CS 4410

Automata, Computability, and Formal Language

Dr. Xuejun Liang

Chapter 2

Finite Automata

1. Deterministic Finite Accepters
    - Deterministic Accepters and Transition Graphs
    - Languages and Dfas
    - Regular Language
2. Nondeterministic Finite Accepters
    - Definition of a Nondeterministic Accepter
    - Why Nondeterministic
3. Equivalence of Deterministic and Nondeterministic Finite Accepters

Learning Objectives

At the conclusion of the chapter, the student will be able to:

- Describe the components of a deterministic finite accepter (dfa)
- State whether an input string is accepted by a dfa
- Describe the language accepted by a dfa
- Construct a dfa to accept a specific language
- Show that a particular language is regular
- Describe the differences between deterministic and nondeterministic finite automata (nfa)
- State whether an input string is accepted by a nfa
- Construct a nfa to accept a specific language
- Transform an arbitrary nfa to an equivalent dfa

Deterministic Finite Accepters

Definition 2.1

A deterministic finite accepter or dfa is defined by the quintuple

$$M = (Q, \Sigma, \delta, q_0, F)$$

where Q is a finite set of internal states,

Σ is a finite set of symbols called the input alphabet,

$\delta: Q \times \Sigma \rightarrow Q$ is a total function called the transition function,

$q_0 \in Q$ is the initial state,

$F \subseteq Q$ is a set of final states.

Example 2.1 $M = (\{q_0, q_1, q_2\}, \{0,1\}, \delta, q_0, \{q_1\})$, where δ is given by

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

	0	**1**
q_0	q_0	q_1
q_1	q_0	q_2
q_2	q_2	q_1

Transition Graphs

A DFA can be visualized with a *Transition Graph*

Transition Graph of a dfa $M = (Q, \Sigma, \delta, q_0, F)$

Vertex labeled with q_i: state $q_i \in Q$,

Edge from q_i to q_j labeled with a: transition $\delta(q_i, a) = q_j$.

The graph below represents the dfa in Example 2.1:

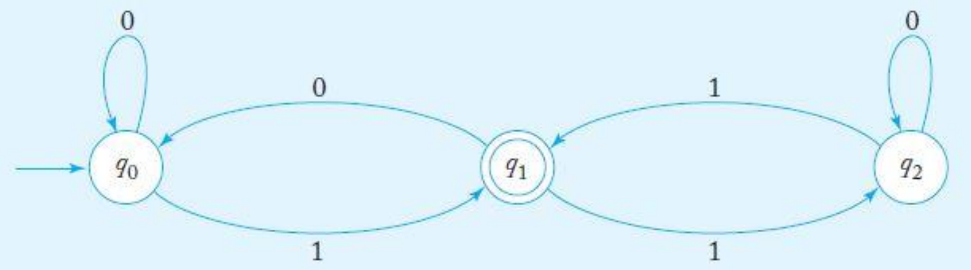


FIGURE 2.1

	0	1
q_0	q_0	q_1
q_1	q_0	q_2
q_2	q_2	q_1

Processing Input with a DFA

- A DFA starts by processing the leftmost input symbol with its control in state q_0. The transition function determines the next state, based on current state and input symbol
- The DFA continues processing input symbols until the end of the input string is reached
- The input string is *accepted* if the automaton is in a final state after the last symbol is processed. Otherwise, the string is *rejected*.
- For example,

  the dfa in example 2.1

  accepts the string 111

  but rejects the string 110

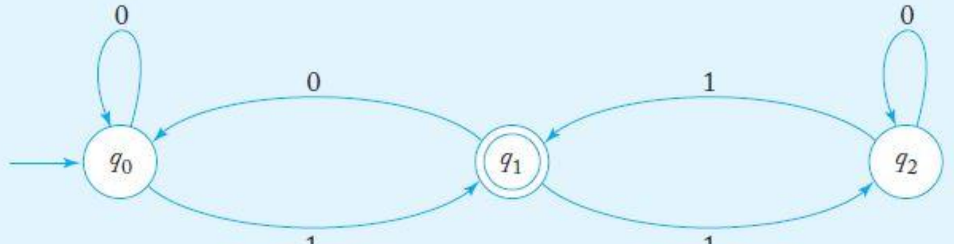


FIGURE 2.1

Extended Transition Function

- For a given dfa, the extended transition function δ^* accepts as input a dfa state and an input string. The value of the function is the state of the automaton after the string is processed.
- Formally, the extended transition function

