RMIT UNIVERSITY

Computing Theory
COSC 1107/1105
Final Exercise Answers

1 Assessment details

1. Consider the grammar derivations below.

$S \Rightarrow aSbb \Rightarrow aaSbbbb \Rightarrow aa\#bbbb$
$S \Rightarrow ccBd \Rightarrow ccccBdd \Rightarrow cccc\#dd$
$S \Rightarrow ccBd \Rightarrow ccAd \Rightarrow cc1A1d \Rightarrow cc1\#1d$
$S \Rightarrow A \Rightarrow 2A2 \Rightarrow 2C2 \Rightarrow 2cCc2 \Rightarrow 2c\#c2$

(a) From the above derivations, construct rules that must exist in any context-free grammar G for which these derivations are correct.

Answer: From the first derivation, we can see the rules must include $S \rightarrow aSbb$ and $S \rightarrow \#$. From the second derivation, we can add the rules $S \rightarrow ccBd$, $B \rightarrow ccBd$ and $B \rightarrow \#$. From the third derivation, we can add the rules $B \rightarrow A$, $A \rightarrow 1A1$ and $A \rightarrow \#$. From the fourth derivation, we can add the rules $S \rightarrow A$, $A \rightarrow 2A2$, $A \rightarrow C$, $C \rightarrow cCc$ and $C \rightarrow \#$.

$S \rightarrow aSbb \mid ccBd \mid A \mid \#$
$B \rightarrow ccBd \mid A \mid \#$
$A \rightarrow 1A1 \mid 2A2 \mid C \mid \#$
$C \rightarrow cCc \mid \#$

(b) Assuming that these are all the rules in G, give $L(G)$ in set notation.

Answer: $L(G) = \{a^i c^{2j} w c^k \# c^k w^R d^j b^{2i} \mid i, j, k \geq 0, w \in \{1,2\}^*\}$

(c) Is the grammar G regular? Explain your answer.

Answer: G is not regular. The rule $S \rightarrow aSbb$ is not in the form where the right-hand side is either the empty string, a single character, or a single character followed by a single variable.

(d) Is there a deterministic finite state automaton for $L(G)$? Explain your answer.

Answer: No. $L(G)$ is not a regular language, as we will need a stack to keep track of the a's and c'as well as $w \in \{1,2\}^*$.

(e) Construct a context-free grammar for the language L_2 below.

$L_2 = \{w_1\ a^{2i}\ c^j\ a^l\ d^k\ b^i\ w_2 \mid w_1, w_2 \in \{1,2\}^*, |w_1| = |w_2|, i \geq 1, j, k, l \geq 0, j < k\}$

Answer: One such grammar is below.

$S \rightarrow 1S1 \mid 1S2 \mid 2S1 \mid 2S2 \mid A$
$A \rightarrow aaAb \mid aaBb$
$B \rightarrow cBd \mid Bd \mid Cd$
$C \rightarrow aC \mid \lambda$

(f) Explain why there is not necessarily a context-free grammar for $\overline{L_2}$.

Answer: Unlike regular languages, the complement of a context-free language is not necessarily context-free. So for any given context-free language, we cannot assume that there is necessarily a context-free grammar for its complement.

2. Legolas Brownbean, a friend of a more famous Legolas, has written the following discussion. There are 6 incorrect statements in the paragraph below. Identify all 6 incorrect statements and justify each of your answers.

Note: A single sentence counts as one statement.

"Algorithms are an expression of knowledge that makes computers work. In order to solve a particular computational problem, such as finding the factors of a given number, or sorting a list of students by their student number, an algorithm must be developed and implemented. It comes as a great surprise to many people that there are some problems for which there is no algorithm possible. These are known as undecidable problems and include the Halting problem for Turing machines, the busy beaver problem, the tile problem and the vertex cover problem. Alan Turing was the first to show there was an undecidable problem, and since that time many other problems have been shown also to be undecidable.

This is done by reducing a problem (say A) with unknown status to a problem known to be undecidable (say B).

