The last example suggests the conjecture that if a language \( L \) is regular, so are \( L^2, L^3, \ldots \). We will see later that this is indeed correct.

**EXERCISES**

1. Which of the strings 0001, 01101, 00001101 are accepted by the dfa in Figure 2.1?
2. Translate the graph in Figure 2.5 into \( \delta \)- notation.
3. For \( \Sigma = \{a, b\} \), construct dfa’s that accept the sets consisting of
   (a) all strings of even length.
   (b) all strings of length greater than 5.
   (c) all strings with an even number of \( a \)’s.
   (d) all strings with an even number of \( a \)’s and an odd number of \( b \)’s.
4. For \( \Sigma = \{a, b\} \), construct dfa’s that accept the sets consisting of
   (a) all strings with exactly one \( a \).
   (b) all strings with at least two \( a \)’s.
   (c) all strings with no more than two \( a \)’s.
   (d) all strings with at least one \( b \) and exactly two \( a \)’s.
   (e) all the strings with exactly two \( a \)’s and more than three \( b \)’s.
5. Give dfa’s for the languages
   (a) \( L = \{ab^4wb^2 : w \in \{a, b\}^*\} \).
   (b) \( L = \{ab^n a^m : n \geq 3, m \geq 2\} \).
   (c) \( L = \{w_1 abbw_2 : w_1 \in \{a, b\}^*, w_2 \in \{a, b\}^*\} \).
   (d) \( L = \{bab^n : n \geq 1, n \neq 4\} \).
6. With \( \Sigma = \{a, b\} \), give a dfa for \( L = \{w_1 aw_2 : |w_1| \geq 3, |w_2| \leq 4\} \).
7. Find dfa’s for the following languages on \( \Sigma = \{a, b\} \).
   (a) \( L = \{w : |w| \mod 3 \neq 0\} \).
(b) \( L = \{ w : \mid w \mid \text{mod } 5 = 0 \} \).
(c) \( L = \{ w : n_a (w) \text{ mod } 3 < 1 \} \).
(d) \( L = \{ w : n_a (w) \text{ mod } 3 < n_b (w) \text{ mod } 3 \} \).
(e) \( L = \{ w : (n_a (w) - n_b (w)) \text{ mod } 3 = 0 \} \).
(f) \( L = \{ w : (n_a (w) + 2n_b (w)) \text{ mod } 3 < 1 \} \).
(g) \( L = \{ w : \mid w \mid \text{mod } 3 = 0, \mid w \mid \neq 5 \} \).

8. A run in a string is a substring of length at least two, as long as possible, and consisting entirely of the same symbol. For instance, the string \( abbbabaab \) contains a run of \( b \)'s of length three and a run of \( a \)'s of length two. Find dfa’s for the following languages on \( \{a, b\} \):
   (a) \( L = \{ w : w \text{ contains no runs of length less than three} \} \).
   (b) \( L = \{ w : \text{every run of } a \text{'s has length either two or three} \} \).
   (c) \( L = \{ w : \text{there are at most two runs of } a \text{'s of length three} \} \).
   (d) \( L = \{ w : \text{there are exactly two runs of } a \text{'s of length 3} \} \).

9. Show that if we change Figure 2.6, making \( q_3 \) a nonfinal state and making \( q_0, q_1, q_2 \) final states, the resulting dfa accepts \( \overline{L} \).

10. Generalize the observation in the previous exercise. Specifically, show that if \( M = (Q, \Sigma, \delta, q_0, F) \) and \( \overline{M} = (Q, \Sigma, \delta, q_0, Q - F) \) are two dfa’s, then \( \overline{L(M)} = L (\overline{M}) \).

11. Consider the set of strings on \( \{0, 1\} \) defined by the requirements below. For each, construct an accepting dfa.
   (a) Every 00 is followed immediately by a 1. For example, the strings 101, 0010, 0010011001 are in the language, but 0001 and 00100 are not.
   (b) All strings that contain the substring 000, but not 0000.
   (c) The leftmost symbol differs from the rightmost one.
   (d) Every substring of four symbols has, at most, two 0’s. For example, 001110 and 011001 are in the language, but 10010 is not because one of its substrings, 0010, contains three zeros.
   (e) All strings of length five or more in which the third symbol from
the right end is different from the leftmost symbol.

