MCS-265: The Theory of Computation Handout #R1
San Skulrattanakulchai
Gustavus Adolphus College February 13, 2014

Finite Automata

Sipser Ch 1: p31–44, p47–54

A *deterministic finite automaton (DFA)* M is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$, where

1. Q is a finite set of *states*,
2. Σ is an *alphabet*,
3. $\delta : Q \times \Sigma \to Q$ is a *transition function*,
4. $q_0 \in Q$ is the *start state*, and
5. $F \subseteq Q$ is the set of *accept* or *final states.*

A string $w \in \Sigma^*$ of length n is *accepted* by M if and only if there exists a sequence of states $r_0, r_1, \ldots, r_n$ such that

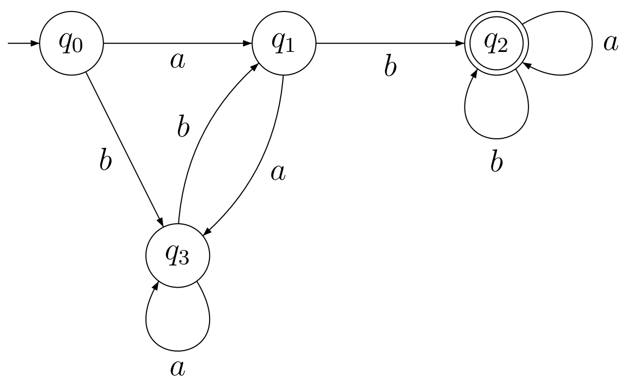
1. $r_0 = q_0$,
2. $r_n \in F$, and
3. $\delta(r_i, w_{i+1}) = r_{i+1}$ for $i = 0, 1, \ldots, n-1$.

The set of all strings accepted by M is the *language recognized by* M, written $L(M)$, i.e., $L(M) = \{w \in \Sigma^* : M \text{ accepts } w\}$.

A DFA can be represented pictorially by a *state diagram*. A *state diagram* is basically a (multi)digraph where vertices represent states, and edges correspond to state transitions (each edge is labelled by a symbol causing that transition). A start state is indicated by a an arrow originating from nowhere pointing into it. A final state is indicated by double circle.

In terms of diagram, a string w of length n is accepted by the DFA if and only if there exists a directed path that begins from the start state and ends in some final state such that the sequence of labels on the edges of the path is $w_1, w_2, \ldots, w_n$.

Example:



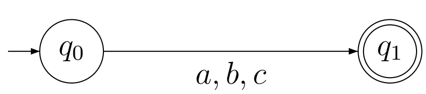
This diagram represents the DFA $M = (Q, \Sigma, \delta, q_0, F)$, where

1. $Q = \{q_0, q_1, q_2, q_3\}$,
2. $\Sigma = \{a, b\}$,
3. δ is given by the following table

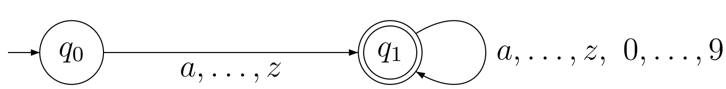
δ	a	b
q_0	q_1	q_3
q_1	q_3	q_2
q_2	q_2	q_2
q_3	q_3	q_1

4. q_0 is the start state, and
5. $F = \{q_2\}$.

Note: We use comma-separated list of symbols as a shorthand for parallel edges, each labelled by a symbol in the list.



We may even use ellipsis to stand for understood omitted symbols.



Sipser also uses Σ to represent a list of all symbols from the alphabet.

A *nondeterministic finite automaton (NFA)* M is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$, where

1. Q is a finite set of *states*,
2. Σ is an *alphabet*,
3. $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow 2^Q$ is the *transition function*,
4. $q_0 \in Q$ is the *start state*, and
5. $F \subseteq Q$ is the set of *accept* or *final states.*

A string $w \in \Sigma^*$ of length n is *accepted* by M if and only if we can write $w = y_1 y_2 \ldots y_m$, where each $y_i = \varepsilon$ or $y_i \in \Sigma$, and there exists a sequence of states $r_0, r_1, \ldots, r_m$ such that

1. $r_0 = q_0$,
2. $r_m \in F$, and
3. $r_{i+1} \in \delta(r_i, y_{i+1})$ for $i = 0, 1, \ldots, m-1$.

The set of all strings accepted by M is the language $L(M)$ *recognized by* M, i.e., $L(M) = \{w \in \Sigma^* : M \text{ accepts } w\}$. Note that $m \neq n$ is possible. (Why?)

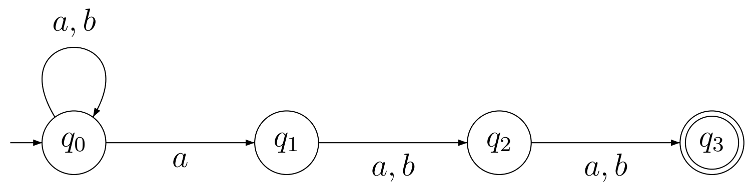
Notes:

1. In a state diagram for a DFA where Σ has n symbols, every state has exactly n edges leaving it, one edge per symbol in Σ. In a state diagram for an NFA, on the other hand, some state may have more or fewer than n edges leaving it. Moreover, two edges leaving the same state may have the same label, and some edge may be labelled with ε.

2. If a string w is accepted by a DFA, then there exists a unique path from the start state to a final state that traces out w. On the other hand, if a string w is accepted by a NFA, then there exists at least one path (may be more) from the start state to some final state that traces out w.

Example:

Let L_3 be the language of all strings over $\Sigma = \{a, b\}$ whose 3rd symbol from the right end is a. Here is an NFA recognizing L_3.



A DFA recognizing L_3 will have to memorize the last 3 symbols seen, i.e., it needs 2^3 states (in general, $|\Sigma|^3$ states).

Exercise. Design a DFA that recognizes L_3.