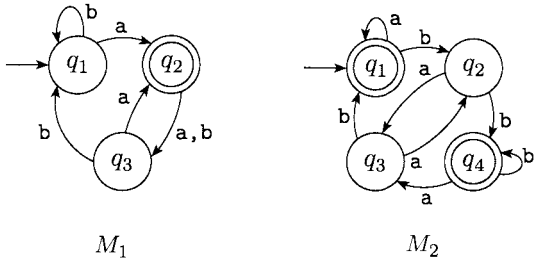
## EXERCISES

[A]**1.1** The following are the state diagrams of two DFAs, $M_1$ and $M_2$. Answer the following questions about each of these machines.




$M_1$ $M_2$

**a.** What is the start state?

**b.** What is the set of accept states?

**c.** What sequence of states does the machine go through on input aabb?

**d.** Does the machine accept the string aabb?

**e.** Does the machine accept the string $\varepsilon$?

[A]**1.2** Give the formal description of the machines $M_1$ and $M_2$ pictured in Exercise 1.1.

**1.3** The formal description of a DFA $M$ is $(\{q_1, q_2, q_3, q_4, q_5\}, \{\mathtt{u}, \mathtt{d}\}, \delta, q_3, \{q_3\})$, where $\delta$ is given by the following table. Give the state diagram of this machine.

| | u | d |
|---|---|---|
| $q_1$ | $q_1$ | $q_2$ |
| $q_2$ | $q_1$ | $q_3$ |
| $q_3$ | $q_2$ | $q_4$ |
| $q_4$ | $q_3$ | $q_5$ |
| $q_5$ | $q_4$ | $q_5$ |

**1.4** Each of the following languages is the intersection of two simpler languages. In each part, construct DFAs for the simpler languages, then combine them using the construction discussed in footnote 3 (page 46) to give the state diagram of a DFA for the language given. In all parts $\Sigma = \{\mathtt{a}, \mathtt{b}\}$.

**a.** $\{w \mid w$ has at least three a's and at least two b's$\}$

[A]**b.** $\{w \mid w$ has at exactly two a's and at least two b's$\}$

**c.** $\{w \mid w$ has an even number of a's and one or two b's$\}$

[A]**d.** $\{w \mid w$ has an even number of a's and each a is followed by at least one b$\}$

**e.** $\{w \mid w$ starts with an a and has at most one b$\}$

**f.** $\{w \mid w$ has an odd number of a's and ends with a b$\}$

**g.** $\{w \mid w$ has even length and an odd number of a's$\}$

**1.5** Each of the following languages is the complement of a simpler language. In each part, construct a DFA for the simpler language, then use it to give the state diagram of a DFA for the language given. In all parts $\Sigma = \{\mathtt{a}, \mathtt{b}\}$.

- **[A]a.** $\{w \mid w$ does not contain the substring $\mathtt{ab}\}$
- **[A]b.** $\{w \mid w$ does not contain the substring $\mathtt{baba}\}$
- **c.** $\{w \mid w$ contains neither the substrings $\mathtt{ab}$ nor $\mathtt{ba}\}$
- **d.** $\{w \mid w$ is any string not in $\mathtt{a}^*\mathtt{b}^*\}$
- **e.** $\{w \mid w$ is any string not in $(\mathtt{ab}^+)^*\}$
- **f.** $\{w \mid w$ is any string not in $\mathtt{a}^* \cup \mathtt{b}^*\}$
- **g.** $\{w \mid w$ is any string that doesn't contain exactly two $\mathtt{a}$'s$\}$
- **h.** $\{w \mid w$ is any string except $\mathtt{a}$ and $\mathtt{b}\}$

**1.6** Give state diagrams of DFAs recognizing the following languages. In all parts the alphabet is $\{\mathtt{0},\mathtt{1}\}$

- **a.** $\{w \mid w$ begins with a $\mathtt{1}$ and ends with a $\mathtt{0}\}$
- **b.** $\{w \mid w$ contains at least three $\mathtt{1}$s$\}$
- **c.** $\{w \mid w$ contains the substring $\mathtt{0101}$, i.e., $w = x\mathtt{0101}y$ for some $x$ and $y\}$
- **d.** $\{w \mid w$ has length at least 3 and its third symbol is a $\mathtt{0}\}$
- **e.** $\{w \mid w$ starts with $\mathtt{0}$ and has odd length, or starts with $\mathtt{1}$ and has even length$\}$
- **f.** $\{w \mid w$ doesn't contain the substring $\mathtt{110}\}$
- **g.** $\{w \mid$ the length of $w$ is at most $5\}$
- **h.** $\{w \mid w$ is any string except $\mathtt{11}$ and $\mathtt{111}\}$
- **i.** $\{w \mid$ every odd position of $w$ is a $\mathtt{1}\}$
- **j.** $\{w \mid w$ contains at least two $\mathtt{0}$s and at most one $\mathtt{1}\}$
- **k.** $\{\varepsilon, \mathtt{0}\}$
- **l.** $\{w \mid w$ contains an even number of $\mathtt{0}$s, or contains exactly two $\mathtt{1}$s$\}$
- **m.** The empty set
- **n.** All strings except the empty string

**1.7** Give state diagrams of NFAs with the specified number of states recognizing each of the following languages. In all parts the alphabet is $\{\mathtt{0},\mathtt{1}\}$.

- **[A]a.** The language $\{w \mid w$ ends with $\mathtt{00}\}$ with three states
- **b.** The language of Exercise 1.6c with five states
- **c.** The language of Exercise 1.6l with six states
- **d.** The language $\{\mathtt{0}\}$ with two states
- **e.** The language $\mathtt{0}^*\mathtt{1}^*\mathtt{0}^+$ with three states
- **[A]f.** The language $\mathtt{1}^*(\mathtt{001}^+)^*$ with three states
- **g.** The language $\{\varepsilon\}$ with one state
- **h.** The language $\mathtt{0}^*$ with one state

**1.8** Use the construction given in the proof of Theorem 1.45 to give the state diagrams of NFAs recognizing the union of the languages described in

- **a.** Exercises 1.6a and 1.6b.
- **b.** Exercises 1.6c and 1.6f.

**1.9** Use the construction given in the proof of Theorem 1.47 to give the state diagrams of NFAs recognizing the concatenation of the languages described in

**a.** Exercises 1.6g and 1.6i.

**b.** Exercises 1.6b and 1.6m.

**1.10** Use the construction given in the proof of Theorem 1.49 to give the state diagrams of NFAs recognizing the star of the language described in

**a.** Exercise 1.6b.

**b.** Exercise 1.6j.

**c.** Exercise 1.6m.

[A]**1.11** Prove that every NFA can be converted to an equivalent one that has a single accept state.

**1.12** Let $D = \{w \mid w$ contains an even number of a's and an odd number of b's and does not contain the substring ab$\}$. Give a DFA with five states that recognizes $D$ and a regular expression that generates $D$. (Suggestion: Describe $D$ more simply.)

**1.13** Let $F$ be the language of all strings over $\{0,1\}$ that do not contain a pair of 1s that are separated by an odd number of symbols. Give the state diagram of a DFA with 5 states that recognizes $F$. (You may find it helpful first to find a 4-state NFA for the complement of $F$.)

**1.14** **a.** Show that, if $M$ is a DFA that recognizes language $B$, swapping the accept and nonaccept states in $M$ yields a new DFA that recognizes the complement of $B$. Conclude that the class of regular languages is closed under complement.

**b.** Show by giving an example that, if $M$ is an NFA that recognizes language $C$, swapping the accept and nonaccept states in $M$ doesn't necessarily yield a new NFA that recognizes the complement of $C$. Is the class of languages recognized by NFAs closed under complement? Explain your answer.

**1.15** Give a counterexample to show that the following construction fails to prove Theorem 1.49, the closure of the class of regular languages under the star operation.[8] Let $N_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ recognize $A_1$. Construct $N = (Q_1, \Sigma, \delta, q_1, F)$ as follows. $N$ is supposed to recognize $A_1^*$.

**a.** The states of $N$ are the states of $N_1$.

**b.** The start state of $N$ is the same as the start state of $N_1$.

**c.** $F = \{q_1\} \cup F_1$.
The accept states $F$ are the old accept states plus its start state.

**d.** Define $\delta$ so that for any $q \in Q$ and any $a \in \Sigma_\varepsilon$,

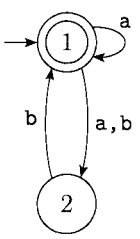
$$\delta(q,a) = \begin{cases} \delta_1(q,a) & q \notin F_1 \text{ or } a \neq \varepsilon \\ \delta_1(q,a) \cup \{q_1\} & q \in F_1 \text{ and } a = \varepsilon. \end{cases}$$

(Suggestion: Show this construction graphically, as in Figure 1.50.)

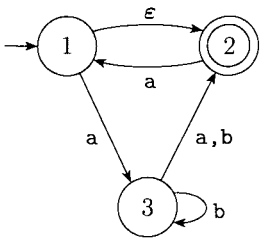
---

[8] In other words, you must present a finite automaton, $N_1$, for which the constructed automaton $N$ does not recognize the star of $N_1$'s language.

**1.16** Use the construction given in Theorem 1.39 to convert the following two nondeterministic finite automata to equivalent deterministic finite automata.