(f) All strings in which the leftmost two symbols and the rightmost two symbols are identical.

(g) All strings of length four or greater in which the leftmost two symbols are the same, but different from the rightmost symbol.

12. Construct a dfa that accepts strings on \{0, 1\} if and only if the value of the string, interpreted as a binary representation of an integer, is zero modulo five. For example, 0101 and 1111, representing the integers 5 and 15, respectively, are to be accepted.

13. Show that the language \( L = \{vwv : v, w \in \{a, b\}^*, |v| = 3\} \) is regular.

14. Show that \( L = \{a^n : n \geq 3\} \) is regular.

15. Show that the language \( L = \{a^n : n \geq 0, n \neq 3\} \) is regular.

16. Show that the language \( L = \{a^n : n \text{ is either a multiple of three or a multiple of 5}\} \) is regular.

17. Show that the language \( L = \{a^n : n \text{ is a multiple of three, but not a multiple of 5}\} \) is regular.

18. Show that the set of all real numbers in \( \mathbb{C} \) is a regular language.

19. Show that if \( L \) is regular, so is \( L - \{\lambda\} \).

20. Show that if \( L \) is regular, so is \( L \cup \{aa\} \), for all \( a \in \Sigma \).

21. Use (2.1) and (2.2) to show that

\[
\delta^*(q, wv) = \delta^*(\delta^*(q, w), v)
\]

for all \( w, v \in \Sigma^* \).

22. Let \( L \) be the language accepted by the automaton in Figure 2.2.

Find a dfa that accepts \( L^3 \).

23. Let \( L \) be the language accepted by the automaton in Figure 2.2.

Find a dfa for the language \( L^2 - L \).

24. Let \( L \) be the language in Example 2.5. Show that \( L^* \) is regular.

25. Let \( G_M \) be the transition graph for some dfa \( M \). Prove the following:

(a) If \( L (M) \) is infinite, then \( G_M \) must have at least one cycle for which there is a path from the initial vertex to some vertex in the
cycle and a path from some vertex in the cycle to some final vertex.

(b) If $L(M)$ is finite, then no such cycle exists.

26. Let us define an operation $\text{truncate}$, which removes the rightmost symbol from any string. For example, $\text{truncate}(aaaba)$ is $aaab$. The operation can be extended to languages by

$$\text{truncate}(L) = \{\text{truncate}(w) : w \in L\}.$$

Show how, given a dfa for any regular language $L$, one can construct a dfa for $\text{truncate}(L)$. From this, prove that if $L$ is a regular language not containing $\lambda$, then $\text{truncate}(L)$ is also regular.

27. While the language accepted by a given dfa is unique, there are normally many dfa’s that accept a language. Find a dfa with exactly six states that accepts the same language as the dfa in Figure 2.4.

28. Can you find a dfa with three states that accepts the language of the dfa in Figure 2.4? If not, can you give convincing arguments that no such dfa can exist?
In the same vein, nondeterminism is an effective mechanism for describing some complicated languages concisely. Notice that the definition of a grammar involves a nondeterministic element. In 

\[ S \rightarrow aSb|\lambda \]

we can at any point choose either the first or the second production. This lets us specify many different strings using only two rules.

Finally, there is a technical reason for introducing nondeterminism. As we will see, certain theoretical results are more easily established for nfa’s than for dfa’s. Our next major result indicates that there is no essential difference between these two types of automata. Consequently, allowing nondeterminism often simplifies formal arguments without affecting the generality of the conclusion.

**EXERCISES**

1. Construct an nfa that accepts all integer numbers in C. Explain why your construct is an nfa.

2. Prove in detail the claim made in the previous section that if in a transition graph there is a walk labeled \( w \), there must be some walk labeled \( w \) of length no more than \(|\Lambda + (1 + \Lambda) | w |\).

