0, R]$
$[r, 1, s, 1, R]$
$[r, B, w, B, L]$
$[w, 0, w, 0, R]$
$[w, B, w, B, R]$

(B is the designated blank symbol.)

(a) Construct a finite-state automaton that accepts the set of strings on which M diverges. If such an automaton does not exist, prove it.

Advice for Answer: M diverges on: $0^*(10)^*$ and it is appropriate to construct a deterministic automaton, so as to be able to convert it in the answer to part (b).

Answer: Figure 359.

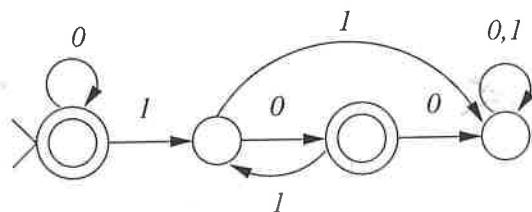


Figure 359:

(b) Construct a finite-state automaton that accepts the set of strings on which M halts. If such an automaton does not exist, prove it.

Answer: Figure 360.

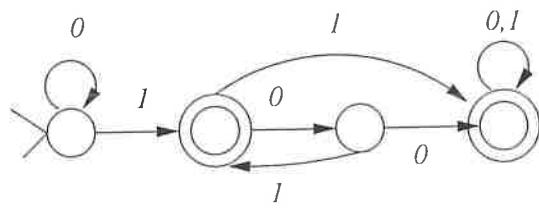


Figure 360:

(c) Dangerous Professor has told her students to write a program that operates as follows:

INPUT: an arbitrary string x over $\{0, 1\}$;

OUTPUT: **yes** if x is a string containing at least one letter, such that M diverges on x ;
no otherwise.

Explain the algorithm that should be employed by this program, or prove that it does not exist.

Answer: The following constructions are algorithmic:

- construct a finite automaton M_1 that accepts the set of all non-empty strings over Σ ;
- construct a finite automaton M_2 that accepts the intersection of languages accepted by M_1 and the automaton given on Figure 359;
- simulate M_2 by a Turing machine or an equivalent computer.

Problem 981 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$Q = \{q, p, s, t, v, w\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, a, b, c\}$;
and δ is defined by the following transition set:

$[q, a, q, a, R]$
$[q, b, p, b, R]$
$[q, c, q, c, R]$

$[p, a, p, a, R]$
$[p, b, s, b, R]$
$[p, c, p, c, R]$

$[s, a, t, a, L]$
$[s, b, t, b, L]$
$[s, c, t, c, L]$
$[s, B, t, B, L]$

$[t, a, v, a, L]$
$[t, b, v, b, L]$
$[t, c, w, c, L]$
$[t, B, v, B, L]$

$[w, a, w, a, R]$
$[w, b, w, b, R]$
$[w, c, w, c, R]$
$[w, B, w, B, R]$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.)

Let L be the set of strings on which the Turing machine M diverges.

(a) Draw a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 361 (which represents a finite automaton that accepts \emptyset .)



Figure 361:

To confirm that M never diverges, verify the following observations. If M does not reach s , it cannot diverge. M enters state s only and just after seeing the second b .

In all cases, M immediately steps back to the left and positions the head on that b , in state t . Hence, M always enters state v at this point, and in v it halts.

(b) Describe the construction of a Turing Machine T which halts on every input and also accepts exactly the strings that represent Turing Machines that accept some string contained in L . In short:

$$((\exists x \in L(\tau) \cap L)) \implies (T(\tau) \downarrow \text{ and accept})$$

$$(\neg(\exists x \in L(\tau) \cap L)) \implies (T(\tau) \downarrow \text{ and reject})$$

If such a Turing machine T does not exist, prove it.

Answer: On every input, T (immediately) halts and rejects.

To clarify, observe that T decides the set of Turing Machines whose languages have the property:

has a non-empty intersection with the empty set.

Since no set has a non-empty intersection with the empty set, there is no Turing Machine whose language has this property—the property is trivial.

Problem 982 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$Q = \{q, p, s, t, v, w\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, a, b, c\}$; and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, q, a, R] \\ & [q, b, p, b, R] \\ & [q, c, q, c, R] \end{aligned}$$

$$\begin{aligned} & [p, a, p, a, R] \\ & [p, b, s, b, R] \\ & [p, c, p, c, R] \end{aligned}$$

$$\begin{aligned} & [s, a, t, a, L] \\ & [s, b, t, b, L] \\ & [s, c, t, c, L] \\ & [s, B, t, B, L] \end{aligned}$$

$$\begin{aligned} & [t, a, v, a, L] \\ & [t, b, w, b, L] \\ & [t, c, v, c, L] \\ & [t, B, v, B, L] \end{aligned}$$

$$\begin{aligned} & [w, a, w, a, R] \\ & [w, b, w, b, R] \\ & [w, c, w, c, R] \\ & [w, B, w, B, R] \end{aligned}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.)

Let L be the set of strings on which the Turing machine M halts.

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(a \cup c)^*(\lambda \cup b)(a \cup c)^*$$

To confirm that M diverges on every string that has two (or more) b 's, observe that M enters state s only and just after seeing the second b . In all cases, M immediately steps back to the left and positions the head on that b , in state t . Hence, M always enters state w at this point, and from w it diverges.

(b) Describe the construction of a Turing Machine T which halts on every input and also accepts exactly the strings that begin with a substring that belongs to L . In short:

$$((\exists x, y)(x \in L \wedge \tau = xy)) \implies (T(\tau) \downarrow \text{ and accept})$$

$$(\neg(\exists x, y)(x \in L \wedge \tau = xy)) \implies (T(\tau) \downarrow \text{ and reject})$$

If such a Turing machine T does not exist, prove it.

Answer: On every input, T (immediately) halts and accepts.

To clarify, observe that L contains λ , which means that every string has a prefix that belongs to L , which in turn means that T should accept it.

Problem 983 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that: $Q = \{q, p, s, t, v, w\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, a, b, c\}$; $F = \{q, p\}$; and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, q, a, R] \\ & [q, b, p, b, R] \\ & [q, c, q, c, R] \end{aligned}$$

$$\begin{aligned} & [p, a, p, a, R] \\ & [p, b, s, b, R] \\ & [p, c, p, c, R] \end{aligned}$$

$$\begin{aligned} & [s, a, t, a, L] \\ & [s, b, t, b, L] \\ & [s, c, t, c, L] \\ & [s, B, t, B, L] \end{aligned}$$

$$\begin{aligned} & [t, a, v, a, L] \\ & [t, b, w, b, L] \\ & [t, c, v, c, L] \\ & [t, B, v, B, L] \end{aligned}$$

$$\begin{aligned} & [w, a, w, a, R] \\ & [w, b, w, b, R] \\ & [w, c, w, c, R] \\ & [w, B, w, B, R] \end{aligned}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.)

Let L be the set of strings on which the Turing machine M (halts and) rejects.

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

\emptyset

To confirm that M never rejects if it halts, observe that M halts in one of its accepting states if it does not see more than one b . Right after M sees the second b (if any), M enters state s , and then immediately steps back to the left and positions the head on this second b , in state t . Hence, M always enters state w at this point, and from w it diverges.

(b) Describe the construction of a Turing Machine T which halts on every input and also accepts exactly those strings that represent Turing Machines whose language is contained in L . In short:

$$(L(\tau) \subseteq L) \implies (T(\tau) \downarrow \text{ and accept})$$

$$(L(\tau) \not\subseteq L) \implies (T(\tau) \downarrow \text{ and reject})$$

If such a Turing machine T does not exist, prove it.

Answer: The machine T does not exist. If T existed, it would decide the set of Turing Machines whose languages satisfy the nontrivial property:

is contained in \emptyset

In other words, T would decide the set of Turing Machines whose languages satisfy the nontrivial property:

is empty

This property is evidently nontrivial, since the empty set has it, but Σ^* does not. Hence, by Rice's Theorem, T does not exist.

Problem 984 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$Q = \{q, p, r, s, t, v\}; \quad \Sigma = \{0, 1\}; \quad \Gamma = \{B, 0, 1\};$$

and δ is defined by the following transition set:

$$\begin{aligned} &[q, 0, p, 0, R] \\ &[q, 1, q, 1, R] \end{aligned}$$

$$\begin{aligned} &[p, 0, p, 0, R] \\ &[p, 1, p, 1, R] \\ &[p, B, t, B, L] \end{aligned}$$

$$\begin{aligned} &[t, 0, v, 0, R] \\ &[t, 1, r, 1, L] \end{aligned}$$

$$\begin{aligned} &[r, 0, v, 0, R] \\ &[r, 1, s, 1, R] \end{aligned}$$

$$\begin{aligned} &[s, 0, s, 0, R] \\ &[s, 1, s, 1, R] \\ &[s, B, s, B, R] \end{aligned}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.)

Let L be the set of strings on which the Turing machine M diverges.

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$1^*0(0 \cup 1)^*11$$

(b) Describe the construction of a Turing Machine T which halts on every input and also accepts exactly those strings on which the Turing Machine M diverges. In short:

$$(M(\tau) \nearrow) \implies (T(\tau) \downarrow \text{ and accept})$$

$$(M(\tau) \searrow) \implies (T(\tau) \downarrow \text{ and reject})$$

If such a Turing machine T does not exist, prove it.

Answer: Employ the known algorithms to construct, in sequence: a finite automaton M_1 equivalent to the regular expression obtained in the answer to part (a), a deterministic finite automaton M_2 equivalent to M_1 , a Turing Machine M_3 that simulates M_2 to decide L — M_3 should accept when M_2 accepts and reject when M_2 rejects.

Problem 985 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$Q = \{q, p, r, t\}; \quad \Sigma = \{0, 1\}; \quad \Gamma = \{B, 0, 1\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, 0, p, 0, R] \\ & [q, 1, q, 1, R] \\ & [q, B, q, B, R] \end{aligned}$$

$$\begin{aligned} & [p, 0, r, 0, R] \\ & [p, 1, r, 1, R] \\ & [p, B, p, B, R] \end{aligned}$$

$$\begin{aligned} & [r, 0, r, 0, R] \\ & [r, 1, t, 1, L] \end{aligned}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.)

Let L be the set of strings on which the Turing machine M halts.

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$1^*0(0 \cup 1)(0 \cup 1)^*$$

(b) Describe the construction of a Turing Machine T which halts on every input and also accepts strings that represent Turing Machines that halt exactly when M halts. In short:

$$(L(M) = L(\tau)) \implies (T(\tau) \downarrow \text{ and accept})$$

$$(L(M) \neq L(\tau)) \implies (T(\tau) \downarrow \text{ and reject})$$

If such a Turing machine T does not exist, prove it.

Answer: This Turing Machine does not exist. If it existed, it would decide whether an arbitrary Turing Machine accepts a language with the property:

is equal to L

where L is the language defined in part (a). This property is non-trivial (since L has it, but \emptyset does not.) By Rice's Theorem, there is no algorithm to decide whether the language accepted by an arbitrary Turing machine has a given non-trivial property.

Problem 986 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that: $Q = \{q, p, r, s, t\}$; $\Sigma = \{0, 1\}$; $\Gamma = \{B, 0, 1\}$; $F = \{t\}$; and δ is defined by the following

transition set:

$$\begin{aligned} & [q, 0, p, 0, R] \\ & [q, 1, q, 1, R] \end{aligned}$$

$$\begin{aligned} & [p, 0, r, 0, R] \\ & [p, 1, r, 1, R] \end{aligned}$$

$$\begin{aligned} & [r, 0, s, 0, R] \\ & [r, 1, s, 1, R] \\ & [r, B, t, B, R] \end{aligned}$$

$$\begin{aligned} & [s, 0, s, 0, R] \\ & [s, 1, s, 1, R] \\ & [s, B, s, B, R] \end{aligned}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol. M accepts by final state.)

Let $L = L(M)$ be the set of strings which the Turing machine M accepts.

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$1^*0(0 \cup 1)$$

(b) Describe the construction of a Turing Machine T which halts on every input and also accepts strings that represent Turing Machines that halt on all strings which M accepts. In short:

$$(L(M) \subseteq L(\tau)) \implies (T(\tau) \downarrow \text{ and accept})$$

$$(L(M) \not\subseteq L(\tau)) \implies (T(\tau) \downarrow \text{ and reject})$$

If such a Turing machine T does not exist, prove it.

Answer: This Turing Machine does not exist. If it existed, it would decide whether an arbitrary Turing Machine accepts a language with the property:

contains $L(M)$ as a subset

where L is the language defined in part (a). This property is non-trivial (since $L(M)$ itself has it, but \emptyset does not.) By Rice's Theorem, there is no algorithm to decide whether the language accepted by an arbitrary Turing machine has a given non-trivial property.

Problem 987 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, r\} \\ \Sigma &= \{a, b, c\} \\ \Gamma &= \{B\} \\ F &= \{r\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, r, B] \\ & [q, a, \lambda, r, BB] \\ & [q, a, \lambda, r, BBB] \\ & [r, b, B, r, \lambda] \\ & [r, c, \lambda, r, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

- (a) List 6 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer:

$$accbc, acccb, abccb, abcbccbc, abbb, acbcbc$$

- (b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, K\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aKbK \mid aKbKbK \mid aKbKbKbK \\ K &\rightarrow cK \mid \lambda \end{aligned}$$

- (c) Dangerous Professor has told her students to write a program that operates as follows:

INPUT: String w .

OUTPUT: **yes** if w represents a Turing Machine that accepts the language accepted by the pushdown automaton M (defined at the beginning of this problem); **no** otherwise.

Explain the algorithm that should be employed by this program, or state that it does not exist, and prove it.

Answer: This algorithm does not exist. If it existed, it would decide the set of Turing machines that accept exactly those languages that have the non-trivial property:

is equal to L

(where L defined at the beginning of this problem) which is impossible, by Rice's Theorem. To see that this property is non-trivial, observe that the language L has it, while the language \emptyset does not.

Problem 988 Consider the following Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$\begin{aligned} Q &= \{q, r, s, p, v, t, z, x\}; \\ \Sigma &= \{a, b, c\}; \\ \Gamma &= \{B, a, b, c\}; \\ F &= \{x\}; \end{aligned}$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, r, a, R] \\ & [r, b, s, b, R] \\ & [s, c, t, c, R] \\ & [t, a, p, a, R] \\ & [t, b, p, b, R] \\ & [t, c, p, c, R] \\ & [p, a, p, a, R] \\ & [p, b, p, b, R] \\ & [p, c, p, c, R] \\ & [p, B, v, B, L] \\ & [v, a, z, a, L] \\ & [z, b, x, b, R] \end{aligned}$$

(B is the designated blank symbol. M accepts by final state.)

Let L be the set of strings accepted by M .

- (a) List 6 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: See the answer to part (b).

- (b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$abc(a \cup b \cup c)^* ba$$

- (c) Dangerous Professor has told her students to write a program that operates as follows:

INPUT: String w over $\{a, b, c\}$.

OUTPUT: **yes** if w is a string that represents a Turing Machine which accepts at least one string accepted by the Turing Machine M (defined at the beginning of this problem); **no** otherwise.

Explain the algorithm that should be employed by this program, or state that it does not exist, and prove it.

Answer: This algorithm does not exist. If it existed, it would decide the set of Turing machines that accept exactly those languages with the the non-trivial property:

has a non-empty intersection with L

(where L defined at the beginning of this problem) which is impossible, by Rice's Theorem. To see that this property is non-trivial, observe that the language L has it, while the language \emptyset does not.

Problem 989 Consider the following Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$Q = \{q, p, s, t, v\}$; $\Sigma = \{0, 1\}$; $\Gamma = \{1, 0, B\}$;
and δ is defined by the following transition set:

$$\begin{array}{l} [q, 0, q, 0, R] \\ [q, 1, p, 1, R] \end{array}$$

$$\begin{array}{l} [p, 0, p, 0, R] \\ [p, 1, t, 1, L] \end{array}$$

$$\begin{array}{l} [t, 0, s, 0, R] \\ [t, 1, v, 1, L] \end{array}$$

$$\begin{array}{l} [s, 0, s, 0, R] \\ [s, 1, s, 1, R] \\ [s, B, s, B, R] \end{array}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.)

Let L be the set of string on which M diverges.

- (a) List 6 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: See the answer to part (b).

- (b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$0^* 10^* 01(0 \cup 1)^*$$

- (c) Dangerous Professor has told her students to write a program that operates as follows:

INPUT: String w over $\{0, 1\}$.

OUTPUT: **yes** if w is a string such that the Turing Machine M , (defined at the beginning of this problem) halts on input w ;
no otherwise.

Explain the algorithm that should be employed by this program, or state that it does not exist and prove it.

Answer:

1. convert the regular expression constructed in the answer to part (a) into an equivalent finite automaton F_1 ;
2. convert the automaton F_1 into an equivalent deterministic automaton F_2 ;
3. simulate F_2 and return the negation of its decision.

Problem 990 Consider the following Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$\begin{array}{l} Q = \{q, s, x, t, p, y, z, v\}; \\ \Sigma = \{0, 1\}; \end{array}$$

$$\Gamma = \{B, 0, 1\};$$

and δ is defined by the following transition set:

$$\begin{array}{l} [q, 0, q, 0, R] \\ [q, 1, s, 1, R] \\ [q, B, x, B, R] \end{array}$$

$$\begin{array}{l} [s, 0, q, 0, R] \\ [s, 1, t, 1, L] \\ [s, B, x, B, R] \end{array}$$

$$\begin{array}{l} [t, 0, q, 0, R] \\ [t, 1, p, 1, R] \\ [t, B, x, B, R] \end{array}$$

$$\begin{array}{l} [p, 0, p, 0, R] \\ [p, 1, p, 1, R] \\ [p, B, y, B, L] \end{array}$$

$$\begin{array}{l} [y, 1, z, 1, L] \\ [y, 0, x, 0, R] \end{array}$$

$$\begin{array}{l} [z, 1, x, 1, R] \\ [z, 0, v, 0, R] \end{array}$$

$$\begin{array}{l} [x, 0, x, 0, R] \\ [x, 1, x, 1, R] \\ [x, B, x, B, R] \end{array}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.)

Let L be the set of string on which M halts.

- (a) List 6 distinct strings that belong to L . If this is impossible, state it and explain why.

Advice for Answer: See the answer to part (b).

- (b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(0 \cup 1)^* 111 (0 \cup 1)^* 01$$

- (c) Dangerous Professor has told her students to write a program that operates as follows:

INPUT: String w over $\{0, 1\}$.

OUTPUT: **yes** if w is a string such that the Turing Machine represented by w halts exactly when the machine M (defined at the beginning of this problem) halts;
no otherwise.

Explain the algorithm that should be employed by this program, or state that it does not exist and prove it.

Answer: This algorithm does not exist. If it existed, it would decide the set of Turing machines that accept (by halting) exactly those languages that have the non-trivial property:

is equal to L

(where L defined at the beginning of this problem) which is impossible, by Rice's Theorem. To see that this property is non-trivial, observe that the language L has it, while the language \emptyset does not.

Problem 991 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where: $Q = \{q, s\}$; $\Sigma = \{a, b, c, d\}$; $\Gamma = \{A, B, K\}$; $F = \{s\}$; and the transition function δ is defined as follows:

$$\begin{aligned} & [q, d, \lambda, q, KA] \\ & [q, d, \lambda, q, KB] \\ & [q, d, \lambda, q, KK] \\ & [q, \lambda, \lambda, s, \lambda] \\ & [s, a, A, s, \lambda] \\ & [s, b, B, s, \lambda] \\ & [s, c, K, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) List 4 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer: $dac, ddacbc, dddcacbccc, \lambda$

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S\}$, and P is:

$$S \rightarrow dSac \mid dSbc \mid dScc \mid \lambda$$

(c) Describe the construction of a Turing Machine T which halts on every input and accepts exactly the strings that represent Turing Machines that accept languages contained in L . In short:

$$(L(\tau) \subseteq L) \implies (T(\tau) \downarrow \text{ and accept})$$

$$\text{otherwise} \implies (T(\tau) \downarrow \text{ and reject})$$

If such a Turing machine T does not exist, prove it.

Answer: This Turing machine does not exist. If it existed, it would decide the set of Turing machines whose languages have the nontrivial property:

is a subset of L

To see that the property is nontrivial, recall that \emptyset has it, but Σ^* does not. By Rice's Theorem, such a machine is impossible.

Problem 992 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where: $Q = \{q, s\}$; $\Sigma = \{a, b, c, d\}$; $\Gamma = \{B, K, D\}$; $F = \{q\}$; and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, s, KB] \\ & [q, a, \lambda, s, KD] \\ & [s, a, \lambda, s, \lambda] \\ & [s, b, B, s, \lambda] \\ & [s, c, K, q, \lambda] \\ & [s, d, D, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) List 4 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer: $abcadc, aabaacaaaabac, adacaaaabc, \lambda$

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(aa^*(b \cup d)a^*c)^*$$

(c) Describe the construction of a Turing Machine T which halts on every input and also accepts exactly the set of all strings that are contained in Σ^* but are not elements of L . In short:

$$(\tau \in \overline{L}) \implies (T(\tau) \downarrow \text{ and accept})$$

$$\text{otherwise} \implies (T(\tau) \downarrow \text{ and reject})$$

If such a Turing machine T does not exist, prove it.

Answer: All of the following steps are algorithmic: Convert the regular expression obtained in the answer to part (b) to a finite automaton. Convert this automaton to a deterministic one. Convert this deterministic automaton to one that accepts the complement of the original language. Simulate the latter automaton by a Turing machine which decides as the automaton does.

Problem 993 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, s, t\}$; $\Sigma = \{a, b, c, d\}$; $\Gamma = \{A, B, K, D\}$; $F = \{t\}$; and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, q, KA] \\ & [q, b, \lambda, q, KB] \\ & [q, \lambda, \lambda, s, \lambda] \\ & [s, \lambda, \lambda, t, \lambda] \\ & [s, c, \lambda, s, KK] \\ & [s, d, \lambda, s, KD] \\ & [t, a, A, t, \lambda] \\ & [t, b, B, t, \lambda] \\ & [t, c, K, t, \lambda] \\ & [t, d, D, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) List 4 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer: $\lambda, aac, abbcac, ccc$

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow aSac \mid bSbc \mid A \\ A &\rightarrow cAcc \mid dAdc \mid \lambda \end{aligned}$$

(c) Describe the construction of a Turing Machine T which halts on every input and also accepts exactly the strings that represent Turing Machines that accept some subset of the language accepted by the pushdown automaton M . In short:

$$(L(\tau) \subseteq L(M)) \implies (T(\tau) \downarrow \text{ and accept})$$

$$(L(\tau) \not\subseteq L(M)) \implies (T(\tau) \downarrow \text{ and reject})$$

If such a Turing machine T does not exist, prove it.

Answer: This is impossible. To prove it, observe that the property " $\subseteq L(M)$ " is nontrivial, since \emptyset has this property, but Σ^* does not have it. If T existed, it would decide the set of all those Turing Machines whose languages have the nontrivial property " $\subseteq L(M)$ ", which is impossible, by Rice's Theorem.