(a)




(b)

**1.17**

**a.** Give an NFA recognizing the language $(01 \cup 001 \cup 010)^*$.

**b.** Convert this NFA to an equivalent DFA. Give only the portion of the DFA that is reachable from the start state.

**1.18** Give regular expressions generating the languages of Exercise 1.6.

**1.19** Use the procedure described in Lemma 1.55 to convert the following regular expressions to nondeterministic finite automata.

**a.** $(0 \cup 1)^*000(0 \cup 1)^*$

**b.** $(((00)^*(11)) \cup 01)^*$

**c.** $\emptyset^*$

**1.20** For each of the following languages, give two strings that are members and two strings that are *not* members—a total of four strings for each part. Assume the alphabet $\Sigma = \{a,b\}$ in all parts.

**a.** $a^*b^*$

**b.** $a(ba)^*b$

**c.** $a^* \cup b^*$

**d.** $(aaa)^*$

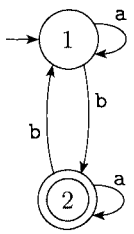
**e.** $\Sigma^*a\Sigma^*b\Sigma^*a\Sigma^*$

**f.** $aba \cup bab$

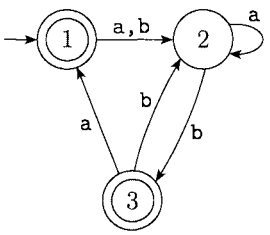
**g.** $(\varepsilon \cup a)b$

**h.** $(a \cup ba \cup bb)\Sigma^*$

**1.21** Use the procedure described in Lemma 1.60 to convert the following finite automata to regular expressions.




(a)




(b)

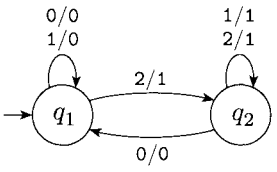
**1.22** In certain programming languages, comments appear between delimiters such as `/#` and `#/`. Let $C$ be the language of all valid delimited comment strings. A member of $C$ must begin with `/#` and end with `#/` but have no intervening `#/`. For simplicity, we'll say that the comments themselves are written with only the symbols `a` and `b`; hence the alphabet of $C$ is $\Sigma = \{\texttt{a}, \texttt{b}, \texttt{/}, \texttt{\#}\}$.

**a.** Give a DFA that recognizes $C$.

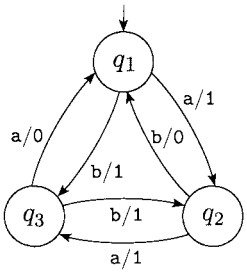
**b.** Give a regular expression that generates $C$.

**A1.23** Let $B$ be any language over the alphabet $\Sigma$. Prove that $B = B^+$ iff $BB \subseteq B$.

**1.24** A ***finite state transducer*** (FST) is a type of deterministic finite automaton whose output is a string and not just *accept* or *reject*. The following are state diagrams of finite state transducers $T_1$ and $T_2$.




$T_1$




$T_2$

Each transition of an FST is labeled with two symbols, one designating the input symbol for that transition and the other designating the output symbol. The two symbols are written with a slash, /, separating them. In $T_1$, the transition from $q_1$ to $q_2$ has input symbol `2` and output symbol `1`. Some transitions may have multiple input–output pairs, such as the transition in $T_1$ from $q_1$ to itself. When an FST computes on an input string $w$, it takes the input symbols $w_1 \cdots w_n$ one by one and, starting at the start state, follows the transitions by matching the input labels with the sequence of symbols $w_1 \cdots w_n = w$. Every time it goes along a transition, it outputs the corresponding output symbol. For example, on input `2212011`, machine $T_1$ enters the sequence of states $q_1, q_2, q_2, q_2, q_2, q_1, q_1, q_1$ and produces output `1111000`. On input `abbb`, $T_2$ outputs `1011`. Give the sequence of states entered and the output produced in each of the following parts.

**a.** $T_1$ on input `011`

**b.** $T_1$ on input `211`

**c.** $T_1$ on input `121`

**d.** $T_1$ on input `0202`

**e.** $T_2$ on input `b`

**f.** $T_2$ on input `bbab`

**g.** $T_2$ on input `bbbbbb`

**h.** $T_2$ on input $\varepsilon$

**1.25** Read the informal definition of the finite state transducer given in Exercise 1.24. Give a formal definition of this model, following the pattern in Definition 1.5 (page 35). Assume that an FST has an input alphabet $\Sigma$ and an output alphabet $\Gamma$ but not a set of accept states. Include a formal definition of the computation of an FST. (Hint: An FST is a 5-tuple. Its transition function is of the form $\delta\colon Q \times \Sigma \longrightarrow Q \times \Gamma$.)

**1.26** Using the solution you gave to Exercise 1.25, give a formal description of the machines $T_1$ and $T_2$ depicted in Exercise 1.24.

**1.27** Read the informal definition of the finite state transducer given in Exercise 1.24. Give the state diagram of an FST with the following behavior. Its input and output alphabets are $\{0,1\}$. Its output string is identical to the input string on the even positions but inverted on the odd positions. For example, on input 0000111 it should output 1010010.

**1.28** Convert the following regular expressions to NFAs using the procedure given in Theorem 1.54. In all parts $\Sigma = \{\mathtt{a}, \mathtt{b}\}$.

**a.** $\mathtt{a}(\mathtt{abb})^* \cup \mathtt{b}$

**b.** $\mathtt{a}^+ \cup (\mathtt{ab})^+$

**c.** $(\mathtt{a} \cup \mathtt{b}^+)\mathtt{a}^+\mathtt{b}^+$

**1.29** Use the pumping lemma to show that the following languages are not regular.

**A a.** $A_1 = \{0^n 1^n 2^n \mid n \geq 0\}$

**b.** $A_2 = \{www \mid w \in \{\mathtt{a}, \mathtt{b}\}^*\}$

**A c.** $A_3 = \{\mathtt{a}^{2^n} \mid n \geq 0\}$ (Here, $\mathtt{a}^{2^n}$ means a string of $2^n$ a's.)

**1.30** Describe the error in the following "proof" that $0^*1^*$ is not a regular language. (An error must exist because $0^*1^*$ *is* regular.) The proof is by contradiction. Assume that $0^*1^*$ is regular. Let $p$ be the pumping length for $0^*1^*$ given by the pumping lemma. Choose $s$ to be the string $0^p1^p$. You know that $s$ is a member of $0^*1^*$, but Example 1.73 shows that $s$ cannot be pumped. Thus you have a contradiction. So $0^*1^*$ is not regular.

## PROBLEMS

**1.31** For any string $w = w_1 w_2 \cdots w_n$, the ***reverse*** of $w$, written $w^{\mathcal{R}}$, is the string $w$ in reverse order, $w_n \cdots w_2 w_1$. For any language $A$, let $A^{\mathcal{R}} = \{w^{\mathcal{R}} \mid w \in A\}$. Show that if $A$ is regular, so is $A^{\mathcal{R}}$.

**1.32** Let

$$\Sigma_3 = \left\{ \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \cdots, \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \right\}.$$

$\Sigma_3$ contains all size 3 columns of 0s and 1s. A string of symbols in $\Sigma_3$ gives three rows of 0s and 1s. Consider each row to be a binary number and let

$$B = \{w \in \Sigma_3^* \mid \text{the bottom row of } w \text{ is the sum of the top two rows}\}.$$

For example,

$$\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} \in B, \qquad \text{but} \qquad \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \notin B.$$

Show that $B$ is regular. (Hint: Working with $B^{\mathcal{R}}$ is easier. You may assume the result claimed in Problem 1.31.)

**1.33** Let

$$\Sigma_2 = \left\{ \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right\}.$$

Here, $\Sigma_2$ contains all columns of 0s and 1s of height two. A string of symbols in $\Sigma_2$ gives two rows of 0s and 1s. Consider each row to be a binary number and let

$$C = \{w \in \Sigma_2^* | \text{ the bottom row of } w \text{ is three times the top row}\}.$$

For example, $\begin{bmatrix} 0 \\ 0 \end{bmatrix}\begin{bmatrix} 0 \\ 1 \end{bmatrix}\begin{bmatrix} 1 \\ 1 \end{bmatrix}\begin{bmatrix} 0 \\ 0 \end{bmatrix} \in C$, but $\begin{bmatrix} 0 \\ 1 \end{bmatrix}\begin{bmatrix} 0 \\ 1 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \end{bmatrix} \notin C$. Show that $C$ is regular. (You may assume the result claimed in Problem 1.31.)

**1.34** Let $\Sigma_2$ be the same as in Problem 1.33. Consider each row to be a binary number and let

$$D = \{w \in \Sigma_2^* | \text{ the top row of } w \text{ is a larger number than is the bottom row}\}.$$

For example, $\begin{bmatrix} 0 \\ 0 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \end{bmatrix}\begin{bmatrix} 1 \\ 1 \end{bmatrix}\begin{bmatrix} 0 \\ 0 \end{bmatrix} \in D$, but $\begin{bmatrix} 0 \\ 0 \end{bmatrix}\begin{bmatrix} 0 \\ 1 \end{bmatrix}\begin{bmatrix} 1 \\ 1 \end{bmatrix}\begin{bmatrix} 0 \\ 0 \end{bmatrix} \notin D$. Show that $D$ is regular.