  $$\delta^* : Q \times \Sigma^* \rightarrow Q$$

  can be recursively defined by

  $$\delta^*(q, \lambda) = q$$

  $$\delta^*(q, wa) = \delta(\delta^*(q, w), a)$$

- Sample values of δ^* for the dfa in example 2.1,

  $\delta^*(q_0, 1001) = q_1$

  $\delta^*(q_1, 000) = q_0$

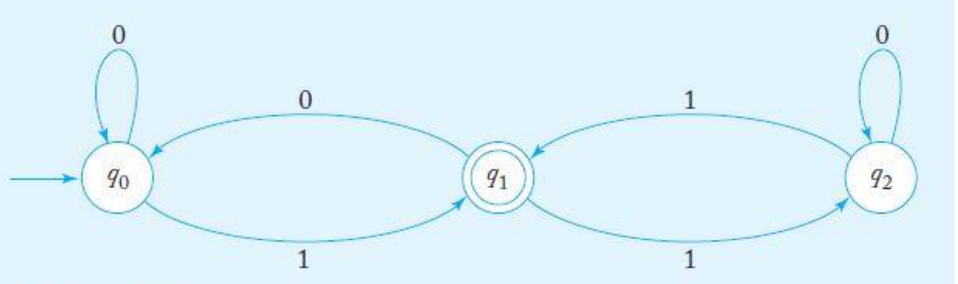


FIGURE 2.1

Languages Accepted by a DFA

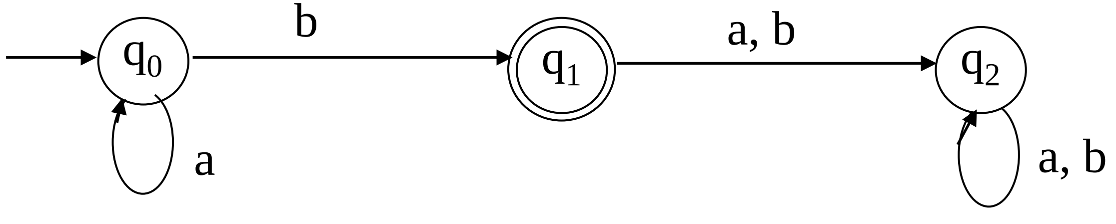
The language accepted by a dfa M is the set of all strings accepted by M. More precisely, the set of all strings w such that $\delta^*(q_0, w)$ results in a final state

Definition 2.2

The language accepted by a dfa M = (Q, Σ, δ, q_0, F) is the set of all strings on Σ accepted by M. In formal notation

$$L(M) = \{w \in \Sigma^* : \delta^*(q_0, w) \in F\}.$$

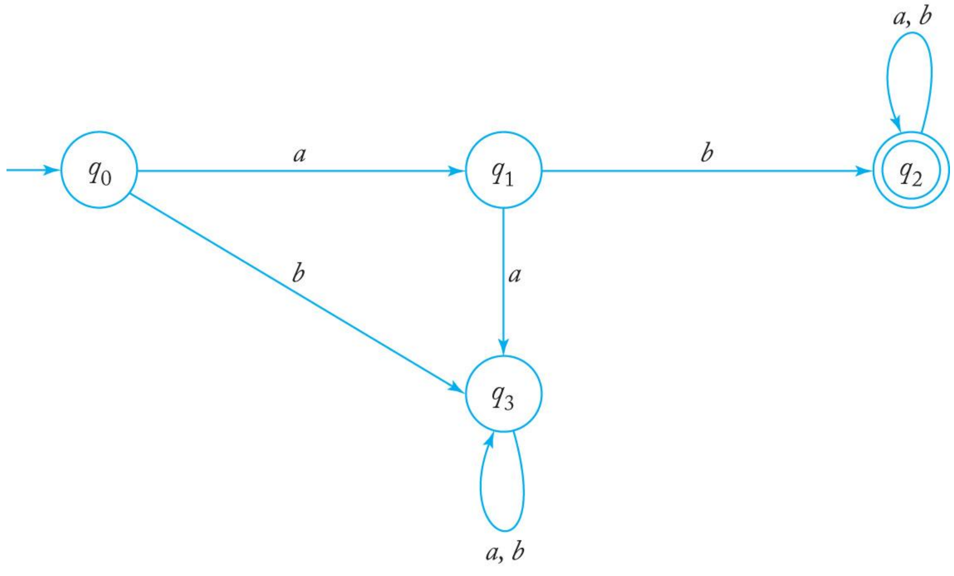
Example 2.2 M is given as below, L(M) = ?