Problem 994 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where: $Q = \{q, s, t\}$; $\Sigma = \{a, b, c, d\}$; $\Gamma = \{A, B, K, D\}$; $F = \{q\}$; and the transition function δ is defined as follows:

$$\begin{aligned} &[q, d, \lambda, t, D] \\ &[t, \lambda, \lambda, s, \lambda] \\ &[t, a, \lambda, t, KAB] \\ &[s, a, A, s, \lambda] \\ &[s, b, B, s, \lambda] \\ &[s, c, K, s, \lambda] \\ &[s, d, D, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) List 4 distinct strings that belong to L . If this is impossible, state it and explain why.

$$\lambda, dd, dabacd, daabacbacddd$$

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow \lambda \mid SS \mid dAd \\ A &\rightarrow aAbac \mid \lambda \end{aligned}$$

(c) Describe the construction of a Turing Machine T which halts on every input and also accepts exactly those strings that are accepted by the pushdown automaton M . In short:

$$\tau \in L(M) \implies (T(\tau) \downarrow \text{ and accept})$$

$$\tau \notin L(M) \implies (T(\tau) \downarrow \text{ and reject})$$

If such a Turing machine T does not exist, prove it.

Answer: Simulate M and decide as M does.

(Since M is nondeterministic, simulate it with a nondeterministic Turing Machine, and then simulate this Turing Machine by M_u)

Problem 995 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that: $Q = \{q, p, r, s, t, v\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, a, b, c\}$; and δ is defined by the following transition set:

$$\begin{aligned} &[q, a, p, a, R] \\ &[q, b, q, b, R] \\ &[q, c, q, c, R] \end{aligned}$$

$$\begin{aligned} &[p, a, p, a, R] \\ &[p, b, p, b, R] \\ &[p, c, p, c, R] \\ &[p, B, t, B, L] \end{aligned}$$

$$\begin{aligned} &[t, a, v, a, R] \\ &[t, b, r, b, L] \\ &[t, c, v, c, R] \end{aligned}$$

$$\begin{aligned} &[r, a, v, a, R] \\ &[r, b, v, b, R] \\ &[r, c, s, c, R] \end{aligned}$$

$$\begin{aligned} &[s, a, s, a, R] \\ &[s, b, s, b, R] \\ &[s, c, s, c, R] \\ &[s, B, s, B, R] \end{aligned}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.)

Let L be the set of strings on which the Turing machine M diverges.

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(b \cup c)^* a (a \cup b \cup c)^* cb$$

(b) Describe the construction of a Turing Machine T which halts on every input and also accepts exactly those strings on which the Turing Machine M diverges. In short:

$$M(\tau) \nearrow \rightarrow (T(\tau) \searrow \text{ and accept})$$

$$M(\tau) \searrow \rightarrow (T(\tau) \searrow \text{ and reject})$$

If such a Turing machine T does not exist, prove it.

Answer: Convert the regular expression obtained in the answer to part (a) into an NFA, and then convert this NFA into DFA. Finally, simulate this DFA and decide as it does.

Problem 996 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that: $Q = \{q, p, r\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, a, b, c\}$; and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, p, a, R] \\ & [q, b, q, b, R] \\ & [q, c, q, c, R] \\ & [q, B, q, B, R] \end{aligned}$$

$$\begin{aligned} & [p, a, r, a, R] \\ & [p, b, r, b, R] \\ & [p, c, r, c, R] \\ & [p, B, p, B, R] \end{aligned}$$

$$\begin{aligned} & [r, a, r, a, R] \\ & [r, b, r, b, R] \\ & [r, c, r, c, R] \end{aligned}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.)

Let L be the set of strings on which the Turing machine M halts.

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(b \cup c)^* a (a \cup b \cup c) (a \cup b \cup c)^*$$

(b) Describe the construction of a Turing Machine T which halts on every input and also accepts strings that represent Turing Machines that halt exactly when M halts. In short:

$$L(M) = L(\tau) \rightarrow (T(\tau) \searrow \text{ and accept})$$

$$L(M) \neq L(\tau) \rightarrow (T(\tau) \searrow \text{ and reject})$$

If such a Turing machine T does not exist, prove it.

Answer: Impossible, by Rice's Theorem. Property:

"is equal to $L(M)$ "

is non-trivial (since $L(M)$ has it, but, say, \emptyset does not.) Hence, there does not exist an algorithm to recognize Turing Machines whose languages have this property.

Problem 997 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that: $Q = \{q, p, r, s, t\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, a, b, c\}$; $F = \{t\}$; and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, p, a, R] \\ & [q, b, q, b, R] \\ & [q, c, q, c, R] \end{aligned}$$

$$\begin{aligned} & [p, a, r, a, R] \\ & [p, b, r, b, R] \\ & [p, c, r, c, R] \end{aligned}$$

$$\begin{aligned} & [r, a, s, a, R] \\ & [r, b, s, b, R] \\ & [r, c, s, c, R] \\ & [r, B, t, B, R] \end{aligned}$$

$$\begin{aligned} & [s, a, s, a, R] \\ & [s, b, s, b, R] \\ & [s, c, s, c, R] \\ & [s, B, s, B, R] \end{aligned}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.)

Let L be the set of strings which the Turing machine M accepts.

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(b \cup c)^* a (a \cup b \cup c)$$

(b) Describe the construction of a Turing Machine T which halts on every input and also accepts strings that represent Turing Machines that halt on at least one string on which M halts. In short:

$$L(M) \cap L(\tau) \neq \emptyset \rightarrow (T(\tau) \searrow \text{ and accept})$$

$$L(M) \cap L(\tau) = \emptyset \rightarrow (T(\tau) \searrow \text{ and reject})$$

If such a Turing machine T does not exist, prove it.

Answer: Impossible, by Rice's Theorem. Property:

"has non-empty intersection with $L(M)$ "

is non-trivial (since $L(M)$ has it, but, say, \emptyset does not.) Hence, there does not exist an algorithm to recognize Turing Machines whose languages have this property.

Problem 998 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where:

$$Q = \{q, s, t, v\}, \Sigma = \{a, b, c, d\}, \Gamma = \{A, B, Z\}, F = \{v\}$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, c, \lambda, t, Z] \\ & [q, d, \lambda, t, Z] \\ & [t, a, \lambda, t, AB] \\ & [t, c, \lambda, s, \lambda] \\ & [s, b, B, s, \lambda] \\ & [s, c, Z, v, \lambda] \\ & [s, d, A, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) List 5 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer: $ccc, dcc, cacbdc, dacbdc, caacbdbdc$

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B\}$, and P is:

$$\begin{aligned} S &\rightarrow cAc \mid dAc \\ A &\rightarrow aAbd \mid c \end{aligned}$$

(c) Dangerous Professor has told her students to write a program that operates as follows:

INPUT: an arbitrary string x over $\{a, b, c, d\}$.

OUTPUT: **yes** if x is a string that belongs to the language $L \cap c^*$;
no otherwise.

Explain the algorithm that should be employed by this program, or prove that it does not exist.

Answer: The following constructions are algorithmic:

- construct a finite automaton M_2 that accepts $L_2 = c^*$;
- construct a pushdown automaton M_1 from the pushdown automaton M and the finite automaton M_2 such that M_1 accepts $L \cap L_2$;
- simulate M_1 by a Turing machine or an equivalent computer.

Problem 999 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where:

$$Q = \{q, s, t\}, \Sigma = \{0, 1\}, \Gamma = \{A, B, Z\}, F = \{q\}$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, 0, \lambda, t, Z] \\ & [q, 1, \lambda, t, Z] \\ & [t, 0, \lambda, s, AB] \\ & [t, 1, Z, q, \lambda] \\ & [s, 0, B, s, \lambda] \\ & [s, 0, Z, q, \lambda] \\ & [s, 1, A, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) List 5 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer: 01, 11, 00010, 10010, 0101

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$((0 \cup 1)(0010 \cup 1))^*$$

(c) Dangerous Professor has told her students to write a program that operates as follows:

INPUT: an arbitrary string x over $\{0, 1\}$.

OUTPUT: **yes** if x is a Turing machine that accepts the language $L \cup 1^*$;
no otherwise.

Explain the algorithm that should be employed by this program, or prove that it does not exist.

Answer: This algorithm does not exist. If it existed, it would decide whether its argument Turing machine accepts a language with the non-trivial property of being

equal to $L \cup 1^*$

By Rice's Theorem, this is impossible.

Problem 1000 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where: $Q = \{q, s, t\}; \Sigma = \{a, b, c, d\}; \Gamma = \{A, B, K, D\}; F = \{s, t\}$; and the transition func-

tion δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, q, KA] \\ & [q, b, \lambda, q, KB] \\ & [q, d, \lambda, q, KD] \\ & [q, \lambda, \lambda, s, \lambda] \\ & [q, \lambda, \lambda, t, \lambda] \\ & [s, a, A, s, \lambda] \\ & [s, b, B, s, \lambda] \\ & [s, c, K, s, \lambda] \\ & [s, d, D, s, \lambda] \\ & [t, a, D, t, \lambda] \\ & [t, b, K, t, \lambda] \\ & [t, c, B, t, \lambda] \\ & [t, d, A, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) List 4 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer:

aac, bbc, bcb, dab

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, I)$, where $\Sigma = \{a, b, c, d\}$, $V = \{I, S, T\}$, and the production set P is:

$$\begin{aligned} I &\rightarrow S \mid T \\ S &\rightarrow aSac \mid bSbc \mid dSdc \mid \lambda \\ T &\rightarrow aTdb \mid bTcb \mid dTab \mid \lambda \end{aligned}$$

(c) Let L_1 be the set of all strings over the alphabet $\{a, b, c, d\}$ that begin with a and end with c .

Describe the construction of a Turing Machine T which halts on every input and also accepts exactly the strings that represent Turing Machines that accept the strings that belong to L but not to L_1 . In short:

$$(L(\tau) \subseteq L \setminus L_1) \implies (T(\tau) \downarrow \text{and accept})$$

$$(L(\tau) \subseteq L \setminus L_1) \implies (T(\tau) \downarrow \text{and reject})$$

If such a Turing machine T does not exist, prove it.

Answer: Machine T does not exist. If T existed, it would decide the set of Turing Machines whose languages satisfy the nontrivial property:

is contained in $L \setminus L_1$

This property is nontrivial, since the empty set has it, but L does not. Hence, by Rice's Theorem, T does not exist.

Problem 1001 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where: $Q = \{q, s, t, v\}$; $\Sigma = \{a, b, c, d\}$; $\Gamma = \{A, B, K, D\}$; $F = \{q\}$; and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, s, KA] \\ & [q, b, \lambda, t, KB] \\ & [q, d, \lambda, v, KD] \\ & [s, a, A, s, \lambda] \\ & [s, a, K, q, \lambda] \\ & [t, b, B, t, \lambda] \\ & [t, b, K, q, \lambda] \\ & [v, d, D, v, \lambda] \\ & [v, d, K, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) List 4 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer:

λ, aaa, bbb, ddd

Advice for Answer: L is given by the regular expression:

$$(aaa \cup bbb \cup ddd)^*$$

(b) Construct a state transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 362.

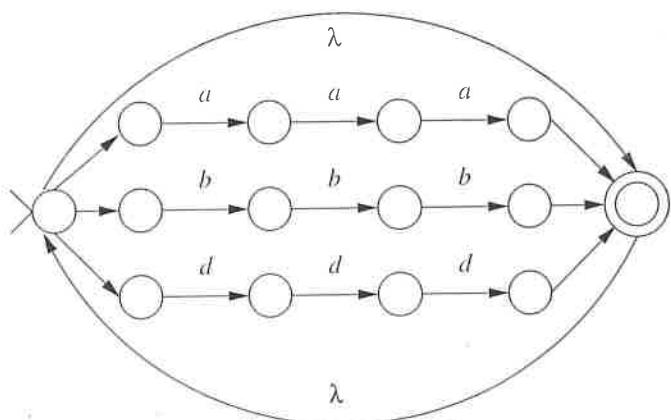


Figure 362:

(c) Let L_1 be the set of all strings over the alphabet $\{a, b, c, d\}$ that begin with a and end with d .

Describe the construction of a Turing Machine T which halts on every input and also accepts exactly the set of

all strings that belong to L but also belong to L_1 . In short:

$$(\tau \in (L \cap L_1) \implies (T(\tau) \downarrow \text{ and accept})$$

$$(\tau \notin L \cap L_1) \implies (T(\tau) \downarrow \text{ and reject})$$

If such a Turing machine T does not exist, prove it.

Answer: To construct T , proceed as follows.

- apply the known algorithm to convert the regular expression $a(a \cup b \cup c \cup d)^*d$, which represents L_1 , into a finite automaton, say M_1 ;
- let M be the automaton given on Figure 362; apply the known algorithm to construct a finite automaton M_2 from finite automata M and M_1 , such that M_2 accepts the intersection $L(M) \cap L(M_1) = L \cap L_1$;
- apply the known algorithm to convert the automaton M_2 into an equivalent deterministic finite automaton M_3 ;
- apply the known algorithm to convert M_3 into an equivalent Turing machine T .

Problem 1002 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where: $Q = \{q, s, t, v\}$; $\Sigma = \{a, b, c, d\}$;

$\Gamma = \{A, B, K, D\}$; $F = \{v\}$; and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, s, KA] \\ & [q, b, \lambda, t, KB] \\ & [q, d, \lambda, v, KD] \\ & [s, a, A, s, \lambda] \\ & [s, a, K, q, \lambda] \\ & [t, b, B, t, \lambda] \\ & [t, b, K, q, \lambda] \\ & [v, d, D, v, \lambda] \\ & [v, d, K, v, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) List 4 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer:

$$- ddd, aaaddd, bbbddd, aaaaaaddd$$

Advice for Answer: L is given by the regular expression:

$$(aaa \cup bbb)^*ddd$$

- (b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aaaS \mid bbbS \mid ddd$$

- (c) Describe the construction of a Turing Machine T which halts on every input and also accepts exactly the strings that are not contained in the language L . In short:

$$(\tau \in L(M)) \implies (T(\tau) \downarrow \text{ and reject})$$

$$(\tau \notin L(M)) \implies (T(\tau) \downarrow \text{ and accept})$$

If such a Turing machine T does not exist, prove it.

Answer: To construct T , proceed as follows.

- apply the known algorithm to convert the regular expression produced in the answer to part (a) (which represents L), into a finite automaton, say M_1 ;
- apply the known algorithm to convert the automaton M_1 into an equivalent deterministic finite automaton M_2 ;
- apply the known algorithm to convert M_2 into a deterministic finite automaton M_3 that accepts $\overline{L(M_2)} = \overline{L}$;
- apply the known algorithm to convert M_3 into an equivalent Turing machine T .

Problem 1003 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where: $Q = \{q, s, t\}$; $\Sigma = \{a, b, c, d\}$;

$\Gamma = \{A, B, K, D\}$; $F = \{q\}$; and the transition function δ is defined as follows:

$$\begin{aligned} & [q, d, \lambda, t, D] \\ & [t, \lambda, \lambda, s, \lambda] \\ & [t, a, \lambda, t, KAAB] \\ & [s, a, A, s, \lambda] \\ & [s, b, B, s, \lambda] \\ & [s, c, K, s, \lambda] \\ & [s, d, D, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) List 4 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer:

$$\lambda, dd, dddd, dabaacd$$

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, T\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow \lambda \mid SS \mid dTd \\ T &\rightarrow aTbaac \mid \lambda \end{aligned}$$

(c) State precisely a non-trivial property of recursively enumerable languages that the language L has, while the language $(abcd)^*$ does not have it. Show that the property is non-trivial and that L indeed has it, while $(abcd)^*$ does not. If this is impossible, state it and explain why.

Answer: One such property is:

every non-empty string begins with d

The property is non-trivial by definition because at least one language has it, while at least one language does not have it. Precisely, the language L indeed has it, as is evident from the grammar constructed in the answer to part (b), which shows that every string of L is a concatenation of zero or more strings that begin (and end) with d . On the other hand, every string of the language $(abcd)^*$ is a concatenation of zero or more strings that begin with a .

(d) State precisely a trivial property of recursively enumerable languages that the language L has, while the language $(abcd)^*$ does not have it. Show that the property is trivial and that L indeed has it, while $(abcd)^*$ does not. If this is impossible, state it and explain why.

Answer: This is impossible. By definition, a trivial property is either true or false for all recursively enumerable languages at once, so it is impossible that one recursively enumerable language (say L) would have it while another one (say $(abcd)^*$) would not have it—if this happened then such a property would by definition be non-trivial.

Problem 1004 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where: $Q = \{q, s\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{A, B, K, Z\}$; $F = \{q\}$; and the transition function δ is defined as follows:

$$\begin{aligned} &[q, a, \lambda, s, ZAA] \\ &[q, a, \lambda, s, ZBBB] \\ &[q, a, \lambda, s, ZKKKK] \\ &[s, a, A, s, \lambda] \\ &[s, b, B, s, \lambda] \\ &[s, c, K, s, \lambda] \\ &[s, \lambda, Z, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) List 4 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer:

$$aaa, abbb, acccc, aaaabbb$$

(b) Construct a state transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 363.

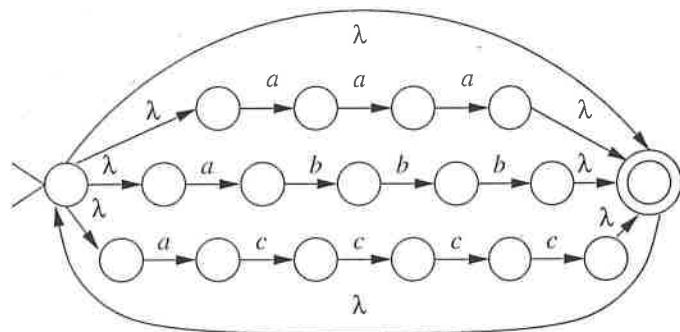


Figure 363: Finite Automaton T

(c) Describe the construction of a Turing Machine T which halts on every input and also accepts exactly those strings that are accepted by the pushdown automaton M . In short:

$$(\tau \in L(M)) \implies (T(\tau) \downarrow \text{ and accept})$$

$$(\tau \notin L(M)) \implies (T(\tau) \downarrow \text{ and reject})$$

If such a Turing machine T does not exist, prove it.

Answer: Employ the known algorithm to convert the finite automaton T , constructed in the answer to part (a), to a deterministic finite automaton T_1 ; simulate this automaton T_1 , then accept when T_1 accepts and reject when T_1 rejects.

Problem 1005 Let:

$$L = \{(ab)^n c^{k+1} (gc)^{\ell+2} \mid n, k, \ell \geq 0\}$$

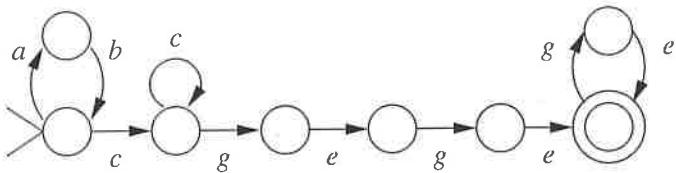
(a) Construct a state transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 364.

(b) Consider the following problem:

INPUT: String w .

OUTPUT: **yes** if w is a string that belongs to the complement of the language L ;
no otherwise.

Figure 364: Finite Automaton M

Explain the algorithm that should be employed by this program, or state that it does not exist, and prove it.

Answer: Simulate the finite automaton M given on Figure 364 (which is already deterministic in that it contains at most one transition for any symbol in any state). Accept if M rejects and reject if M accepts.

(c) Consider the following problem:

INPUT: String w .

OUTPUT: yes if w is a Turing Machine that accepts the complement of the language L ;

no otherwise.

Explain the algorithm that should be employed by this program, or state that it does not exist, and prove it.

Answer: This algorithm does not exist. If it existed, it would decide whether an arbitrary Turing Machine accepts a language with the property:

is equal to \bar{L}

where L is the language defined in part (a). This property is non-trivial (since \bar{L} has it, but \emptyset does not.) By Rice's Theorem, there is no algorithm to decide whether the language accepted by an arbitrary Turing machine has a given non-trivial property.

Problem 1006 Let L be the language defined by the regular expression:

$$(bb \cup cc)^*((a \cup ba)cd)^*$$

(a) Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:
 $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$,
and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow AA \mid \lambda \mid bb \mid cc \\ B &\rightarrow BB \mid \lambda \mid Dcd \\ D &\rightarrow a \mid ba \end{aligned}$$

(b) Is it possible to write a computer program (algorithm) that operates as follows:

INPUT: An arbitrary Turing machine ζ accepting strings over $\{a, b, c, d\}$.

OUTPUT: yes if ζ accepts L and no if ζ does not accept L .

Explain your answer briefly.

Answer: No. The property is equal to L is a non-trivial property, since (evidently) language \bar{L} has it, and (for example) the empty set \emptyset does not have it. By Rice's theorem, there is no algorithm that determines if a language accepted by a given Turing machine has a given non-trivial property.

Problem 1007 Let L_3 be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, p, F)$$

where:

$$\begin{aligned} Q &= \{p, q, r, s\} \\ \Sigma &= \{0, 1\} \\ \Gamma &= \{A, T\} \\ F &= \{s\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} &[p, \lambda, \lambda, q, T] \\ &[q, \lambda, T, s, \lambda] \\ &[q, 1, \lambda, r, AA] \\ &[r, 0, A, r, \lambda] \\ &[r, \lambda, T, q, T] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

(a) **Answer:** Write a complete formal definition of a context-free grammar G that generates L_3 . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where:

$\Sigma = \{0, 1\}$, $V = \{S\}$, and the rule set P is:

$$S \rightarrow SS \mid \lambda \mid 100$$

(b) Write a regular expression that defines L_3 . If such regular expression does not exist, prove it.

Answer:

$$(100)^*$$

(c) Describe the algorithm that should be employed by a program that solves the following problem:

INPUT: An arbitrary binary string x .

QUESTION: Does x represent a Turing Machine that accepts L_3 ?

Explain your answer. If such algorithm does not exist, prove it.

Answer: Such algorithm does not exist. The property:

is equal to L_3

is non-trivial—observe that L_3 has this property, while the empty set does not. By Rice's theorem, there is no algorithm to decide if an arbitrary Turing machine (general program) accepts a language with a given non-trivial property.

Problem 1008 Consider a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$Q = \{q, s, t\};$$

$$\Sigma = \{a, b\};$$

$$\Gamma = \{B, a, b\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, s, a, R] \\ & [q, b, q, b, R] \\ & [s, a, t, a, R] \\ & [s, b, t, b, R] \\ & [t, a, t, a, R] \\ & [t, b, t, b, R] \\ & [t, B, t, B, R] \end{aligned}$$

(a) Write a complete formal definition of a finite automaton M_1 , such that M_1 accepts exactly those strings that M accepts by halting. If such finite automaton M_1 does not exist, prove it.

Answer: $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$, where:

$Q_1 = \{q_1, s_1\}$; $\Sigma = \{a, b\}$; $F_1 = \{q_1, s_1\}$ and δ_1 is defined by the following transition set:

$$\begin{aligned} & [q_1, a, s_1] \\ & [q_1, b, q_1] \end{aligned}$$

To verify the construction, observe that M accepts by halting the language:

$$b^*(a \cup \lambda)$$

(b) Does there exist an algorithm that solves the following problem:

INPUT: An arbitrary Turing machine M and an arbitrary finite automaton M_1 .

QUESTION: Does M_1 accept the language accepted by M ?

Explain your answer briefly.