**1.35** Let $\Sigma_2$ be the same as in Problem 1.33. Consider the top and bottom rows to be strings of 0s and 1s and let

$$E = \{w \in \Sigma_2^* | \text{ the bottom row of } w \text{ is the reverse of the top row of } w\}.$$

Show that $E$ is not regular.

**1.36** Let $B_n = \{\mathtt{a}^k | \text{ where } k \text{ is a multiple of } n\}$. Show that for each $n \geq 1$, the language $B_n$ is regular.

**1.37** Let $C_n = \{x | \ x \text{ is a binary number that is a multiple of } n\}$. Show that for each $n \geq 1$, the language $C_n$ is regular.

**1.38** An ***all***-NFA $M$ is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ that accepts $x \in \Sigma^*$ if *every* possible state that $M$ could be in after reading input $x$ is a state from $F$. Note, in contrast, that an ordinary NFA accepts a string if *some* state among these possible states is an accept state. Prove that all-NFAs recognize the class of regular languages.

**1.39** The construction in Theorem 1.54 shows that every GNFA is equivalent to a GNFA with only two states. We can show that an opposite phenomenon occurs for DFAs. Prove that for every $k > 1$ a language $A_k \subseteq \{0,1\}^*$ exists that is recognized by a DFA with $k$ states but not by one with only $k - 1$ states.

**1.40** Say that string $x$ is a ***prefix*** of string $y$ if a string $z$ exists where $xz = y$ and that $x$ is a ***proper prefix*** of $y$ if in addition $x \neq y$. In each of the following parts we define an operation on a language $A$. Show that the class of regular languages is closed under that operation.

**[A]a.** $NOPREFIX(A) = \{w \in A | \text{ no proper prefix of } w \text{ is a member of } A\}$.

**b.** $NOEXTEND(A) = \{w \in A | \ w \text{ is not the proper prefix of any string in } A\}$.

**1.41** For languages $A$ and $B$, let the ***perfect shuffle*** of $A$ and $B$ be the language

$$\{w | \ w = a_1 b_1 \cdots a_k b_k, \text{ where } a_1 \cdots a_k \in A \text{ and } b_1 \cdots b_k \in B, \text{ each } a_i, b_i \in \Sigma\}.$$

Show that the class of regular languages is closed under perfect shuffle.

**1.42** For languages $A$ and $B$, let the ***shuffle*** of $A$ and $B$ be the language

$$\{w | \ w = a_1 b_1 \cdots a_k b_k, \text{ where } a_1 \cdots a_k \in A \text{ and } b_1 \cdots b_k \in B, \text{ each } a_i, b_i \in \Sigma^*\}.$$

Show that the class of regular languages is closed under shuffle.

**1.43** Let $A$ be any language. Define $DROP\text{-}OUT(A)$ to be the language containing all strings that can be obtained by removing one symbol from a string in $A$. Thus, $DROP\text{-}OUT(A) = \{xz \mid xyz \in A \text{ where } x, z \in \Sigma^*, y \in \Sigma\}$. Show that the class of regular languages is closed under the DROP-OUT operation. Give both a proof by picture and a more formal proof by construction as in Theorem 1.47.

A**1.44** Let $B$ and $C$ be languages over $\Sigma = \{0, 1\}$. Define

$$B \overset{1}{\leftarrow} C = \{w \in B \mid \text{for some } y \in C, \text{ strings } w \text{ and } y \text{ contain equal numbers of 1s}\}.$$

Show that the class of regular languages is closed under the $\overset{1}{\leftarrow}$ operation.

*1.45 Let $A/B = \{w \mid wx \in A \text{ for some } x \in B\}$. Show that if $A$ is regular and $B$ is any language then $A/B$ is regular.

**1.46** Prove that the following languages are not regular. You may use the pumping lemma and the closure of the class of regular languages under union, intersection, and complement.

**a.** $\{0^n 1^m 0^n \mid m, n \geq 0\}$

A**b.** $\{0^m 1^n \mid m \neq n\}$

**c.** $\{w \mid w \in \{0,1\}^* \text{ is not a palindrome}\}$[9]

**d.** $\{wtw \mid w, t \in \{0,1\}^+\}$

**1.47** Let $\Sigma = \{1, \#\}$ and let

$$Y = \{w \mid w = x_1 \# x_2 \# \cdots \# x_k \text{ for } k \geq 0, \text{ each } x_i \in 1^*, \text{ and } x_i \neq x_j \text{ for } i \neq j\}.$$

Prove that $Y$ is not regular.

**1.48** Let $\Sigma = \{0,1\}$ and let

$D = \{w \mid w \text{ contains an equal number of occurrences of the substrings } 01 \text{ and } 10\}$.

Thus $101 \in D$ because 101 contains a single 01 and a single 10, but $1010 \notin D$ because 1010 contains two 10s and one 01. Show that $D$ is a regular language.

**1.49**

**a.** Let $B = \{1^k y \mid y \in \{0,1\}^* \text{ and } y \text{ contains at least } k \text{ 1s, for } k \geq 1\}$. Show that $B$ is a regular language.

**b.** Let $C = \{1^k y \mid y \in \{0,1\}^* \text{ and } y \text{ contains at most } k \text{ 1s, for } k \geq 1\}$. Show that $C$ isn't a regular language.

A**1.50** Read the informal definition of the finite state transducer given in Exercise 1.24. Prove that no FST can output $w^{\mathcal{R}}$ for every input $w$ if the input and output alphabets are $\{0,1\}$.

**1.51** Let $x$ and $y$ be strings and let $L$ be any language. We say that $x$ and $y$ are ***distinguishable by L*** if some string $z$ exists whereby exactly one of the strings $xz$ and $yz$ is a member of $L$; otherwise, for every string $z$, we have $xz \in L$ whenever $yz \in L$ and we say that $x$ and $y$ are ***indistinguishable by L***. If $x$ and $y$ are indistinguishable by $L$ we write $x \equiv_L y$. Show that $\equiv_L$ is an equivalence relation.

---

[9] A ***palindrome*** is a string that reads the same forward and backward.

A* **1.52** **Myhill–Nerode theorem.** Refer to Problem 1.51. Let $L$ be a language and let $X$ be a set of strings. Say that $X$ is ***pairwise distinguishable by L*** if every two distinct strings in $X$ are distinguishable by $L$. Define the ***index of L*** to be the maximum number of elements in any set that is pairwise distinguishable by $L$. The index of $L$ may be finite or infinite.

- **a.** Show that, if $L$ is recognized by a DFA with $k$ states, $L$ has index at most $k$.
- **b.** Show that, if the index of $L$ is a finite number $k$, it is recognized by a DFA with $k$ states.
- **c.** Conclude that $L$ is regular iff it has finite index. Moreover, its index is the size of the smallest DFA recognizing it.

**1.53** Let $\Sigma = \{0, 1, +, =\}$ and

$$ADD = \{x{=}y{+}z \mid x, y, z \text{ are binary integers, and } x \text{ is the sum of } y \text{ and } z\}.$$

Show that $ADD$ is not regular.

**1.54** Consider the language $F = \{\mathtt{a}^i\mathtt{b}^j\mathtt{c}^k \mid i, j, k \geq 0 \text{ and if } i = 1 \text{ then } j = k\}$.

- **a.** Show that $F$ is not regular.
- **b.** Show that $F$ acts like a regular language in the pumping lemma. In other words, give a pumping length $p$ and demonstrate that $F$ satisfies the three conditions of the pumping lemma for this value of $p$.
- **c.** Explain why parts (a) and (b) do not contradict the pumping lemma.

**1.55** The pumping lemma says that every regular language has a pumping length $p$, such that every string in the language can be pumped if it has length $p$ or more. If $p$ is a pumping length for language $A$, so is any length $p' \geq p$. The ***minimum pumping length*** for $A$ is the smallest $p$ that is a pumping length for $A$. For example, if $A = \mathtt{01}^*$, the minimum pumping length is 2. The reason is that the string $s = \mathtt{0}$ is in $A$ and has length 1 yet $s$ cannot be pumped, but any string in $A$ of length 2 or more contains a 1 and hence can be pumped by dividing it so that $x = \mathtt{0}$, $y = \mathtt{1}$, and $z$ is the rest. For each of the following languages, give the minimum pumping length and justify your answer.

- A **a.** $\mathtt{0001}^*$
- A **b.** $\mathtt{0}^*\mathtt{1}^*$
- **c.** $\mathtt{001} \cup \mathtt{0}^*\mathtt{1}^*$
- A **d.** $\mathtt{0}^*\mathtt{1}^+\mathtt{0}^+\mathtt{1}^* \cup \mathtt{10}^*\mathtt{1}$
- **e.** $(\mathtt{01})^*$
- **f.** $\varepsilon$
- **g.** $\mathtt{1}^*\mathtt{01}^*\mathtt{01}^*$
- **h.** $\mathtt{10}(\mathtt{11}^*\mathtt{0})^*\mathtt{0}$
- **i.** $\mathtt{1011}$
- **j.** $\Sigma^*$

* **1.56** If $A$ is a set of natural numbers and $k$ is a natural number greater than 1, let

$$B_k(A) = \{w \mid w \text{ is the representation in base } k \text{ of some number in } A\}.$$

Here, we do not allow leading 0s in the representation of a number. For example, $B_2(\{3, 5\}) = \{\mathtt{11}, \mathtt{101}\}$ and $B_3(\{3, 5\}) = \{\mathtt{10}, \mathtt{12}\}$. Give an example of a set $A$ for which $B_2(A)$ is regular but $B_3(A)$ is not regular. Prove that your example works.