From this reduction it is simple to show that problem A must be undecidable using the Pumping Lemma. The existence of undecidable problems means that certain properties of programs cannot be guaranteed. So anyone faced with an undecidable problem should refuse to work on it, as no instance of any such problem can ever be solved. The Church-Turing thesis, which states that anything computable can be performed by a Turing machine, indicates that we cannot avoid undecidable problems by changing technologies.

Undecidable problems are only an issue for nondeterministic Turing machines, as anything that can be executed on a deterministic Turing machine is decidable.

Unrestricted grammars also do not have any issues with undecidable problems. Note though that NP-complete and other intractable problems are all decidable. The limits on computing with such problems is practical rather than fundamental." **Answer:**

(a) The vertex cover problem is NP-complete, and is hence decidable. So it is incorrect to list this as an undecidable problem.

(b) Reductions are done from a known undecidable problem to the problem whose status is unknown. This shows that if the unknown problem is decidable, then so is the problem known to be undecidable, which is a contradiction. This statement has this the wrong way round.

(c) The Pumping lemma is a property of regular (and context-free) languages. It is not involved in any undecidability proofs.

(d) Algorithms for undecidable problems cannot be guaranteed to always terminate. But some instances of them can still be solved, so this statement is incorrect.

(e) Deterministic Turing machines are as powerful as nondeterministic ones, and so have exactly the same undecidability issues.

(f) Unrestricted grammars are equivalent to (nondeterministic) Turing machines, and so the same undecidability issues arise.

3. (a) Katie the Koala loves playing Platypus tournaments. She is particularly fond of the 4-player variety, for which a tournament of n machines will require $n(n+1)(n+2)(n+3)/24$ matches. She ran her own tournament on her own laptop for 200 machines, which took 97.65 seconds. Encouraged by this, she resolves to run the largest tournament that she can in 10 hours. Calculate the largest number of machines for which Katie can run a tournament in this amount of time.

Answer: For 200 machines, this takes 68,685,050 games and so the rate per game is 97.65/(68685050) seconds. 10 hours is $10 \times 60 \times 60 = 36,000$ seconds, which means Katie can play a tournament of up to 881 machines, as this will take $881 \times 882 \times 883 \times 884 \times 97.65/(24 \times 68685050) = 35,929$ seconds. A quick check for $n = 882$ shows that this value exceeds $36,000$ seconds. So the answer is 881.

(b) Katie is disappointed by the previous result and so turns to playing the same 4-player game but in a different tournament format suggested by her friend the Wombat. She works out that using this format the tournament takes $(1.1)^n/(1.05)^n$ matches for n machines where $n \geq 200$, using the Platypus game as in the previous question. Calculate the largest number of machines for which Katie can run a Wombat-style tournament in 10 hours.

Answer: Using the same rate of 97.65/(68685050) seconds per game, we find that Katie can play a tournament of up to 514 machines this way, as this will take $(1.1)^{514}/(1.05)^{514} \times 97.65/(68685050) = 34462.79$ seconds. A quick check for $n = 515$ shows that this value exceeds $36,000$ seconds. So the answer is 514.

(c) Never one to quit, Katie asks some friends for help. They work out that they have 10 identical computers between them, each of which can run a 4-player game in exactly the same time as Katie, and they agree to divide up the matches equally between them. Katie decides that they will attempt to collectively run a tournament for 2,000 machines in the original 4-player format. How long will each machine have to run to complete each person's share (ie 10%) of the matches? We will call this time t_1.