Find a DFA to Accept a Language

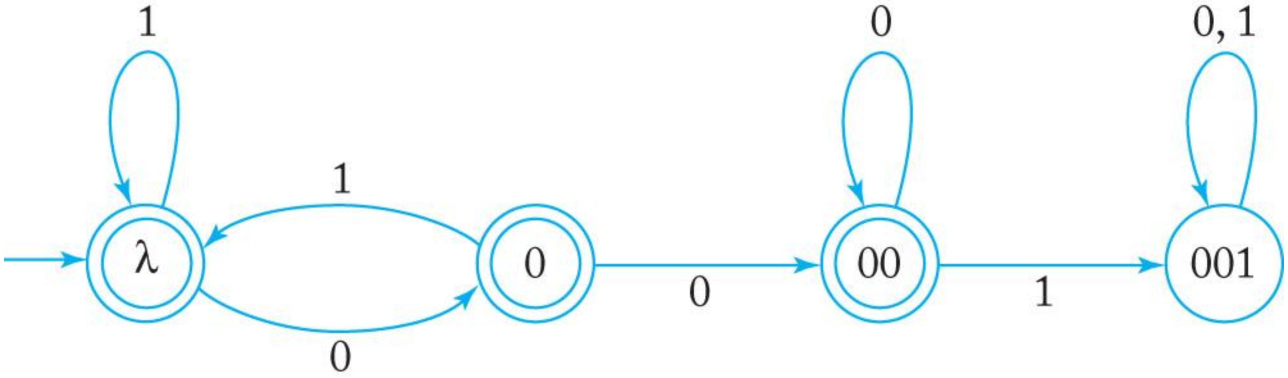
Theorem 2.1 Let M = (Q, Σ, δ, q_0, F) a dfa, and let G_M be its associated transition graph. Then $\delta^*(q_i, w) = q_j$ if and only if there is in G_M a walk with label w from q_i to q_j.

Example 2.3 Find a dfa that accepts all strings on Σ = {a, b} starting with the prefix ab.



Find a DFA to Accept a Language

Example 2.4 Find a dfa that accepts all strings on Σ = {0, 1}, except those containing the substring 001.



Regular Languages

Definition 2.3

A language L is called regular if and only if there exists some deterministic finite accepter M such that L = L(M).

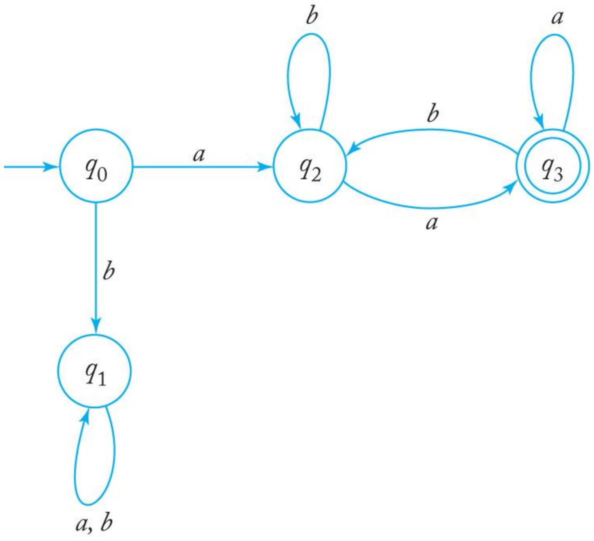
Therefore, to show that a language is regular, one must construct a DFA to accept it.

Example 2.5

Show that the language

L = $\{awa : w \in \{a, b\}^*\}$

is regular.



Regular Languages

Example 2.6: Let L={awa : $w \in \{a, b\}^*$}. Show that L^2 is regular.