CHAPTER 3

FINITE AUTOMATA AND REGULAR GRAMMARS

3.1 THE FINITE AUTOMATON

In Chapter 2, we were introduced to a generating scheme—the grammar. Grammars are finite specifications for languages. In this chapter we shall see another method of finitely specifying infinite languages—the recognizer. We shall consider what is undoubtedly the simplest recognizer, called a finite automaton. The finite automaton (fa) cannot define all languages defined by grammars, but we shall show that the languages defined are exactly the type 3 languages. In later chapters, the reader will be introduced to recognizers for type 0, 1, and 2 languages. Here we shall define a finite automaton as a formal system, then give the physical meaning of the definition.

A *finite automaton* M over an alphabet Σ is a system $(K, \Sigma, \delta, q_0, F)$, where K is a finite, nonempty set of *states*, Σ is a finite *input alphabet*, δ is a mapping of $K \times \Sigma$ into K, q_0 in K is the *initial state*, and $F \subseteq K$ is the set of *final states*.

Our model in Fig. 3.1 represents a finite control which reads symbols from a linear input tape in a sequential manner from left to right. The set of states K consists of the states of the finite control. Initially, the finite control is in state q_0 and is scanning the leftmost symbol of a string of symbols in Σ which appear on the input tape. The interpretation of $\delta(q, a) = p$, for q and p in K and a in Σ, is that M, in state q and scanning the input symbol a, moves its input head one cell to the right and goes to state p.

The mapping δ is from $K \times \Sigma$ to K. We can extend δ to domain† $K \times \Sigma^*$ by defining a mapping $\hat{\delta}$ as follows:

$$\hat{\delta}(q, \epsilon) = q$$

$$\hat{\delta}(q, xa) = \delta(\hat{\delta}(q, x), a) \qquad \text{for each } x \text{ in } \Sigma^* \text{ and } a \text{ in } \Sigma.$$

Thus the interpretation of $\hat{\delta}(q, x) = p$ is that M, starting in state q with the string x written on the input tape, will be in state p when the input head moves right from the portion of the input tape containing x. Since δ and $\hat{\delta}$

† The *domain* of a mapping is the set of valid arguments for the mapping. The set of values which the mapping could take is called the *range*.

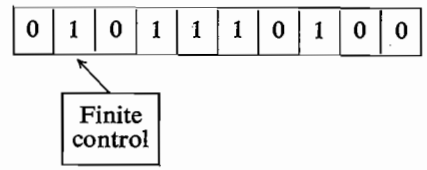


Fig. 3.1. A finite automaton.

agree wherever δ is defined, no confusion will arise if we fail to distinguish between δ and $\hat{\delta}$. Thus, for the remainder of the book, we shall use δ for both δ and $\hat{\delta}$.

A sentence x is said to be *accepted* by M if $\delta(q_0, x) = p$ for some p in F. The set of all x accepted by M is designated $T(M)$. That is,

$$T(M) = \{x | \delta(q, x) \text{ is in } F\}.$$

Any set of strings accepted by a finite automaton is said to be *regular*.

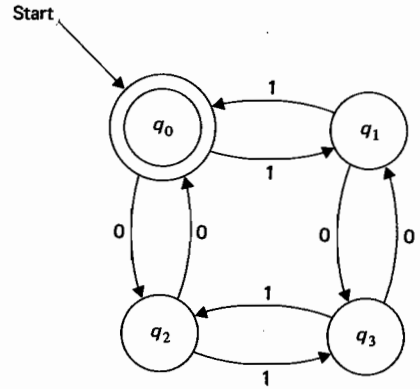
Example 3.1. The specifications for a finite automaton are given in Fig. 3.2(a). A *state diagram* for the automaton is shown in Fig. 3.2(b). The state diagram consists of a node for every state and a directed line from state q to state p with label a (in Σ) if the finite automaton, in state q, scanning the input symbol a, would go to state p. Final states, i.e., states in F, are indicated by a double circle. The initial state is marked by an arrow labeled start.