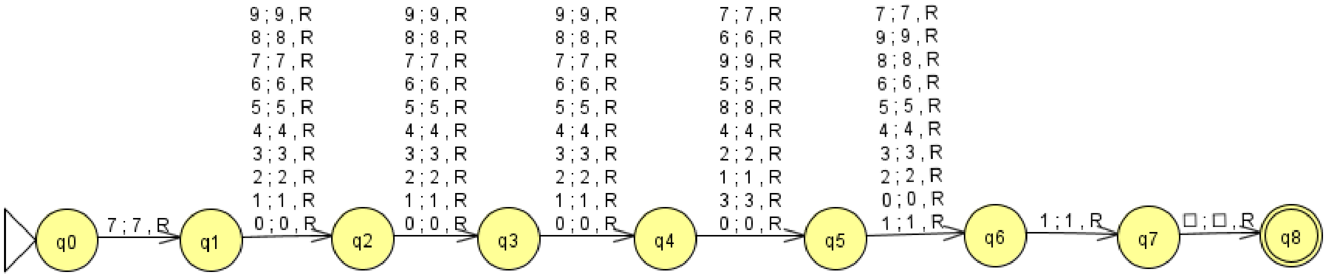
Answer: A tournament of 2,000 machines will require 668,668,500,500 matches, so each of the group will have to run 66,866,850,050 matches. At the rate of 97.65 seconds for 68,685,050 matches this means 97.65 * 66,866,850,050 /68,685,050 = 95,065 seconds = 26.4 hours.

(d) Flushed with their success, the group of friends that each one of them will dedicate the same time as calculated in the previous question for running as large a Wombat-style tournament as they can. Calculate the number of machines for which they can complete a Wombat-style tournament this way. In other words, what is the largest number of machines for which they can run a Wombat-style tournament in time t_1?

Answer: Each person will have 95,065 seconds of computation so that makes a total of 950,650 seconds. When $n = 585$ for a Wombat-style tournament, this gives $(1.1)^{585}/(1.05)^{585} \times 97.65/(68685050) = 937110.69$ seconds. A quick check for $n = 586$ shows this value exceeds 950,650 seconds. So the answer is 585.

4. (a) Student numbers at the University of Minas Tirith are strings of length 7 over the alphabet $0, 1, 2, 3, 4, 5, 6, 7, 8, 9$. The first digit of each number must be 7 and the final digit must be 1. Construct a Turing machine which recognises such student numbers. Any string of length 6 or less should be rejected, as should any string of length 8 or more.

Answer: One such machine is below.


q0
7;7,R
q1
9;9,R 8;8,R 7;7,R 6;6,R 5;5,R 4;4,R 3;3,R 2;2,R 1;1,R 0;0,R
q2
9;9,R 8;8,R 7;7,R 6;6,R 5;5,R 4;4,R 3;3,R 2;2,R 1;1,R 0;0,R
q3
9;9,R 8;8,R 7;7,R 6;6,R 5;5,R 4;4,R 3;3,R 2;2,R 1;1,R 0;0,R
q4
7;7,R 6;6,R 9;9,R 5;5,R 8;8,R 4;4,R 2;2,R 1;1,R 3;3,R 0;0,R
q5
7;7,R 9;9,R 8;8,R 6;6,R 5;5,R 4;4,R 3;3,R 2;2,R 0;0,R 1;1,R
q6
1;1,R
q7
□;□,R
q8

(b) Which of the following machines could also be used to recognise such numbers? Briefly justify each of your answers.

- A non-deterministic PDA
- A deterministic PDA
- A non-deterministic finite state automaton (NFA)
- A deterministic finite state automaton (DFA)
- A linear-bounded automaton

Answer: This language is regular, as it is finite (and the length of each string in the language is exactly 7). This means that any of the above automata could be used to recognise it.

(c) Given a Minas Tirith student number (as in the previous question), the internal string is the 5 digits of the number excluding the 7 at the start and the 1 at the end. For example, given the student number 7654321, the internal string is 65432. The Steward of the university, Denethor the Demented, announces that he is especially interested in student numbers whose internal string is either a palindrome or contains at least three 9s, which he calls Palantir numbers. Note that the string has to be a valid student number as well in order to be a Palantir number. For example, 7643461 and 7299991 are both Palantir numbers, and 7666441 and 7992281 are valid student numbers but not Palantir numbers, and 199991 and 63399332 are neither.

Which of the following machines can be used to recognise valid student numbers that are also Palantir numbers? Briefly justify each of your answers. You do not have to explicitly construct any machines in your answer.