3. Find a dfa that accepts the language defined by the nfa in Figure 2.8.

4. Find a dfa that accepts the complement of the language defined by the nfa in Figure 2.8.

5. In Figure 2.9, find \( \delta^* (q_0, 1011) \) and \( \delta^* (q_1, 01) \).

6. In Figure 2.10, find \( \delta^* (q_0, a) \) and \( \delta^* (q_1, \lambda) \).

7. For the nfa in Figure 2.9, find \( \delta^* (q_0, 1010) \) and \( \delta^* (q_1, 00) \).

8. Design an nfa with no more than five states for the set \{abab^n : n \geq 0\} \cup \{aba^n : n \geq 0\}. 


9. Construct an nfa with three states that accepts the language \{ab, abc\}*

10. Do you think Exercise 9 can be solved with fewer than three states?

11. (a) Find an nfa with three states that accepts the language

\[ L = \{a^n : n \geq 1\} \cup \{b^m a^k : m \geq 0, k \geq 0\}. \]

(b) Do you think the language in part (a) can be accepted by an nfa with fewer than three states?

12. Find an nfa with four states for \( L = \{a^n : n \geq 0\} \cup \{b^m a : n \geq 1\} \).

13. Which of the strings 00, 01001, 10010, 000, 0000 are accepted by the following nfa?

![Diagram of an NFA]

14. What is the complement of the language accepted by the nfa in Figure 2.10?

15. Let \( L \) be the language accepted by the nfa in Figure 2.8. Find an nfa that accepts \( L \cup \{a^5\} \).

16. Find an nfa for \( L^* \), where \( L \) is the language in Exercise 15.

17. Find an nfa that accepts \( \{a\}^* \) and is such that if in its transition graph a single edge is removed (without any other changes), the resulting automaton accepts \( \{a\} \).

18. Can Exercise 17 be solved using a dfa? If so, give the solution; if not, give convincing arguments for your conclusion.

19. Consider the following modification of Definition 2.6. An nfa with multiple initial states is defined by the quintuple
\[ M = (Q, \Sigma, \delta, Q_0, F), \]
where \( Q_0 \subseteq Q \) is a set of possible initial states. The language accepted by such an automaton is defined as

\[ L(M) = \{ w : \delta^*(q_0, w) \text{ contains } q_f, \text{ for any } q_0 \in Q_0, q_f \in F \}. \]

Show that for every nfa with multiple initial states there exists an nfa with a single initial state that accepts the same language.

20. Suppose that in Exercise 19 we made the restriction \( Q_0 \cap F = \emptyset \). Would this affect the conclusion?

21. Use Definition 2.5 to show that for any nfa

\[ \delta^* (q, wv) = \bigcup_{p \in \delta^*(q, w)} \delta^* (p, v), \]

for all \( q \in Q \) and all \( w, v \in \Sigma^* \).

22. An nfa in which (a) there are no \( \lambda \)-transitions, and (b) for all \( q \in Q \) and all \( a \in \Sigma, \delta (q, a) \), it contains at most one element, is sometimes called an incomplete dfa. This is reasonable because the conditions make it such that there is never any choice of moves.

For \( \Sigma = \{a, b\} \), convert the incomplete dfa below into a standard dfa.

23. Let \( L \) be a regular language on some alphabet \( \Sigma \), and let \( \Sigma_1 \subset \Sigma \) be a smaller alphabet. Consider \( L_1 \), the subset of \( L \) whose elements are made up only of symbols from \( \Sigma_1 \), that is,
Show that $L_1$ is also regular.

$L_1 = L \cap \Sigma_1^*$. 
we have the partially constructed automaton shown in Figure 2.15. Since there are still some missing edges, we continue until we obtain the complete solution in Figure 2.16.

One important conclusion we can draw from Theorem 2.2 is that every language accepted by an nfa is regular.

EXERCISES

1. Use the construction of Theorem 2.2 to convert the nfa in Figure 2.10 to a dfa. Can you see a simpler answer more directly?

2. Convert the nfa in Exercise 13, Section 2.2, into an equivalent dfa.

3. Convert the nfa defined by

\[
\begin{align*}
\delta(q_0, a) &= \{q_0, q_1\} \\
\delta(q_1, b) &= \{q_1, q_2\} \\
\delta(q_2, a) &= \{q_2\}
\end{align*}
\]

with initial state \(q_0\) and final state \(q_2\) into an equivalent dfa.

4. Convert the nfa defined by

\[
\begin{align*}
\delta(q_0, a) &= \{q_0, q_1\} \\
\delta(q_1, b) &= \{q_0, q_2\} \\
\delta(q_2, a) &= \{q_1\} \\
\delta(q_0, \lambda) &= \{q_2\}
\end{align*}
\]

with initial state \(q_0\) and final state \(q_2\) into an equivalent dfa.