Consider the state diagram of Fig. 3.2(b). Suppose that 110101 is the input to M. Since $\delta(q_0, 1) = q_1$ and $\delta(q_1, 1) = q_0$, $\delta(q_0, 11) = q_0$. We might comment that thus, 11 is in $T(M)$, but we are interested in 110101. Now $\delta(q_0, 0) = q_2$, so $\delta(q_0, 110) = q_2$. Next $\delta(q_2, 1) = q_3$, so $\delta(q_0, 1101) = q_3$. Finally, $\delta(q_3, 0) = q_1$ and $\delta(q_1, 1) = q_0$, so $\delta(q_0, 110101) = q_0$, and thus 110101 is in $T(M)$. It is easily shown that $T(M)$ is the set of all sentences in $\{0, 1\}^*$ containing both an even number of 0's and an even number of 1's.

$M = (K, \Sigma, \delta, q_0, F)$
$\Sigma = \{0, 1\}$
$K = \{q_0, q_1, q_2, q_3\}$
$F = \{q_0\}$

$\delta(q_0, 0) = q_2$ $\quad \delta(q_0, 1) = q_1$
$\delta(q_1, 0) = q_3$ $\quad \delta(q_1, 1) = q_0$
$\delta(q_2, 0) = q_0$ $\quad \delta(q_2, 1) = q_3$
$\delta(q_3, 0) = q_1$ $\quad \delta(q_3, 1) = q_2$

(a)



(b)

Fig. 3.2. A finite automaton accepting the set of strings with an even number of 0's and an even number of 1's. (a) A finite automaton. (b) State diagram of the finite automaton.

3.2 EQUIVALENCE RELATIONS AND FINITE AUTOMATA

A *binary relation* R on a set S is a set of pairs of elements in S. If (a, b) is in R, then we are accustomed to seeing this fact written as aRb.

Example 3.2. For a familiar example, consider the relation "less than" usually denoted by the symbol $<$ on the set of integers. In the formal sense, this relation is the set: $\{(i, j) \mid i \text{ is less than } j\}$. Thus $3 < 4$, $2 < 17$, etc.

We are going to be concerned with some relations on sets of strings over a finite alphabet.

A binary relation R over a set S is said to be:

1. *reflexive* if for each s in S, sRs,
2. *symmetric* if for s and t in S, sRt implies tRs,
3. *transitive* if for s, t, and u in S, sRt and tRu imply sRu.

A relation which is reflexive, symmetric, and transitive is called an *equivalence relation*. An example of an equivalence relation over the set of positive integers is the relation E, given by: iEj if and only if $|i - j|$ is divisible by 3.

An important property of equivalence relations is that if R is an equivalence relation on the set S then we can divide S into k disjoint subsets, called *equivalence classes*, for some k between 1 and infinity, inclusive, such that aRb if and only if a and b are in the same subset.

The proof is simple. Define $[a]$ to be $\{b|aRb\}$. For any a and b in S, either $[a] = [b]$, or $[a]$ and $[b]$ are disjoint. Otherwise, let c be in $[a]$ and $[b]$, and d be in $[b]$ but not $[a]$. That is, aRc, bRc, and bRd, but not aRd. By symmetry, we have cRb. By transitivity, we can show cRd and aRd. The latter statement is a contradiction. The distinct sets that are $[a]$ for some a in S are the equivalence classes. Clearly, a and b are in the same set if and only if they are equivalent.

Example 3.3. The relation E given by iEj if and only if $|i - j|$ is divisible by 3 divides the set of positive integers into three classes $\{1, 4, 7, 10, \ldots\}$, $\{2, 5, 8, 11, \ldots\}$, and $\{3, 6, 9, 12, \ldots\}$. Any two elements from the same class are equivalent ($1E4$, $3E6$, etc.), and any two elements from different classes fail to satisfy the equivalence relation (not $7E9$, $1E5$, etc.).

The *index* of an equivalence relation is the number of equivalence classes generated. Thus the equivalence relation E has index 3.