- A deterministic Turing machine
- A non-deterministic PDA
- A non-deterministic finite state automaton (NFA)
- A deterministic finite state automaton (DFA)

Answer: This language is also regular, as it is also finite. So there is DFA for this language, and hence any of the other automata could also be used to recognise it.

(d) Some time later, Denethor changes his mind about both Minas Tirith student numbers and Palantir numbers. He decrees that from now on student numbers can be of any odd length, provided the first digit is 7 and the final digit is 1. He also decrees that Palantir numbers must be palindromes only. For example, 711 and 798654321 are now both valid student numbers, but 7992281 is no longer a Palantir number, whereas 71234543211 is a Palantir number.

Given these changes, which of the following machines can be used to recognise valid student numbers that are also Palantir numbers? Briefly justify each of your answers. You do not have to explicitly construct any machines in your answer.

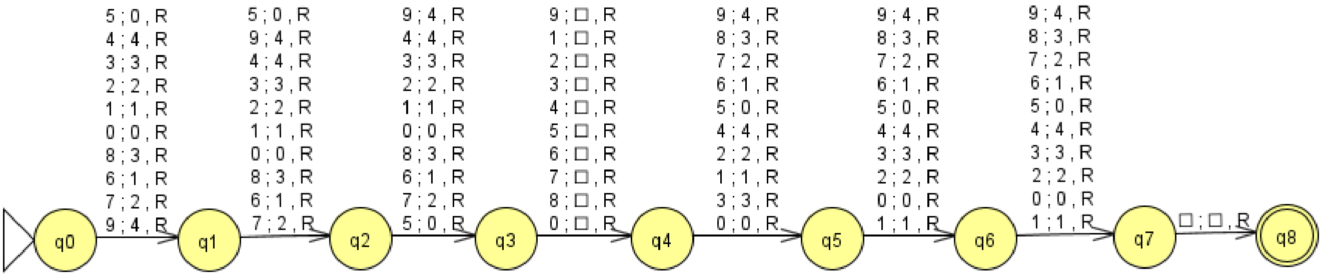
- A deterministic Turing machine
- A non-deterministic PDA
- A non-deterministic finite state automaton (NFA)
- A deterministic finite state automaton (DFA)

Answer: As student numbers can now be arbitrarily long, the language of such numbers is infinite. In addition, there requirement for palindromes means that a stack must be used to keep track of the first part of the palindrome and check it against the second half. Hence neither NFAs nor DFAs can be used. However, non-deterministic PDAs and any type of Turing machine could be used.

5. (a) Construct a Turing machine M_1 which given as input a string of length 7 over $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ performs as follows.

- The fourth digit, whatever it is, is replaced with a blank.
- Any of the digits 5,6,7,8 or 9 is replaced with 0,1,2,3 or 4 respectively.
- The machine must terminate on all strings over $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}^*$ (including the empty string).
- The machine only halts in an accepting state if the input string has exactly 7 digits. Strings of any other length are rejected.