5. Convert the nfa defined by
6. Carefully complete the arguments in the proof of Theorem 2.2. Show in detail that if the label of \( \delta_D(q_0, w) \) contains \( q_f \), then \( \delta_N(q_0, w) \) also contains \( q_f \).

7. Is it true that for any nfa \( M = (Q, \Sigma, \delta, q_0, F) \), the complement of \( L(M) \) is equal to the set \( \{ w \in \Sigma^* : \delta^*(q_0, w) \cap F = \emptyset \} \)? If so, prove it. If not, give a counterexample.

8. Is it true that for every nfa \( M = (Q, \Sigma, \delta, q_0, F) \), the complement of \( L(M) \) is equal to the set \( \{ w \in \Sigma^* : \delta^*(q_0, w) \cap (Q - F) \neq \emptyset \} \)? If so, prove it; if not, give a counterexample.

9. Prove that for every nfa with an arbitrary number of final states there is an equivalent nfa with only one final state. Can we make a similar claim for dfa’s?

10. Find an nfa without \( \lambda \)-transitions and with a single final state that accepts the set \( \{ a \} \cup \{ b^n : n \geq 2 \} \).

11. Let \( L \) be a regular language that does not contain \( \lambda \). Show that an nfa exists without \( \lambda \)-transitions and with a single final state that accepts \( L \).

12. Define a dfa with multiple initial states in an analogous way to the corresponding nfa in Exercise 18, Section 2.2. Does an equivalent dfa with a single initial state always exist?

13. Prove that all finite languages are regular.

14. Show that if \( L \) is regular, so is \( L^R \).

15. Give a simple verbal description of the language accepted by the dfa

\[
\begin{align*}
\delta(q_0, a) &= \{ q_0, q_1 \} \\
\delta(q_1, b) &= \{ q_1, q_2 \} \\
\delta(q_2, a) &= \{ q_2 \} \\
\delta(q_1, \lambda) &= \{ q_1, q_2 \}
\end{align*}
\]

with initial state \( q_0 \) and final state \( q_2 \) into an equivalent dfa.
in Figure 2.16. Use this to find another DFA, equivalent to the given one, but with fewer states.

16. Let $L$ be any language. Define $even (w)$ as the string obtained by extracting from $w$ the letters in even-numbered positions; that is, if

$$w = a_1a_2a_3a_4...,$$

then

$$even (w) = a_2a_4....$$

Corresponding to this, we can define a language

$$even (L) = \{ even (w) : w \in L \}.$$ 

Prove that if $L$ is regular, so is $even (L)$.

17. From a language $L$, we create a new language, $chopleft (L)$, by removing the leftmost symbol of every string in $L$. Specifically,

$$chopleft (L) = \{ w : vw \in L, \text{ with } |v| = 1 \}.$$ 

Show that if $L$ is regular, then $chopleft (L)$ is also regular.

18. From a language $L$, we create a new language $chopright (L)$, by removing the rightmost symbol of every string in $L$. Specifically,

$$chopright (L) = \{ w : wv \in L, \text{ with } |v| = 1 \}.$$ 

Show that if $L$ is regular, then $chopright (L)$ is also regular.
1. Consider the dfa with initial state $q_0$, final state $q_2$ and

$$\delta(q_0, a) = q_2 \quad \delta(q_0, b) = q_2$$
$$\delta(q_1, a) = q_2 \quad \delta(q_1, b) = q_2$$
$$\delta(q_2, a) = q_3 \quad \delta(q_2, b) = q_3$$
$$\delta(q_3, a) = q_3 \quad \delta(q_3, b) = q_1$$

Find a minimal equivalent dfa.

2. Minimize the number of states in the dfa in Figure 2.16.

3. Find minimal dfa’s for the following languages. In each case, prove that the result is minimal.
   (a) $L = \{a^n b^m : n \geq 1, m \geq 2\}$.
   (b) $L = \{a^n b : n \geq 1\} \cup \{b^n a : n \geq 1\}$.
   (c) $L = \{a^n : n \geq 0, n \neq 2\}$.
   (d) $L = \{a^n : n \neq 3 \text{ and } n \neq 4\}$.
   (e) $L = \{a^n : n \mod 3 = 1\} \cup \{a^n : n \mod 5 = 1\}$.