***1.57** If $A$ is any language, let $A_{\frac{1}{2}-}$ be the set of all first halves of strings in $A$ so that

$$A_{\frac{1}{2}-} = \{x \mid \text{ for some } y,\ |x| = |y| \text{ and } xy \in A\}.$$

Show that, if $A$ is regular, then so is $A_{\frac{1}{2}-}$.

***1.58** If $A$ is any language, let $A_{\frac{1}{3}-\frac{1}{3}}$ be the set of all strings in $A$ with their middle thirds removed so that

$$A_{\frac{1}{3}-\frac{1}{3}} = \{xz \mid \text{ for some } y,\ |x| = |y| = |z| \text{ and } xyz \in A\}.$$

Show that, if $A$ is regular, then $A_{\frac{1}{3}-\frac{1}{3}}$ is not necessarily regular.

***1.59** Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA and let $h$ be a state of $M$ called its "home". A ***synchronizing sequence*** for $M$ and $h$ is a string $s \in \Sigma^*$ where $\delta(q, s) = h$ for every $q \in Q$. (Here we have extended $\delta$ to strings, so that $\delta(q, s)$ equals the state where $M$ ends up when $M$ starts at state $q$ and reads input $s$.) Say that $M$ is ***synchronizable*** if it has a synchronizing sequence for some state $h$. Prove that, if $M$ is a $k$-state synchronizable DFA, then it has a synchronizing sequence of length at most $k^3$. Can you improve upon this bound?

**1.60** Let $\Sigma = \{\mathtt{a}, \mathtt{b}\}$. For each $k \geq 1$, let $C_k$ be the language consisting of all strings that contain an $\mathtt{a}$ exactly $k$ places from the right-hand end. Thus $C_k = \Sigma^*\mathtt{a}\Sigma^{k-1}$. Describe an NFA with $k + 1$ states that recognizes $C_k$, both in terms of a state diagram and a formal description.

**1.61** Consider the languages $C_k$ defined in Problem 1.60. Prove that for each $k$, no DFA can recognize $C_k$ with fewer than $2^k$ states.

**1.62** Let $\Sigma = \{\mathtt{a}, \mathtt{b}\}$. For each $k \geq 1$, let $D_k$ be the language consisting of all strings that have at least one $\mathtt{a}$ among the last $k$ symbols. Thus $D_k = \Sigma^*\mathtt{a}(\Sigma \cup \varepsilon)^{k-1}$. Describe a DFA with at most $k + 1$ states that recognizes $D_k$, both in terms of a state diagram and a formal description.

**1.63**

**a.** Let $A$ be an infinite regular language. Prove that $A$ can be split into two infinite disjoint regular subsets.

**b.** Let $B$ and $D$ be two languages. Write $B \Subset D$ if $B \subseteq D$ and $D$ contains infinitely many strings that are not in $B$. Show that, if $B$ and $D$ are two regular languages where $B \Subset D$, then we can find a regular language $C$ where $B \Subset C \Subset D$.

**1.64** Let $N$ be an NFA with $k$ states that recognizes some language $A$.

**a.** Show that, if $A$ is nonempty, $A$ contains some string of length at most $k$.

**b.** Show that, by giving an example, that part (a) is not necessarily true if you replace both $A$'s by $\overline{A}$.

**c.** Show that, if $\overline{A}$ is nonempty, $\overline{A}$ contains some string of length at most $2^k$.

**d.** Show that the bound given in part (c) is nearly tight; that is, for each $k$, demonstrate an NFA recognizing a language $A_k$ where $\overline{A_k}$ is nonempty and where $\overline{A_k}$'s shortest member strings are of length exponential in $k$. Come as close to the bound in (c) as you can.

*1.65 Prove that, for each $n > 0$, a language $B_n$ exists where

**a.** $B_n$ is recognizable by a NFA that has $n$ states, and

**b.** if $B_n = A_1 \cup \cdots \cup A_k$, for regular languages $A_i$, then at least one of the $A_i$ requires a DFA with exponentially many states.

## SELECTED SOLUTIONS

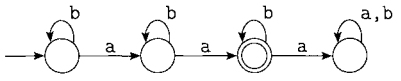
*1.1* For $M_1$: **(a)** $q_1$; **(b)** $\{q_2\}$; **(c)** $q_1, q_2, q_3, q_1, q_1$; **(d)** No; **(e)** No
For $M_2$: **(a)** $q_1$; **(b)** $\{q_1, q_4\}$; **(c)** $q_1, q_1, q_1, q_2, q_4$; **(d)** Yes; **(e)** Yes

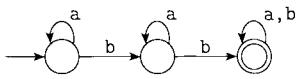
*1.2* $M_2 = (\{q_1, q_2, q_3\}, \{\mathtt{a}, \mathtt{b}\}, \delta_1, q_1, \{q_2\})$.
$M_3 = (\{q_1, q_2, q_3, q_4\}, \{\mathtt{a}, \mathtt{b}\}, \delta_2, q_1, \{q_1, q_4\})$.
The transition functions are

| $\delta_1$ | a | b |
|---|---|---|
| $q_1$ | $q_2$ | $q_1$ |
| $q_2$ | $q_3$ | $q_3$ |
| $q_3$ | $q_2$ | $q_1$ |

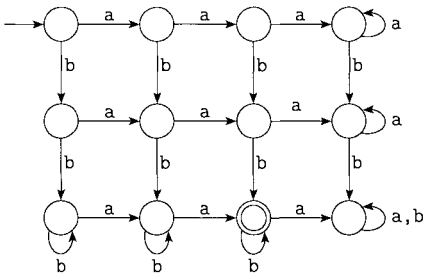
| $\delta_2$ | a | b |
|---|---|---|
| $q_1$ | $q_1$ | $q_2$ |
| $q_2$ | $q_3$ | $q_4$ |
| $q_3$ | $q_2$ | $q_1$ |
| $q_4$ | $q_3$ | $q_4$ |

*1.4* **(b)** The following are DFAs for the two languages $\{w \mid w$ has exactly two a's$\}$ and $\{w \mid w$ has at least two b's$\}$:

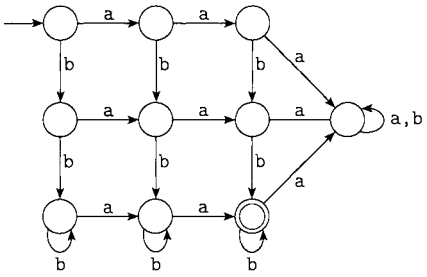


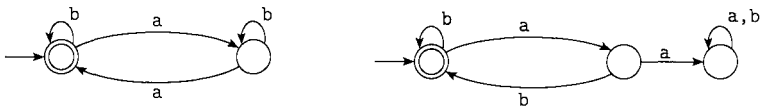

Combining them using the intersection construction gives the DFA:



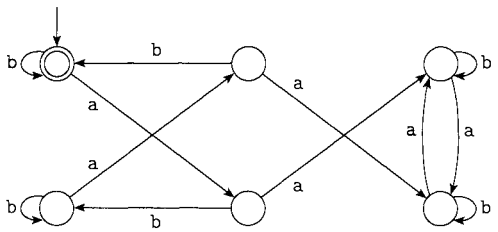

Though the problem doesn't request you to simplify the DFA, certain states can be combined to give



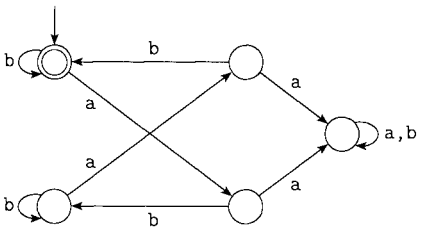

**(d)** These are DFAs for the two languages $\{w|\ w$ has an even number of a's$\}$ and $\{w|$ each a is followed by at least one b$\}$:




Combining them using the intersection construction gives the DFA:

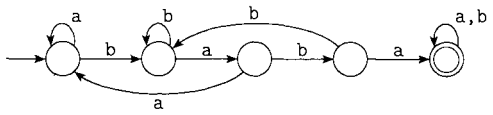



Though the problem doesn't request you to simplify the DFA, certain states can be combined to give

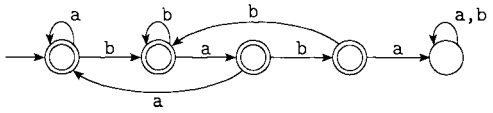


*1.5* **(a)** The left-hand DFA recognizes $\{w \mid w$ contains ab$\}$. The right-hand DFA recognizes its complement, $\{w \mid w$ doesn't contain ab$\}$.

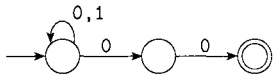

**(b)** This DFA recognizes $\{w \mid w$ contains baba$\}$.



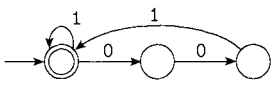

This DFA recognizes $\{w \mid w$ does not contain baba$\}$.