Consider the finite automaton of Example 3.1. For x and y in $\{0, 1\}^*$, let (x, y) be in R if and only if $\delta(q_0, x) = \delta(q_0, y)$. The relation R is reflexive, symmetric, and transitive, since "$=$" has these properties, and thus, R is an equivalence relation. R divides the set $\{0, 1\}^*$ into four equivalence classes corresponding to the four states. In addition, if xRy, then $xz\ R\ yz$ for all z in $\{0, 1\}^*$, since

$$\delta(q_0, xz) = \delta(\delta(q_0, x), z) = \delta(\delta(q_0, y), z) = \delta(q_0, yz).$$

Such an equivalence relation is said to be *right invariant*. We see that every finite automaton induces a right invariant equivalence relation defined as R was defined, on its set of input strings. This result is formalized in the following theorem.

Theorem 3.1. The following three statements are equivalent:

1. The set $L \subseteq \Sigma^*$ is accepted by some finite automaton.
2. L is the union of some of the equivalence classes of a right invariant equivalence relation of finite index.
3. Let equivalence relation R be defined by: xRy if and only if for all z in Σ^*, xz is in L exactly when yz is in L. Then R is of finite index.

Proof. (1) $\Rightarrow$ (2). Assume that L is accepted by some fa $M = (K, \Sigma, \delta, q_0, F)$. Let R' be the equivalence relation $xR'y$ if and only if $\delta(q_0, x) = \delta(q_0, y)$. R' is right invariant since, for any z, if $\delta(q_0, x) = \delta(q_0, y)$, then

$$\delta(q_0, xz) = \delta(q_0, yz).$$

The index of R' is finite since the index is at most the number of states in K. Furthermore, L is the union of those equivalence classes which include an element x such that $\delta(q_0, x)$ is in F.

(2) $\Rightarrow$ (3). We show that any equivalence relation R' satisfying (2) is a refinement of R; that is, every equivalence class of R' is entirely contained in some equivalence class of R. Thus the index of R cannot be greater than the index of R' and so is finite. Assume that $xR'y$. Then since R' is right invariant, for each z in Σ^*, $xzR'yz$, and thus yz is in L if and only if xz is in L. Thus xRy, and hence, the equivalence class of x in R' is contained in the equivalence class of x in R. We conclude that each equivalence class of R' is contained within some equivalence class of R.

(3) $\Rightarrow$ (1). Assume that xRy. Then for each w and z in Σ^*, xwz is in L if and only if ywz is in L. Thus $xwRyw$, and R is right invariant. Now let K' be the finite set of equivalence classes of R and $[x]$ the element of K' containing x. Define $\delta'([x], a) = [xa]$. The definition is consistent, since R is right invariant. Let $q_0' = [\epsilon]$ and let $F' = \{[x] \mid x \in L\}$. The finite automaton $M' = (K', \Sigma, \delta', q_0', F')$ accepts L since $\delta'(q_0', x) = [x]$, and thus x is in $T(M')$ if and only if $[x]$ is in F'.

Theorem 3.2. The minimum state automaton accepting L is unique up to an isomorphism (i.e., a renaming of the states) and is given by M' of Theorem 3.1.

Proof. In the proof of Theorem 3.1 we saw that any fa $M = (K, \Sigma, \delta, q_0, F)$ accepting L defines an equivalence relation which is a refinement of R. Thus the number of states of M is greater than or equal to the number of states of M' of Theorem 3.1. If equality holds, then each of the states of M can be identified with one of the states of M'. That is, let q be a state of M. There

must be some x in Σ^*, such that $\delta(q_0, x) = q$, otherwise q could be removed from K, and a smaller automaton found. Identify q with the state $\delta'(q_0', x)$, of M'. This identification will be consistent. If $\delta(q_0, x) = \delta(q_0, y) = q$, then, by Theorem 3.1, x and y are in the same equivalence class of R. Thus $\delta'(q_0', x) = \delta'(q_0', y)$.

3.3 NONDETERMINISTIC FINITE AUTOMATA

We now introduce the notion of a nondeterministic finite automaton. It will turn out that any set accepted by a nondeterministic finite automaton can also be accepted by a deterministic finite automaton.