Answer: No—by Rice's Theorem, there does not exist an algorithm to decide if an arbitrary Turing machine accepts a language with a given non-trivial property. The property “IS ACCEPTED BY M_1 ” is non-trivial, since $L(M_1)$ has this property, while $\overline{L(M_1)}$ does not have it.

Problem 1009 Describe the algorithm that should be employed by a program that solves the following problem:

INPUT: An arbitrary Turing Machine M .

QUESTION: Does the language accepted by M have a regular context-free grammar?

Explain your answer. If such algorithm does not exist, prove it.

Answer: This algorithm does not exist. The property:

has a regular grammar

is non-trivial, since regular languages have this property, while languages that are not regular do not have it. By Rice's Theorem, given a Turing Machine, it is undecidable if the language accepted by it has any non-trivial property.

Problem 1010 Does there exist an algorithm that solves the following problem:

INPUT: An arbitrary context-free grammar G and an arbitrary compiler \mathcal{P} .

QUESTION: Is the language specified by G equal to the set of programs that compile under \mathcal{P} ?

Explain your answer briefly.

Answer: No. Compiler \mathcal{P} is a program written for a general purpose computer, which is equivalent to a Turing Machine. Hence, the question is equivalent to asking if an arbitrary recursively enumerable language has the property:

is equal to $L(G)$

which is a non-trivial property. By Rice's Theorem, this question does not have an algorithmic answer.

Problem 1011 (a) Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$Q = \{q, r, s, t\};$$

$$\Sigma = \{a, b, c\};$$

$$\Gamma = \{B, a, b, c\};$$

$$F = \{q\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, r, a, R] \\ & [q, b, s, b, R] \\ & [q, c, t, b, R] \\ & [r, a, r, a, R] \\ & [s, a, r, b, R] \\ & [t, B, q, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

Let L be the set of strings over Σ which are accepted by M . Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 365.

(b) Dangerous Professor has given a homework to her class. For the homework, each student has to write a JAVA program which simulates the Turing machine defined in part (a). The Professor has hired a grader to grade this homework. The grader is intimidated by 40 programs and has decided to write a program to automate his grading task. Describe precisely the algorithm that the grader should employ. If such an algorithm does not exist, prove it.

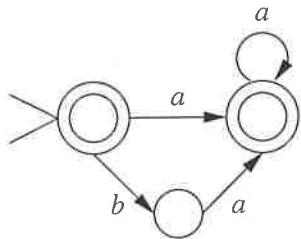


Figure 365:

Answer: Such algorithm does not exist. If it existed, it would decide if an arbitrary JAVA program, which is equivalent to a Turing machine, has a non-trivial property of being equivalent to another Turing machine. By Rice's Theorem, this is impossible.

Problem 1012 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$Q = \{q, r, s\};$$

$$\Sigma = \{a, b, c\};$$

$$\Gamma = \{B, a, b, c\};$$

$$F = \{s\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, r, a, R] \\ & [q, b, q, b, R] \\ & [q, c, q, c, R] \end{aligned}$$

$$\begin{aligned} & [r, a, q, a, R] \\ & [r, b, s, b, R] \\ & [r, c, q, c, R] \\ & [r, B, q, B, R] \end{aligned}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) M accepts by final state. B is the designated blank symbol.)

Let $L(M)$ be the language accepted by M .

(a) Write a complete formal definition of a Turing machine T_1 such that T_1 accepts input τ if τ is a Turing machine that accepts $L(M)$, and T_1 rejects input τ if τ is not a Turing machine that accepts $L(M)$, for all $\tau \in \Sigma^*$. In short:

$$(L(\tau) = L(M)) \rightarrow (T_1(\tau) \searrow \text{and accepts})$$

and also:

$$(L(\tau) \neq L(M)) \rightarrow (T_1(\tau) \searrow \text{and rejects})$$

If such a Turing machine does not exist, prove it.

Answer: Such a Turing Machine does not exist. If it existed, it would decide the set of those Turing Machines whose languages have the nontrivial property of being equal to $L(M)$. By Rice's Theorem, this is impossible.

(b) Write a complete formal definition of a Turing machine T_2 such that T_2 halts on input τ if τ is not a word

that belongs to $L(M)$, and T_2 diverges on input τ if τ is a word that belongs to $L(M)$, for all $\tau \in \Sigma^*$. In short:

$$(\tau \notin L(M)) \rightarrow (T_2(\tau) \nearrow)$$

and also:

$$(\tau \in L(M)) \rightarrow (T_2(\tau) \nearrow)$$

If such a Turing machine does not exist, prove it.

Answer: $T_2 = (Q, \Sigma, \Gamma, \delta, q)$ where: $Q = \{q, r, s\}$; $\Sigma = \{a, b, c\}$; $\Gamma = \{B, a, b, c\}$; and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, r, a, R] \\ & [q, b, q, b, R] \\ & [q, c, q, c, R] \end{aligned}$$

$$\begin{aligned} & [r, a, q, a, R] \\ & [r, b, s, b, R] \\ & [r, c, q, c, R] \\ & [r, B, q, B, R] \end{aligned}$$

$$\begin{aligned} & [s, a, s, a, R] \\ & [s, b, s, b, R] \\ & [s, c, s, c, R] \\ & [s, B, s, B, R] \end{aligned}$$

Problem 1013 Consider the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$Q = \{q, t, s\}; \Sigma = \{a, b\}; \Gamma = \{A, Z\}; F = \{q\};$$

and the transition function δ is defined as follows:

$$\begin{aligned} & [q, \lambda, \lambda, t, Z] \\ & [t, \lambda, \lambda, s, \lambda] \\ & [t, b, \lambda, t, AA] \\ & [s, a, A, s, \lambda] \\ & [s, a, ZA, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$, where $n \geq 2$, is popped from the stack by an individual transition, then the rightmost symbol X_n is popped first, while the leftmost symbol X_1 is popped last.)

Let $L(M)$ be the language accepted by M .

(a) List five different strings that belong to $L(M)$, or explain why this is impossible.

Answer: Observe that the general template for the strings of $L(M)$ is:

$$(b^m a^{2m})^\ell \text{ for } m, \ell \geq 0$$

whence the answer:

$$\lambda, baa, baabbaaa, bbbaaaaaabbaabbaaaa, bbaaaabaa$$

- (b) Write a complete formal definition of a context-free grammar G such that G generates the language comprising exactly those words that belong to $L(M)$. In short, for any $\chi \in \Sigma^*$:

$$(\chi \in L(M)) \iff (\chi \in L(G))$$

If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b\}$, $V = \{S, E\}$, and P is:

$$\begin{aligned} S &\rightarrow \lambda \mid SS \mid E \\ E &\rightarrow bEaa \mid \lambda \end{aligned}$$

- (c) Write a complete formal definition of a Turing Machine T such that T decides the language comprising Turing Machines that accept $L(M)$. In short, for any $\chi \in \Sigma^*$:

$$(L(\chi) = L(M)) \rightarrow (T(\chi) \downarrow \text{ and accepts})$$

and also:

$$(L(\chi) \neq L(M)) \rightarrow (T(\chi) \downarrow \text{ and rejects})$$

If such a Turing machine does not exist, prove it.

Answer: Such a Turing Machine does not exist. If it existed, it would decide the set of those Turing Machines whose languages have the nontrivial property of being equal to $L(M)$. By Rice's Theorem, this is impossible.

Problem 1014 (a) Give an example of a language that is not regular but has an infinite regular subset and an infinite regular superset. Give a precise definition of these three languages and explain your answer briefly. If such a language does not exist, explain why.

Answer: One example is the complement $\overline{L_2}$ of the language L_2 , defined as follows:

$$L_2 = \{a^k b^k \mid k \geq 0\}$$

L_2 is a canonical non-regular language, as is readily proved by Pumping Lemma. Hence, its complement $\overline{L_2}$ is not regular. However, $\overline{L_2}$ contains infinite regular subsets, like a^* or b^* . Straightforwardly, $\overline{L_2}$ is a subset of $(a \cup b)^*$, which is evidently regular, and thereby a regular superset of $\overline{L_2}$.

(b) Give an example of a language that is not context-free but has an infinite context-free subset and an infinite context-free superset. Give a precise definition of all these three languages and explain your answer briefly. If such a language does not exist, explain why.

Answer: The language:

$$L_4 = \{a^k b^m c^k d^m \mid k, m \geq 0\}$$

is not context-free, as is readily proved by Pumping Lemma. However, its subset:

$$L_2 = \{a^k c^k \mid k \geq 0\}$$

(obtained by fixing $m = 0$ in the template for L_4) is a canonical context-free language. Straightforwardly, L_4 is a subset of $\{a, b, c, d\}^*$, which is evidently regular and thereby context-free.

(c) Give an example of a language that is not recursively enumerable but has a recursively enumerable complement. Give a precise definition of this language and explain your answer briefly. If such a language does not exist, explain why.

Answer: One example is the complement $\overline{L_H}$ of the halting-problem language, defined as follows:

$$L_H = \{(M, w) \mid (M, w) \downarrow\}$$

L_H is the set of all pairs (machine,input) such that the machine halts on the input. L_H is recursively enumerable—the universal Turing machine accepts it. However, since L_H is known not to be decidable (recursive), its complement $\overline{L_H}$ cannot be recursively enumerable.

Problem 1015 (a) Give an example of a recursive language whose complement is not recursive. Give a precise definition of this language and explain your answer. If such a language does not exist, explain why.

Answer: Impossible. If a language is recursive, its complement is also recursive. (Recall that if a Turing machine M decides L , then M thereby also decides \overline{L} —just interchange the answers: yes for no and vice versa.)

(b) Give an example of a non-recursive but recursively enumerable language whose complement is not recursively enumerable. Give a precise definition of this language and explain your answer. If such a language does not exist, explain why.

Answer: The halting-problem language:

$$L_H = \{(M, w) \mid (M, w) \downarrow\}$$

which is the set of all pairs (machine,input) such that the machine halts on the input. L_H is recursively enumerable—the universal Turing machine accepts it. Since L_H is not recursive, its complement cannot be recursively enumerable.

(c) Give an example of a non-recursive but recursively enumerable language whose complement is recursively enumerable. Give a precise definition of this language and explain your answer. If such a language does not exist, explain why.

Answer: Impossible. If both L and \overline{L} are recursively enumerable, then both L and \overline{L} are recursive. (Recall that if M accepts L and if \overline{M} accepts \overline{L} , then a machine that decides both L and \overline{L} operates by simulating both M and \overline{M} in lockstep and decides as soon as one of them halts.)

(d) Give an example of a regular language whose complement is not regular. Give a precise definition of this language and explain your answer. If such a language does not exist, explain why.

Answer: Impossible. If a language is regular, its complement is also regular. (Recall that if a deterministic finite automaton M accepts L , then an automaton that accepts \overline{L} is obtained from M by interchanging accepting states to non-accepting and vice versa.)

(e) Give an example of an infinite context-free language that has a subset which is not context-free. Give a precise definition of both languages and explain your answer. If such a language does not exist, explain why.

Answer: $\{a, b, c\}^*$ is trivially context-free. Its proper subset:

$$L_3 = \{a^m b^m c^n \mid m \geq 0\}$$

is a canonical non-context-free language.

(f) Give an example of a recursive language that has a subset which is not recursively enumerable. Give a precise definition of both languages and explain your answer. If such a language does not exist, explain why.

Answer: $\{0, 1\}^*$ is trivially recursive. Its proper subset $\overline{L_H}$, the complement of the halting-problem language, is not recursively enumerable.

Problem 1016 (a) Give an example of a language that is not regular but has a regular complement. Give a precise definition of this language and explain your answer briefly. If such a language does not exist, explain why.

(b) Give an example of a language that is not context-free but has a context-free complement. Give a precise definition of this language and explain your answer briefly. If such a language does not exist, explain why.

(c) Give an example of a language that is not recursive but has a recursive complement. Give a precise definition of this language and explain your answer briefly. If such a language does not exist, explain why.

Problem 1017 For each of the following six languages, give the *best possible* classification of that language into one of the following classes:

regular
context-free
recursive
recursively enumerable
not recursively enumerable

Your classification of a language is the best possible if the class which you indicate indeed contains that language, while no other listed class which is a proper subset of your class contains that language.

1. The language consisting of pairs $(R(M), w)$, where $R(M)$ is a representation of Turing machine M and M halts on input w .

Answer: Recursively enumerable.

2. The language consisting of representations of Turing machines that do not halt on blank tape.

Answer: Not recursively enumerable.

3. \emptyset .

Answer: Regular.

4. $\{0, 1, 111, 00\}$.

Answer: Regular.

5. The language generated by the grammar:
 $G = (\{S\}, \{a, b\}, P, S)$, where P is given by:
 $S \rightarrow aS \mid bS \mid \lambda$.

Answer: Regular.

6. The union of two arbitrary recursive languages.

Answer: Recursive.

Problem 1018 You are given two Turing machines, M_1 and M_2 , such that M_1 decides language L_1 and M_2 decides language L_2 .

Is $L_1 \cap L_2$ a recursive language?

If your answer is “yes”, prove it by describing an appropriate Turing machine. If your answer is “no”, prove it by showing why such a Turing machine does not exist.

Answer: Yes, $L_1 \cap L_2$ is a recursive language. A Turing machine M' that accepts it and halts on every input operates as follows. On a given input string w , M' emulates the operation of M_1 until M_1 halts; then it emulates the operation of M_2 on the same input string w until M_2 halts. (Note that both M_1 and M_2 , by assumption, halt on every input.) M' accepts w if and only if both M_1 and M_2 have accepted w .

Problem 1019 Let $\Sigma = \{a, b\}$. Construct a recursively enumerable language L over Σ that is not recursive, such that its complement \overline{L} is recursively enumerable but not recursive. Explain your answer. If such a language does not exist, prove it.

Problem 1020 You are given two Turing machines, M_1 and M_2 , such that M_1 accepts language L_1 and M_2 decides language L_2 .

Is $L_1 \setminus L_2$ a recursively enumerable language?

If your answer is “yes”, prove it by describing an appropriate Turing machine. If your answer is “no”, prove it by showing that such a Turing machine does not exist.

Answer: Yes— $L_1 \setminus L_2$ is recursively enumerable. Let $L = L_1 \setminus L_2$. A Turing machine M that accepts L operates as follows. Given an input string w , M first simulates M_2 until M_2 halts, which must happen because M_2 decides language L_2 . If M_2 accepts then M rejects, because $w \in L_2$ implies $w \notin L$. If M_2 rejects, then M simulates M_1 , and halts and accepts if and when M_1 halts and accepts, which occurs if and only if $w \in L_1$. Since also $w \notin L_2$, M indeed accepts if and only if $w \in L_1 \setminus L_2$.

Problem 1021 You are given two Turing machines, M_1 and M_2 , such that M_1 accepts language L_1 and M_2 accepts language L_2 .

Is $L_1 \cup L_2$ a recursively enumerable language?

If your answer is “yes”, prove it by describing an appropriate Turing machine. If your answer is “no”, prove it by showing why such a Turing machine does not exist.

Answer: Yes, $L_1 \cup L_2$ is a recursively enumerable language. A Turing machine M' that accepts it operates as follows. On a given input string w , M' emulates simultaneously the operation of M_1 and M_2 . Say, M' emulates M_1 on its even steps, and M' emulates M_2 on its odd steps. M' halts and accepts w if and only if one of the machines M_1 , M_2 halts and accepts w .

Problem 1022 (a)

Let M_1 be a Turing machine that decides language L_1 ; let M_2 be a Turing machine that accepts language L_2 ; let M_3 be a Turing machine that accepts language L_3 .

Describe a Turing machine M that accepts language:

$$L_1 \cap (L_2 \cup L_3)$$

If such Turing machine does not exist, prove it.

Answer: M simulates M_1 until M_1 halts. If M_1 rejects then M also rejects. If M_1 accepts them M continues as follows. M simulates both M_2 and M_3 in parallel and halts and accepts if and when any of them accepts.

(b)

Let M_1 be a Turing machine that accepts language L_1 ; let M_2 be a Turing machine that decides language L_2 ; let M_3 be a Turing machine that decides language L_3 .

Describe a Turing machine M that accepts language:

$$L_1 \setminus (L_2 \cap L_3)$$

If such Turing machine does not exist, prove it.

Answer: M simulates both M_2 and M_3 in parallel until they both halt. If both M_2 and M_3 accept, then M rejects. Otherwise, M simulates M_1 and accepts if and when M_1 accepts.

(c)

Let M_1 be a Turing machine that decides language L_1 ; let M_2 be a Turing machine that accepts language L_2 ; let M_3 be a Turing machine that accepts language L_3 .

Describe a Turing machine M that decides language:

$$L_1 \setminus (L_2 \cap L_3)$$

If such Turing machine does not exist, prove it.

Answer: Impossible. If M existed, we could set:

$$L_1 = \Sigma^*$$

$$L_2 = L_H$$

$$L_3 = \Sigma^*$$

where:

$$L_H = \{(M, w) \mid (M, w) \searrow\}$$

yielding:

$$L_2 \cap L_3 = L_H$$

Hence, M would decide:

$$L \setminus L_H = \overline{L_H}$$

which is impossible, since $\overline{L_H}$ is not even recursively enumerable.

Problem 1023 Let L_1 be a recursively enumerable language which is not recursive; and let L_2 be a recursive language.

(a) Is $L_1 \setminus L_2$ a recursive language?

If your answer is “yes”, prove it by describing an appropriate Turing machine. If your answer is “no”, prove it by showing that such a Turing machine does not exist.

Answer: No. Otherwise, we could set:

$$L_1 = L_H$$

$$L_2 = \emptyset$$

where:

$$L_H = \{(M, w) \mid (M, w) \searrow\}$$

yielding:

$$L_1 \setminus L_2 = L_H$$

and claim that L_H is recursive, which is a contradiction.

(b) Is $L_1 \setminus L_2$ a recursively enumerable language?

If your answer is “yes”, prove it by describing an appropriate Turing machine. If your answer is “no”, prove it by showing that such a Turing machine does not exist.

Answer: Yes. There exists a machine M_1 that accepts L_1 and a machine M_2 that decides L_2 . To accept $L_1 \setminus L_2$, simulate M_2 until it halts. If M_2 accepts, then reject. If M_2 rejects, then simulate M_1 and accept if and when M_1 accepts.

(c) Is $L_2 \setminus L_1$ a recursive language?

If your answer is “yes”, prove it by describing an appropriate Turing machine. If your answer is “no”, prove it by showing that such a Turing machine does not exist.

Answer: No. Otherwise, we could set:

$$L_2 = \Sigma^*$$

$$L_1 = L_H$$

and claim that:

$$\Sigma^* \setminus L_H = \overline{L_H}$$

is recursive. This is false, since $\overline{L_H}$ is not even recursively enumerable.

(d) Is $L_2 \setminus L_1$ a recursively enumerable language?

If your answer is “yes”, prove it by describing an appropriate Turing machine. If your answer is “no”, prove it by showing that such a Turing machine does not exist.

Answer: No---by the answer given in part (c).

Problem 1024 Let L_1 be a recursively enumerable language, and let L_2 be a recursive language. Describe a Turing machine M that accepts $L_1 \setminus L_2$. If such M does not exist, explain why.

Problem 1025 Let M_1 and M_2 be two arbitrary Turing machines over input alphabet Σ . For each of the following six questions, determine if the answer is always yes, always no, or sometimes yes. Justify your answer in each case.

(a) Is $L(M_1) = \emptyset$?

Advice for Answer: Sometimes.

(b) Is $L(M_2) = \Sigma^*$?

Advice for Answer: Sometimes.

(c) Is $L(M_1)$ recursive?

Advice for Answer: Sometimes.

(d) Is $L(M_2)$ recursively enumerable?

Advice for Answer: Always.

(e) Is $L(M_1) = L(M_2)$?

Advice for Answer: Sometimes.

(f) Is $L(M_1) \cup L(M_2)$ recursively enumerable?

Advice for Answer: Always.

Problem 1026 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q)$$

such that:

$$Q = \{q, r, t, v\};$$

$$\Sigma = \{a, b, c\};$$

$$\Gamma = \{B, a, b, c\};$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, t, a, R] \\ & [q, b, r, b, R] \\ & [q, c, t, c, R] \\ & [q, B, t, B, R] \end{aligned}$$

$$\begin{aligned} & [r, a, t, a, R] \\ & [r, b, v, b, R] \\ & [r, c, t, c, R] \\ & [r, B, t, c, R] \end{aligned}$$

$$\begin{aligned} & [t, a, t, a, R] \\ & [t, b, t, b, R] \\ & [t, c, t, c, R] \\ & [t, B, t, B, R] \end{aligned}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) B is the designated blank symbol.)

Let L be the set of strings on which the Turing machine M halts.

(a) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$bb(a \cup b \cup c)^*$$

(b) Is the language L decidable (recursive?) Explain your answer.

Answer: Yes. L is regular, as is confirmed in the answer to part (a), and every regular language is recursive (decidable.)

(c) Is the language \bar{L} (the complement of L) decidable (recursive?) Explain your answer.

Answer: Yes. \bar{L} is regular, as the complement of the regular language L , and every regular language is recursive (decidable.)

(d) Is the language L context-free? Explain your answer.

Answer: Yes. L is regular, and every regular language is context-free.

(e) Is the language \bar{L} (the complement of L) context-free? Explain your answer.

Answer: Yes. \bar{L} is regular, and every regular language is context-free.

Problem 1027 Assume that $\Sigma = \{0, 1\}$. State the cardinality of each of the following sets. (For a finite set, state the exact number. For an infinite set, specify if it is countable or not.)

Answer:

(a) The set of all regular languages over Σ that cannot be generated by any context-free grammar is empty.

(b) The set of all context-free languages over Σ that cannot be represented by any regular expression is infinite and countable.

(c) The set of all languages over Σ is infinite and uncountable.

(d) The set of all regular languages over Σ is infinite and countable.

(e) The set of all languages over Σ that are not regular is infinite and uncountable.

(f) The set of all finite languages over Σ is infinite and countable.

(g) The set of all context-free languages over Σ is infinite and countable.

(h) The set of all total functions from $N = \{0, 1, \dots\}$ to Σ is infinite and uncountable.

(i) The set of all total functions from $N = \{0, 1, \dots\}$ to Σ that can be computed by some Turing Machine is infinite and countable.

(j) The set of all total functions from $N = \{0, 1, \dots\}$ to Σ that cannot be computed by any Turing Machine is infinite and uncountable.

(k) The set of all regular languages over Σ that are not decidable is empty and has the cardinality equal to zero.

(l) The set of all recursively enumerable languages over Σ is infinite and countable.

Problem 1028 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where: $Q = \{q, s\}$; $\Sigma = \{a, b, c, d\}$; $\Gamma = \{A, B, K, D, E\}$; $F = \{q\}$; and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, s, EBKA] \\ & [q, d, \lambda, s, EDDD] \\ & [s, a, A, s, \lambda] \\ & [s, b, B, s, \lambda] \\ & [s, c, K, s, \lambda] \\ & [s, d, D, s, \lambda] \\ & [s, d, E, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) List 4 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer:

$$aacbd, ddd, aacbdaacbd, aacbddddddd$$

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$$(aacbd \cup ddd)^*$$

(c) State the cardinality of the language L . If this cardinality is finite, state the exact number; if it is infinite, specify whether it is countable or uncountable.

Answer: Infinite and countable.