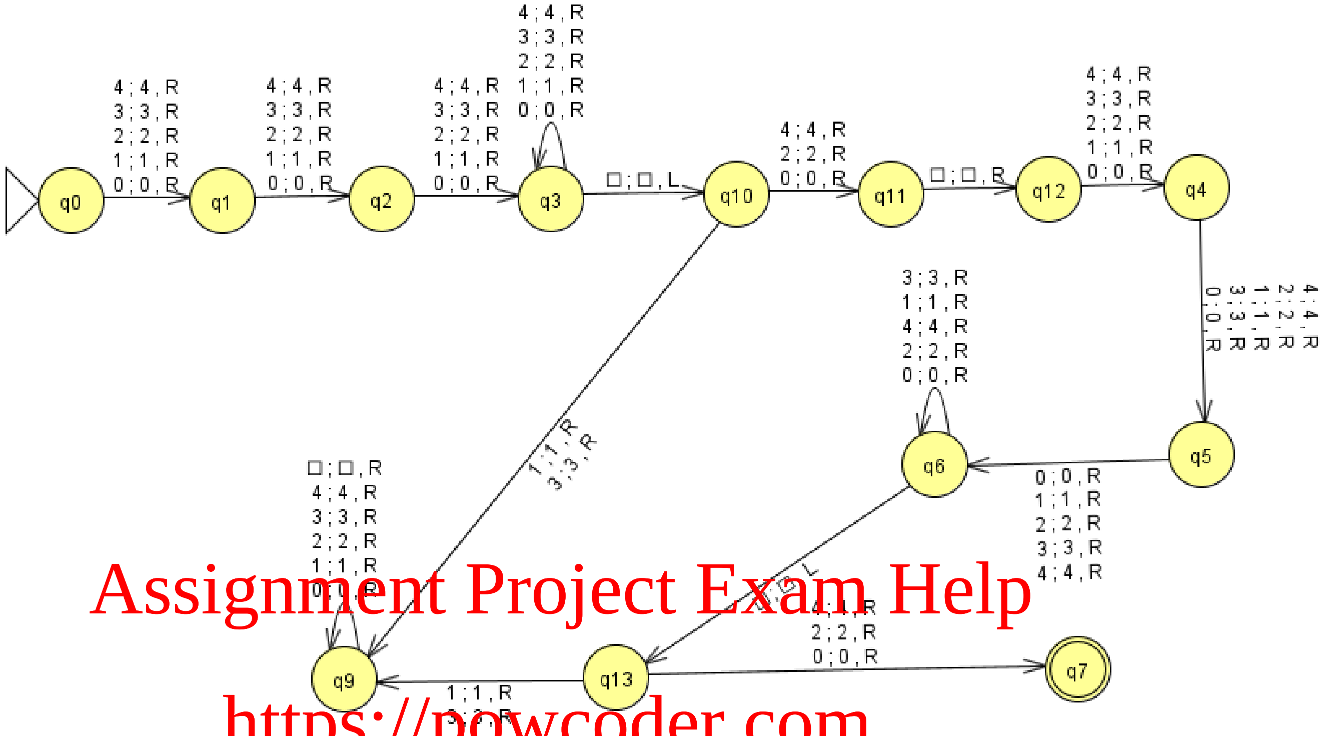
Answer: One such machine is below.



(b) Construct a Turing machine M_2 which given as input two string of digits over $\{0, 1, 2, 3, 4\}$ separated by a blank performs as follows.

- If either string has length less than three, the machine halts with rejection.
- If the final digit of both strings is 0, 2 or 4, then machine halts with acceptance.
- If the final digit of either string is 1 or 3, the machine loops forever (i.e. it does not terminate).

Answer: One such machine is below.



(c) Consider the machine formed by combining M_1 and M_2 as above in sequence into a new machine M_3, so that M_3 first performs as M_1 does, rewinds the tape head to the appropriate point on the tape and then performs as M_2 does. Clearly the behaviour of M_3 will depend on the behaviour of M_1 and M_2.

- For which inputs does M_3 halt with rejection?
- For which inputs does M_3 halt with acceptance?
- For which inputs does M_3 not halt?

In each case, justify your answer.

Answer: M_1 takes an input of any length over the alphabet $\{0,1,2,3,4,5,6,7,8,9\}$, and rejects any string which is not of length 7. Otherwise M_1 replaces the string with two strings of length 3 separated by a blank, where each string is over the alphabet $\{0,1,2,3,4\}$. So M_1 terminates on any input. M_2 takes two strings of length at least 3 separated by a blank and will terminate if both of the strings input to it end in 0,2, or 4, and otherwise will loop forever. Combining these properties into M_3 as above means that M_3 with halt with rejection for any string over the alphabet $\{0,1,2,3,4,5,6,7,8,9\}$ which is not of length 7. For strings of length 7, M_3 will halt with acceptance if both the third and the seventh character of the string is one of 0,2,4,5,7 or 9. For all other strings of length 7, M_3 will not halt.

