## 2.2 Nondeterministic Finite Accepters

[1] 

### Definition of a Nondeterministic Accepter

Nondeterminism means a choice of moves for an automaton. Rather than prescribing a unique move in each situation, we allow a set of possible moves. Formally, we achieve this by defining the transition function so that its range is a set of possible states.

## Definition 2.4

A **nondeterministic finite accepter** or **nfa** is defined by the quintuple

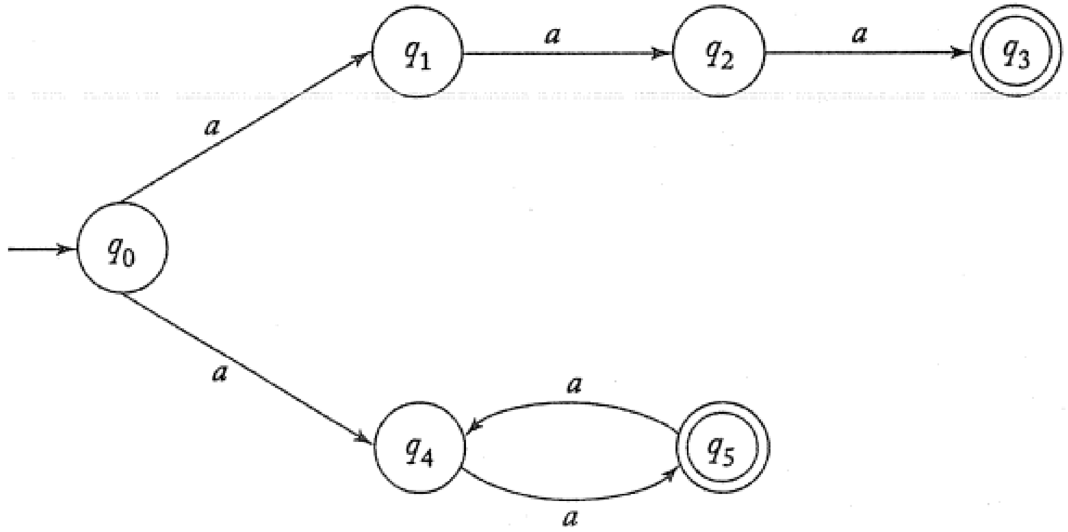
$$M = (Q, \Sigma, \delta, q_0, F),$$

where $Q, \Sigma, q_0, F$ are defined as for deterministic finite accepters, but

$$\delta : Q \times (\Sigma \cup \{\lambda\}) \to 2^Q.$$

**Example 2.7** Consider the transition graph in Figure 2.8. It describes a nondeterministic accepter since there are two transitions labeled $a$ out of $q_0$.

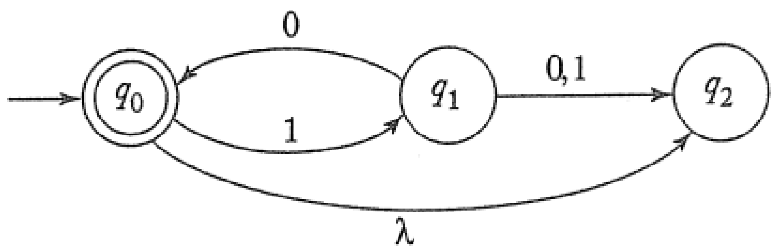
**Figure 2.8**



**Example 2.8**

A nondeterministic automaton is shown in Figure 2.9. It is nondeterministic not only because several edges with the same label originate from one vertex, but also because it has a $\lambda$-transition. Some transitions, such as $\delta(q_2, 0)$, are unspecified in the graph. This is to be interpreted as a transition to the empty set, that is, $\delta(q_2, 0) = \varnothing$. The automaton accepts strings $\lambda$, 1010, and 101010, but not 110 and 10100. Note that for 10 there are two alternative walks, one leading to $q_0$, the other to $q_2$. Even though $q_2$ is not a final state, the string is accepted because one walk leads to a final state.

**Figure 2.9**



Again, the transition function can be extended so its second argument is a string. We require of the extended transition function $\delta^*$ that if

$$\delta^*(q_i, w) = Q_j,$$

then $Q_j$ is the set of all possible states the automaton may be in, having started in state $q_i$ and having read $w$. A recursive definition of $\delta^*$, analogous to (2.1) and (2.2), is possible, but not particularly enlightening. A more easily appreciated definition can be made through transition graphs.