(d) State the cardinality of the language $L \cap a^*$. If this cardinality is finite, state the exact number; if it is infinite, specify whether it is countable or uncountable.

Answer: One. (Since $L \cap a^* = \{\lambda\}$.)

(e) State the cardinality of the class that contains all regular languages over Σ . If this cardinality is finite, state the exact number; if it is infinite, specify whether it is countable or uncountable.

Answer: Infinite and countable.

(f) State the cardinality of the class that contains all languages over Σ that are not decidable but are recursively enumerable and also have a recursively enumerable complement. If this cardinality is finite, state the exact number; if it is infinite, specify whether it is countable or uncountable.

Answer: Zero. (If a language is recursively enumerable and has a recursively enumerable complement, then it is decidable, together with its complement.)

Problem 1029 Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where: $Q = \{q, s\}$; $\Sigma = \{a, b, c, d\}$; $\Gamma = \{A, B, K, D\}$; $F = \{s\}$; and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, s, \lambda] \\ & [q, a, \lambda, q, DAA] \\ & [q, a, \lambda, q, DBBB] \\ & [q, a, \lambda, q, DKKKK] \\ & [s, a, A, s, \lambda] \\ & [s, b, B, s, \lambda] \\ & [s, c, K, s, \lambda] \\ & [s, d, D, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) List 4 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer:

$$a, aaaad, aabbdd, aaccccd$$

(b) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSaad \mid aSbbd \mid aScacd \mid a$$

(c) Describe the construction of a Turing Machine T which halts on every input and also accepts exactly those strings that represent Turing Machines that accept at least one string accepted by the pushdown automaton M . In short:

$$(L(x) \cap L(M) \neq \emptyset) \implies (T(x) \downarrow \text{ and accept})$$

$$(L(x) \cap L(M) = \emptyset) \implies (T(x) \downarrow \text{ and reject})$$

If such a Turing machine T does not exist, prove it.

Answer: This is impossible. If T existed, it would decide the set of Turing Machines whose languages have the following non-trivial property:

"has a non-empty intersection with $L(M)$ ".

This property is non-trivial, since $L(M)$ has it but \emptyset does not. Hence, by Rice's Theorem, there is no algorithm to recognize the Turing Machines whose languages have this property.

Problem 1030 Consider the Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that: $Q = \{q, p, r, t, s, v, w, e, f\}$; $\Sigma = \{0, 1\}$; $\Gamma = \{B, 0, 1\}$; $F = \{f\}$; and δ is defined by the following transition set:

$$\begin{aligned} & [q, 0, p, 0, R] \\ & [q, 1, e, 1, R] \\ & [q, B, e, B, R] \end{aligned}$$

$$\begin{aligned} & [p, 0, r, 0, R] \\ & [p, 1, e, 1, R] \\ & [p, B, e, B, R] \end{aligned}$$

$$\begin{aligned} & [r, 0, t, 0, R] \\ & [r, 1, e, 1, R] \\ & [r, B, e, B, R] \end{aligned}$$

$$\begin{aligned} & [t, 0, t, 0, R] \\ & [t, 1, t, 1, R] \\ & [t, B, s, B, L] \end{aligned}$$

$$\begin{aligned} & [s, 0, v, 0, L] \\ & [v, 1, w, 1, L] \\ & [w, 1, f, 1, L] \end{aligned}$$

$$\begin{aligned} & [e, 0, e, 0, R] \\ & [e, 1, e, 1, R] \\ & [e, B, e, B, R] \end{aligned}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) M accepts by final state. B is the designated blank symbol.)

Let L be the set of strings which the Turing machine M accepts.

(a) List 4 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer:

000110, 0000110, 0001110, 00000110

(b) Draw a state transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 366.

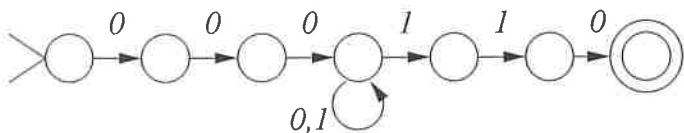


Figure 366:

(c) Describe the construction of a Turing Machine T which halts on every input and also accepts exactly those strings that are not accepted by the Turing Machine M . In short:

$$(x \notin L(M)) \implies (T(x) \downarrow \text{ and accept})$$

$$(x \in L(M)) \implies (T(x) \downarrow \text{ and reject})$$

If such a Turing machine T does not exist, prove it.

Answer: Convert the automaton constructed in part (b) to a deterministic one, simulate it, and return the negation of its decision.

Problem 1031 Consider the Turing machine (already defined in Problem 1030):

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that: $Q = \{q, p, r, t, s, v, w, e, f\}$; $\Sigma = \{0, 1\}$; $\Gamma = \{B, 0, 1\}$; $F = \{f\}$; and δ is defined by the following transition set:

$$\begin{aligned} & [q, 0, p, 0, R] \\ & [q, 1, e, 1, R] \\ & [q, B, e, B, R] \end{aligned}$$

$$\begin{aligned} & [p, 0, r, 0, R] \\ & [p, 1, e, 1, R] \\ & [p, B, e, B, R] \end{aligned}$$

$$\begin{aligned} & [r, 0, t, 0, R] \\ & [r, 1, e, 1, R] \\ & [r, B, e, B, R] \end{aligned}$$

$$\begin{aligned} & [t, 0, t, 0, R] \\ & [t, 1, t, 1, R] \\ & [t, B, s, B, L] \end{aligned}$$

$$\begin{aligned} & [s, 0, v, 0, L] \\ & [v, 1, w, 1, L] \\ & [w, 1, f, 1, L] \end{aligned}$$

$$\begin{aligned} & [e, 0, e, 0, R] \\ & [e, 1, e, 1, R] \\ & [e, B, e, B, R] \end{aligned}$$

(Assume that M is defined so as to have an one-way infinite tape (infinite to the right side only.) M accepts by final state. B is the designated blank symbol.)

Let L be the set of strings which the Turing machine M rejects.

(a) List 4 distinct strings that belong to L . If this is impossible, state it and explain why.

Answer:

0001, 000010, 0000100, 00000010

(b) Write a regular expression that defines L . If such a regular expression does not exist, prove it.

Answer:

$000(0 \cup 1)^*(1 \cup 00 \cup 010) \cup 0000 \cup 00010$

(c) Describe the construction of a Turing Machine T which halts on every input and also accepts exactly those

strings that represent Turing Machines that accept every string accepted by M . In short:

$$(L(M) \subseteq L(x)) \implies (T(x) \downarrow \text{ and accept})$$

$$(L(M) \not\subseteq L(x)) \implies (T(x) \downarrow \text{ and reject})$$

If such a Turing machine T does not exist, prove it.

Answer: This is impossible. If T existed, it would decide the set of Turing Machines whose languages have the following non-trivial property:

"contains $L(M)$ as subset".

This property is non-trivial, since $L(M)$ has it but \emptyset does not. Hence, by Rice's Theorem, there is no algorithm to recognize the Turing Machines whose languages have this property.

Problem 1032 Let L be the set of those strings over the alphabet $\{a, b, c\}$ that contain exactly two b 's.

Let L_1 be the set of those strings over the alphabet $\{a, b, c\}$ that have an odd length.

(a) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 367.

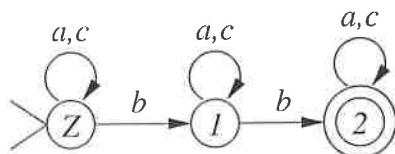


Figure 367:

(b) Construct a state-transition graph of a finite automaton that accepts $L \cap L_1$. If such an automaton does not exist, prove it.

Answer: See Figure 368.

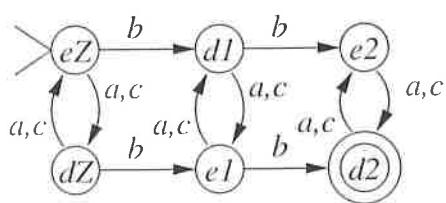


Figure 368:

(c) Explain how to construct an algorithm that solves the following problem:

INPUT:

two finite automata: M and M_1 , and a string x .

OUTPUT:

yes if x is a string that belongs to the intersection of languages defined by $L(M)$ and $L(M_1)$;

no otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist. If it existed, it would decide whether the argument Turing Machine x accepts a language with the non-trivial property "is equal to $L(M) \cap L(M_1)$ ". This property is non-trivial because the language $L(M) \cap L(M_1)$ has this property (as it indeed is equal to itself) while its complement $\overline{L(M) \cap L(M_1)}$ does not have this property (because it is not equal to $L(M) \cap L(M_1)$). By Rice's Theorem, such a decision is impossible.

Problem 1033 Let L be the set of those strings over the alphabet $\{a, b, c\}$ that contain exactly one b .

Let L_1 be the set of those strings over the alphabet $\{a, b, c\}$ whose length is divisible by 3.

(a) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it.

Answer: See Figure 369.

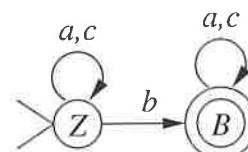


Figure 369:

(b) Construct a state-transition graph of a finite automaton that accepts $L \cap L_1$. If such an automaton does not exist, prove it.

Answer: See Figure 370.

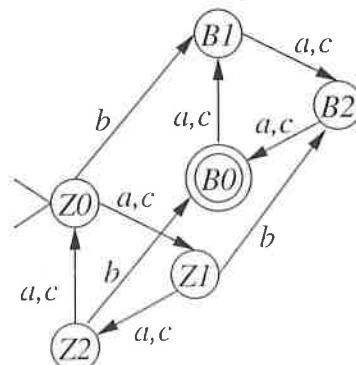


Figure 370:

(c) Explain how to construct an algorithm that solves the following problem:

INPUT:

two finite automata: M and M_1 , and a string x .

OUTPUT:

yes if x is a string that belongs to the intersection of languages defined by $L(M)$ and $L(M_1)$;

no otherwise.

If this algorithm does not exist, prove it.

Answer: The algorithm simulates the two argument automata M and M_1 on the argument string x , and returns yes if and only if both of these two simulations return yes.

Problem 1034 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, t\}$, $\Sigma = \{a, b, c\}$, $\Gamma = \{B, Z\}$, $F = \{t\}$ and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, q, ZBB] \\ & [q, \lambda, \lambda, t, \lambda] \\ & [t, b, B, t, \lambda] \\ & [t, c, Z, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSbbc \mid \lambda$$

(b) Explain how to construct an algorithm that solves the following problem:

INPUT: A Turing Machine T .

OUTPUT: A push-down automaton P , such that

$$L(P) = L(T)$$

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist. A language accepted by a Turing Machine need not be context free (and thereby accepted by some pushdown automaton), and it is undecidable whether it is context free.

(c) Explain how to construct an algorithm that solves the following problem:

INPUT: A non-deterministic push-down automaton P .

OUTPUT: A Turing Machine T such that

$$L(T) = L(P)$$

If this algorithm does not exist, prove it.

Answer: The output Turing Machine T simulates the argument pushdown automaton P and decides exactly as P does.

Problem 1035 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where:

$Q = \{q, s, t\}$, $\Sigma = \{a, b, c, d\}$, $\Gamma = \{B, K, Z\}$, $F = \{q\}$ and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, s, ZKK] \\ & [q, a, \lambda, t, ZBB] \\ & [t, b, B, t, \lambda] \\ & [t, d, Z, q, \lambda] \\ & [s, c, K, s, \lambda] \\ & [s, d, Z, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow \lambda \mid SS \mid accd \mid abbd$$

(c) Is the automaton M deterministic? Explain your answer.

Answer: No. In state q , while looking at input symbol a , regardless of the stack content, the automaton is able to perform any one of the first two transitions.

(d) Is $L(M)$ decidable? Explain your answer.

Answer: Yes. L is accepted by the pushdown automaton M , and thereby decidable.

(e) Is $L(M)$ accepted by any deterministic finite automaton? Explain your answer.

Answer: Yes. By inspection of the grammar constructed in the answer to part (a), we conclude that L is regular, hence accepted by some deterministic finite automaton.

(f) Is $\overline{L(M)}$ (the complement of $L(M)$) context free? Explain your answer.

Answer: Yes. As is explained in the answer to part (e), L is regular, and thus has a regular complement, which in turn is context free, because every regular language is context free.

Problem 1036 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, t\}$, $\Sigma = \{a, b, c\}$, $\Gamma = \{B, K, Z\}$, $F = \{q\}$ and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, t, ZKK] \\ & [q, a, \lambda, t, ZBB] \\ & [t, b, B, t, \lambda] \\ & [t, c, K, t, \lambda] \\ & [t, \lambda, Z, q, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack

string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow \lambda \mid SS \mid acc \mid abb$$

(b) Explain how to construct an algorithm that solves the following problem:

INPUT: A non-deterministic push-down automaton P and a string x .

OUTPUT: yes if $x \in L(P)$;

no otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm simulates P on input x and decides exactly as P decides.

(c) Explain how to construct an algorithm that solves the following problem:

INPUT: A non-deterministic push-down automaton P and a string x .

OUTPUT: yes if x is a Turing Machine such that

$$L(x) = L(P)$$

no otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist. If it existed, it would decide whether the argument Turing Machine x accepts a language with the non-trivial property “is equal to $L(P)$ ”. This property is non-trivial because the language $L(P)$ has this property (as it indeed is equal to itself) while its complement $\overline{L(P)}$ does not have this property (because it is not equal to $L(P)$). By Rice’s Theorem, such a decision is impossible.

Problem 1037 Let L be the language accepted by the pushdown automaton: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ where: $Q = \{q, t\}$, $\Sigma = \{a, b, c, d\}$, $\Gamma = \{B, K, Z\}$, $F = \{t\}$ and the transition function δ is defined as follows:

$$\begin{aligned} & [q, a, \lambda, q, ZK] \\ & [q, a, \lambda, q, ZB] \\ & [q, \lambda, \lambda, t, \lambda] \\ & [t, b, B, t, \lambda] \\ & [t, c, K, t, \lambda] \\ & [t, d, Z, t, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack. Furthermore, if an arbitrary stack string, say $X_1 \dots X_n \in \Gamma^*$ where $n \geq 2$, is pushed on the stack by an individual transition, then the leftmost

symbol X_1 is pushed first, while the rightmost symbol X_n is pushed last.)

(a) Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aScd \mid aSbd \mid \lambda$$

(b) Explain how to construct an algorithm that solves the following problem:

INPUT: A context-free grammar G .

OUTPUT: A context-free grammar G_1 such that

$$L(G_1) = \overline{L(G)}$$

If this algorithm does not exist, prove it.

Answer: Impossible. The complement of the language given by the argument context free grammar need not be context free and it is undecidable whether it is context free.

(c) Explain how to construct an algorithm that solves the following problem:

INPUT: A context-free grammar G .

OUTPUT: A Turing Machine T such that

$$L(T) = \overline{L(G)}$$

If this algorithm does not exist, prove it.

Answer: The output Turing machine converts the argument grammar G into an equivalent pushdown automaton P , simulates P , and decides exactly as P does.

Problem 1038 Let L be the language accepted (by final state) by the Turing Machine: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ such that: $Q = \{q, r, s, t, v, x, y\}$; $\Sigma = \{0, 1\}$; $\Gamma = \{B, 0, 1, Z\}$; $F = \{t\}$; and δ is defined by the following transition set:

$$\begin{aligned} & [q, 0, r, Z, R] \\ & [q, 1, r, Z, R] \\ & [r, 0, s, Z, R] \\ & [r, 1, s, Z, R] \\ & [s, 0, v, Z, R] \\ & [s, 1, v, Z, R] \\ & [v, 0, v, 0, R] \\ & [v, 1, v, 1, R] \\ & [v, B, x, B, L] \\ & [x, 1, y, B, L] \\ & [y, 0, t, 0, R] \end{aligned}$$

(where B is the designated blank symbol.)

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

$$(0 \cup 1)(0 \cup 1)(0 \cup 1)(0 \cup 1)^* 01$$

(b) Let L' be the set of strings which the Turing Machine M rejects. Is L' decidable? Prove your answer.

Answer: L' is decidable. To see this, observe (by inspection of the code) that the Turing Machine M never diverges. Hence, the language L' rejected by M is equal to the complement \bar{L} of the language L accepted by M . As is demonstrated in the answer to part (a), L is regular. Hence; the complement $\bar{L} = L'$ is also regular, and thereby decidable, since every regular language is decidable.

(c) Explain how to construct an algorithm that solves the following problem:

INPUT: A string x over $\{0, 1\}$.

OUTPUT: **yes** if x is a Turing Machine such that $L(x)$ contains at least three strings accepted by M ;
no otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist. If it existed, it would decide whether the argument Turing Machine x accepts a language with the non-trivial property “contains three strings that belong to L ”. This property is non-trivial because the language L has this property (as it indeed contains all of its infinitely many strings) while the empty set does not have this property (because it does not contain any strings and in particular does not contain three strings from L .) By Rice’s Theorem, such a decision is impossible.

(d) Explain how to construct an algorithm that solves the following problem:

INPUT: A string x over $\{0, 1\}$.

OUTPUT: **yes** if x belongs to L' , the language defined in part (b);
no otherwise.

If this algorithm does not exist, prove it.

Answer: The algorithm converts the regular expression proposed in the answer to part (a) into an equivalent finite automaton, simulates that automaton on input x , and decides exactly the opposite.

Problem 1039 Let L be the language accepted (by final state) by the Turing Machine: $M = (Q, \Sigma, \Gamma, \delta, q, F)$ such that: $Q = \{q, r, s, t, v, x, y\}$; $\Sigma = \{0, 1\}$; $\Gamma = \{B, 0, 1, Z\}$; $F = \{t\}$; and δ is defined by the following transition set:

$[q, 1, r, Z, R]$
$[r, 1, s, Z, R]$
$[s, 0, v, Z, R]$
$[v, 0, v, 0, R]$
$[v, 1, v, 1, R]$
$[v, B, x, B, L]$
$[x, 0, y, B, L]$
$[y, Z, t, B, R]$

(where B is the designated blank symbol.)

(a) Write a regular expression that defines L . If such regular expression does not exist, prove it.

Answer:

1100

(b) Is \bar{L} (the complement of the language L) decidable? Prove your answer.

Answer: Yes. As is demonstrated in the answer to part (a), L is regular, and as such must have a regular complement, which in turn is decidable, because every regular language is decidable.

(c) Explain how to construct an algorithm that solves the following problem:

INPUT: A string x over $\{0, 1\}$.

OUTPUT: **yes** if x belongs to \bar{L} (the complement of the language L);
no otherwise.

If this algorithm does not exist, prove it.

Answer: The algorithm converts the regular expression proposed in the answer to part (a) into an equivalent finite automaton, simulates that automaton on input x , and decides exactly the opposite.

(d) Explain how to construct an algorithm that solves the following problem:

INPUT: A string x over $\{0, 1\}$.

OUTPUT: **yes** if x is a Turing Machine that accepts \bar{L} (the complement of the language L);
no otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist. If it existed, it would decide whether the argument Turing Machine x accepts a language with the non-trivial property “is equal to \bar{L} ”. This property is non-trivial because the language \bar{L} has this property (as it indeed is equal to itself) while its complement L does not have this property (because it is not equal to \bar{L} .) By Rice’s Theorem, such a decision is impossible.

Problem 1040 Prove that the following problem is undecidable by constructing a reduction from the halting problem:

INPUT: A representation of a Turing machine M .

QUESTION: Is $L(M) = \Sigma^*$?

Problem 1041 State carefully six different undecidable problems. For the statement of each problem, use the following form:

INPUT:

QUESTION:

Problem 1042 Let L be the set of strings over the alphabet $\{a\}$ whose length gives an odd remainder after the division by 4.

- (a) Construct a state-transition graph of a finite automaton that accepts L . If such an automaton does not exist, prove it, and do not answer parts (b) and (c).

Answer: See Figure 371.

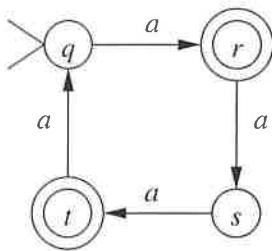


Figure 371:

- (b) Describe briefly an algorithm that solves the following problem:

INPUT: An arbitrary finite automaton F .

OUTPUT: A Turing Machine M that decides whether its input is a word that belongs to the language $L(F)$ accepted by F . Precisely:

$$(F(x) \text{ accepts}) \implies (M(x) \downarrow \text{ and accepts})$$

and also:

$$(F(x) \text{ rejects}) \implies (M(x) \downarrow \text{ and rejects})$$

If such an algorithm does not exist, explain why. If such an algorithm exists, illustrate its application, employing the automaton which you constructed in the answer to the part (a) as the input.

Answer: The required algorithm converts a finite automaton F to an equivalent Turing Machine M . To this end, first arrange for F to be deterministic—if necessary, apply the algorithm that converts a non-deterministic finite automaton to a deterministic one. Next, M inherits from F the set of states, alphabet, initial state, and the set of final states. To construct the transition function of M , rewrite every tuple $[q, a, p]$ of F into: $[q, a, p, a, R]$.

The automaton constructed in the part (a) has the following transition function:

$$\begin{aligned} & [q, a, r] \\ & [r, a, s] \\ & [s, a, t] \\ & [t, a, q] \end{aligned}$$

and the complete definition of the equivalent Turing machine is:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

such that:

$$\begin{aligned} Q &= \{q, r, s, t\}; \\ \Sigma &= \{a\}; \\ \Gamma &= \{B, a\}; \\ F &= \{r, t\}; \end{aligned}$$

and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, r, a, R] \\ & [r, a, s, a, R] \\ & [s, a, t, a, R] \\ & [t, a, q, a, R] \end{aligned}$$

- (c) Describe an algorithm that solves the following problem:

INPUT: An arbitrary finite automaton F .

OUTPUT: A Turing Machine M that decides whether its input is a Turing Machine that accepts the language $L(F)$ accepted by F . Precisely:

$$(L(x) = L(F)) \implies (M(x) \downarrow \text{ and accepts})$$

and also:

$$(L(x) \neq L(F)) \implies (M(x) \downarrow \text{ and rejects})$$

If such an algorithm does not exist, explain why. If such an algorithm exists, illustrate its application, employing the automaton which you constructed in the answer to the part (a) as the input.

Answer: This algorithm does not exist. The property of being equal to $L(F)$ is a non-trivial property—by Rice's theorem, there is no algorithm that can recognize Turing machines that accept languages with this property.

Problem 1043 Let L_0 be the language accepted by halting by the Turing machine: $M = (Q, \Sigma, \Gamma, \delta, q)$, such that: $Q = \{q, r, s, t\}$; $\Sigma = \{a, b\}$; $\Gamma = \{B, a, b, X, Y\}$; and δ is defined by the following transition set:

$$\begin{aligned} & [q, a, q, a, R] \\ & [q, b, r, X, R] \\ & [q, B, q, B, R] \end{aligned}$$

$$\begin{aligned} & [r, a, r, a, R] \\ & [r, b, s, Y, R] \\ & [r, B, r, B, R] \end{aligned}$$

$$\begin{aligned} & [s, a, s, a, R] \\ & [s, b, t, b, R] \end{aligned}$$

$$\begin{aligned} & [t, a, t, a, R] \\ & [t, b, t, b, R] \\ & [t, B, t, B, R] \end{aligned}$$

(where B is the designated blank symbol.)