(d) An evil wizard now changes M_1 to another machine M_0. This is the same as M_1 except that rather than deterministically replacing 5, 6, 7, 8 or 9 with 0, 1, 2, 3 or 4, M_0 *non-deterministically* replaces each of these digits with one of 0, 1, 2, 3, or 4. For example, if there is a transition such as $\delta(q_i, 5) = (q_j, 0, R)$ in M_1, there are 5 transitions in M_0, i.e. $\delta(q_i, 5) = \{(q_j, 0, R), (q_j, 1, R), (q_j, 2, R), (q_j, 3, R), (q_j, 4, R), \}$ in M_0.

The same combination technique as above is used to combine M_0 and M_2 into M_4.

- For which inputs does M_4 halt with rejection?
- For which inputs does M_4 halt with acceptance?
- For which inputs does M_4 not halt?

In each case, justify your answer.

Answer: M_4, just like M_3, will reject any strings which are not of length 7. For all strings to M_4 over the alphabet $\{0, 1, 2, 3, 4\}$, M_4 will behave in the same way as M_3. For each character in $\{5, 6, 7, 8, 9\}$ there is now a trace in M_4 which (due to the way M_0 is defined) will replace this character with 0. So M_4 will halt with rejection for any string over the alphabet $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ which is not of length 7. For strings of length 7, M_4 will halt with acceptance if both the third and the seventh character of the string is one of 0,2,4,5,6,7,8 or 9. For all other strings of length 7, M_4 will not halt (ie M_4 will not halt if the third or the seventh character of the input string is 1 or 3).

6. (a) Prove that the language $L_1 = \{a^i b^j c^j d^i | i, j \geq 0\}$ is not regular by using the Pumping Lemma. Use the string $a^n b^n c^n d^n$ at an appropriate point in the proof.

Answer:

Assume that L_1 is regular. Then the Pumping Lemma applies, and so there is an $n \geq 1$ such that for any $w \in L_1$ with $|w| \geq n$, there exist strings x, y and z such that $w = xyz$, $|xy| \leq n$, $y \neq \lambda$ and $xy^i z \in L_1$ for all $i \geq 0$.

Let $w = a^n b^n c^n d^n$, and so $w \in L_1$ and $|w| > n$. So $xyz = a^n b^n c^n d^n$ and as $|xy| \leq n$, we must have that $y = a^j$ for some $1 \leq j \leq n$.

Consider $xy^2 z$. As $y = a^j$, we have $xy^2 z = a^{n+j} b^n c^n d^n$ and so $xy^2 z \notin L_1$. This is a contradiction, and so we conclude that L_1 is not regular. **QED**

(b) Write out the proof of the same result which uses the string $b^{2n} c^{2n}$ and $i = 4$ instead. Which steps are different? Which steps remain the same?

Answer: The proof is below. The differences are highlighted with a box $\boxed{\text{like}}$ this.

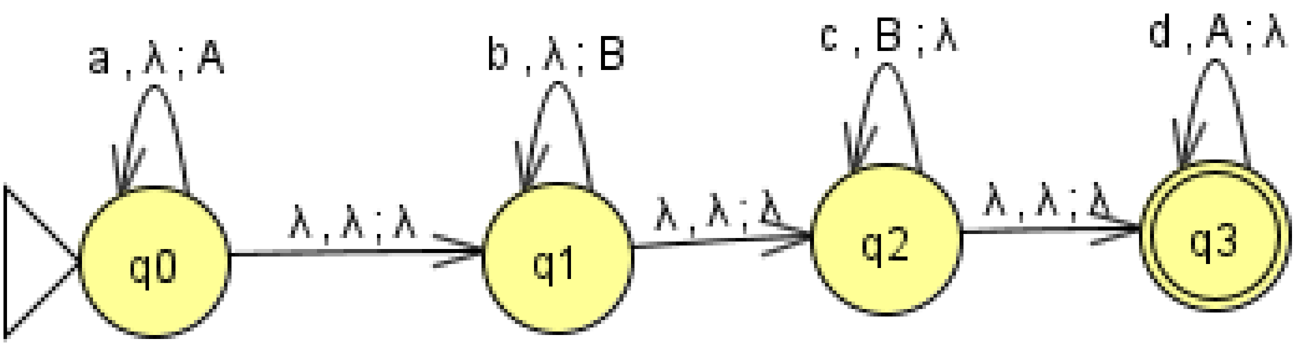
Assume that L_1 is regular. Then the Pumping Lemma applies, and so there is an $n \geq 1$ such that for any $w \in L_1$ with $|w| \geq n$, there exist strings x, y and z such that $w = xyz$, $|xy| \leq n$, $y \neq \lambda$ and $xy^i z \in L_1$ for all $i \geq 0$.