---

**Definition 2.5**

For an nfa, the extended transition function is defined so that $\delta^*(q_i, w)$ contains $q_j$ if and only if there is a walk in the transition graph from $q_i$ to $q_j$ labeled $w$. This holds for all $q_i, q_j \in Q$, and $w \in \Sigma^*$.

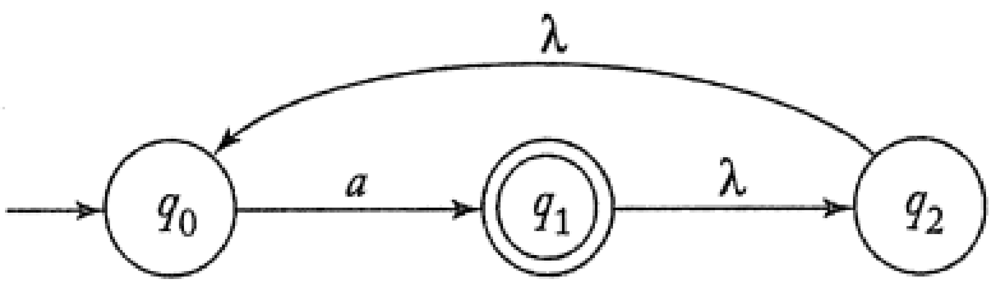
---

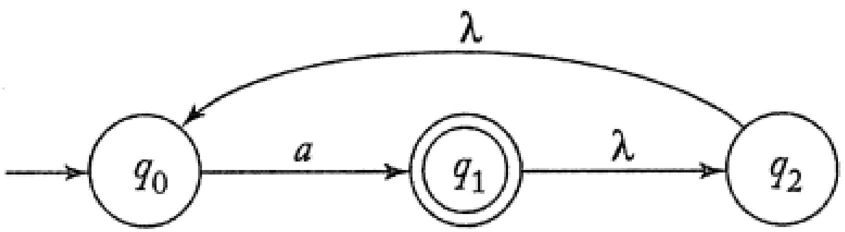
**Example 2.9** Figure 2.10 represents an nfa. It has several $\lambda$-transitions and some undefined transitions such as $\delta(q_2, a)$.

Suppose we want to find $\delta^*(q_1, a)$ and $\delta^*(q_2, \lambda)$. There is a walk labeled $a$ involving two $\lambda$-transitions from $q_1$ to itself. By using some of the $\lambda$-edges twice, we see that there are also walks involving $\lambda$-transitions to $q_0$ and $q_2$. Thus,

$$\delta^*(q_1, a) = \{q_0, q_1, q_2\}.$$

**Figure 2.10**






Since there is a $\lambda$-edge between $q_2$ and $q_0$, we have immediately that $\delta^*(q_2, \lambda)$ contains $q_0$. Also, since any state can be reached from itself by making no move, and consequently using no input symbol, $\delta^*(q_2, \lambda)$ also contains $q_2$. Therefore,

$$\delta^*(q_2, \lambda) = \{q_0, q_2\}.$$

Using as many $\lambda$-transitions as needed, you can also check that

$$\delta^*(q_2, aa) = \{q_0, q_1, q_2\}.$$

**Definition 2.6**

The language $L$ accepted by an nfa $M = (Q, \Sigma, \delta, q_0, F)$ is defined as the set of all strings accepted in the above sense. Formally,

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

In words, the language consists of all strings $w$ for which there is a walk labeled $w$ from the initial vertex of the transition graph to some final vertex.

**Example 2.10** What is the language accepted by the automaton in Figure 2.9? It is easy to see from the graph that the only way the nfa can stop in a final state is if the input is either a repetition of the string 10 or the empty string. Therefore, the automaton accepts the language $L = \{(10)^n : n \geq 0\}$.

What happens when this automaton is presented with the string $w = 110$? After reading the prefix 11, the automaton finds itself in state $q_2$, with the transition $\delta(q_2, 0)$ undefined. We call such a situation a **dead configuration**, and we can visualize it as the automaton simply stopping without further action. But we must always keep in mind that such visualizations are imprecise and carry with them some danger of misinterpretation. What we can say precisely is that

$$\delta^*(q_0, 110) = \varnothing.$$

Thus, no final state can be reached by processing $w = 110$, and hence the string is not accepted.

## [2] Why Nondeterminism?