*1.7* **(a)**




**(f)**




*1.11* Let $N = (Q, \Sigma, \delta, q_0, F)$ be any NFA. Construct an NFA $N'$ with a single accept state that accepts the same language as $N$. Informally, $N'$ is exactly like $N$ except it has $\varepsilon$-transitions from the states corresponding to the accept states of $N$, to a new accept state, $q_{\text{accept}}$. State $q_{\text{accept}}$ has no emerging transitions. More formally, $N' = (Q \cup \{q_{\text{accept}}\}, \Sigma, \delta', q_0, \{q_{\text{accept}}\})$, where for each $q \in Q$ and $a \in \Sigma$

$$\delta'(q,a) = \begin{cases} \delta(q,a) & \text{if } a \neq \varepsilon \text{ or } q \notin F \\ \delta(q,a) \cup \{q_{\text{accept}}\} & \text{if } a = \varepsilon \text{ and } q \in F \end{cases}$$

and $\delta'(q_{\text{accept}}, a) = \emptyset$ for each $a \in \Sigma_\varepsilon$.

*1.23* We prove both directions of the "iff."
($\rightarrow$) Assume that $B = B^+$ and show that $BB \subseteq B$.
For every language $BB \subseteq B^+$ holds, so if $B = B^+$, then $BB \subseteq B$.
($\leftarrow$) Assume that $BB \subseteq B$ and show that $B = B^+$.
For every language $B \subseteq B^+$, so we need to show only $B^+ \subseteq B$. If $w \in B^+$, then $w = x_1 x_2 \cdots x_k$ where each $x_i \in B$ and $k \geq 1$. Because $x_1, x_2 \in B$ and $BB \subseteq B$, we have $x_1 x_2 \in B$. Similarly, because $x_1 x_2$ is in $B$ and $x_3$ is in $B$, we have $x_1 x_2 x_3 \in B$. Continuing in this way, $x_1 \cdots x_k \in B$. Hence $w \in B$, and so we may conclude that $B^+ \subseteq B$.

The latter argument may be written formally as the following proof by induction.
Assume that $BB \subseteq B$.
*Claim:* For each $k \geq 1$, if $x_1, \ldots, x_k \in B$, then $x_1 \cdots x_k \in B$.
*Basis:* Prove for $k = 1$. This statement is obviously true.
*Induction step:* For each $k \geq 1$, assume that the claim is true for $k$ and prove it to be true for $k + 1$.
If $x_1, \ldots, x_k, x_{k+1} \in B$, then by the induction assumption, $x_1 \cdots x_k \in B$. Therefore $x_1 \cdots x_k x_{k+1} \in BB$, but $BB \subseteq B$, so $x_1 \cdots x_{k+1} \in B$. That proves the induction step and the claim. The claim implies that, if $BB \subseteq B$, then $B^* \subseteq B$.

***1.29*** **(a)** Assume that $A_1 = \{0^n 1^n 2^n \mid n \geq 0\}$ is regular. Let $p$ be the pumping length given by the pumping lemma. Choose $s$ to be the string $0^p 1^p 2^p$. Because $s$ is a member of $A_1$ and $s$ is longer than $p$, the pumping lemma guarantees that $s$ can be split into three pieces, $s = xyz$, where for any $i \geq 0$ the string $xy^iz$ is in $A_1$. Consider two possibilities:

1. The string $y$ consists only of 0s, only of 1s, or only of 2s. In these cases the string $xyyz$ will not have equal numbers of 0s, 1s, and 2s. Hence $xyyz$ is not a member of $A_1$, a contradiction.
2. The string $y$ consists of more than one kind of symbol. In this case $xyyz$ will have the 0s, 1s, or 2s out of order. Hence $xyyz$ is not a member of $A_1$, a contradiction.

Either way we arrive at a contradiction. Therefore, $A_1$ is not regular.

**(c)** Assume that $A_3 = \{a^{2^n} \mid n \geq 0\}$ is regular. Let $p$ be the pumping length given by the pumping lemma. Choose $s$ to be the string $a^{2^p}$. Because $s$ is a member of $A_1$ and $s$ is longer than $p$, the pumping lemma guarantees that $s$ can be split into three pieces, $s = xyz$, satisfying the three conditions of the pumping lemma.

The third condition tells us that $|xy| \leq p$. Furthermore, $p < 2^p$ and so $|y| < 2^p$. Therefore $|xyyz| = |xyz| + |y| < 2^p + 2^p = 2^{p+1}$. The second condition requires $|y| > 1$ so $2^p < |xyyz| < 2^{p+1}$. The length of $xyyz$ cannot be a power of 2. Hence $xyyz$ is not a member of $A_3$, a contradiction. Therefore, $A_3$ is not regular.

***1.40*** Let $M = (Q, \Sigma, \delta, q_0, F)$ be an NFA recognizing $A$, where $A$ is some regular language. Construct $M' = (Q', \Sigma, \delta', q_0', F')$ recognizing $NOPREFIX(A)$ as follows:

1. $Q' = Q$.
2. For $r \in Q'$ and $a \in \Sigma$ define $\delta'(r, a) = \begin{cases} \delta(r, a) & \text{if } r \notin F \\ \emptyset & \text{if } r \in F. \end{cases}$
3. $q_0' = q_0$.
4. $F' = F$.

***1.44*** Let $M_B = (Q_B, \Sigma, \delta_B, q_B, F_B)$ and $M_C = (Q_C, \Sigma, \delta_C, q_C, F_C)$ be DFAs recognizing $B$ and $C$ respectively. Construct NFA $M = (Q, \Sigma, \delta, q_0, F)$ that recognizes $B \stackrel{1}{\leftarrow} C$ as follows. To decide whether its input $w$ is in $B \stackrel{1}{\leftarrow} C$, the machine $M$ checks that $w \in B$, and in parallel, nondeterministically guesses a string $y$ that contains the same number of 1s as contained in $w$ and checks that $y \in C$.

1. $Q = Q_B \times Q_C$.
2. For $(q, r) \in Q$ and $a \in \Sigma$ define

$$\delta((q,r),a) = \begin{cases} \{(\delta_B(q,0),r)\} & \text{if } a = 0 \\ \{(\delta_B(q,1),\delta_C(r,1))\} & \text{if } a = 1 \\ \{(q,\delta_C(r,0))\} & \text{if } a = \varepsilon. \end{cases}$$

3. $q_0 = (q_B, q_C)$.
4. $F = F_B \times F_C$.

***1.46*** **(b)** Let $B = \{0^m 1^n \mid m \neq n\}$. Observe that $\overline{B} \cap 0^*1^* = \{0^k 1^k \mid k \geq 0\}$. If $B$ were regular, then $\overline{B}$ would be regular and so would $\overline{B} \cap 0^*1^*$. But we already know that $\{0^k 1^k \mid k \geq 0\}$ isn't regular, so $B$ cannot be regular.

Alternatively, we can prove $B$ to be nonregular by using the pumping lemma directly, though doing so is trickier. Assume that $B = \{0^m 1^n \mid m \neq n\}$ is regular. Let $p$ be the pumping length given by the pumping lemma. Observe that $p!$ is divisible by all integers from 1 to $p$, where $p! = p(p-1)(p-2)\cdots 1$. The string $s = 0^p 1^{p+p!} \in B$, and $|s| \geq p$. Thus the pumping lemma implies that $s$ can be divided as $xyz$ with $x = 0^a$, $y = 0^b$, and $z = 0^c 1^{p+p!}$, where $b \geq 1$ and $a + b + c = p$. Let $s'$ be the string $xy^{i+1}z$, where $i = p!/b$. Then $y^i = 0^{p!}$ so $y^{i+1} = 0^{b+p!}$, and so $xyz = 0^{a+b+c+p!}1^{p+p!}$. That gives $xyz = 0^{p+p!}1^{p+p!} \notin B$, a contradiction.

***1.50*** Assume to the contrary that some FST $T$ outputs $w^{\mathcal{R}}$ on input $w$. Consider the input strings 00 and 01. On input 00, $T$ must output 00, and on input 01, $T$ must output 10. In both cases the first input bit is a 0 but the first output bits differ. Operating in this way is impossible for an FST because it produces its first output bit before it reads its second input. Hence no such FST can exist.

***1.52*** **(a)** We prove this assertion by contradiction. Let $M$ be a $k$-state DFA that recognizes $L$. Suppose for a contradiction that $L$ has index greater than $k$. That means some set $X$ with more than $k$ elements is pairwise distinguishable by $L$. Because $M$ has $k$ states, the pigeonhole principle implies that $X$ contains two distinct strings $x$ and $y$, where $\delta(q_0, x) = \delta(q_0, y)$. Here $\delta(q_0, x)$ is the state that $M$ is in after starting in the start state $q_0$ and reading input string $x$. Then, for any string $z \in \Sigma^*$, $\delta(q_0, xz) = \delta(q_0, yz)$. Therefore either both $xz$ and $yz$ are in $L$ or neither are in $L$. But then $x$ and $y$ aren't distinguishable by $L$, contradicting our assumption that $X$ is pairwise distinguishable by $L$.