4. Show that the automaton generated by procedure reduce is deterministic.

5. Show that if $L$ is a nonempty language such that any $w$ in $L$ has length at least $n$, then any dfa accepting $L$ must have at least $n + 1$ states.

6. Prove or disprove the following conjecture. If $M = (Q, \Sigma, \delta, q_0, F)$ is a minimal dfa for a regular language $L$, then $\widehat{M} = (Q, \Sigma, \delta, q_0, Q - F)$ is a minimal dfa for $\overline{L}$.

7. Show that indistinguishability is an equivalence relation but that distinguishability is not.
8. Show the explicit steps of the suggested proof of the first part of Theorem 2.4, namely, that $\overline{M}$ is equivalent to the original dfa.

9. Prove the following: If the states $q_a$ and $q_b$ are indistinguishable, and if $q_a$ and $q_c$ are distinguishable, then $q_b$ and $q_c$ must be distinguishable.
expressions. We say the two regular expressions are equivalent if they
denote the same language. One can derive a variety of rules for
simplifying regular expressions (see Exercise 20 in the following exercise
section), but since we have little need for such manipulations we will not
pursue this.

**EXERCISES**

1. Find all strings in $L ((a + bb)^*)$ of length five.
2. Find all strings in $L ((ab + b)^* b (a + ab)^*)$ of length less than four.
3. Find an nfa that accepts the language $L (aa^* (ab + b))$.
4. Find an nfa that accepts the language $L (aa^* (a + b))$.
5. Does the expression $((0 + 1) (0 + 1)^*)^* 00 (0 + 1)^*$ denote the
language in Example 3.5?
6. Show that $r = (1 + 01)^* (0+1^*)^* 00 (0 + 1)^*$ also denotes the language in Example 3.6. Find two other equivalent expressions.
7. Find a regular expression for the set $\{a^n b^m : n \geq 3, m \text{ is odd}\}$.
8. Find a regular expression for the set $\{a^n b^m : (n + m) \text{ is odd}\}$.
9. Give regular expressions for the following languages.
   (a) $L_1 = \{a^n b^m, n \geq 3, m \leq 4\}$.
   (b) $L_2 = \{a^n b^m : n < 4, m \leq 4\}$.
   (c) The complement of $L_1$.
   (d) The complement of $L_2$.
10. What languages do the expressions $(\emptyset^*)^*$ and $a \emptyset$ denote?
11. Give a simple verbal description of the language $L ((aa)^* b (aa)^* + a (aa)^* ba (aa)^*)$.
12. Give a regular expression for $L^R$, where $L$ is the language in Exercise
2.
13. Give a regular expression for \( L = \{a^n b^m : n \geq 2, m \geq 1, nm \geq 3\} \).
14. Find a regular expression for \( L = \{a b^n w : n \geq 4, w \in \{a, b\}^+\} \).
15. Find a regular expression for the complement of the language in Example 3.4.
16. Find a regular expression for \( L = \{v w v : w \in \{a, b\}^*, |v| = 2\} \).
17. Find a regular expression for \( L = \{v w v : w \in \{a, b\}^*, |v| \leq 4\} \).
18. Find a regular expression for

\[ L = \{w \in \{0, 1\}^* : w \text{ has exactly one pair of consecutive zeros}\} \].

19. Give regular expressions for the following languages on \( \Sigma = \{a, b, c\} \):

   (a) All strings containing exactly two \( a \)'s.
   (b) All strings containing no more than three \( a \)'s.
   (c) All strings that contain at least one occurrence of each symbol in \( \Sigma \).

20. Write regular expressions for the following languages on \{0, 1\}:

   (a) All strings ending in 10.
   (b) All strings not ending in 10.
   (c) All strings containing an odd number of 0’s.

21. Find regular expressions for the following languages on \{a, b\}:

   (a) \( L = \{w : |w| \mod 3 \neq 0\} \).
   (b) \( L = \{w : n_a (w) \mod 3 = 0\} \).
   (c) \( L = \{w : n_a (w) \mod 5 > 0\} \).