- (a) Write a regular expression that defines the set of strings that M accepts (by halting.) If such regular expression does not exist, prove it.

Answer:

$$a^*ba^*ba^*$$

- (b) Does M halt on input $ababa$? If your answer is “yes” write the configuration in which M halts. If your answer is “no”, write the configuration which M assumes after exactly 6 steps.

Answer: Yes— M halts on input $ababa$ in the configuration:

$$aXaYas$$

(c) Describe the algorithm that should be employed by a program that solves the following problem:

INPUT: An arbitrary string x over Σ .

QUESTION: Does x represent a finite automaton that accepts L_6 ?

Explain your answer. If such algorithm does not exist, prove it.

Answer: Yes. First, employ the known algorithm to convert the regular expression for L_6 , obtained in part (a), to an equivalent finite automaton M_6 . Next, apply a sequence of algorithmic constructions, based on the closure properties of regular sets, to arrive at the finite automaton M_0 that accepts the language:

$$L(M_0) = (L(M_6) \cap \overline{L(x)}) \cup (L(x) \cap \overline{L(M_6)})$$

$L(M_0)$ is empty if and only if x accepts exactly L_6 . To test if $L(M_0) = \emptyset$, test all the words over Σ of length less than k , where k is the number of states of M_0 . If all these words are rejected, the answer is yes; otherwise the answer is no.

Problem 1044 Let L be a language accepted by a finite automaton:

$$M = (Q, \Sigma, \delta, q_0, F)$$

such that:

$$\begin{aligned} |Q| &= k \\ |\Sigma| &= s \end{aligned}$$

for some positive integers k and s .

(a) Describe a procedure for determining if L is empty. If such a procedure does not exist, prove it.

(b) Describe a procedure for determining if L is infinite. If such a procedure does not exist, prove it.

(c) Describe a procedure for determining if L is equal to Σ^* . If such a procedure does not exist, prove it.

Problem 1045 Let L be a language accepted by a Turing machine:

$$M = (Q, \Sigma, \Gamma, \delta, q_0)$$

such that:

$$\begin{aligned} |Q| &= k \\ |\Sigma| &= s \end{aligned}$$

for some positive integers k and s .

(a) Describe a procedure for determining if L is empty. If such a procedure does not exist, prove it.

(b) Describe a procedure for determining if L is infinite. If such a procedure does not exist, prove it.

(c) Describe a procedure for determining if L is equal to Σ^* . If such a procedure does not exist, prove it.

Problem 1046 Does there exist an algorithm that solves the following problem:

INPUT: An arbitrary context-free grammar G .

QUESTION: Is the language generated by G regular?

Explain your answer briefly.

Answer: No—this problem is undecidable, which is proved by a series of reductions from the Halting Problem.

Problem 1047 Let L be the set of strings over alphabet $\{a, b, c\}$ that do not begin with the substring aa .

(a) Draw a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 372.

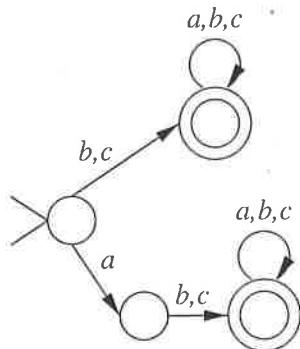


Figure 372:

(b) Is it possible to write a computer program (algorithm) that operates as follows:

INPUT: An arbitrary finite automaton μ accepting strings over $\{a, b, c\}$.

OUTPUT: yes if μ accepts L and no if μ does not accept L .

Explain your answer briefly.

Answer: Yes. Such an algorithm first constructs a finite automaton, say F , that accepts the language:

$$L(F) = (L(M) \cap \overline{L(\mu)}) \cup (L(\mu) \cap \overline{L(M)})$$

where M is the finite automaton constructed in the answer to part (a). This construction is algorithmic, based on the closure properties of regular sets. Next, the algorithm tests if:

$$L(F) = \emptyset$$

which happens if and only if F does not accept any word of length not exceeding $k - 1$, where k is the number of the states of F . If an exhaustive test of all such words indeed produces no acceptances, then the final answer is yes; otherwise it is no.

Problem 1048 Let L_1 be the language defined by the regular expression:

$$((a \cup c)^* b^* ab \cup bca)^*$$

Describe precisely a procedure (algorithm, method) to solve the following problem:

INPUT: A description of a finite automaton M .

QUESTION: Is the language accepted by automaton M equal to L_1 ?

If such a procedure does not exist, prove it.

Problem 1049 Let L_1 be the language defined by the regular expression:

$$a^* b^*$$

Describe precisely a procedure (algorithm, method) to solve the following problem:

INPUT: A description of a Turing machine M .

QUESTION: Is the language accepted by the Turing machine M equal to L_1 ?

If such a procedure does not exist, prove it.

Problem 1050 Let L be the language generated by the context-free grammar $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S, B, D, E\}$, and P is:

$$\begin{aligned} S &\rightarrow cS \mid aB \mid bD \mid a \\ B &\rightarrow cB \mid bD \mid a \mid b \\ D &\rightarrow a \mid b \\ E &\rightarrow aS \mid cS \mid bE \mid \lambda \end{aligned}$$

(a) Construct a state-transition graph of a finite automaton that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 373.

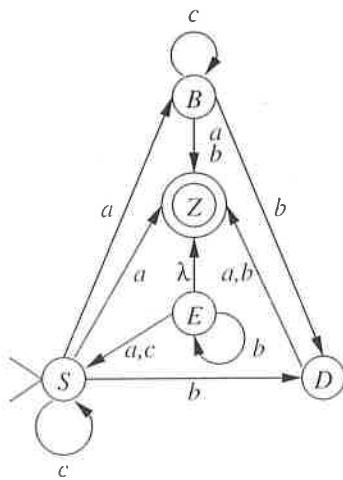


Figure 373:

(b) Does there exist an algorithm that solves the following problem:

INPUT: An arbitrary regular context-free grammar G , and an arbitrary finite automaton M .

QUESTION: Does G generate the language accepted by M ?

Explain your answer briefly.

Answer: Yes. First, convert G into a finite automaton M' , which is algorithmic because G is regular. Next, compare the two finite automata, M and M' , for equality. To do this, construct the finite automaton that accepts:

$$L_x = (L(M) \cap \overline{L(M')}) \cup (L(M') \cap \overline{L(M)})$$

which is algorithmic, via closure properties of regular sets. Converting this automaton into its deterministic equivalent is also algorithmic. Finally, decide if:

$$L_x = \emptyset$$

by testing all the strings of length not exceeding η , where η is the number of states of the deterministic finite automaton that accepts L_x .

(c) Does there exist an algorithm that solves the following problem:

INPUT: An arbitrary context-free grammar G , and an arbitrary finite automaton M .

QUESTION: Does G generate the language accepted by M ?

Explain your answer briefly.

Answer: No—it is undecidable for an arbitrary context-free grammar whether the language generated by it is equal to a particular regular language.

Problem 1051 Consider the finite automaton M represented by the state transition graph given on Figure 291.

(a) Draw the generalized expression graph obtained from the automaton M after the elimination of node F , according to the algorithm for conversion of finite automata to regular expressions. If it is impossible to eliminate node F , prove it.

Answer: See Figure 374.

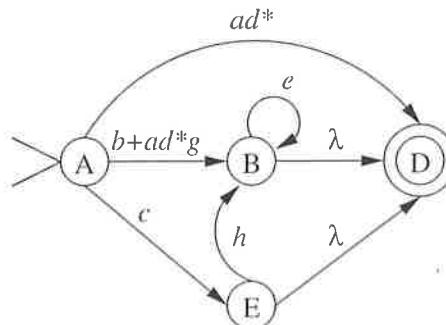


Figure 374:

(b) Explain the construction of an algorithm that operates as follows.

INPUT: A finite automaton A .

OUTPUT: A pushdown automaton P that accepts the language accepted by A .

Explain your answer briefly.

If such an algorithm does not exist, state it and explain why.

Answer: A (deterministic) finite automaton is straightforwardly converted into a pushdown automaton—the simulating pushdown automaton P behaves exactly as the simulated finite automaton A , ignoring the stack. Precisely, if $A = (Q, \Sigma, \delta, q, F)$, then $P = (Q, \Sigma, \emptyset, \delta_P, q, F)$, where the new transition function δ_P is obtained by sending every tuple $[s, a, t]$ of δ to the tuple $[s, a, \lambda, t, \lambda]$ of δ_P , for any $s, t \in Q$, and $a \in \Sigma$.

(c) Explain the construction of an algorithm that operates as follows.

INPUT: A finite automaton A .

OUTPUT: A deterministic Turing Machine T that accepts the language accepted by A .

If such an algorithm does not exist, state it and explain why.

Answer: A deterministic finite automaton is straightforwardly converted into a deterministic Turing Machine—the simulating Turing machine T behaves exactly as the simulated finite automaton A , moving right-only read-only on the tape. Precisely, if $A = (Q, \Sigma, \delta, q, F)$, then $T = (Q, \Sigma, (\Sigma \cup \{B\}), \delta_T, q, F)$, where the new transition function δ_T is obtained by sending every tuple $[s, a, t]$ of δ to the tuple $[s, a, t, a, R]$ of δ_T , for any $s, t \in Q$, and $a \in \Sigma$.

Problem 1052 Explain how to construct an algorithm that solves the following problem:

INPUT: A regular context-free grammar G_1 over $\Sigma = \{0, 1\}$;

OUTPUT: A context-free grammar G_2 such that:

$$L(G_2) = \overline{L}$$

If this algorithm does not exist, prove it.

Answer: The required algorithm is obtained as a sequence of the following algorithmic conversions and constructions.

- convert the input regular context-free grammar G_1 into a finite automaton F ;
- convert the finite automaton F into an equivalent deterministic finite automaton F_1 ;
- from the finite automaton F_1 , construct a deterministic finite automaton F_2 that accepts $\overline{L(F_1)}$;
- convert the deterministic finite automaton F_2 into an equivalent (regular) context-free grammar G_2 .

Problem 1053 Explain how to construct an algorithm that solves the following problem:

INPUT: A Turing Machine M_1 over $\Sigma = \{0, 1\}$ that decides a language L ;

OUTPUT: A Turing Machine M_2 that decides the language:

$$\overline{L}$$

If this algorithm does not exist, prove it.

Answer: The required algorithm simulates M_1 until M_1 makes its decision. Then, if M_1 decides yes, the algorithm returns no; if M_1 decides no, the algorithm returns yes.

Problem 1054 Explain how to construct an algorithm that solves the following problem:

INPUT: Two Turing Machines, M_1 and M_2 , over $\Sigma = \{0, 1\}$, which decide languages L_1 and L_2 , respectively.

OUTPUT: A Turing Machine M , such that M decides the language:

$$L_1 \setminus L_2$$

If this algorithm does not exist, prove it.

Answer: The required algorithm simulates M_1 until M_1 makes its decision. Then, if M_1 decides no, the algorithm returns no. If M_1 decides yes, the algorithm simulates M_2 until M_2 makes its decision. Then, if M_2 decides no, the algorithm returns yes; if M_2 decides yes, the algorithm returns no.

Problem 1055 Explain how to construct an algorithm that solves the following problem:

INPUT: Two context-free grammars, G_1 and G_2 , over $\Sigma = \{0, 1\}$, which generate languages L_1 and L_2 , respectively.

OUTPUT: A context-free grammar G , such that:

$$L(G) = L_1 \cap L_2$$

If this algorithm does not exist, prove it.

Answer: Given two context-free grammars, it is undecidable whether the intersection of their languages is context-free. If the required algorithm existed, it would decide this undecidable question. Hence, the required algorithm does not exist.

Problem 1056 Explain how to construct an algorithm that solves the following problem:

INPUT: A Turing Machine M_1 over $\Sigma = \{0, 1\}$ that decides a language L , and another Turing Machine M_2 .

OUTPUT: yes if

$$L(M_2) = LL;$$

no otherwise.

If this algorithm does not exist, prove it.

Answer: This is impossible by Rice's Theorem. Precisely, the property “*is equal to LL*” is a non-trivial property of recursively enumerable languages. Hence, it is impossible to have an algorithm to recognize Turing Machines that accept a language with this particular property.

Problem 1057 Explain how to construct an algorithm that solves the following problem:

INPUT: A Turing Machine M_1 over $\Sigma = \{0, 1\}$ that decides a language L ;

OUTPUT: A Turing Machine M_2 that decides the language:

$$LL$$

If this algorithm does not exist, prove it.

Answer: On any input $x \in \Sigma^*$, the required Turing Machine M_2 operates as follows.

Let n be the length of the input string x .

- generate all $n + 1$ possible distinct representations of x as a concatenation of two strings: $x = yz$;
- for each of the $n + 1$ possible distinct representations, test whether:

$$y \in L \wedge z \in L$$

by simulating M_1 on y and then on x , and return **yes** if and when the first (conjunction) test returns **true**;

- otherwise, if all $n + 1$ tests fail, return **no**.

(Observe that all tests in the middle stage of the algorithm must terminate, since M_1 decides the language L .)

Problem 1058 Explain how to construct an algorithm that solves the following problem:

INPUT: Two regular expressions: e and e_1 , and a string x .

OUTPUT: **yes** if x is a Turing Machine that accepts the intersection of languages defined by e and e_1 ;

no otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist. If it existed, it would decide whether the language accepted by the input Turing machine x has the nontrivial property of being “equal to the intersection of languages defined by e and e_1 ”. However, by Rice’s Theorem, there is no algorithm to recognize those Turing machines that accept a language with any non-trivial property.

Problem 1059 Explain how to construct an algorithm that solves the following problem:

INPUT: Two regular expressions: e and e_1 , and a string x .

OUTPUT: **yes** if x is a string that belongs to the intersection of languages defined by e and e_1 ;

no otherwise.

If this algorithm does not exist, prove it.

Answer: Perform the following algorithmic steps:

1. convert e into an equivalent finite automaton A ;
2. convert e_1 into an equivalent finite automaton A_1 ;
3. apply the “product construction” to obtain a finite automaton P that accepts

$$L(A) \cap L(A_1)$$

4. on input x , simulate P and decide as P decides.

Problem 1060 Explain how to construct an algorithm that solves the following problem:

INPUT: A context-free grammar G and a regular grammar G_1 .

OUTPUT: A context-free grammar G' such that

$$L(G') = L(G) \cap L(G_1)$$

If this algorithm does not exist, prove it.

Answer: Perform the following algorithmic steps:

1. convert G into an equivalent pushdown automaton M ;
2. convert G_1 into an equivalent finite automaton M_1 ;
3. apply the “product construction” to obtain the pushdown automaton M' that accepts

$$L(M) \cap L(M_1) = L(G) \cap L(G_1) = L(G')$$

4. convert M' into an equivalent context-free grammar G' .

Problem 1061 Explain how to construct an algorithm that solves the following problem:

INPUT: Two context-free grammars: G and G_1 .

OUTPUT: A push-down automaton M such that

$$L(M) = L(G) \cap L(G_1)$$

If this algorithm does not exist, prove it.

Answer: Such algorithm does not exist: the input context-free languages $L(G)$ and $L(G_1)$ need not have a context-free intersection.

Problem 1062 (a) Explain how to construct an algorithm that solves the following problem:

INPUT: A non-deterministic push-down automaton M and a string x .

OUTPUT: **yes** if $x \in L(M)$;
no otherwise.

If this algorithm does not exist, prove it.

Answer: Perform the following algorithmic steps:

1. construct a (non-deterministic, multitape) Turing Machine T that decides $L(M)$ by simulating the pushdown automaton M ;

2. employ the Universal Turing Machine M_U to simulate T on input x and decide as M_U does.

(b) Explain how to construct an algorithm that solves the following problem:

INPUT: A non-deterministic push-down automaton M and a string x .

OUTPUT: **yes** if x is a Turing Machine such that $L(x) = L(M)$;

no otherwise.

If this algorithm does not exist, prove it.

Answer: This algorithm does not exist. If it existed, it would decide whether the language accepted by the input Turing machine x has the nontrivial property of being “equal to $L(M)$ ”. However, by Rice’s Theorem, there is no algorithm to recognize those Turing machines that accept a language with any non-trivial property.

Problem 1063 (a) Explain how to construct an algorithm that solves the following problem:

INPUT: A Turing Machine T .

OUTPUT: A push-down automaton M , such that

$$L(T) = L(M)$$

If this algorithm does not exist, prove it.

Answer: Such algorithm does not exist: the input Turing Machine need not accept a context-free language. Moreover, it is impossible to tell whether T accepts a context-free language: the property of being context-free is nontrivial, and (by Rice’s Theorem) there is no algorithm to recognize those Turing machines that accept a language with any non-trivial property.

(b) Explain how to construct an algorithm that solves the following problem:

INPUT: A non-deterministic push-down automaton M .

OUTPUT: A Turing Machine T such that

$$L(T) = L(M)$$

If this algorithm does not exist, prove it.

Answer: Convert the push-down automaton M into a Turing Machine C (possibly a non-deterministic, multi-tape one); the required output machine T is obtained by instantiating M_U to simulate C on any given input.

Problem 1064 (a) Explain how to construct an algorithm that solves the following problem:

INPUT: A context-free grammar G .

OUTPUT: A context-free grammar G_1 such that

$$L(G_1) = \overline{L(G)}$$

If this algorithm does not exist, prove it.

Answer: Such algorithm does not exist: the language generated by the input context-free grammar G need not have a context-free complement.

(b) Explain how to construct an algorithm that solves the following problem:

INPUT: A context-free grammar G .

OUTPUT: A Turing Machine T such that

$$L(T) = \overline{L(G)}$$

If this algorithm does not exist, prove it.

Answer: Perform the following algorithmic steps:

1. convert G into an equivalent pushdown automaton M ;
2. construct a (non-deterministic, multitape) Turing Machine T that decides $L(M)$ by simulating the pushdown automaton M , so that T rejects exactly when M accepts, and accepts otherwise.

Problem 1065 Students have written their programming homework. Their assignment was to write a class `Final_Exam`, declared as follows.

```
class Final_Exam
{
public:
    bool answer (int);
};
```

Each student has written their own implementation:

```
bool Final_Exam::answer (int question)
{
    /* student's own code here */
}
```

The class is normally employed as follows.

```
#include <iostream>
int main (int argc, char* argv[])
{
    Final_Exam problem;
    int n;

    cin >> n;
    cout << problem.answer (n);
}
```

This programming language is somewhat “magic” in that it is able to represent and manipulate correctly integers of arbitrary (unbounded) size. (In other words, no integer is too big or too small.)

You are helping Professor Lupin to grade this homework. He has established (proved) that student Hermione Granger has implemented `Final_Exam::answer` so that it calculates the correct answer (which is sometimes `true`

but sometimes **false** in less than ten seconds for any integer argument.

(a) Describe how to write a program in this programming language that operates as follows.

INPUT:

code to implement `Final_Exam::answer`, submitted by an individual student x ;
code to implement `Final_Exam::answer`, submitted by Hermione;

OUTPUT: **yes** if the implementation submitted by x returns some result for every integer argument;

no if the implementation submitted by x computes for infinitely many steps (without returning the result) for some integer argument(s).

If this program cannot be written, prove it.

Answer: This program cannot be written—if it existed it would decide whether the program submitted by x halts, which is known to be impossible.

(b) Describe how to write a program in this programming language that operates as follows.

INPUT:

positive integer N ;
code to implement `Final_Exam::answer`, submitted by an individual student x ;
code to implement `Final_Exam::answer`, submitted by Hermione;

OUTPUT: **yes** if the implementation submitted by x returns the result faster than Hermione's implementation for any positive integer input less than N ;

no otherwise.

If this program cannot be written, prove it.

Answer: The program should operate as follows.

For every positive integer k , such that $1 \leq k \leq N$, run Hermione's code and note the running time, say t_k time units, spent by Hermione's code on input k . Then, run the code submitted by x for exactly t_k units on input k . If the code submitted by x completes by time t_k and gives the correct result for every positive integer k such that $1 \leq k \leq N$, then return **yes**; otherwise return **no**.

(c) Describe how to write a program in this programming language that operates as follows.

INPUT:

positive integer N ;
code to implement `Final_Exam::answer`, submitted by an individual student x ;
code to implement `Final_Exam::answer`, submitted by Hermione;

OUTPUT: **yes** if the implementation submitted by x returns **true** on input N while Hermione's implementation returns **false** on the same input N ;

no otherwise.

If this program cannot be written, prove it.

Answer: This program cannot be written.

Let L_x be the set of those integers (more precisely, the set of strings that represent (in binary) such integers) that yield the result of **true** if presented as input to the code submitted by x . Let n_0 be an arbitrary positive integer input that leads Hermione's code to return **false**. If this algorithm existed, then it would answer the following question:

INPUT:

C++ code that defines language L_x ;

QUESTION: Does L_x contain n_0 ?

However, the property

contains n_0

is nontrivial, since, say $(0 \cup 1)^*$ has this property while \emptyset does not have it. By Rice's Theorem, it is impossible to write a program that decides whether a C++ program defines a language having any nontrivial property.

(d) Describe how to write a program in this programming language that operates as follows.

INPUT:

positive integer N ;
code to implement `Final_Exam::answer`, submitted by an individual student x ;
code to implement `Final_Exam::answer`, submitted by Hermione;

OUTPUT: **yes** if Hermione's implementation and the implementation submitted by x return identical answers for every integer input k such that $N \leq k \leq 2N$.

no otherwise;

If this program cannot be written, prove it.

Answer: This program cannot be written.

Let L_x be the set of those integers (more precisely, the set of strings that represent (in binary) such integers) that yield the result of **true** if presented as input to the code submitted by x . Let n_0 be an arbitrary positive integer input such that that leads Hermione's code to return **true**. If this algorithm existed, then it would answer the following question:

INPUT:

C++ code that defines language L_x ;

QUESTION: Does L_x contain n_0 ?

However, the property

contains n_0

is nontrivial, since, say $(0 \cup 1)^*$ has this property while \emptyset does not have it. By Rice's Theorem, it is impossible to write a program that decides whether a C++ program defines a language having any nontrivial property.

Problem 1066 (a) Let L be the set of all strings over the alphabet $\{a, b, c\}$ that are different from the string bca . Construct a state-transition graph of a finite

automaton that accepts L . If such automaton does not exist, prove it.

Answer: See Figure 375.

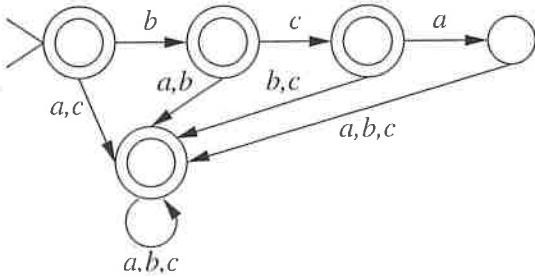


Figure 375:

(b) Dangerous Professor has given a homework to her class. For the homework, each student has to construct a deterministic finite automaton with exactly 10 states, which operates over the alphabet $\{0, 1\}$, and accepts an infinite language (of student's choice). The Professor has hired a grader to grade this homework. The grader is intimidated by 40 automata and has decided to write a program to automate his grading task. Describe precisely the algorithm that the grader should employ. If such an algorithm does not exist, prove it.