**(b)** Let $X = \{s_1, \ldots, s_k\}$ be pairwise distinguishable by $L$. We construct DFA $M = (Q, \Sigma, \delta, q_0, F)$ with $k$ states recognizing $L$. Let $Q = \{q_1, \ldots, q_k\}$, and define $\delta(q_i, a)$ to be $q_j$, where $s_j \equiv_L s_i a$ (the relation $\equiv_L$ is defined in Problem 1.51). Note that $s_j \equiv_L s_i a$ for some $s_j \in X$; otherwise, $X \cup s_i a$ would have $k + 1$ elements and would be pairwise distinguishable by $L$, which would contradict the assumption that $L$ has index $k$. Let $F = \{q_i \mid s_i \in L\}$. Let the start state $q_0$ be the $q_i$ such that $s_i \equiv_L \varepsilon$. $M$ is constructed so that, for any state $q_i$, $\{s \mid \delta(q_0, s) = q_i\} = \{s \mid s \equiv_L s_i\}$. Hence $M$ recognizes $L$.

**(c)** Suppose that $L$ is regular and let $k$ be the number of states in a DFA recognizing $L$. Then from part (a) $L$ has index at most $k$. Conversely, if $L$ has index $k$, then by part (b) it is recognized by a DFA with $k$ states and thus is regular. To show that the index of $L$ is the size of the smallest DFA accepting it, suppose that $L$'s index is *exactly* $k$. Then, by part (b), there is a $k$-state DFA accepting $L$. That is the smallest such DFA because if it were any smaller, then we could show by part (a) that the index of $L$ is less than $k$.

***1.55*** **(a)** The minimum pumping length is 4. The string 000 is in the language but cannot be pumped, so 3 is not a pumping length for this language. If $s$ has length 4 or more, it contains 1s. By dividing $s$ onto $xyz$, where $x$ is 000 and $y$ is the first 1 and $z$ is everything afterward, we satisfy the pumping lemma's three conditions.

**(b)** The minimum pumping length is 1. The pumping length cannot be 0 because the string $\varepsilon$ is in the language and it cannot be pumped. Every nonempty string in the language can be divided into $xyz$, where $x = \varepsilon$ and $y$ is the first character and $z$ is the remainder. This division satisfies the three conditions.

**(d)** The minimum pumping length is 3. The pumping length cannot be 2 because the string 11 is in the language and it cannot be pumped. Let $s$ be a string in the language of length at least 3. If $s$ is generated by $0^*1^+0^+1^*$, we can write it as $xyz$, where $x$ is the empty string, $y$ is the first symbol of $s$, and $z$ is the remainder of $s$. Breaking $s$ up in this way shows that it can be pumped. If $s$ is generated by $10^*1$, we can write it as $xyz$, where $x = 1$ and $y = 0$ and $z$ is the remainder of $s$. This division gives a way to pump $s$.

## EXERCISES

**2.1** Recall the CFG $G_4$ that we gave in Example 2.4. For convenience, let's rename its variables with single letters as follows.

$$\begin{aligned} E &\to E + T \mid T \\ T &\to T \times F \mid F \\ F &\to (E) \mid \mathtt{a} \end{aligned}$$

Give parse trees and derivations for each string.

**a.** a
**b.** a+a
**c.** a+a+a
**d.** ((a))

**2.2** **a.** Use the languages $A = \{\mathtt{a}^m\mathtt{b}^n\mathtt{c}^n \mid m, n \geq 0\}$ and $B = \{\mathtt{a}^n\mathtt{b}^n\mathtt{c}^m \mid m, n \geq 0\}$ together with Example 2.36 to show that the class of context-free languages is not closed under intersection.

**b.** Use part (a) and DeMorgan's law (Theorem 0.20) to show that the class of context-free languages is not closed under complementation.

**A2.3** Answer each part for the following context-free grammar $G$.

$$\begin{aligned} R &\to XRX \mid S \\ S &\to \mathtt{a}T\mathtt{b} \mid \mathtt{b}T\mathtt{a} \\ T &\to XTX \mid X \mid \varepsilon \\ X &\to \mathtt{a} \mid \mathtt{b} \end{aligned}$$

**a.** What are the variables of $G$?
**b.** What are the terminals of $G$?
**c.** Which is the start variable of $G$?
**d.** Give three strings in $L(G)$.
**e.** Give three strings *not* in $L(G)$.
**f.** True or False: $T \Rightarrow \mathtt{aba}$.
**g.** True or False: $T \stackrel{*}{\Rightarrow} \mathtt{aba}$.
**h.** True or False: $T \Rightarrow T$.
**i.** True or False: $T \stackrel{*}{\Rightarrow} T$.
**j.** True or False: $XXX \stackrel{*}{\Rightarrow} \mathtt{aba}$.
**k.** True or False: $X \stackrel{*}{\Rightarrow} \mathtt{aba}$.
**l.** True or False: $T \stackrel{*}{\Rightarrow} XX$.
**m.** True or False: $T \stackrel{*}{\Rightarrow} XXX$.
**n.** True or False: $S \stackrel{*}{\Rightarrow} \varepsilon$.
**o.** Give a description in English of $L(G)$.

**2.4** Give context-free grammars that generate the following languages. In all parts the alphabet $\Sigma$ is $\{0,1\}$.

**A a.** $\{w \mid w$ contains at least three 1s$\}$
**b.** $\{w \mid w$ starts and ends with the same symbol$\}$
**c.** $\{w \mid$ the length of $w$ is odd$\}$
**A d.** $\{w \mid$ the length of $w$ is odd and its middle symbol is a 0$\}$
**e.** $\{w \mid w = w^{\mathcal{R}}$, that is, $w$ is a palindrome$\}$
**f.** The empty set

**2.5** Give informal descriptions and state diagrams of pushdown automata for the languages in Exercise 2.4.

**2.6** Give context-free grammars generating the following languages.

**[A]a.** The set of strings over the alphabet {a,b} with more a's than b's

**b.** The complement of the language $\{\mathtt{a}^n\mathtt{b}^n \mid n \geq 0\}$

**[A]c.** $\{w\#x \mid w^{\mathcal{R}}$ is a substring of $x$ for $w, x \in \{0,1\}^*\}$

**d.** $\{x_1\#x_2\#\cdots\#x_k \mid k \geq 1$, each $x_i \in \{\mathtt{a},\mathtt{b}\}^*$, and for some $i$ and $j$, $x_i = x_j^{\mathcal{R}}\}$

**[A]2.7** Give informal English descriptions of PDAs for the languages in Exercise 2.6.

**[A]2.8** Show that the string `the girl touches the boy with the flower` has two different leftmost derivations in grammar $G_2$ on page 101. Describe in English the two different meanings of this sentence.

**2.9** Give a context-free grammar that generates the language

$$A = \{\mathtt{a}^i\mathtt{b}^j\mathtt{c}^k \mid i = j \text{ or } j = k \text{ where } i, j, k \geq 0\}.$$

Is your grammar ambiguous? Why or why not?

**2.10** Give an informal description of a pushdown automaton that recognizes the language $A$ in Exercise 2.9.

**2.11** Convert the CFG $G_4$ given in Exercise 2.1 to an equivalent PDA, using the procedure given in Theorem 2.20.

**2.12** Convert the CFG $G$ given in Exercise 2.3 to an equivalent PDA, using the procedure given in Theorem 2.20.

**2.13** Let $G = (V, \Sigma, R, S)$ be the following grammar. $V = \{S, T, U\}$; $\Sigma = \{0, \#\}$; and $R$ is the set of rules:

$$\begin{aligned} S &\rightarrow TT \mid U \\ T &\rightarrow 0T \mid T0 \mid \# \\ U &\rightarrow 0U00 \mid \# \end{aligned}$$

**a.** Describe $L(G)$ in English.

**b.** Prove that $L(G)$ is not regular.

**2.14** Convert the following CFG into an equivalent CFG in Chomsky normal form, using the procedure given in Theorem 2.9.

$$\begin{aligned} A &\rightarrow BAB \mid B \mid \varepsilon \\ B &\rightarrow 00 \mid \varepsilon \end{aligned}$$

**2.15** Give a counterexample to show that the following construction fails to prove that the class of context-free languages is closed under star. Let $A$ be a CFL that is generated by the CFG $G = (V, \Sigma, R, S)$. Add the new rule $S \rightarrow SS$ and call the resulting grammar $G'$. This grammar is supposed to generate $A^*$.

**2.16** Show that the class of context-free languages is closed under the regular operations, union, concatenation, and star.

**2.17** Use the results of Problem 2.16 to give another proof that every regular language is context free, by showing how to convert a regular expression directly to an equivalent context-free grammar.

## PROBLEMS

[A]**2.18** **a.** Let $C$ be a context-free language and $R$ be a regular language. Prove that the language $C \cap R$ is context free.

**b.** Use part (a) to show that the language $A = \{w \mid w \in \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}^*$ and contains equal numbers of a's, b's, and c's$\}$ is not a CFL.

*2.19 Let CFG $G$ be

$$S \rightarrow \mathtt{a}S\mathtt{b} \mid \mathtt{b}Y \mid Y\mathtt{a}$$
$$Y \rightarrow \mathtt{b}Y \mid \mathtt{a}Y \mid \varepsilon$$

Give a simple description of $L(G)$ in English. Use that description to give a CFG for $\overline{L(G)}$, the complement of $L(G)$.

**2.20** Let $A/B = \{w \mid wx \in A$ for some $x \in B\}$. Show that, if $A$ is context free and $B$ is regular, then $A/B$ is context free.

*2.21 Let $\Sigma = \{\mathtt{a},\mathtt{b}\}$. Give a CFG generating the language of strings with twice as many a's as b's. Prove that your grammar is correct.