Let $\boxed{w = b^{2n} c^{2n}}$, and so $w \in L_1$ and $|w| > n$. So $\boxed{xyz = b^{2n} c^{2n}}$ and as $|xy| \leq n$, we must have that $\boxed{y = b^j}$ for some $1 \leq j \leq n$.

Consider $\boxed{xy^4 z}$. As $\boxed{y = b^j}$, we have $\boxed{xy^4 z = b^{2n+3j} c^{2n}}$ and so $\boxed{xy^4 z \notin L_1}$. This is a contradiction, and so we conclude that L_1 is not regular. **QED**

(c) Give a PDA for the language $L_1 = \{a^i b^j c^j d^i | i, j \geq 0\}$. your machine deterministic or nondeterministic? Briefly explain your answer.

Answer: One such PDA is below. This machine is nondeterministic, as it contains λ transitions.

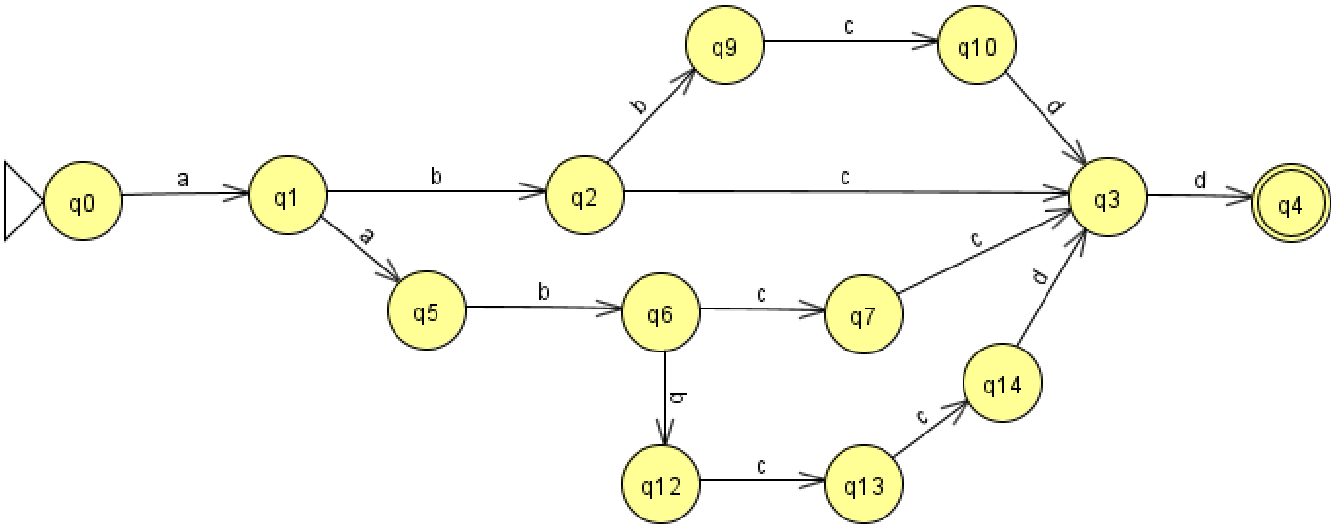


(d) Is there a context-free grammar for the language $L_2 = \{a^i b^j c^i d^j | i, j \geq 1\}$? Explain your answer.