Answer: To evaluate an individual automaton, the grader should let it run, giving it as input each binary string of length at least 10 but less than 20. There are $2^{20} - 2^{10}$ such strings. As soon as one such string is accepted, the automaton can be declared correct. If none of these strings is accepted, the automaton is incorrect, since the language it accepts is not infinite.

Problem 1067 (a) Let L be the set of palindromes (strings that are equal to their reversal) over the alphabet $\{a, b, c\}$. Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSa \mid bSb \mid cSc \mid a \mid b \mid c \mid \lambda$$

(b) Dangerous Professor has given a homework to her class. For the homework, each student has to write a grammar described in part (a). The Professor has hired a grader to grade this homework. The grader is intimidated by 40 grammars and has decided to write a program to automate his grading task. Describe precisely the algorithm that the grader should employ. If such an algorithm does not exist, prove it.

Answer: Such algorithm does not exist. If it existed, it would decide if an arbitrary context-free grammar is equal to the given one. However, this is a well-known undecidable problem.

Problem 1068 (a) Let:

$$L = \{d^m a^k b^m b^k c^j \mid j, k, m \geq 0\}$$

Write a complete formal definition of a context-free grammar that generates L . If such a grammar does not exist, prove it.

Answer: Observe that the template of the strings which belong to L can be rewritten as:

$$d^m a^k b^k b^m c^j$$

whence the grammar: $G = (V, \Sigma, P, S)$, where $\Sigma = \{a, b, c, d\}$, $V = \{S, A, B, D\}$, and the production set P is:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow dAb \mid D \\ D &\rightarrow aDb \mid \lambda \\ B &\rightarrow cB \mid \lambda \end{aligned}$$

(b) Dangerous Professor has given a homework to her class. For the homework, each student has to construct a context-free grammar with at least 10 variables, over terminal alphabet $\{0, 1\}$, which accepts a non-empty language (of student's choice). The Professor has hired a grader to grade this homework. The grader is intimidated by 40 grammars and has decided to write a program to automate his grading task. Describe precisely the algorithm that the grader should employ. If such an algorithm does not exist, prove it.

Answer: The algorithm to decide if a given context-free grammar generates a non-empty language employs an iterative procedure of marking grammar symbols. Initially, only terminals and λ are marked. Then, productions are considered repeatedly, and the variable at the left-hand side of a production is marked just when all the symbols at the right-hand side of the production are marked. This procedure continues until no more variables can be marked. The language generated by the grammar is non-empty if and only if the start symbol is marked.

Problem 1069 The context-free grammar G has suffered a transmission error, so it is not clear what its start symbol is. Otherwise, G is defined as follows.

$G = (V, \Sigma, P, \dots)$, where $\Sigma = \{a, b, c\}$, $V = \{A, B, D, E, Z\}$, and the production set P is:

$$\begin{aligned} A &\rightarrow AA \mid B \\ B &\rightarrow BE \mid E \\ D &\rightarrow ZDZ \mid E \mid cD \mid c \mid Z \\ E &\rightarrow EE \mid B \\ Z &\rightarrow a \mid b \end{aligned}$$

(a) Which variable(s) of G could be declared as the start symbol so that the language generated by G is not empty? If such a variable does not exist, state it.

Answer: D, Z .

(b) Which variable(s) of G could be declared as the start symbol so that the language generated by G is empty? If such a variable does not exist, state it.

Answer: A, B, E .

(c) Prove the answer you have given in part (b).

Advice for Answer: Apply the marking algorithm for testing emptiness of context-free grammars. The algorithm marks exactly symbols D and Z .

Problem 1070 (a) Let L be the language accepted by the pushdown automaton:

$$M = (Q, \Sigma, \Gamma, \delta, q, F)$$

where:

$$\begin{aligned} Q &= \{q, s\} \\ \Sigma &= \{a, b, c, d\} \\ \Gamma &= \{A, Z, V\} \\ F &= \{s\} \end{aligned}$$

and the transition function δ is defined as follows:

$$\begin{aligned} [q, a, \lambda, q, Z] \\ [q, b, \lambda, q, V] \\ [q, c, \lambda, q, A] \\ [q, \lambda, \lambda, s, \lambda] \\ [s, d, A, s, \lambda] \\ [s, b, Z, s, \lambda] \\ [s, c, V, s, \lambda] \end{aligned}$$

(Recall that M is defined so as to accept by final state and empty stack.)

Write a complete formal definition of a context-free grammar that generates L . If such grammar does not exist, prove it.

Answer: $G = (V, \Sigma, P, S)$, where

$\Sigma = \{a, b, c, d\}$, $V = \{S\}$, and the production set P is:

$$S \rightarrow aSb \mid bSc \mid cSd \mid \lambda$$

(b) Dangerous Professor has given a homework to her class. For the homework, each student has to construct a pushdown automaton which accepts a non-empty language (of student's choice) over the alphabet $\{0, 1\}$. The Professor has hired a grader to grade this homework. The grader is intimidated by 40 automata and has decided to write a program to automate his grading task. Describe precisely the algorithm that the grader should employ. If such an algorithm does not exist, prove it.

Answer: First, convert the push-down automaton into an equivalent context-free grammar—this conversion is algorithmic, and the algorithm is given in our textbook. Next, test if this grammar generates a non-empty languages—this algorithm is given in the answer to Problem 1068.

Problem 1071 Dangerous Professor has told her students to write a program that operates as follows:

INPUT: A pair of strings x, y over $\{0, 1\}$.

OUTPUT: **yes** if x is a Turing machine that accepts y , and **no** otherwise.

Explain the algorithm that should be employed by this program, or prove that it does not exist.

Answer: This algorithm does not exist, for it is precisely the solution to the Halting Problem.

Problem 1072 Dangerous Professor is organizing a Lab for Practice Programming in Abstract Models. You have been hired to write software that will automate the performance of a variety of tasks associated with the operations of the Lab. Now, you have to submit a proposal to the Professor, explaining the specific procedures which you will apply to write programs that solve individual tasks.

For each of the following tasks, explain the procedure that will be performed by your program that implements the task. You may assume that the Professor is familiar with all the algorithms (and all the basic results) that you have studied in this course—you are allowed to apply these algorithms and results without explaining them, as long as you refer to them correctly and precisely. If a program that implements an individual task cannot be written, prove it.

(a) Check student programs (coded in JAVA), so as to exclude those programs that print flattering messages for the Professor. (The Professor has a list of such unacceptable messages.)

Answer: The required program does not exist. It is undecidable whether a program written for a Turing machine or an equivalent computer prints an individual symbol (or a sequence of symbols.)

(b) Check student programs (coded in JAVA), so as to exclude those programs that do not print student name some time during the first sixty seconds of execution. (These students will suffer a penalty of at least one point.)

Answer: The required program should run every student submission for sixty seconds and eliminate those that have not printed the name by then.

(c) Check student programs (coded in C++), so as to exclude those programs that use the specific shade of midnight blue color while drawing output images (until we get the new version of the color support software—the present one has a bug.)

Answer: The required program does not exist. It is undecidable whether a program written for a Turing machine or an equivalent computer prints an individual symbol (or a sequence of symbols that results in an appearance of a color, etc.)

(d) Check whether regular expressions submitted by students are syntactically correct (legal.)

Answer: First, write a grammar for the language of regular expressions, such as: $G = (V, \Sigma, P, S)$, where: $\Sigma = \{0, 1, \Lambda, \Phi, (,), \cup, \cdot, *, \cdot\}$, $V = \{S\}$, and P is:

$$S \rightarrow (S \cup S) \mid (S \cdot S) \mid (S^*) \mid 0 \mid 1 \mid \Lambda \mid \Phi$$

Next, apply the known algorithm to convert this grammar to an equivalent push-down automaton. Finally, simulate this push-down automaton on each student submission and decide accordingly.

- (e) Check whether student programs (coded as TURING MACHINES) accept the language represented by the assigned regular expression.

Answer: The required program does not exist. Being equal to the language represented by the assigned regular expression is a non-trivial property of languages. By Rice's theorem, there is no algorithm to recognize those Turing Machines that accept languages with this property.

- (f) Convert regular expressions submitted by students into equivalent deterministic finite automata.

Answer: The required program should implement the algorithm for conversion of a regular expression into an equivalent nondeterministic finite automaton, and then apply the algorithm for conversion of a nondeterministic finite automaton into an equivalent deterministic finite automaton.

Problem 1073 (a) Let G be a grammar that defines the C++ programming language. Describe the algorithm that should be employed by a program that solves the following problem:

INPUT: An arbitrary string p of characters from the legal C++ character set.

QUESTION: Does p represent a valid C++ program?

Explain your answer. If such algorithm does not exist, prove it.

Answer: The program should simulate the push-down automaton which accepts all strings that are valid C++ programs; this automaton is obtained by an algorithmic conversion of the grammar G .

(b) Let G be a grammar that defines the C++ programming language. In an introductory programming class, students write C++ programs for verifying if a sequence of five integers is sorted. Describe the algorithm that should be employed by a program that solves the following problem:

INPUT: An arbitrary student program p and an arbitrary sequence s of five integers.

QUESTION: Does p correctly verify s ?

Explain your answer. If such algorithm does not exist, prove it.

Answer: This algorithm does not exist, by Rice's Theorem, even for a fixed s . Let r be any program that correctly verifies if a given fixed sequence s of five integers is sorted. (Observe that, for a given fixed sequence, r may simply write the answer and do nothing else.) The property of being equivalent to r is non-trivial, since r has it and many other programs do not.

(c) Let G be a grammar that defines the C++ programming language. In an introductory programming class, students write C++ programs for sorting a sequence of integers of arbitrary length. There are only two grades: pass and fail. A student program passes if it correctly sorts the benchmark input sequence within two seconds

of processor time; otherwise the program receives the grade fail. Describe the algorithm that should be employed by a program that solves the following problem:

INPUT: An arbitrary student program p and an arbitrary integer sequence s .

QUESTION: Does p pass if tested on s as the benchmark? Explain your answer. If such algorithm does not exist, prove it.

Answer: Execute the student program p on input s for two seconds. After two seconds have expired, if the program p has arrived to the correct answer, output pass, else output fail.

(d) Let G be a grammar that defines the C++ programming language, as implemented by a major software vendor, named X , and let P be the compiler manufactured by this vendor. At a major university, students are writing C++ compilers in their software design courses, as programming exercises. Company X is interested in hiring students that write good compilers. Company X evaluates a compiler as good if it compiles the same set of programs as its own compiler P . Describe the algorithm that should be employed by a program that solves the following problem:

INPUT: Compiler P developed by X , and an arbitrary compiler P_1 written by a student.

QUESTION: Is P_1 a good compiler?

Explain your answer. If such algorithm does not exist, prove it.

Answer: This algorithm does not exist, by Rice's Theorem, even for a fixed P . The property of being equal to the set of strings accepted by P is non-trivial, and we cannot decide if the language accepted by P_1 has this property.

(e) The scenario is identical to that given in part (d), except that Company X evaluates a compiler as good if it compiles the same set of programs as its own compiler P and, additionally, is never slower than P by a factor greater than 2. Describe the algorithm that should be employed by a program that solves the following problem:

INPUT: Compiler P developed by X , and an arbitrary compiler P_1 written by a student.

QUESTION: Is P_1 a good compiler?

Explain your answer. If such algorithm does not exist, prove it.

Answer: This algorithm does not exist, for the reasons stated in part (d)—the new condition is a conjunction of two non-trivial properties.

Problem 1074 You are a manager of a computer center. The profit of your center largely depends on running time-consuming programs that output complicated drawings. Presently, your only high-quality printer is damaged in the following way: it cannot print letter A ,

but is otherwise perfect. However, if a letter A is missing from its place in the output, the output is invalid, and your company may even be taken to court if such an output is handed back to a client. The printer cannot be repaired for at least a month. The head of your center, who wishes to accept as many client jobs as possible under the circumstances, is asking you to write a program that will analize the source code of the clients' programs and select those that do not print A . Only these selected programs will be compiled and run. What will you do?

Answer: The proposed program cannot be written. Precisely, given a representation of a Turing machine and one of its tape symbols, it is undecidable if the machine ever writes the symbol.

Problem 1075 You are a head of a research project. The representatives of your funding agency are coming to visit you the day after tomorrow, and you have to demonstrate your results to them. Your students have prepared five demo-programs and disappeared to a vacation. Your collaborator is telling you that the computer hardware has been damaged in the following way: if the computer operates continuously for two hours, a fuse burns down and everything stops. Although the fuse is cheap, and the computer can be restarted immediately, the embarrassment is so great that you cannot allow this to happen during the demo. The computer cannot be repaired for at least a week. You know only that each demo-program takes between one hour and three hours to complete, but you do not know the exact time requirement of any of them. Your collaborator is asking you to select only those demo-programs that do not take longer than two hours to complete. Only these selected programs will be presented during the demo. What will you do?

Answer: Run the five programs and measure their running time. Since the five of them together take at most 15 hours, there is enough time to do that before the demo. Since the programs are deterministic, they will take the same time during the demo as they do during the testing. Select those that complete in less than two hours.

Problem 1076 You are teaching an introductory programming course. For their final grade, your students must submit the complete code for a programming project. Many of them are careless, however, and they do not write their I.D. number in the output of the program. This wastes a lot of your time, since such unidentified projects cannot be graded anyway. You have prepared a list of I.D. numbers of the 300 students who are enrolled in the course. You are requesting from your assistant to make a program, which preprocesses the student submissions in order to select just those that will print in their output **one** of the I.D. numbers from your list. These selected submissions are then indeed tested. Describe informally what the assistant should do, and justify your description carefully.

Answer: The assistant should point out that the construction of the required program is impossible. Precisely, it is unsolvable whether a given program writes a given symbol during its execution. Hence, not even the appearance of the first symbol of the first I.D. number in the output is decidable.

Problem 1077 You are the owner of a general-purpose computing facility, which consists of Turing machines. The fastest and most frequently used of your machines, namely machine M , is damaged in the following way: Whenever M enters its state q_{59} , it catches fire and destroys its tape; otherwise it is perfect. While you are waiting for a maintenance crew to arrive and fix it, you want your engineer to make a program to run on another (usually available) Turing machine, M' . This program should simply preprocess the programs brought in by the clients, and find those programs that never use state q_{59} when running on M . Those programs are then indeed executed on the machine M .

Describe informally what the engineer should do, and justify your description carefully.

Answer: The engineer should point out that the construction of the required program is impossible. Precisely, it is unsolvable whether a given Turing machine enters a given state during its computation.

Problem 1078 For each of the following claims, circle the word "yes" that follows the claim if the claim is correct, and circle the word "no" that follows the claim if the claim is not correct.

1. The union of any two regular languages is context-free.

Answer: Yes, in fact it is regular.

2. The intersection of any two regular languages is regular.

Answer: Yes.

3. The union of any two context-free languages is regular.

Answer: No—but it is context-free.

4. The intersection of any two context-free languages is context-free.

Answer: No—some context-free languages do not have a context-free intersection.

5. Every subset of a regular language is regular.

Answer: No—all languages (including non-regular ones) are subsets of Σ^* , which is certainly regular.

6. Every regular language is context-free.

Answer: Yes.

7. Some context-free languages are not regular.

Answer: Yes.

8. Some context-free languages are regular.
Answer: Yes.
9. Every non-deterministic finite automaton has an equivalent deterministic finite automaton.
Answer: Yes—the construction is algorithmic.
10. Some non-deterministic push-down automata have equivalent regular expressions.
Answer: Yes—some context-free languages are regular.
11. Some finite automata have equivalent regular grammars.
Answer: Yes—in fact all of them do.
12. Every finite automaton has an equivalent regular grammar.
Answer: Yes.
13. Every regular grammar has an equivalent regular expression.
Answer: Yes—the construction is algorithmic.
14. Every context-free grammar has an equivalent non-deterministic finite automaton.
Answer: No—some context-free languages are not regular.
15. Every regular expression has an equivalent deterministic finite automaton.
Answer: Yes—the construction is algorithmic.
16. Every non-deterministic push-down automaton has an equivalent regular grammar.
Answer: No—some context-free languages are not regular.
17. Every subset of a context-free language is context-free.
Answer: No—all languages (including those that are not context-free) are subsets of Σ^* , which is certainly regular, and thereby context-free.
18. The intersection of any regular language and any context-free language is context-free.
Answer: Yes.
19. The intersection of any regular language and any context-free language is regular.
Answer: No—but it is context-free.
20. The union of any regular language and any context-free language is regular.
Answer: No—but it is context-free.
21. Every context-free language contains a regular subset.
Answer: Yes—all languages contain \emptyset , which is certainly regular.

22. The complement of every context-free language is context-free.
Answer: No—some context-free languages do not have a context-free complement.
23. The complement of every regular language is context-free.
Answer: Yes—in fact it is regular.
24. The complement of every regular language is regular.
Answer: Yes.
25. Every recursively enumerable language is accepted by final state by some Turing machine.
Answer: Yes—by definition.
26. Some recursively enumerable languages are not accepted by halting by any Turing machine.
Answer: No—by definition.
27. Every recursive language is recursively enumerable.
Answer: Yes—by definition.

Problem 1079 Fill each empty box that follows an enumerated claim with one of the following four symbols: V 0 $>$ — so that the meaning of the symbols is as follows.

- V means that the preceding claim is always true;
- 0 means that the preceding claim is always false;
- $>$ means that the preceding claim may be true but may be false; however, there exists an algorithm to determine if it is true.
- — means that the preceding claim may be true but may be false, and there does not exist an algorithm to determine if it is true.

Problem assumptions:

$\Sigma = \{a, b, c\}$ is an alphabet.

F is an arbitrary deterministic finite automaton over Σ .

P is an arbitrary non-deterministic pushdown automaton over Σ .

M is an arbitrary Turing machine over Σ , which accepts by halting.

Claims:

1. F accepts string abc . _____ $>$
2. P accepts string abc . _____ $>$
3. M accepts string abc . _____ —
4. F accepts a regular language. _____ V
5. M accepts a regular language. _____ —

6. F accepts a context-free language. V
7. The complement of the language accepted by F is not context-free. 0
8. M does not halt on input aa . —
9. F does not halt on input aa . 0
10. P accepts a context-free language. V
11. The complement of the language accepted by F is regular. V
12. M accepts a decidable language. —
13. The complement of the language accepted by M is decidable. —
14. The complement of the language accepted by M is recursively enumerable. —
15. The complement of the language accepted by P is decidable. V

Specific instructions for Problems 1080–1121:

Fill the empty box at the end of each of the numbered claims with one of the following three signs:

✓ — ?

so that the signs have the following meaning.

- ✓ means that the preceding claim is always true;
- — means that the preceding claim is always false;
- ? means that the preceding claim may be true but may be false, depending on the values of the variable(s) appearing in the claim.

Problem 1080 Problem variables:

G is a regular grammar over the alphabet $\Sigma = \{a, b, c\}$.
 $L(G)$ is the language generated by G .

Claims:

1. $L(G)$ is infinite. ?
2. $L(G)$ is regular. ✓
3. $L(G)$ is context-free. ✓
4. $L(G)$ may be non-regular, but there exists an algorithm to determine whether $L(G)$ is regular. —
5. $L(G)$ is accepted by some nondeterministic pushdown automaton, and there exists an algorithm to construct this automaton. ✓

6. $L(G)$ may be accepted by some deterministic finite automaton, but there is no algorithm to construct this automaton. —

Problem 1081 Problem variables:

G is a context-free grammar over the alphabet $\Sigma = \{a, b, c\}$.

$L(G)$ is the language generated by G .

F is a finite automaton over the alphabet $\Sigma = \{a, b, c\}$.

$L(F)$ is the language accepted by F .

Claims:

1. $L(F) \neq L(G)$. ?
2. $L(F)$ may be equal to $L(G)$, but there is no algorithm to determine whether $L(F) = L(G)$. ✓
3. $L(F)$ may be equal to a^* , but there is no algorithm to determine whether $L(F) = a^*$. —
4. $L(G)$ may be equal to a^* , and there exists an algorithm to determine whether $L(G) = a^*$. —
5. $L(G)$ may be empty, but there is no algorithm to determine whether $L(G) = \emptyset$. —
6. $L(F)$ may be empty, but there exists an algorithm to determine whether $L(F) = \emptyset$. ✓

Problem 1082 Problem variables:

F is a finite automaton over the alphabet $\Sigma = \{a, b, c\}$.
 $L(F)$ is the language accepted by F .

e is a regular expression over the alphabet $\Sigma = \{a, b, c\}$.
 L_e is the language represented by e .

Claims:

1. $L(F) \cup L_e$ is equivalent to some regular expression, and there exists an algorithm to construct this regular expression. ✓
2. $\overline{L(F)}$ may be regular but there is no algorithm to determine whether $\overline{L(F)}$ is regular. —
3. $L(F) \cap L_e$ may be non-regular but it is context-free. —
4. $L(F) = (L(F))^*$. ?
5. $L(F) \subseteq (L(F))^*$. ✓
6. $L(F) \subset (L(F))^*$. ?

Problem 1083 Problem variables:

G_1 and G_2 are two context-free grammars over the alphabet $\Sigma = \{a, b, c\}$.

$L(G_1)$ is the language generated by G_1 , and $L(G_2)$ is the language generated by G_2 .

Claims:

1. $L(G_1) \cup L(G_2)$ is accepted by some pushdown automaton, and there exists an algorithm to construct this pushdown automaton. _____ ?
2. $\overline{L(G_1)}$ may be regular but there is no algorithm to determine whether $\overline{L(G_1)}$ is regular. _____ ?
3. $\overline{L(G_1)}$ may be context-free and there exists an algorithm determine whether $\overline{L(G_1)}$ is context-free. _____ -
4. $L(G_1) \cap L(G_2)$ is regular. _____ ?
5. $L(G_1) \cap L(G_2)$ is context-free. _____ ?
6. If $L(G_1)$ is regular, then $L(G_1) \cap L(G_2)$ is context-free. _____ ?

Problem 1084 Problem variables:

G_1 is a regular grammar over the alphabet $\Sigma = \{a, b, c\}$.

G_2 is an arbitrary context-free grammar over the alphabet $\Sigma = \{a, b, c\}$.

$L(G_1)$ is the language generated by G_1 , and $L(G_2)$ is the language generated by G_2 .

Claims:

1. G_2 is equivalent to some regular grammar. _____ ?
2. G_1 is equivalent to some context-free grammar that is not regular. _____ ?
3. $L(G_1)$ is a regular language. _____ ?
4. $L(G_2)$ is a regular language. _____ ?
5. $L(G_1)$ may be non-regular and there exists an algorithm to determine whether $L(G_1)$ is regular. _____ -
6. $L(G_2)$ may be non-regular and there is no algorithm to determine whether $L(G_2)$ is regular. _____ ?

Problem 1085 Problem variables:

M_1 and M_2 are two nondeterministic pushdown automata over the alphabet $\Sigma = \{a, b, c\}$.

$L(M_1)$ is the language accepted by M_1 and $L(M_2)$ is the language accepted by M_2 .

Claims:

1. M_1 is equivalent to some deterministic pushdown automaton. _____ ?
2. M_1 is equivalent to some non-deterministic finite automaton and there exists an algorithm to construct this automaton. _____ -
3. M_1 may be equivalent to some deterministic finite automaton, but there is no algorithm to construct this automaton. _____ ?
4. $L(M_1) \cup L(M_2)$ is context-free. _____ ?
5. $L(M_1) \cup L(M_2)$ is regular. _____ ?
6. $L(M_1) \cup L(M_2)$ is decidable. _____ ?