22. Determine whether or not the following claims are true for all regular expressions \( r_1 \) and \( r_2 \). The symbol \( \equiv \) stands for equivalence of regular expressions in the sense that both expressions denote the same language.
23. Give a general method by which any regular expression \( r \) can be changed into \( \hat{r} \) such that \( (L(r))^R = L(\hat{r}) \).

24. Prove rigorously that the expressions in Example 3.6 do indeed denote the specified language.

25. For the case of a regular expression \( r \) that does not involve \( \lambda \) or \( \emptyset \), give a set of necessary and sufficient conditions that \( r \) must satisfy if \( L(r) \) is to be infinite.

26. Formal languages can be used to describe a variety of two-dimensional figures. Chain-code languages are defined on the alphabet \( \Sigma = \{u, d, r, l\} \), where these symbols stand for unit-length straight lines in the directions up, down, right, and left, respectively. An example of this notation is \( urdl \), which stands for the square with sides of unit length. Draw pictures of the figures denoted by the expressions \( (rd)^* \), \( (urddru)^* \), and \( (ruldr)^* \).

27. In Exercise 27, what are sufficient conditions on the expression so that the picture is a closed contour in the sense that the beginning and ending points are the same? Are these conditions also necessary?

28. Find a regular expression that denotes all bit strings whose value, when interpreted as a binary integer, is greater than or equal to 40.

29. Find a regular expression for all bit strings, with leading bit 1, interpreted as a binary integer, with values not between 10 and 30.
A challenging task in such an application is to write an efficient program for recognizing string patterns. Searching a file for occurrences of a given string is a very simple programming exercise, but here the situation is more complicated. We have to deal with an unlimited number of arbitrarily complicated patterns; furthermore, the patterns are not fixed beforehand, but created at run time. The pattern description is part of the input, so the recognition process must be flexible. To solve this problem, ideas from automata theory are often used.

If the pattern is specified by a regular expression, the pattern-recognition program can take this description and convert it into an equivalent nfa using the construction in Theorem 3.1. Theorem 2.2 may then be used to reduce this to a dfa. This dfa, in the form of a transition table, is effectively the pattern-matching algorithm. All the programmer has to do is to provide a driver that gives the general framework for using the table. In this way we can automatically handle a large number of patterns that are defined at run time.

The efficiency of the program must also be considered. The construction of finite automata from regular expressions using Theorems 2.1 and 3.1 tends to yield automata with many states. If memory space is a problem, the state reduction method described in Section 2.4 is helpful.

**EXERCISES**

1. Use the construction in Theorem 3.1 to find an nfa that accepts the language $L (a^*a + ab)$.
2. Use the construction in Theorem 3.1 to find an nfa that accepts the language $L ((aab)^*ab)$.
3. Use the construction in Theorem 3.1 to find an nfa that accepts the language $L (ab^*aa + bba^*ab)$. 
4. Find an NFA that accepts the complement of the language in Exercise 3.

5. Give an NFA that accepts the language $L \left( (a + b)^* b \ (a + bb)^* \right)$.

6. Find DFA’s that accept the following languages:
   
   (a) $L \left( aa^* + aba^* b^* \right)$.
   (b) $L \left( ab \ (a + ab)^* \ (a + aa) \right)$.
   (c) $L \left( (abab)^* + (aaa^* + b)^* \right)$.
   (d) $L \left( ((aa^*)^* b)^* \right)$.
   (e) $L((aa^*)^* + abb)$.

7. Find DFA’s that accept the following languages:
   
   (a) $L = L \left( ab^* a^* \right) \cup L \left( (ab)^* ba \right)$.
   (b) $L = L \left( ab^* a^* \right) \cap L \left( (b)^* ab \right)$.

8. Find the minimal DFA that accepts $L(ab)^* \cup L(a^* bb^*)$.

9. Find the minimal DFA that accepts $L(a^* bb) \cup L(ab^* ba)$.

10. Consider the following generalized transition graph.

(a) Find an equivalent generalized transition graph with only two states.
(b) What is the language accepted by this graph?

11. What language is accepted by the following generalized transition graph?
12. Find regular expressions for the languages accepted by the following automata.
13. Rework Example 3.11, this time eliminating the state OO first.
14. Show how all the labels in Figure 3.14 were obtained.
15. Find a regular expression for the following languages on \{a, b\}.
   