*2.22 Let $C = \{x\#y \mid x, y \in \{\mathtt{0},\mathtt{1}\}^*$ and $x \neq y\}$. Show that $C$ is a context-free language.

*2.23 Let $D = \{xy \mid x, y \in \{\mathtt{0},\mathtt{1}\}^*$ and $|x| = |y|$ but $x \neq y\}$. Show that $D$ is a context-free language.

*2.24 Let $E = \{\mathtt{a}^i\mathtt{b}^j \mid i \neq j$ and $2i \neq j\}$. Show that $E$ is a context-free language.

**2.25** For any language $A$, let $SUFFIX(A) = \{v \mid uv \in A$ for some string $u\}$. Show that the class of context-free languages is closed under the $SUFFIX$ operation.

**2.26** Show that, if $G$ is a CFG in Chomsky normal form, then for any string $w \in L(G)$ of length $n \geq 1$, exactly $2n - 1$ steps are required for any derivation of $w$.

*2.27 Let $G = (V, \Sigma, R, \langle \text{STMT} \rangle)$ be the following grammar.

$$\begin{aligned}
\langle \text{STMT} \rangle &\rightarrow \langle \text{ASSIGN} \rangle \mid \langle \text{IF-THEN} \rangle \mid \langle \text{IF-THEN-ELSE} \rangle \\
\langle \text{IF-THEN} \rangle &\rightarrow \texttt{if condition then } \langle \text{STMT} \rangle \\
\langle \text{IF-THEN-ELSE} \rangle &\rightarrow \texttt{if condition then } \langle \text{STMT} \rangle \texttt{ else } \langle \text{STMT} \rangle \\
\langle \text{ASSIGN} \rangle &\rightarrow \texttt{a:=1}
\end{aligned}$$

$$\Sigma = \{\texttt{if}, \texttt{condition}, \texttt{then}, \texttt{else}, \texttt{a:=1}\}.$$
$$V = \{\langle \text{STMT} \rangle, \langle \text{IF-THEN} \rangle, \langle \text{IF-THEN-ELSE} \rangle, \langle \text{ASSIGN} \rangle\}$$

$G$ is a natural-looking grammar for a fragment of a programming language, but $G$ is ambiguous.

**a.** Show that $G$ is ambiguous.

**b.** Give a new unambiguous grammar for the same language.

*2.28 Give unambiguous CFGs for the following languages.

**a.** $\{w \mid$ in every prefix of $w$ the number of a's is at least the number of b's$\}$

**b.** $\{w \mid$ the number of a's and b's in $w$ are equal$\}$

**c.** $\{w \mid$ the number of a's is at least the number of b's$\}$

*2.29 Show that the language $A$ in Exercise 2.9 is inherently ambiguous.

**2.30** Use the pumping lemma to show that the following languages are not context free.

**a.** $\{0^n1^n0^n1^n \mid n \geq 0\}$

**A b.** $\{0^n\#0^{2n}\#0^{3n} \mid n \geq 0\}$

**A c.** $\{w\#t \mid w \text{ is a substring of } t, \text{ where } w, t \in \{a, b\}^*\}$

**d.** $\{t_1\#t_2\#\cdots\#t_k \mid k \geq 2, \text{ each } t_i \in \{a, b\}^*, \text{ and } t_i = t_j \text{ for some } i \neq j\}$

**2.31** Let $B$ be the language of all palindromes over $\{0,1\}$ containing an equal number of 0s and 1s. Show that $B$ is not context free.

**2.32** Let $\Sigma = \{1, 2, 3, 4\}$ and $C = \{w \in \Sigma^* \mid \text{in } w, \text{ the number of 1s equals the number of 2s, and the number of 3s equals the number of 4s}\}$. Show that $C$ is not context free.

***2.33** Show that $F = \{a^ib^j \mid i = kj \text{ for some positive integer } k\}$ is not context free.

**2.34** Consider the language $B = L(G)$, where $G$ is the grammar given in Exercise 2.13. The pumping lemma for context-free languages, Theorem 2.34, states the existence of a pumping length $p$ for $B$. What is the minimum value of $p$ that works in the pumping lemma? Justify your answer.

**2.35** Let $G$ be a CFG in Chomsky normal form that contains $b$ variables. Show that, if $G$ generates some string with a derivation having at least $2^b$ steps, $L(G)$ is infinite.

**2.36** Give an example of a language that is not context free but that acts like a CFL in the pumping lemma. Prove that your example works. (See the analogous example for regular languages in Problem 1.54.)

***2.37** Prove the following stronger form of the pumping lemma, wherein *both* pieces $v$ and $y$ must be nonempty when the string $s$ is broken up.

If $A$ is a context-free language, then there is a number $k$ where, if $s$ is any string in $A$ of length at least $k$, then $s$ may be divided into five pieces, $s = uvxyz$, satisfying the conditions:

**a.** for each $i \geq 0$, $uv^ixy^iz \in A$,

**b.** $v \neq \varepsilon$ and $y \neq \varepsilon$, and

**c.** $|vxy| \leq k$.

**A 2.38** Refer to Problem 1.41 for the definition of the perfect shuffle operation. Show that the class of context-free languages is not closed under perfect shuffle.

**2.39** Refer to Problem 1.42 for the definition of the shuffle operation. Show that the class of context-free languages is not closed under shuffle.

***2.40** Say that a language is ***prefix-closed*** if the prefix of any string in the language is also in the language. Let $C$ be an infinite, prefix-closed, context-free language. Show that $C$ contains an infinite regular subset.

***2.41** Read the definitions of $NOPREFIX(A)$ and $NOEXTEND(A)$ in Problem 1.40.

**a.** Show that the class of CFLs is not closed under $NOPREFIX$ operation.

**b.** Show that the class of CFLs is not closed under $NOEXTEND$ operation.

***2.42** Let $\Sigma = \{1, \#\}$ and $Y = \{w \mid w = t_1\#t_2\#\cdots\#t_k \text{ for } k \geq 0, \text{ each } t_i \in 1^*, \text{ and } t_i \neq t_j \text{ whenever } i \neq j\}$. Prove that $Y$ is not context free.

**2.43** For strings $w$ and $t$, write $w \triangleq t$ if the symbols of $w$ are a permutation of the symbols of $t$. In other words, $w \triangleq t$ if $t$ and $w$ have the same symbols in the same quantities, but possibly in a different order.

For any string $w$, define $SCRAMBLE(w) = \{t \mid t \triangleq w\}$. For any language $A$, let $SCRAMBLE(A) = \{t \mid t \in SCRAMBLE(w) \text{ for some } w \in A\}$.

**a.** Show that, if $\Sigma = \{0, 1\}$, then the *SCRAMBLE* of a regular language is context free.

**b.** What happens in part (a) if $\Sigma$ contains 3 or more symbols? Prove your answer.

**2.44** If $A$ and $B$ are languages, define $A \diamond B = \{xy \mid x \in A \text{ and } y \in B \text{ and } |x| = |y|\}$. Show that if $A$ and $B$ are regular languages, then $A \diamond B$ is a CFL.

***2.45** Let $A = \{wtw^{\mathcal{R}} \mid w, t \in \{0, 1\}^* \text{ and } |w| = |t|\}$. Prove that $A$ is not a context-free language.

## SELECTED SOLUTIONS

*2.3* **(a)** $R, X, S, T$; **(b)** a, b; **(c)** $R$; **(d)** Three strings in $G$ are ab, ba, and aab; **(e)** Three strings not in $G$ are a, b, and $\varepsilon$; **(f)** False; **(g)** True; **(h)** False; **(i)** True; **(j)** True; **(k)** False; **(l)** True; **(m)** True; **(n)** False; **(o)** $L(G)$ consists of all strings over a and b that are not palindromes.

*2.4* **(a)** $S \rightarrow R1R1R1R$
$R \rightarrow 0R \mid 1R \mid \varepsilon$

**(d)** $S \rightarrow 0 \mid 0S0 \mid 0S1 \mid 1S0 \mid 1S1$

*2.6* **(a)** $S \rightarrow TaT$
$T \rightarrow TT \mid aTb \mid bTa \mid a \mid \varepsilon$

$T$ generates all strings with at least as many a's as b's, and $S$ forces an extra a.

**(c)** $S \rightarrow TX$
$T \rightarrow 0T0 \mid 1T1 \mid \#X$
$X \rightarrow 0X \mid 1X \mid \varepsilon$

*2.7* **(a)** The PDA uses its stack to count the number of a's minus the number of b's. It enters an accepting state whenever this count is positive. In more detail, it operates as follows. The PDA scans across the input. If it sees a b and its top stack symbol is an a, it pops the stack. Similarly, if it scans an a and its top stack symbol is a b, it pops the stack. In all other cases, it pushes the input symbol onto the stack. After the PDA finishes the input, if a is on top of the stack, it accepts. Otherwise it rejects.

**(c)** The PDA scans across the input string and pushes every symbol it reads until it reads a #. If # is never encountered, it rejects. Then, the PDA skips over part of the input, nondeterministically deciding when to stop skipping. At that point, it compares the next input symbols with the symbols it pops off the stack. At any disagreement, or if the input finishes while the stack is nonempty, this branch of the computation rejects. If the stack becomes empty, the machine reads the rest of the input and accepts.