Problem 1086 Problem variables:

M is a nondeterministic pushdown automaton over the alphabet $\Sigma = \{a, b, c\}$.

F is a nondeterministic finite automaton over the alphabet $\Sigma = \{a, b, c\}$.

$L(M)$ is the language accepted by M and $L(F)$ is the language accepted by F .

Claims:

1. $L(M) \cap L(F)$ is context-free. _____ ?
2. $L(M) \cup L(F)$ is regular. _____ ?
3. $L(M)L(F)$ is context-free. _____ ?
4. $\overline{L(M)}$ is regular. _____ ?
5. $\overline{L(M)}$ is context-free. _____ ?
6. $\overline{L(M)}$ need not be context-free but there exists an algorithm to determine whether $\overline{L(M)}$ is context-free. _____ -

Problem 1087 Problem variables:

P is a text for a JAVA program, written by a Queens College student in an advanced programming course, to be run in the Programming Lab.

Claims:

1. P compiles correctly. _____ ?
2. P need not be compiling correctly, but there exists an algorithm to determine whether P compiles correctly. _____ ?
3. P plays music if its input is any odd positive integer. _____ ?

4. P need not play music if its input is some odd positive integer, but there exists an algorithm to determine whether P plays music if its input is any odd positive integer. _____
5. P completes its execution on the input string SPRING 2003 in less than 120 minutes. _____
6. P need not complete its execution on the input string SPRING 2003 in less than 120 minutes, but there exists an algorithm to determine whether P does so. _____

Problem 1088 Problem variables:

M is a Turing Machine over the input alphabet $\Sigma = \{a, b, c\}$.

L is the set of input strings on which the Turing machine M diverges.

Claims:

1. L is recursively enumerable. _____
2. \overline{L} (the complement of L) is recursively enumerable.
3. L is decidable. _____
4. L may be equal to a^* , but there is no algorithm to determine whether $L = a^*$. _____
5. L need not be decidable, but there exists an algorithm to determine whether L is decidable. _____
6. L is regular. _____
7. L may be accepted by a finite automaton, and there exists an algorithm to construct this automaton.

Problem 1089 Problem variables:

M is a Turing Machine over the input alphabet $\Sigma = \{a, b, c\}$.

L is the set of input strings on which the Turing machine M diverges.

Claims:

1. \overline{L} (the complement of L) is recursively enumerable.
2. L is decidable. _____
3. If L is recursively enumerable, then L is decidable.
4. If L is recursively enumerable, then \overline{L} is decidable.

5. L need not be decidable, and there is no algorithm to determine whether L is decidable. _____
6. L is context-free. _____
7. L may be accepted by a pushdown automaton, and there exists an algorithm to construct this automaton. _____

Problem 1090 Problem variables:

M is a Turing Machine over the input alphabet $\Sigma = \{a, b, c\}$.

L is the set of input strings that the Turing machine M accepts by halting.

Claims:

1. L is recursively enumerable. _____
2. \overline{L} (the complement of L) is recursively enumerable.
3. L is decidable. _____
4. L is recursively enumerable, but need not be decidable. _____
5. There may exist a Turing Machine M' that accepts L by final state, but there is no algorithm to construct this machine. _____
6. There may exist a Turing Machine M'' that accepts L and halts on every input, but there is no algorithm to construct this machine. _____
7. L cannot be accepted by any finite automaton.

Problem 1091 Problem variables:

M is a Turing Machine over the input alphabet $\Sigma = \{a, b, c\}$.

L is the set of input strings that the Turing machine M accepts by halting.

Claims:

1. L is recursively enumerable. _____
2. \overline{L} (the complement of L) is recursively enumerable.
3. L is decidable. _____
4. L is recursively enumerable but need not be decidable. _____
5. There exists a Turing Machine M' that accepts L by final state, and there exists an algorithm to construct this machine. _____

6. There exists a Turing Machine M'' that accepts L and halts on every input, and there exists an algorithm to construct this machine. _____

7. L is regular. _____ ?

Problem 1092 Problem variables:

F is a finite automaton over the alphabet $\Sigma = \{a, b, c\}$.
 $L(F)$ is the language accepted by F (and $\overline{L(F)}$ is the complement of $L(F)$.)

Claims:

1. $L(F)$ is infinite. _____ ?

2. $L(F)$ can be represented by a context-free grammar.

3. $L(F)$ sometimes can be represented by a context-free grammar, but not always by a regular one.

4. $L(F)$ always can be represented by a regular context-free grammar, and there exists an algorithm to construct this grammar. _____

5. $\overline{L(F)}$ is regular, but not necessarily context-free.

6. $\overline{L(F)}$ is empty. _____ ?

7. $\overline{L(F)}$ need not be empty, but there is no algorithm to determine whether $\overline{L(F)}$ is empty. _____

Problem 1093 Problem variables:

F is a nondeterministic finite automaton over the alphabet $\Sigma = \{a, b, c\}$.

$L(F)$ is the language accepted by F (and $\overline{L(F)}$ is the complement of $L(F)$.)

Claims:

1. $L(F)$ is accepted by some deterministic finite automaton. _____

2. $L(F)$ can be represented by a regular expression.

3. $\overline{L(F)}$ is regular, but not necessarily context-free.

4. $L(F)$ sometimes can be represented by a context-free grammar, but not always by a regular one.

5. $L(F)$ always can be represented by a regular context-free grammar, but there is no algorithm to construct this grammar. _____

6. $\overline{L(F)}$ is empty. _____ ?

7. $\overline{L(F)}$ need not be empty, but there exists an algorithm to determine whether $\overline{L(F)}$ is empty. _____

Problem 1094 Problem variables:

F is a finite automaton over the alphabet $\Sigma = \{a, b, c\}$.
 $L(F)$ is the language accepted by F .

e is a regular expression over the alphabet $\Sigma = \{a, b, c\}$.
 L_e is the language represented by e .

Claims:

1. $L(F)$ may be equal to a^* , but there is no algorithm to determine whether $L = a^*$. _____

2. $L(F) \cap L_e$ is regular. _____

3. $L(F) \cap L_e$ is context-free. _____

4. $L(F) \cap L_e$ may be finite, and there exists an algorithm to determine whether $L(F) \cap L_e$ is finite.

5. $L(F) \cap L_e$ is accepted by some deterministic finite automaton, and there exists an algorithm to construct this automaton. _____

6. $\overline{L(F)}$ (the complement of $L(F)$) need not be regular, but it is context-free. _____

7. There exists always a finite automaton that accepts $L(F) \cup L_e$, but not always a deterministic one. _____

Problem 1095 Problem variables:

F_1 and F_2 are finite automata over the alphabet $\Sigma = \{a, b, c\}$.

L_1 is the language accepted by F_1 , and L_2 is the language accepted by F_2 .

Claims:

1. L_1 may be a subset of L_2 , and there exists an algorithm to determine whether $L \subseteq L_2$. _____

2. $L_1 \cup L_2$ need not be regular. _____

3. $L_1 \cup L_2$ is regular, but need not have a regular complement. _____

4. $L_1 \cap L_2$ may be finite, but there is no algorithm to determine whether $L_1 \cap L_2$ is finite. _____

5. $L_1 \cap L_2$ is accepted by some deterministic finite automaton, and there exists an algorithm to construct this automaton. _____

6. $\overline{L_1}$ (the complement of L_1) is infinite. _____ ?

7. There exists always a finite automaton that accepts $\overline{L_1}$, but there is no algorithm to construct this automaton. _____

Problem 1096 Problem variables:

G_1 and G_2 are two context-free grammars over the alphabet $\Sigma = \{a, b, c\}$.

L_1 is the language generated by G_1 , and L_2 is the language generated by G_2 .

Claims:

1. L_1 may be equal to a^* , but there is no algorithm to determine whether $L_1 = a^*$. _____
2. $L_1 \cap L_2$ is regular. _____
3. $L_1 \cap L_2$ is context-free. _____
4. $L_1 \cup L_2$ is accepted by some pushdown automaton, and there exists an algorithm to construct this automaton. _____
5. $L_1 \cap L_2$ is accepted by some pushdown automaton, and there exists an algorithm to construct this automaton. _____
6. $\overline{L_1}$ (the complement of L_1) is context-free. _____
7. There exists always a pushdown automaton that accepts L_1^* , but not always a deterministic one. _____

Problem 1097 Problem variables:

G_1 and G_2 are two context-free grammars over the alphabet $\Sigma = \{a, b, c\}$.

L_1 is the language generated by G_1 , and L_2 is the language generated by G_2 .

Claims:

1. L_1 may be equal to L_2 , but there is no algorithm to determine whether $L_1 = L_2$. _____
2. $L_1 \cap L_2$ is decidable. _____
3. $L_1 \cap L_2$ is context-free. _____
4. $L_1 \cap L_2$ is accepted by some pushdown automaton, but not necessarily a deterministic one. _____
5. $\overline{L_1}$ (the complement of L_1) is context-free. _____
6. L_1^* is accepted by some pushdown automaton, and there exists an algorithm to construct this automaton. _____
7. There exists always a pushdown automaton that accepts $L_1 L_2$, but not always a deterministic one. _____

Problem 1098 Problem variables:

G is a context-free grammar over the alphabet $\Sigma = \{a, b, c\}$.

$L(G)$ is the language generated by G .

e is a regular expression over the alphabet $\Sigma = \{a, b, c\}$.

L_e is the language represented by e .

Claims:

1. G is regular. _____
2. G need not be regular, and there is no algorithm to determine whether G is regular. _____
3. $L(G)$ is regular. _____
4. $L(G)$ need not be regular, and there is no algorithm to determine whether $L(G)$ is regular. _____
5. $L(G)$ may be equal to a^* , but there is no algorithm to determine whether $L(G) = a^*$. _____
6. $L(G) \cap L_e$ is regular. _____
7. $L(G) \cap L_e$ is context-free. _____

Problem 1099 Problem variables:

G is a context-free grammar over the alphabet $\Sigma = \{a, b, c\}$.

$L(G)$ is the language generated by G .

e is a regular expression over the alphabet $\Sigma = \{a, b, c\}$.

L_e is the language represented by e .

Claims:

1. G is regular. _____
2. G need not be regular, but there exists an algorithm to determine whether G is regular. _____
3. $L(G)$ is regular. _____
4. $L(G)$ need not be regular, but there exists an algorithm to determine whether $L(G)$ is regular. _____
5. $L(G)$ may be equal to a^* , and there exists an algorithm to determine whether $L(G) = a^*$. _____
6. $L(G) \cap L_e$ is not regular. _____
7. $L(G) \cap L_e$ is not context-free. _____

Problem 1100 Problem variables:

M_1 and M_2 are two nondeterministic pushdown automata over the alphabet $\Sigma = \{a, b, c\}$.

L_1 is the language accepted by M_1 and L_2 is the language accepted by M_2 .

Claims:

1. M_1 is equivalent to some deterministic pushdown automaton. _____
2. M_1 can be simulated by a Turing Machine. _____
3. M_1 can be simulated by a deterministic Turing Machine. _____
4. M_1 can be simulated by a deterministic Turing Machine, but only if M_1 itself is deterministic. _____
5. M_1 need not be deterministic, but there exists an algorithm to convert it to a deterministic pushdown automaton. _____
6. L_1^* is regular. _____
7. $L_1 L_2$ is accepted by a pushdown automaton, and there exists an algorithm to construct this automaton. _____

Problem 1101 Problem variables:

M is a nondeterministic pushdown automaton over the alphabet $\Sigma = \{a, b, c\}$.

L is the language accepted by M .

Claims:

1. L is accepted by some deterministic pushdown automaton. _____
2. \overline{L} (the complement of L) need not be decidable, and there is no algorithm to determine whether it is decidable. _____
3. \overline{L} is recursively enumerable. _____
4. \overline{L} is context-free. _____
5. \overline{L} is regular. _____
6. There exists a Turing Machine that diverges exactly when the input word is an element of L . _____
7. L is accepted by some deterministic Turing Machine that halts on every input, and there exists an algorithm to construct this machine. _____

Problem 1102 Problem variables:

M is a nondeterministic pushdown automaton over the alphabet $\Sigma = \{a, b, c\}$.

L is the language accepted by M .

Claims:

1. L is regular. _____
2. L need not be decidable, but there exists an algorithm to determine whether it is decidable. _____

3. L is accepted by some Turing Machine that halts on every input. _____
4. L need not be accepted by any Turing Machine that halts on every input, but there exists an algorithm to determine whether L is accepted by such a Turing Machine. _____
5. L is accepted by some deterministic Turing Machine that halts on every input, and there exists an algorithm to construct this machine. _____
6. \overline{L} is decidable. _____
7. \overline{L} is recursively enumerable. _____

Problem 1103 Problem variables:

THEORIST is a JAVA program that implements a virtual student: an interactive animation of a student in Queens College, especially well suited for simulations of test taking in the subjects of Computation Theory.

Claims:

1. THEORIST is compliant with the Java language standard. _____
2. THEORIST need not be compliant with the Java language standard, and there is no algorithm to determine whether it is compliant. _____
3. THEORIST can solve Problem 640, for any input automaton. _____
4. THEORIST need not be able to solve Problem 640 correctly for every input automaton, but there exists an algorithm to determine whether THEORIST solves it correctly. _____
5. Dangerous Professor can give a homework to her students to implement that function of the THEORIST that solves Problem 839 correctly, for any input Turing Machine. _____
6. THEORIST can state the Pumping Lemma correctly within the exam time interval of 120 minutes. _____
7. THEORIST need not be able to state the Pumping Lemma correctly within the exam time interval of 120 minutes, but there exists an algorithm to determine whether THEORIST accomplishes this. _____

Problem 1104 Problem variables:

THEORIST is a JAVA program that implements a virtual student: an interactive animation of a student in Queens College, especially well suited for simulations of test taking in the subjects of Computation Theory.

Claims:

1. THEORIST is compliant with the Java language standard. _____ ?
2. THEORIST need not be compliant with the Java language standard, but there exists an algorithm to determine whether it is compliant. _____
3. THEORIST can solve Problem 839, for any Turing Machine. _____ -
4. THEORIST need not be able to solve Problem 839 correctly for every input Turing Machine, but there exists an algorithm to determine whether THEORIST solves it correctly. _____ -
5. Dangerous Professor can give a homework to her students to implement that function of the THEORIST that solves Problem 640 correctly, for any finite automaton. _____
6. THEORIST can write its student ID within the exam time interval of 120 minutes. _____ ?
7. THEORIST need not be able to write its student ID within the exam time interval of 120 minutes, and there is no algorithm to determine whether THEORIST accomplishes this. _____ -

Problem 1105 Problem variables:

G is a context-free grammar over the alphabet $\Sigma = \{a, b, c\}$.

$L(G)$ is the language generated by G .

F is a finite automaton over the alphabet $\Sigma = \{a, b, c\}$.

$L(F)$ is the language accepted by F .

Claims:

1. $L(F) = L(G)$. _____ ?
2. $L(F)$ may be equal to $L(G)$, and there is no algorithm to determine whether $L(F) = L(G)$. _____
3. $L(F)$ may be equal to a^* , but there is no algorithm to determine whether $L(F) = a^*$. _____ -
4. $L(G)$ may be equal to a^* , and there exists an algorithm to determine whether $L(G) = a^*$. _____ -
5. $L(G)$ may be empty, and there exists an algorithm to determine whether $L(G) = \emptyset$. _____
6. $L(F)$ may be empty, and there is no algorithm to determine whether $L(F) = \emptyset$. _____ -

Problem 1106 Problem variables:

G_1 and G_2 are two context-free grammars over the alphabet $\Sigma = \{a, b, c\}$.

$L(G_1)$ is the language generated by G_1 , and $L(G_2)$ is the language generated by G_2 .

Claims:

1. $\overline{L(G_1)}$ (the complement of $L(G_1)$) may be regular but there is no algorithm to determine whether $\overline{L(G_1)}$ is regular. _____
2. $\overline{L(G_1)}$ is context-free. _____ ?
3. $\overline{L(G_1)}$ need not be context-free, but there exists an algorithm determine whether $\overline{L(G_1)}$ is context-free. _____ -
4. $L(G_1) \cap L(G_2)$ is accepted by some pushdown automaton, and there exists an algorithm to construct this pushdown automaton. _____ -
5. $L(G_1) \cap L(G_2)$ may be accepted by some pushdown automaton, but there is no algorithm to construct this pushdown automaton. _____
6. $L(G_1) \cap L(G_2)$ is regular. _____ ?
7. $L(G_1) \cap L(G_2)$ is context-free. _____ ?
8. $L(G_1)L(G_2)$ is context-free. _____
9. If $L(G_1)$ is regular, then $L(G_1) \cap L(G_2)$ is context-free. _____

Problem 1107 Problem variables:

G_1 is a regular grammar over the alphabet $\Sigma = \{a, b, c\}$. G_2 is an arbitrary context-free grammar over the alphabet $\Sigma = \{a, b, c\}$.

$L(G_1)$ is the language generated by G_1 , and $L(G_2)$ is the language generated by G_2 .

Claims:

1. G_2 is equivalent to some regular grammar. _____ ?
2. G_1 is equivalent to some context-free grammar that is not regular. _____
3. $L(G_1)$ is a regular language. _____
4. $L(G_2)$ is a regular language. _____ ?
5. $\overline{L(G_1)}$ (the complement of $L(G_1)$) is a regular language. _____
6. $\overline{L(G_2)}$ is a regular language. _____ ?
7. $L(G_1)$ may be non-regular and there exists an algorithm to determine whether $L(G_1)$ is regular. _____ -
8. $L(G_2)$ may be non-regular and there is no algorithm to determine whether $L(G_2)$ is regular. _____

Problem 1108 Problem variables:

M_1 and M_2 are two nondeterministic pushdown automata over the alphabet $\Sigma = \{a, b, c\}$.

$L(M_1)$ is the language accepted by M_1 and $L(M_2)$ is the language accepted by M_2 .

Claims:

1. M_1 is equivalent to some deterministic pushdown automaton. _____
2. M_1 is equivalent to some non-deterministic finite automaton and there exists an algorithm to construct this automaton. _____
3. M_1 may be equivalent to some deterministic finite automaton, but there is no algorithm to construct this automaton. _____
4. $L(M_1) \cap L(M_2)$ is context-free. _____
5. $L(M_1) \cap L(M_2)$ is regular. _____
6. $L(M_1) \cup L(M_2)$ is decidable. _____
7. $L(M_1) \cup L(M_2)$ is a recursively enumerable language. _____

Problem 1109 Problem variables:

M_1 and M_2 are two nondeterministic pushdown automata over the alphabet $\Sigma = \{a, b, c\}$.

$L(M_1)$ is the language accepted by M_1 and $L(M_2)$ is the language accepted by M_2 .

Claims:

1. M_1 is equivalent to some deterministic pushdown automaton. _____
2. M_1 is equivalent to some non-deterministic finite automaton. _____
3. M_1 may be equivalent to some deterministic finite automaton, and there is an algorithm to construct this automaton. _____
4. $L(M_1)^*$ is context-free. _____
5. $L(M_1) \cup L(M_2)$ is context-free. _____
6. $L(M_1) \cup L(M_2)$ is regular. _____
7. $L(M_1) \cup L(M_2)$ need not be regular, but there is an algorithm to determine whether $L(M_1) \cup L(M_2)$ is regular. _____

8. $L(M_1) \cup L(M_2)$ need not be context-free, but there is an algorithm to determine whether $L(M_1) \cup L(M_2)$ is context-free. _____

Problem 1110 Problem variables:

M is a nondeterministic pushdown automaton over the alphabet $\Sigma = \{a, b, c\}$.

F is a nondeterministic finite automaton over the alphabet $\Sigma = \{a, b, c\}$.

$L(M)$ is the language accepted by M and $L(F)$ is the language accepted by F .

Claims:

1. $L(M) \cap L(F)$ is context-free. _____
2. $L(M) \cap L(F)$ need not be context-free but there exists an algorithm to determine whether it is context-free. _____
3. $L(M) \cup L(F)$ is regular. _____
4. $L(M) \cup L(F)$ is context-free. _____
5. $L(M)L(F)$ is context-free. _____
6. $\overline{L(M)}$ (the complement of $L(M)$) is a regular language. _____
7. $\overline{L(M)}$ is context-free. _____
8. $\overline{L(M)}$ need not be context-free but there exists an algorithm to determine whether $\overline{L(M)}$ is context-free. _____

Problem 1111 Problem variables:

M is a nondeterministic pushdown automaton over the alphabet $\Sigma = \{a, b, c\}$.

F is a nondeterministic finite automaton over the alphabet $\Sigma = \{a, b, c\}$.

$L(M)$ is the language accepted by M and $L(F)$ is the language accepted by F .

Claims:

1. $L(M)L(F)$ is context-free. _____
2. $L(M)L(F)$ need not be context-free but there exists an algorithm to determine whether it is context-free. _____
3. $L(M) \cap L(F)$ is regular. _____
4. $L(M) \cup L(F)$ is regular. _____
5. $L(M)L(F)$ is regular. _____

6. $\overline{L(F)}$ (the complement of $L(F)$) is context-free. _____
7. $\overline{L(M)}$ need not be context-free but there exists an algorithm to determine whether $\overline{L(M)}$ is context-free. _____

Problem 1112 Problem variables:

P is a text of a C++ program for sorting arbitrary sequences of letters, written by a Queens College student in an advanced programming course, to be run in the Programming Lab.

Claims:

1. P contains compile-time errors. _____
2. P may compile correctly, but there is no algorithm to determine whether P compiles correctly. _____
3. P gives an incorrect result on some input sequences. _____
4. P gives an incorrect result on some input sequences of length less than 200. _____
5. P may give an incorrect result on some input sequences of length less than 200, but there exists an algorithm to determine whether P gives an incorrect result on some such sequences. _____
6. P completes its execution on the input string FINAL 2003 in less than 120 minutes. _____
7. P need not complete its execution on the input string FINAL 2003 in less than 120 minutes, but there exists an algorithm to determine whether P does so. _____

Problem 1113 Problem variables:

P is a text of a C++ program that computes the Fibonacci sequence for an arbitrary integer argument, written by a Queens College student in an advanced programming course, to be run in the Programming Lab.

Claims:

1. P is compliant with the C++ language standard. _____
2. P need not be compliant with the C++ language standard, but there is an algorithm to determine whether it is compliant. _____
3. P computes the 36th Fibonacci number correctly. _____
4. P need not compute the 36th Fibonacci number correctly, but there is an algorithm to determine whether P computes this Fibonacci number correctly. _____

5. P computes the 36th Fibonacci number correctly in less than 120 minutes. _____

6. P need not compute the 36th Fibonacci number correctly in less than 120 minutes, and there is no algorithm to determine whether P does so. _____

7. P computes every Fibonacci number correctly. _____

8. P need not compute every Fibonacci number correctly, but there is an algorithm to determine whether P computes every Fibonacci number correctly. _____

Problem 1114 Problem variables: M_1 is an arbitrary non-deterministic finite automaton; M_2 is an arbitrary non-deterministic pushdown automaton.