   (a) \( L = \{w : n_a (w) \text{ and } n_b (w) \text{ are both odd}\}. \)
   
   (b) \( L = \{w : (n_a (w) - n_b (w)) \mod 3 = 2\}. \)
   
   (c) \( L = \{w : (n_a (w) - n_b (w)) \mod 3 = 0\}. \)
(d) $L = \{ w : 2n_a(w) + 3n_b(w) \text{ is even} \}$.

16. Prove that the construction suggested by Figures 3.11 and 3.12 generate equivalent generalized transition graphs.

17. Write a regular expression for the set of all C real numbers.

18. Use the construction in Theorem 3.1 to find nfa’s for $L (a \circ)$ and $L (\circ^*)$. Is the result consistent with the definition of these languages?
FIGURE 3.19

Each gives a complete and unambiguous definition of a regular language. The connection between all these concepts is established by the four theorems in this chapter, as shown in Figure 3.19.

EXERCISES

1. Construct a dfa that accepts the language generated by the grammar

   \[ S \rightarrow abA, \]
   \[ A \rightarrow baB, \]
   \[ B \rightarrow aA \mid bb. \]
2. Construct a dfa that accepts the language generated by the grammar

\[ S \rightarrow abS \mid A, \]
\[ A \rightarrow baB, \]
\[ B \rightarrow aA \mid bb. \]

3. Find a regular grammar that generates the language \( L (aa^* (ab + a)^*) \).


5. Construct right- and left-linear grammars for the language

\[ L = \{ a^n b^m : n \geq 3, m \geq 2 \}. \]

6. Construct a right-linear grammar for the language \( L ((aaab^* ab)^*) \).

7. Find a regular grammar that generates the language on \( \Sigma = \{ a, b \} \)
   consisting of all strings with no more than two \( a \)'s.

8. In Theorem 3.5, prove that \( L \left( \widehat{G} \right) = (L(G))^R \).

9. Suggest a construction by which a left-linear grammar can be
   obtained from an nfa directly.

10. Use the construction suggested by the above exercises to construct a
    left-linear grammar for the nfa below.
11. Find a left-linear grammar for the language $L = ((aaab^*ba)^*)$.

12. Find a regular grammar for the language $L = \{a^n b^m : n + m \text{ is odd}\}$.

13. Find a regular grammar that generates the language

$$L = \{w \in \{a, b\}^* : n_a(w) + 3n_b(w) \text{ is odd}\}.$$

14. Find regular grammars for the following languages on $\{a, b\}$:

(a) $L = \{w : n_a(w) \text{ is even}, n_b(w) \geq 4\}$.

(b) $L = \{w : n_a(w) \text{ and } n_b(w) \text{ are both even}\}$.

(c) $L = \{w : (n_a(w) - n_b(w)) \text{ mod } 3 = 1\}$.

(d) $L = \{w : (n_a(w) - n_b(w)) \text{ mod } 3 \neq 1\}$.

(e) $L = \{w : (n_a(w) - n_b(w)) \text{ mod } 3 \neq 0\}$.

(f) $L = \{w : |n_a(w) - n_b(w)| \text{ is odd}\}$.

15. Show that for every regular language not containing $\lambda$ there exists a right-linear grammar whose productions are restricted to the forms

$$A \rightarrow aB,$$

or

$$A \rightarrow a,$$

where $A, B \in V$, and $a \in T$.

16. Show that any regular grammar $G$ for which $L(G) \cap = \emptyset$ must have at least one production of the form

$$A \rightarrow x$$

where $A \in V$ and $x \in T^*$.

17. Find a regular grammar that generates the set of all real numbers in $\mathbb{C}$. 
18. Let $G_1 = (V_1, \Sigma, S_1, P_1)$ be right-linear and $G_2 = (V_2, \Sigma, S_2, P_2)$ be a left-linear grammar, and assume that $V_1$ and $V_2$ are disjoint. Consider the linear grammar $G = (\{S\} \cup V_1 \cup V_2, \Sigma, S, P)$, where $S$ is not in $V_1 \cup V_2$ and $P = \{S \rightarrow S_1 | S_2\} \cup P_1 \cup P_2$. Show that $L(G)$ is regular.