Lecture 4: Finite Automata

Deterministic finite automata (DFA):

A language recognizer.

1. An *input tape* divided into squares: for string input.
2. A machine which can be in a finite number of states — *finite control*.
3. A movable *reading head*.

Figure 2-1

4. Initialization:
   - Reading head at the leftmost square.
   - Finite control in an *initial state*.
5. After reading a symbol,
   - Reading head moves one square to the right.
   - Finite control enters a new state (depending upon the current state and current symbol).
6. After reading all symbols,
   - If finite control is in a *final state*, the input string is *accepted*.
   - Otherwise, the input string is not *accepted*.

Deterministic finite automata

To specify a DFA, we need to specify

1. K — a finite set of states (of the machine),
2. Σ — an alphabet (that the machine can recognize),
3. $s \in K$ — the initial state,
4. $F \subseteq K$ — the set of final states,
5. δ — the transition function:
   - $\delta : K \times \Sigma \to K$,
   - If q is the current state and symbol σ is read, the machine enters state $\delta(q, \sigma)$.

A DFA is a quintuple $M = (K, \Sigma, \delta, s, F)$.

Configuration:

- Status of machine, reading head, and input tape.
- (current state of machine q, unread part of input string).
- A element of $K \times \Sigma^*$.

Configuration displayed in Figure 2-1 is (,).

Figure 2-1

Deterministic finite automata

Given DFA M. One step of computation:

- (q, w) — current configuration, $w = \sigma w'$.
- (q', w') — the next configuration, $q' = \delta(q, \sigma)$.
- (q, w) **yields** (q', w') in **one step**;

$$(q, w) \vdash_M (q', w').$$

- $\vdash_M$: $K \times \Sigma^+ \to K \times \Sigma^*$.

Several steps of computations:

- If $(q, w) = (q_1, w_1) \vdash_M (q_2, w_2) \vdash_M \ldots \vdash_M (q_n, w_n) = (q', w')$, we say (q, w) **yields** (q', w'),

$$(q, w) \vdash_M^* (q', w').$$

- Let $n = 0$,

$$(q, w) \vdash_M^* (q, w).$$

- $\vdash_M^*$ is the reflexive and transitive closure of $\vdash_M$.

Acceptance:

- A string w is **accepted** by M if

$$(s, w) \vdash_M^* (q, e), \text{ for some} q \in F$$

- The **language accepted by M**:

$$L(M) = \{w : w \text{ accepted by } M\}.$$

Example: M

$K = \{q_0, q_1\}$, $s = q_0$, $F = \{q_0\}$

$\Sigma = \{a, b\}$

q	σ	$\delta(q, \sigma)$
q_0	a	q_0
q_0	b	q_1
q_1	a	q_1
q_1	b	q_0

Symbol "b" flips the state, while "a" doesn't.

$$\begin{aligned}(q_0, aabba) &\vdash_M (q_0, abba)\\ &\vdash_M (q_0, bba)\\ &\vdash_M (q_1, ba)\\ &\vdash_M (q_0, a)\\ &\vdash_M (q_0, e)\end{aligned}$$

- So, $(q_0, aabba) \vdash_M^* (q_0, e)$
- M accepts $aabba$.

$L(M) = \{w \in \{a, b\}^* : w \text{ contains even number of } b\text{'}\}$.

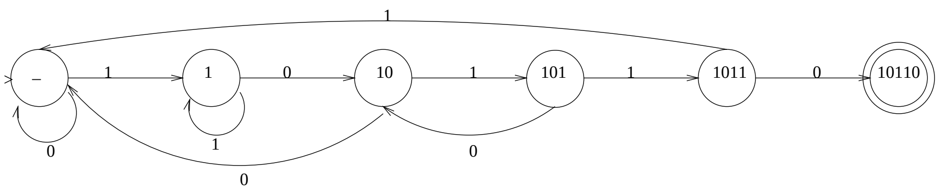
State diagram: pictorial representation of DFA.

- Nodes represent states,
  - initial state indicated by $>$,
  - final states indicated by double circles.
- If $\delta(q, \sigma) = q'$, arrow from q to q', labelled with σ.

Figure 2-2

DFA and pattern matching:

A machine that matches an occurence of 10110 in a string.



Nondeterministic finite automata (NFA)

NFA: an extension of DFA, more powerful.

- In a DFA,
    1. *each* symbol causes a move (even though the state of the machine might remained unchanged after the move),
    2. the next state is completely determined by the current state and current symbol (1 next state).
- In an NFA,
    1. (a) The machine might have many possible moves
       (b) the machine can sometimes move without consuming any symbol
       (c) sometimes there are no possible moves;
       (d) sometimes it take more than one symbol to force the machine to move,
    2. the state is only *partially* determined by the current state and input symbols (>1 next states).

Figures 2-5, 2-6, 2-7

Nondeterministic finite automata (NFA)

Formally, an NFA is a quintuple $M = (K, \Sigma, \delta, s, F)$, where

1. K — a finite set of states,
2. Σ — an alphabet,
3. $s \in K$ — the initial state,
4. $F \subseteq K$ — the set of final states,
5. Δ — the transition relation *not function*
   - $\Delta \subseteq K \times \Sigma^* \times K$.

Nondeterministic finite automata (NFA)

Figure 2-5

1. $K = \{q_0, q_1, q_2\}$, $s = q_0$, $F = \{q_0\}$;
2. $\Sigma = \{a, b\}$;
3. $\Delta = \{(q_0, a, q_1), (q_1, b, q_0), (q_1, b, q_2), (q_2, a, q_0)\}$

1. $K = \{q_0, q_1, q_2\}$, $s = q_0$, $F = \{q_0\}$;
2. $\Sigma = \{a, b\}$;
3. $\Delta = \{(q_0, a, q_1), (q_1, b, q_2), (q_2, a, q_0), (q_2, e, q_0)\}$

Figure 2-6 Figure 2-7

1. $K = \{q_0\}$, $s = q_0$, $F = \{q_0\}$;
2. $\Sigma = \{a, b\}$;
3. $\Delta = \{(q_0, ab, q_0), (q_0, aba, q_0)\}$

Nondeterministic finite automata (NFA)

Given a NFA $M = (K, \Sigma, \delta, s, F)$,

- $(q, w) \vdash_M (q', w')$ iff there exists $u \in \Sigma^*$ s.t.
  - $w = uw'$, and
  - $(q, u, q') \in \Delta$.
- $\vdash_M^*$ is the reflexive and transitive closure of $\vdash_M$.
- A string w is **accepted** by M if

$$(s, w) \vdash_M^* (q, e), \text{ and } q \in F$$

- The **language accepted by M**:

$$L(M) = \{w : w \text{ accepted by } M\}.$$

Nondeterministic finite automata (NFA)

Why bother?

> Languages can be represented much more compactly by using NFA's than using DFA's.

$L = (ab \cup aba)*$
$= e$ and all strings that start with ab and does two b's or three a's in a row.

Figure 2-4 (DFA)

Figures 2-5, 2-6, 2-7 (NFA)