However, the nondeterministic finite automaton is a useful concept in proving theorems. Also, the concept of a nondeterministic device is not an easy one to grasp. It is well to begin with a simple device. Later we deal with nondeterministic devices that are not equivalent to their deterministic counterparts. It is hoped that the study of nondeterministic finite automata will help in the understanding of those devices.

A *nondeterministic finite automaton* M is a system $(K, \Sigma, \delta, q_0, F)$, where K is a finite nonempty set of states, Σ the finite input alphabet, δ is a mapping of $K \times \Sigma$ into subsets of K, q_0 in K is the initial state, and $F \subseteq K$ is the set of final states.

The important difference between the deterministic and nondeterministic case is that $\delta(q, a)$ is a (possibly empty) set of states rather than a single state. The interpretation of $\delta(q, a) = \{p_1, p_2, \ldots, p_k\}$ is that M, in state q, scanning a on its input tape, moves its head one cell to the right and chooses any one of $p_1, p_2, \ldots, p_k$ as the next state.

The mapping δ can be extended to domain $K \times \Sigma^*$ by defining

$$\delta(q, \epsilon) = \{q\} \qquad \text{and} \qquad \delta(q, xa) = \bigcup_{p \text{ in } \delta(q,x)} \delta(p, a),$$

for each x in Σ^*, and a in Σ.

The mapping δ can be further extended to domain $2^K \times \Sigma^*$† by defining

$$\delta(\{p_1, p_2, \ldots, p_k\}, x) = \bigcup_{i=1}^{k} \delta(p_i, x).$$

A sentence x is *accepted* by M if there is a state p in both F and $\delta(q_0, x)$. The set of all x accepted by M is denoted $T(M)$.

Example 3.4. A nondeterministic fa which accepts the set of all sentences with either two consecutive 0's or two consecutive 1's is given in Fig. 3.3. The fa will make many choices upon reading an input string. Thus, suppose that 010110 is the input. After reading the first 0, M may stay in state q_0 or go to q_3. Next, with a 1 input, M can go nowhere from q_3, but from q_0 can

† 2^K, for any set K, denotes the *power set* or set of all subsets of K.

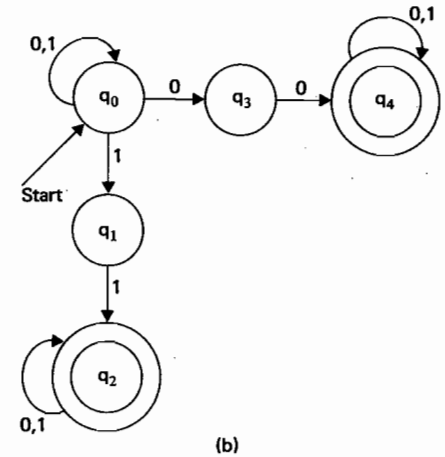
go to q_0 or q_1. Similarly, by the time the fourth input symbol is read, M can still be in only q_0 or q_1. When the fifth symbol, a 1, is read, M can go from q_1 to q_2 and from q_0 to q_0 or q_1. Thus M may be in state q_0, q_1, or q_2. Since there is a sequence of states leading to q_2, 01011 is accepted. Likewise, after the sixth symbol is read, M can be in state q_0, q_2, or q_3. Thus 010110 is also accepted.

$M = (\{q_0, q_1, q_2, q_3, q_4\}, \{0, 1\}, \delta, q_0, \{q_2, q_4\})$

$\delta(q_0, 0) = \{q_0, q_3\}$;
$\delta(q_1, 0) = \varphi$;
$\delta(q_2, 0) = \{q_2\}$;
$\delta(q_3, 0) = \{q_4\}$;
$\delta(q_4, 0) = \{q_4\}$;

$\delta(q_0, 1) = \{q_0, q_1\}$.
$\delta(q_1, 1) = \{q_2\}$.
$\delta(q_2, 1) = \{q_2\}$.
$\delta(q_3, 1) = \varphi$.
$\delta(q_4, 1) = \{q_4\}$.