***2.8*** Here is one derivation:
⟨SENTENCE⟩ ⇒ ⟨NOUN-PHRASE⟩⟨VERB-PHRASE⟩ ⇒
⟨CMPLX-NOUN⟩⟨VERB-PHRASE⟩ ⇒
⟨CMPLX-NOUN⟩⟨CMPLX-VERB⟩⟨PREP-PHRASE⟩ ⇒
⟨ARTICLE⟩⟨NOUN⟩⟨CMPLX-VERB⟩⟨PREP-PHRASE⟩ ⇒
`The boy` ⟨VERB⟩⟨NOUN-PHRASE⟩⟨PREP-PHRASE⟩ ⇒
`The boy` ⟨VERB⟩⟨NOUN-PHRASE⟩⟨PREP⟩⟨CMPLX-NOUN⟩ ⇒
`The boy touches` ⟨NOUN-PHRASE⟩⟨PREP⟩⟨CMPLX-NOUN⟩ ⇒
`The boy touches` ⟨CMPLX-NOUN⟩⟨PREP⟩⟨CMPLX-NOUN⟩ ⇒
`The boy touches` ⟨ARTICLE⟩⟨NOUN⟩⟨PREP⟩⟨CMPLX-NOUN⟩ ⇒
`The boy touches the girl with` ⟨CMPLX-NOUN⟩ ⇒
`The boy touches the girl with` ⟨ARTICLE⟩⟨NOUN⟩ ⇒
`The boy touches the girl with the flower`

Here is another derivation:
⟨SENTENCE⟩ ⇒ ⟨NOUN-PHRASE⟩⟨VERB-PHRASE⟩ ⇒
⟨CMPLX-NOUN⟩⟨VERB-PHRASE⟩ ⇒ ⟨ARTICLE⟩⟨NOUN⟩⟨VERB-PHRASE⟩ ⇒
`The boy` ⟨VERB-PHRASE⟩ ⇒ `The boy` ⟨CMPLX-VERB⟩ ⇒
`The boy` ⟨VERB⟩⟨NOUN-PHRASE⟩ ⇒
`The boy touches` ⟨NOUN-PHRASE⟩ ⇒
`The boy touches` ⟨CMPLX-NOUN⟩⟨PREP-PHRASE⟩ ⇒
`The boy touches` ⟨ARTICLE⟩⟨NOUN⟩⟨PREP-PHRASE⟩ ⇒
`The boy touches the girl` ⟨PREP-PHRASE⟩ ⇒
`The boy touches the girl` ⟨PREP⟩⟨CMPLX-NOUN⟩ ⇒
`The boy touches the girl with` ⟨CMPLX-NOUN⟩ ⇒
`The boy touches the girl with` ⟨ARTICLE⟩⟨NOUN⟩ ⇒
`The boy touches the girl with the flower`

Each of these derivations corresponds to a different English meaning. In the first derivation, the sentence means that the boy used the flower to touch the girl. In the second derivation, the girl is holding the flower when the boy touches her.

***2.18*** **(a)** Let $C$ be a context-free language and $R$ be a regular language. Let $P$ be the PDA that recognizes $C$, and $D$ be the DFA that recognizes $R$. If $Q$ is the set of states of $P$ and $Q'$ is the set of states of $D$, we construct a PDA $P'$ that recognizes $C \cap R$ with the set of states $Q \times Q'$. $P'$ will do what $P$ does and also keep track of the states of $D$. It accepts a string $w$ if and only if it stops at a state $q \in F_P \times F_D$, where $F_P$ is the set of accept states of $P$ and $F_D$ is the set of accept states of $D$. Since $C \cap R$ is recognized by $P'$, it is context free.

**(b)** Let $R$ be the regular language $\mathtt{a}^*\mathtt{b}^*\mathtt{c}^*$. If $A$ were a CFL then $A \cap R$ would be a CFL by part (a). However, $A \cap R = \{\mathtt{a}^n\mathtt{b}^n\mathtt{c}^n \mid n \geq 0\}$, and Example 2.36 proves that $A \cap R$ is not context free. Thus $A$ is not a CFL.

***2.30*** **(b)** Let $B = \{\mathtt{0}^n\#\mathtt{0}^{2n}\#\mathtt{0}^{3n} \mid n \geq 0\}$. Let $p$ be the pumping length given by the pumping lemma. Let $s = \mathtt{0}^p\#\mathtt{0}^{2p}\#\mathtt{0}^{3p}$. We show that $s = uvxyz$ cannot be pumped.

Neither $v$ nor $y$ can contain #, otherwise $xv^2wy^2z$ contains more than two #s. Therefore, if we divide $s$ into three segments by #'s: $\mathtt{0}^p$, $\mathtt{0}^{2p}$, and $\mathtt{0}^{3p}$, at least one of the segments is not contained within either $v$ or $y$. Hence $xv^2wy^2z$ is not in $B$ because the $1:2:3$ length ratio of the segments is not maintained.

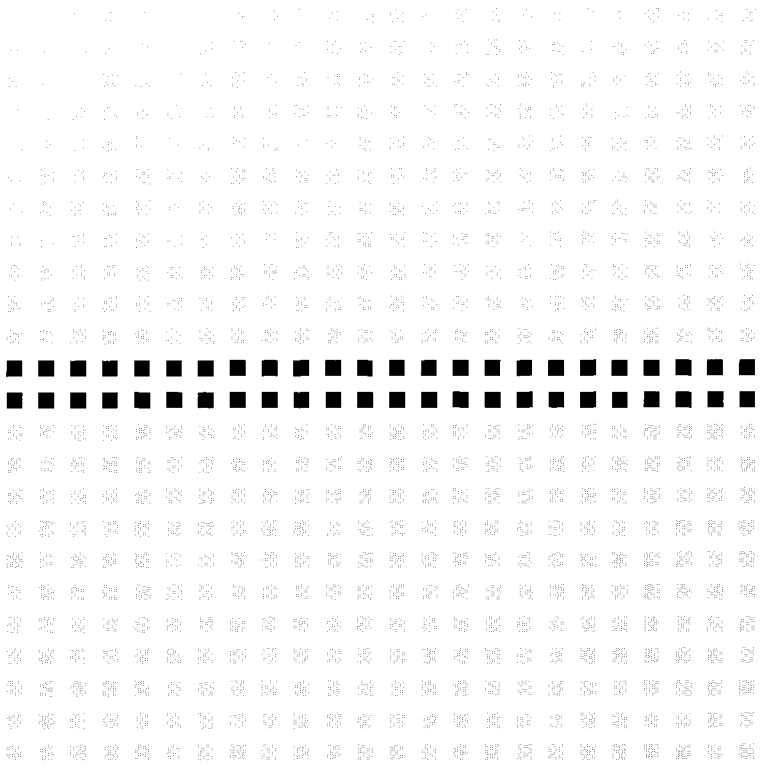
**(c)** Let $C = \{w\#t | \ w$ is a substring of $t$, where $w, t \in \{\mathtt{a}, \mathtt{b}\}^*\}$. Let $p$ be the pumping length given by the pumping lemma. Let $s = \mathtt{a}^p\mathtt{b}^p\#\mathtt{a}^p\mathtt{b}^p$. We show that the string $s = uvxyz$ cannot be pumped.

Neither $v$ nor $y$ can contain #, otherwise $uv^0xy^0z$ does not contain # and therefore is not in $C$. If both $v$ and $y$ are nonempty and occur on the left-hand side of the #, the string $uv^2xy^2z$ cannot be in $C$ because it is longer on the left-hand side of the #. Similarly, if both strings occur on the right-hand side of the #, the string $uv^0xy^0z$ cannot be in $C$ because it is again longer on the left-hand side of the #. If only one of $v$ and $y$ is nonempty (both cannot be nonempty), treat them as if both occurred on the same side of the # as above.

The only remaining case is where both $v$ and $y$ are nonempty and straddle the #. But then $v$ consists of b's and $y$ consists of a's because of the third pumping lemma condition $|vxy| \leq p$. Hence, $uv^2xy^2z$ contains more b's on the left-hand side of the #, so it cannot be a member of $C$.

***2.38*** Let $A$ be the language $\{\mathtt{0}^k\mathtt{1}^k | \ k \geq 0\}$ and let $B$ be the language $\{\mathtt{a}^k\mathtt{b}^{3k} | \ k \geq 0\}$. The perfect shuffle of $A$ and $B$ is the language $C = \{(\mathtt{0a})^k(\mathtt{0b})^k(\mathtt{1b})^{2k} | \ k \geq 0\}$. Languages $A$ and $B$ are easily seen to be CFLs, but $C$ is not a CFL, as follows. If $C$ were a CFL, let $p$ be the pumping length given by the pumping lemma, and let $s$ be the string $(\mathtt{0a})^p(\mathtt{0b})^p(\mathtt{1b})^{2p}$. Because $s$ is longer than $p$ and $s \in C$, we can divide $s = uvxyz$ satisfying the pumping lemma's three conditions. Strings in $C$ contain twice as many 1s as a's. In order for $uv^2xy^2z$ to have that property, the string $vxy$ must contain both 1s and a's. But that is impossible, because they are separated by $2p$ symbols yet the third condition says that $|vxy| \leq p$. Hence $C$ is not context free.

# PART TWO



## COMPUTABILITY THEORY