Answer: No. This language is not context-free, and so there is no context-free grammar for it. The reason is that we need to record both the number of a's and then the number of b's, but then check the number of c's against the number of a's, which means discarding the number of b's. So there can be no PDA for this language, and hence no context-free grammar.

(e) The language $L_3 = \{a^i b^j c^i d^j | 1 \leq i, j \leq 2\}$ is regular. Construct a finite state automaton which accepts L_3 (and only L_3). Your machine can be either deterministic or non-deterministic.

Answer: Note that $L_3 = \{abcd, aabccd, abbcdd, aabbccdd\}$. One such automaton is below.



(f) The language $L_4 = \{a^i b^j c^i d^j | 1 \leq i, j \leq 6\}$ is also regular. If you were to extend your previous machine to accept this language, how many transitions would you need? In other words, if there are k transitions in your machine for L_3, give a formula for your estimated number of transitions required in the machine for L_4.

Answer: The machine above has 15 transitions for a language with 4 strings. L_4 includes all of these, so ...

(g) Give a context-free grammar for $L_5 = \{a^i b^j c^j d^i \mid i, j \geq 1\}$ with at most 4 rules. The number of rules is the number of different right-hand sides in the grammar; for example, the grammar below has 6 rules.

$S \rightarrow aA|Aa|a$
$A \rightarrow aBaa|Baa$
$B \rightarrow abc$

Answer:

$S \rightarrow aSd \mid aAd$
$A \rightarrow bAc \mid bc$

7. Snivelling Sam the Shonky Snake Oil Salesman has heard about your work on Turing machines. This is of great interest to Sam, who wishes to set up a Gallery of Important Turing Machines which are Useful and Brilliant (GITMUB). This will showcase various Turing machines, which must be of the acceptor/rejector type (i.e. Turing machines which accept a given language by terminating in an accepting state). Sam is keen to ensure that he only has the "best and brightest" Turing machines on display from those submitted to GITMUB by the general public, and so insists on the following policy on adding machines. Let the machine to be added to GITMUB be M, and the machines already in GITMUB be $M_1, \ldots, M_n$. If $L(M) \neq L(M_i)$ for all $1 \leq i \leq n$, then M is added to GITMUB. Otherwise, (i.e. $L(M) = L(M_i)$ for some $1 \leq i \leq n$) then Sam makes a choice between M and M_i, and either rejects M as a member of GITMUB, or ejects M_i from GITMUB and replaces it with M (according to what he perceives as the 'inherent artistic merit' of each machine). This means that Sam can boast that GITMUB's machines are all guaranteed to be unique representatives of machines accepting a particular language.

(a) Explain why, under the above policy, GITMUB can never be guaranteed to contain more than one machine.

Answer: The problem of determining whether two Turing machines accept the same language is undecidable. So while the first machine that Sam puts in his collection will trivially accept a distinct language from the other machines already in the gallery. When it comes to adding a second machine, there is no way to guarantee that the language of the second machine is different from that of the first. So if Sam insists on this guarantee, there can never be more than one machine in GITMUB.

(b) Suggest one way that Sam could relax his policy on adding machines to the gallery in order to have a larger collection of machines in GITMUB while remaining as true as possible to his original aim.

Answer: One way to do this is to relax the guarantee from the languages being definitely different to being different as far as can be distinguished in a bounded amount of time. This might include testing each machine with a finite set of strings (with the time taken for each machine bounded), and if no distinction can be found in this manner, then the machines are treated as if they accept the same language, as above.

(c) Sam is also interested in the history of Turing machines, and runs a competition to find the Greatest Ingenious Turing Machine of All Time (the GITMOAT). Sam asks you for your choice of a Turing machine to enter into this competition. What machine would be your choice? Justify your answer.

Answer: Candidates include

- Turing's original universal machine
- Minsky's machine
- Colmerauer's machine
- The minimal one from the Wolfram competition

2 Submission