(a)



(b)

Fig. 3.3. A nondeterministic finite automaton which accepts the set of all sentences containing either two consecutive 0's or two consecutive 1's. (a) Specification. (b) State diagram.

Theorem 3.3. Let L be a set accepted by a nondeterministic finite automaton. Then there exists a deterministic finite automaton that accepts L.

Proof. Let $M = (K, \Sigma, \delta, q_0, F)$ be a nondeterministic fa accepting L. Define a deterministic fa, $M' = (K', \Sigma, \delta', q_0', F')$ as follows. The states of M' are all the subsets of the set of states of M. That is, $K' = 2^K$. M' will keep track of all the states M could be in at any given time. F' is the set of all states in K' containing a state of F. An element of K' will be denoted by $[q_1, q_2, \ldots, q_i]$, where $q_1, q_2, \ldots, q_i$ are in K. Note that $q_0' = [q_0]$.

We define

$$\delta'([q_1, q_2, \ldots, q_i], a) = [p_1, p_2, \ldots, p_j]$$

if and only if

$$\delta(\{q_1, q_2, \ldots, q_i\}, a) = \{p_1, p_2, \ldots, p_j\}.$$

That is, δ' applied to an element Q of K' is computed by applying δ to each state of K represented by $Q = [q_1, q_2, \ldots, q_i]$. On applying δ to each of $q_1, q_2, \ldots, q_i$ and taking the union, we get some new set of states, $p_1, p_2, \ldots, p_j$. This new set of states has a representative, $[p_1, p_2, \ldots, p_j]$ in K', and that element is the value of $\delta'([q_1, q_2, \ldots, q_i], a)$.

It is easy to show by induction on the length of the input string x that

$$\delta'(q_0', x) = [q_1, q_2, \ldots, q_i]$$

if and only if

$$\delta(q_0, x) = \{q_1, q_2, \ldots, q_i\}.$$

The result is trivial for $|x| = 0$, since $q_0' = [q_0]$. Suppose that it is true for $|x| \leqq l$. Then, for a in Σ,

$$\delta'(q_0', xa) = \delta'(\delta'(q_0', x), a).$$

By the inductive hypothesis,

$$\delta'(q_0', x) = [p_1, p_2, \ldots, p_j]$$

if and only if

$$\delta(q_0, x) = \{p_1, p_2, \ldots, p_j\}.$$

But by definition,

$$\delta'([p_1, p_2, \ldots, p_j], a) = [r_1, r_2, \ldots, r_k]$$

if and only if

$$\delta(\{p_1, p_2, \ldots, p_j\}, a) = \{r_1, r_2, \ldots, r_k\}.$$

Thus,

$$\delta'(q_0', xa) = [r_1, r_2, \ldots, r_k]$$

if and only if

$$\delta(q_0, xa) = \{r_1, r_2, \ldots, r_k\}.$$

To complete the proof, we have only to add that $\delta'(q_0', x)$ is in F' exactly when $\delta(q_0, x)$ contains a state of K which is in F. Thus $T(M) = T(M')$.

Since the deterministic and nondeterministic finite automata accept the same sets, we shall not distinguish between them unless it becomes necessary, but shall simply refer to both as finite automata.

Example 3.5. Let $M = (\{q_0, q_1\}, \{0, 1\}, \delta, q_0, \{q_1\})$ be a nondeterministic fa, where:

$$\delta(q_0, 0) = \{q_0, q_1\} \qquad \delta(q_0, 1) = \{q_1\} \qquad \delta(q_1, 0) = \varphi \qquad \delta(q_1, 1) = \{q_0, q_1\}.$$