Claims:

1. M_1 is equivalent to a deterministic finite automaton. _____
2. M_2 is equivalent to a deterministic pushdown automaton. _____
3. There exists an algorithm which on input M_1 outputs a decision whether $L(M_1)$ is empty. _____
4. There exists an algorithm which on input M_2 outputs a decision whether $L(M_2)$ is empty. _____
5. There exists an algorithm which on input M_1 outputs a context-free grammar for the language $L(M_1)$, accepted by M_1 . _____
6. There exists an algorithm which on input M_2 outputs a context-free grammar for the language $L(M_2)$, accepted by M_2 . _____
7. There exists an algorithm which on input M_1 outputs a decision whether $L(M_1)$ is equal to $(a \cup b)^*$. _____
8. There exists an algorithm which on input M_2 outputs a decision whether $L(M_2)$ is equal to $(a \cup b)^*$. _____

Problem 1115 Problem variables:

L_1 and L_2 are arbitrary decidable languages.

L_3 is an arbitrary recursively enumerable language.

Claims:

1. $L_1 \setminus L_3$ is decidable. _____
2. $L_1 \setminus L_3$ is recursively enumerable. _____

3. $L_1 \cup L_3$ is decidable. _____ ?
4. $L_1 \cap L_3$ is recursively enumerable. _____ ✓
5. $L_1 L_2$ is decidable. _____ ✓
6. If L_3 is decidable then its complement $\overline{L_3}$ is not decidable. _____ -

Problem 1116 Problem variables:

M is an arbitrary Turing Machine that accepts by final state. $L(M)$ is the language accepted by M .

Claims:

1. M halts on every input string. _____ ?
2. $L(M)$ is a regular language. _____ ?
3. $L(M)$ may be a regular language but there does not exist an algorithm to determine whether it is regular. _____ ✓
4. There exists a Turing Machine M' that accepts $L(M)$ by halting, and there exists an algorithm to construct M' . _____ ✓
5. $L(M)$ is recursively enumerable. _____ ✓
6. $\overline{L(M)}$ (the complement of $L(M)$) is recursively enumerable. _____ ?

Problem 1117 Problem variables:

G is an arbitrary context-free grammar.

Claims:

1. G is equivalent to a deterministic finite automaton. _____ ?
2. G is equivalent to a non-deterministic pushdown automaton. _____ ✓
3. $L(G)$ is a regular language. _____ ?
4. $\overline{L(G)}$ (the complement of $L(G)$) is a context-free language. _____ ?
5. $\overline{L(G)}$ (the complement of $L(G)$) may be a context-free language, and there exists an algorithm to determine whether it is context-free. _____ -
6. $\overline{L(G)}$ (the complement of $L(G)$) may be a context-free language, but there does not exist an algorithm to determine whether it is context-free. _____ ✓

Problem 1118 Problem variables:

G is an arbitrary context-free grammar.

Claims:

1. G is a regular grammar. _____ ?
2. G may be a regular grammar, and there exists an algorithm to determine whether it is regular. _____ ✓
3. $L(G)$ is a regular language. _____ ?
4. $L(G)$ may be a regular language, and there exists an algorithm to determine whether it is regular. _____ -
5. $L(G)$ may be empty, but there is no algorithm to determine whether it is empty. _____ -
6. $\overline{L(G)}$ (the complement of $L(G)$) is a regular language. _____ ?

Problem 1119 Problem variables:

e is an arbitrary regular expression. $L(e)$ is the language represented by the regular expression e .

Claims:

1. $L(e)$ is not empty. _____ ?
2. $L(e^*)$ is not empty. _____ ✓
3. $L(e)$ may be empty, but there is no algorithm to determine whether it is empty. _____ -
4. There exists a non-deterministic finite automaton equivalent to e , and there exists an algorithm to construct it. _____ ✓
5. There exists a deterministic finite automaton equivalent to e , but there does not exist an algorithm to construct it. _____ -
6. $\overline{L(e)}$ (the complement of $L(e)$) may be regular, but there is no algorithm to determine whether it is regular. _____ -

Problem 1120 Problem variables:

M is an arbitrary Turing Machine, and $L(M)$ is the language accepted by M . F is an arbitrary finite automaton, and $L(F)$ is the language accepted by F . L is the set of strings over the alphabet $\{a, b\}$ that contain exactly one b .

Claims:

1. $L(M)$ is equal to L . _____ ?
2. $L(M)$ may be equal to L and there exists an algorithm to determine whether $L(M)$ is equal to L . _____ -
3. $L(M)$ may be equal to L , but there does not exist an algorithm to determine whether $L(M)$ is equal to L . _____ ✓

4. $L(F)$ is equal to L . _____ ?
5. $L(F)$ may be equal to L and there exists an algorithm to determine whether $L(F)$ is equal to L .
6. $L(F)$ may be equal to L , but there does not exist an algorithm to determine whether $L(F)$ is equal to L . _____ -

a	a	bb	bb
aa	bb	b	b

Figure 377:

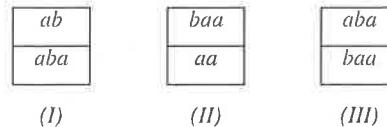


Figure 378:

Problem 1121 Problem variables:

L_1 is an arbitrary decidable language; M_1 is a Turing machine that decides L_1 ; L_2 is an arbitrary recursively enumerable language; M_2 is a Turing machine that accepts L_2 by halting.

Claims:

1. There exists a Turing machine that accepts L_1 by halting. _____
2. There exists a Turing machine that accepts L_2 by final state. _____
3. There exists a Turing machine that accepts L_2 by final state and halts on every input. _____ ?
4. L_1 has a decidable complement. _____
5. L_2 has a decidable complement. _____ ?
6. There exists an algorithm which on input M_2 outputs a decision whether $L(M_2)$ is equal to $(a \cup b)^*$. _____ -
7. There exists an algorithm which on input M_2 outputs a decision whether $L(M_2)$ is decidable. _____ -
8. L_2 is enumerable in a lexicographic (length-first) order. _____ ?
9. L_1 is enumerable in a lexicographic (length-first) order. _____

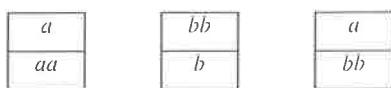
Problem 1122 (a) Show a solution (win) to the instance of the Modified Post's Correspondence Problem represented on Figure 376.

Figure 376:

Answer: The solution is shown on Figure 377.

(b) Prove that the instance of the Modified Post's Correspondence Problem represented on Figure 378 has no solution (win).

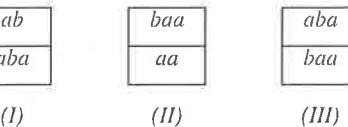


Figure 377:

Figure 378:

Answer: Domino (I) has a trailing unmatched a in the lower row; hence the second domino must have a leading a in the upper row. There are two such dominos: (I) and (III). Domino (I) cannot be used here, as it produces $abab$ in the upper row, and $aba\bar{a}ba$ in the lower row, while these two strings differ at the position indicated by “ $\bar{}$ ”. After putting (III) after (I), we have $ababa$ in the upper row and $ababaa$ in the lower row; hence the next domino again must have a leading a in the upper row, which is equivalent to the situation after the first domino—the lower row retains a trailing unmatched a .

Problem 1123 (a) Find the solution (win) of the following instance of the Post's Correspondence Problem:

1	10111	10
111	10	0

If the solution does not exist, prove it.

Answer:

10111	1	1	10
10	111	111	0

(b) Describe an algorithm that can be employed to answer the question stated in part (a) for an arbitrary instance of the Post's Correspondence Problem. If such algorithm does not exist, explain why.

Answer: Such algorithm does not exist, since the Post's Correspondence Problem is undecidable, which is proved by reduction from the Halting Problem.

Problem 1124 (a) Show a solution (win) to the instance of the Modified Post's Correspondence Problem represented on Figure 379.

Answer: The solution is shown on Figure 381.

(b) Prove that the instance of the Modified Post's Correspondence Problem represented on Figure 380 has no solution (win).

Answer: Domino (I) has a trailing unmatched a in the lower row; hence the second domino must have a leading a in the upper row. There are two such dominos: (I) and (III). Domino (I) cannot be used here, as it produces $abab$ in the upper row, and $aba\bar{a}ba$ in the lower row, while these two strings differ at the position indicated by “ $\bar{}$ ”. If domino (III) is used after (I), we have $ababa$

in the upper row and $ababab$ in the lower row; hence the next domino must have a leading b in the upper row. The only such domino is (II). However, (II) cannot be used here, as it produces $ababab\bar{b}a$ in the upper row, and $ababab\bar{a}a$ in the lower row, while these two strings differ at the position indicated by “ \leftarrow ”.

ab	$bbbb$	bb	ab
abb	bb	bab	bb

Figure 379:

ab	bba	aba
aba	aa	bab

(I) (II) (III)

Figure 380:

ab	bb	ab	$bbbb$
abb	bab	bb	bb

Figure 381:

Problem 1125 Find the solution (win) of the following instance of the Post's Correspondence Problem:

0	101	0111	001
110	10	0101	10

If the solution does not exist, prove it.

Advice for Answer: The solution string is:

10100101110

A Sample Diagonalization Proofs

Problem 1126 Set of subsets of the set N of natural numbers is uncountable.

Proof. Assume the opposite, that the set of all subsets of N is countable. Then, there exists an infinite sequence of sets:

$$S_0, S_1, S_2, \dots$$

in which *every subset* of N appears.

We refer to this sequence to define a subset $Z \subset N$ as follows. For an arbitrary natural number, say n :

$$n \in Z \iff n \notin S_n$$

Since Z is a subset of N , Z has to appear in the sequence S_0, S_1, S_2, \dots . In other words, there exists some index, say ℓ , such that:

$$Z = S_\ell$$

But, then:

$$\ell \in Z \iff \ell \notin S_\ell \iff \ell \notin Z$$

where the first equivalence follows from the definition of Z , and the second equivalence follows from the fact that $Z = S_\ell$. However, the claim:

$$\ell \in Z \iff \ell \notin Z$$

is a contradiction.

In summary, having assumed that the listing of all subsets of N is complete, we employ the listing to construct a subset that cannot be listed, since it by construction differs from every subset in the list.

Problem 1127 Set of all total functions from the set N of natural numbers to the set $\{0, 1\}$ is uncountable.

Proof. Assume the opposite, that the set of all total functions from the set N of natural numbers to the set $\{0, 1\}$ is countable. Then, there exists an infinite sequence of functions:

$$f_0, f_1, f_2, \dots$$

in which *every total function* from N to $\{0, 1\}$ appears.

We refer to this sequence to define a total function $g : N \rightarrow \{0, 1\}$ as follows. For an arbitrary natural number, say n :

$$g(n) = \begin{cases} 0 & \text{iff } f_n(n) = 1 \\ 1 & \text{iff } f_n(n) = 0 \end{cases}$$

Since $f_n(n)$ is defined for every n , so is $g(n)$ —hence, g is a total function from N to $\{0, 1\}$. Then, g has to appear in the sequence f_0, f_1, f_2, \dots . In other words, there exists some index, say ℓ , such that:

$$g = f_\ell$$

meaning that $g(x) = f_\ell(x)$ for every $x \in N$. But, then:

$$g(\ell) = 0 \iff f_\ell(\ell) = 1 \iff g(\ell) = 1$$

where the first equality follows from the definition of g , and the second equality follows from the fact that $g = f_\ell$.

However, the claim:

$$g(\ell) = 0 \iff g(\ell) = 1$$

is a contradiction, since g is always defined (as it is a total function) and its value at any point must be equal to either 0 or 1 (as its codomain is the set $\{0, 1\}$, by definition of g).

In summary, having assumed that the listing of all total functions from N to $\{0, 1\}$ is complete, we employ the listing to construct a function that cannot be listed, since it by construction differs from every function in the list. Precisely, for every $n \in N$, g differs from the n th function in the list in the value it assigns to number n .

B Sample Construction Correctness Proof

Problem 1128 Let $G = (V, \Sigma, P, S)$ be a grammar over alphabet $\Sigma = \{a, b\}$ such that $V = \{S, B\}$ and the set P consists of the following productions:

$$\begin{aligned} S &\rightarrow aS \mid bB \\ B &\rightarrow aB \mid bS \mid \lambda \end{aligned}$$

(a) Prove that every string of the language $L(G)$, generated by the grammar G , contains an odd number of occurrences of letter b .

Answer: We prove the following claim:

For every sentential form $w \in (V \cup \Sigma)^*$ derivable in G the quantity

$$n_w = n_w(b) + 2n_w(B) + n_w(S)$$

is odd, where $n_w(\alpha)$ denotes the number of occurrences of symbol $\alpha \in V \cup \Sigma$ in the sentential form w .

Straightforwardly, if w is a sentence in $L(G)$, that is a sentential form consisting of terminals alone, then $n_w(B) = n_w(S) = 0$, so $n_w = n_w(b)$, and our claim guarantees that $n_w(b)$ is odd, which means that w has an odd number of occurrences of symbol b .

We prove the claim by induction on the length of the derivation of a sentential form in G .

In the base case, consider the sentential forms obtained by derivations of length 1. There are two of them: aS and bB . By inspection:

$$\begin{aligned} n_{(aS)} &= n_{(aS)}(b) + 2n_{(aS)}(B) + n_{(aS)}(S) \\ &= 0 + 0 + 1 = 1 \\ n_{(bB)} &= n_{(bB)}(b) + 2n_{(bB)}(B) + n_{(bB)}(S) \\ &= 1 + 2 \cdot 1 + 0 = 3 \end{aligned}$$

hence, both $n_{(aS)}$ and $n_{(bB)}$ are odd.

Assume that every sentential form w' derivable in G in no more than m steps has the property that $n_{w'}$ is odd. We have to prove that every sentential form w derivable in G in $m+1$ steps has the property that n_w is odd. Let w be one such sentential form, derivable in G in $m+1$ steps. Then, the derivation of w is of the form:

$$S \xrightarrow{m} w' \xrightarrow{1} w$$

In other words, there is a sentential form w' , derived in G in m steps, from which w is derived in a single step. By the inductive hypothesis, as w' is derived in m steps:

$$n_{w'} = n_{w'}(b) + 2n_{w'}(B) + n_{w'}(S)$$

is odd. Now, look at the last production in the derivation of w , which transforms w' to w . This production may be any of the five productions in G . We consider each one of them, keeping track of the symbol it eliminates from w' (the left side), and taking into account the symbols it introduces into w' (the right side) to create w . In summary:

$$\begin{aligned} S \rightarrow aS &\implies n_w = n_{w'} - 1 + 1 = n_{w'} \\ S \rightarrow bB &\implies n_w = n_{w'} - 1 + 1 + 2 \cdot 1 = n_{w'} + 2 \\ B \rightarrow aB &\implies n_w = n_{w'} - 2 \cdot 1 + 2 \cdot 1 = n_{w'} \\ B \rightarrow bS &\implies n_w = n_{w'} - 2 \cdot 1 + 1 + 1 = n_{w'} \\ B \rightarrow \lambda &\implies n_w = n_{w'} - 2 \cdot 1 + 0 = n_{w'} - 2 \end{aligned}$$

In each case, if $n_{w'}$ is odd, so is n_w , whence the claim.

(b) Prove that every string over the alphabet $\{a, b\}$ with an odd number of occurrences of letter b belongs to $L(G)$, the set of strings generated by G .

Answer: We prove the following claim by induction on n :

Every string over the alphabet $\{a, b\}$ which contains exactly $2n+1$ occurrences of letter b is derivable in G from the start symbol S .

The base case, when $n = 0$, corresponds to a string that contains exactly $2 \cdot 0 + 1 = 1$ occurrence of b . Such string is of the form:

$$w_0 = a^k b a^\ell, \text{ where } k, \ell \geq 0$$

The derivation of w_0 proceeds according to the following sequence.

k applications of the rule $S \rightarrow aS$, which generate the sentential form $a^k S$;

one application of the rule $S \rightarrow bB$, which generates the sentential form $a^k b B$;

ℓ applications of the rule $B \rightarrow aB$, which generate the sentential form $a^k b a^\ell B$;

one application of the rule $B \rightarrow \lambda$, which generates the word $a^k b a^\ell = w_0$.

Inductively, assume that for some $n \geq 0$, every string over the alphabet $\{a, b\}$ which contains exactly $2n + 1$ occurrences of letter b is derivable in G from the start symbol S . We need to prove that every string over the alphabet $\{a, b\}$ which contains exactly $2(n + 1) + 1 = 2n + 3$ occurrences of letter b is also derivable in G from the start symbol S .

Let w be an arbitrary string over the alphabet $\{a, b\}$ with exactly $2n + 3$ occurrences of b . Since $2n + 3 \geq 3 > 2$, w certainly contains more than 2 occurrences of b . Hence, w can be written in a form that represents a concatenation of a prefix that contains exactly 2 occurrences of b and a suffix that contains the remaining $2n + 1$ occurrences of b :

$$w = a^k b a^\ell b w', \text{ where } k, \ell \geq 0$$

and w' is a string over the alphabet $\{a, b\}$ with exactly $2n + 1$ occurrences of b . By the inductive hypothesis, there exists a derivation of w' from S , and we are going to employ this derivation to construct a derivation of the entire string w .

The derivation of w proceeds according to the following sequence.

k applications of the rule $S \rightarrow aS$, which generate the sentential form $a^k S$;

one application of the rule $S \rightarrow bB$, which generates the sentential form $a^k b B$;

ℓ applications of the rule $B \rightarrow aB$, which generate the sentential form $a^k b a^\ell B$;

one application of the rule $B \rightarrow bS$, which generates the sentential form $a^k b a^\ell b S$;

the sequence of rule applications that derives w' from S (by the inductive hypothesis), which generates the word $a^k b a^\ell b w' = w$.

C Algorithm Checklist

1. Input: finite sequence of natural numbers;
Output: Gödel number of the sequence.
2. Input: natural number;
Output: sequence of integers that is the pre-image of the input under the Gödel numbering,
NO if the pre-image does not exist.
3. Input: algorithm to decide whether $x \in L_1$ and $x \in \overline{L}_2$ for every string x and given languages L_1 and L_2 ;
Output: algorithm to decide whether $x \in L_1 \cap L_2$, $x \in L_1 \setminus L_2$, $x \in L_1 \cup L_2$, $x \in L_1 L_2$, and $x \in L_1^*$, for every string x .
4. Input: listing of finite language L ;
Output: regular expressions for L and \overline{L} .
5. Input: regular expression;
Output: equivalent context-free grammar.
6. Input: context-free grammars G_1 and G_2 ;
Output: context-free grammars for:
 $L(G_1)^*$, $L(G_1) \cup L(G_2)$, $L(G_1) \cdot L(G_2)$.
7. Input: finite automata F_1 and F_2 ;
Output: finite automata for:
 $L(F_1)^*$, $L(F_1) \cup L(F_2)$, $L(F_1) \cdot L(F_2)$, $L(F_1) \cap L(F_2)$.
8. Input: deterministic finite automaton F ;
Output: deterministic finite automaton for $\overline{L(F)}$.
9. Input: string $w \in \Sigma^*$;
Output: finite automata for sets of strings over Σ that contain w as a substring, prefix, or suffix.
10. Input: integer $m > 1$, alphabet $\Sigma = \{a, b, \dots, z\}$ (of any size) and expression of the form:
 $n = \alpha \cdot (\#a) + \beta \cdot (\#b) + \dots + \zeta \cdot (\#z)$
where $\alpha, \beta, \dots, \zeta$ are integer constants, and
 $(\#a), (\#b), \dots, (\#z)$ represent the number of occurrences of the corresponding letter in a word.
Output: finite automaton that counts $n \bmod m$.
11. Input: non-deterministic finite automaton;
Output: equivalent deterministic finite automaton.
12. Input: regular expression;
Output: equivalent finite automaton.
13. Input: finite automaton;
Output: equivalent regular expression.
14. Input: regular context-free grammar;
Output: equivalent finite automaton.

15. Input: deterministic finite automaton;
Output: equivalent regular context-free grammar.
16. Input: finite automaton;
Output: equivalent pushdown automaton.
17. Input: deterministic finite automaton;
Output: equivalent Turing Machine;
18. Input: deterministic finite automaton F ;
Output: YES if $L(F) \neq \emptyset$; NO otherwise.
19. Input: deterministic finite automaton F ;
Output: YES if $L(F)$ is infinite; NO otherwise.
20. Input: deterministic finite automata F_1 and F_2 ;
Output: YES if $L(F_1) = L(F_2)$;
NO otherwise.
21. Input: pushdown automaton;
Output: equivalent Turing Machine.
22. Input: context-free grammar;
Output: equivalent pushdown automaton.
23. Input: pushdown automaton;
Output: equivalent context-free grammar.
24. Input: pushdown automaton;
Output: equivalent “atomic” pushdown automaton, which pops only one symbol at a time.
25. Input: pushdown automaton that accepts by final state and empty stack;
Output: equivalent pushdown automata that accept by final state only and by empty stack only.
26. Input: pushdown automaton that accepts by final state only;
Output: equivalent pushdown automaton that accepts by final state and empty stack.
27. Input: pushdown automaton that accepts by empty stack only;
Output: equivalent pushdown automaton that accepts by final state and empty stack.
28. Input: pushdown automaton P and finite automaton F ;
Output: pushdown automaton that accepts $L(P) \cap L(F)$.
29. Input: context-free grammar G ;
Output: YES if $L(G) \neq \emptyset$; NO otherwise.
30. Input: context-free grammar G ;
Output: YES if $L(G)$ is infinite; NO otherwise.
31. Input: Turing Machine that accepts by halting;
Output: equivalent Turing Machine that accepts by final state.
32. Input: Turing Machine that accepts by final state;
Output: equivalent Turing Machine that accepts by halting.
33. Input: multi-track Turing Machine;
Output: equivalent standard Turing Machine.
34. Input: Turing Machine with two-way infinite tape;
Output: equivalent standard Turing Machine.
35. Input: multi-tape Turing Machine;
Output: equivalent standard Turing Machine.
36. Input: non-deterministic Turing Machine M and input string w ;
Output: YES if M halts on w .
37. Input: Turing Machine D that decides L ;
Output: Turing Machine X that enumerates L in length-first lexicographic order.
38. Input: Turing Machine X that enumerates L in length-first lexicographic order.
Output: Turing Machine D that decides L .
39. Input: Turing Machine M that accepts L by halting;
Output: Turing Machine E that enumerates L .
40. Input: Turing Machine E that enumerates L .
Output: Turing Machine M that accepts L by halting.
41. Input: Turing Machine M_β that decides whether its input is a Turing Machine that accepts a language that has a non-trivial property β ;
Output: Turing Machine M_H that solves the (unsolvable) Halting Problem for Turing Machines.
42. Input: Turing Machines M_1 and M_2 that accept by halting languages L and \overline{L} , respectively;
Output: Turing Machine that decides L (and \overline{L}).
43. Input: Turing Machines D_1 and D_2 that decide languages L_1 and L_2 , respectively;
Output: Turing Machines that decide:
 $\overline{L_1}$, L_1^* , $L_1 \cup L_2$, $L_1 L_2$, $L_1 \cap L_2$.
44. Input: Turing Machines M_1 and M_2 ;
Output: Turing Machines that accept by halting:
 $L(M_1)^*$, $L(M_1) \cup L(M_2)$, $L(M_1) \cdot L(M_2)$,
 $L(M_1) \cap L(M_2)$.