We can construct a deterministic fa, $M' = (K, \{0, 1\}, \delta', [q_0], F)$, accepting $T(M)$ as follows. K consists of all subsets of $\{q_0, q_1\}$. We denote the elements of K by $[q_0]$, $[q_1]$, $[q_0, q_1]$ and φ. Since $\delta(q_0, 0) = \{q_0, q_1\}$,

$$\delta'([q_0], 0) = [q_0, q_1].$$

Likewise,

$$\delta'([q_0], 1) = [q_1], \ \delta'([q_1], 0) = \varphi \qquad \text{and} \qquad \delta'([q_1], 1) = [q_0, q_1].$$

Naturally, $\delta'(\varphi, 0) = \delta'(\varphi, 1) = \varphi$. Lastly,

$$\delta'([q_0, q_1], 0) = [q_0, q_1],$$

since

$$\delta(\{q_0, q_1\}, 0) = \delta(q_0, 0) \cup \delta(q_1, 0) = \{q_0, q_1\} \cup \varphi = \{q_0, q_1\};$$

and

$$\delta'([q_0, q_1], 1) = [q_0, q_1],$$

since

$$\delta(\{q_0, q_1\}, 1) = \delta(q_0, 1) \cup \delta(q_1, 1) = \{q_1\} \cup \{q_0, q_1\} = \{q_0, q_1\}.$$

The set F of final states is $\{[q_1], [q_0, q_1]\}$.

3.4 FINITE AUTOMATA AND TYPE 3 LANGUAGES

We now turn to the relationship between the languages generated by type 3 grammars and the sets accepted by finite automata.

Theorem 3.4. Let $G = (V_N, V_T, P, S)$ be a type 3 grammar. Then there exists a finite automaton $M = (K, V_T, \delta, S, F)$ with $T(M) = L(G)$.

Proof. M will be a nondeterministic fa. The states of M are the variables of G, plus an additional state A, not in V_N. Thus, $K = V_N \cup \{A\}$. The initial state of M is S. If P contains the production $S \to \epsilon$, then $F = \{S, A\}$. Otherwise, $F = \{A\}$. Recall that S will not appear on the right of any production if $S \to \epsilon$ is in P. The state A is in $\delta(B, a)$ if $B \to a$ is in P. In addition, $\delta(B, a)$ contains all C such that $B \to aC$ is in P. $\delta(A, a) = \varphi$ for each a in V_T.

The fa M, when accepting a sentence x, simulates a derivation of x by the grammar G. We shall show that $T(M) = L(G)$. Let $x = a_1 a_2 \ldots a_n$ be in $L(G)$, $n \geqq 1$. Then

$$S \Rightarrow a_1 A_1 \Rightarrow \cdots \Rightarrow a_1 a_2 \ldots a_{n-1} A_{n-1} \Rightarrow a_1 a_2 \ldots a_{n-1} a_n$$

for some sequence of variables $A_1, A_2, \ldots, A_{n-1}$. From the definition of δ, we can see that $\delta(S, a_1)$ contains A_1, that $\delta(A_1, a_2)$ contains A_2, etc., and that $\delta(A_{n-1}, a_n)$ contains A. Thus x is in $T(M)$, since $\delta(S, x)$ contains A, and A is in F. If ϵ is in $L(G)$, then S is in F, so ϵ is in $T(M)$.

Likewise, if x is in $T(M)$, $|x| \geqq 1$, then there exists a sequence of states $S, A_1, A_2, \ldots, A_{n-1}, A$ such that $\delta(S, a_1)$ contains A_1, $\delta(A_1, a_2)$ contains A_2, and so forth. Thus, P contains rules $S \to a_1 A_1$, $A_1 \to a_2 A_2, \ldots$ and $A_{n-1} \to a_n$. Therefore, $S \Rightarrow a_1 A_1 \Rightarrow a_1 a_2 A_2 \Rightarrow \cdots \Rightarrow a_1 a_2 \ldots a_{n-1} A_{n-1} \Rightarrow a_1 a_2 \ldots a_n$ is a derivation in G and x is in $L(G)$. If ϵ is in $T(M)$, then S is in F, so $S \to \epsilon$ is a production in P, and ϵ is in $L(G)$.

Theorem 3.5. Given a finite automaton M, there exists a type 3 grammar G, such that $L(G) = T(M)$.