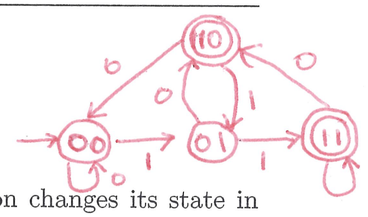
Lecture 2: Finite Automata: Its many forms

Date: August 24, 2023.

Deterministic Finite Automaton (DFA) is a tuple $M = (Q, \Sigma, \delta, s, F)$, where



- Q is a finite set of *states*,
- Σ is a finite set called the *input alphabet* that is used to encode the input,
- $\delta : Q \times \Sigma \to Q$ is the *transition function*, that determines how the automaton changes its state in response to a new input symbol,
- $s \in Q$ is the *initial state* in which the automaton begins its computation, and
- $F \subseteq Q$ is the set of *accept* or *final* states.

$00 \xrightarrow{0} 00 \xrightarrow{1} 01 \xrightarrow{0} 10 \xrightarrow{0} 00$ ✗

A **run** of M on input $x = a_1 a_2 \cdots a_n$, where $a_i \in \Sigma$, is a sequence of states $q_0, q_1, \ldots q_n$ such that (a) $q_0 = s$, and (b) $q_{i+1} = \delta(q_i, a_i)$ for every $i \geq 0$. Such a run corresponds to an **accepting** computation, if the last state belongs to F.

$\mathbf{L}(M) = \{x \in \Sigma^* \mid M \text{ accepts } x\}$.

Nondeterministic Finite Automaton (NFA) is a tuple $N = (Q, \Sigma, \Delta, S, F)$, where

- Q, Σ, $F \subseteq Q$, are the set of states, input alphabet, and final states, respectively, as before,
- $\Delta : Q \times \Sigma \to 2^Q$ is transition function, which given a current state, and input symbol, determines the set of *possible* next states of the automaton, and
- $S \subseteq Q$ is set of possible initial/start states in which the machine could begin.

A **run** of N on input $x = a_1 a_2 \cdots a_n$ is a sequence of states $q_0, q_1, \ldots q_n$ such that (a) $q_0 \in S$, and (b) $q_{i+1} \in \Delta(q_i, a_i)$ for every $i \geq 0$. An accepting run is one where $q_n \in F$. And an input x is **accepted** if N has **some** accepting run on x. $\mathbf{L}(N)$ is the collection of all strings accepted by N.

Universal Finite Automaton (UFA) is a tuple $U = (Q, \Sigma, \Delta, S, F)$ where Q, Σ, S, and F are the set of states, input alphabet, initial states, and final states, respectively, just like for NFAs. The transition function $\Delta : Q \times \Sigma \to (2^Q \setminus \{\emptyset\})$ maps a current state and input symbol to a *non-empty* set of next states.

U **accepts** input x if *every* run of U on x ends in an accepting state. $\mathbf{L}_\forall(U)$ is the collection all inputs x accepted by U.

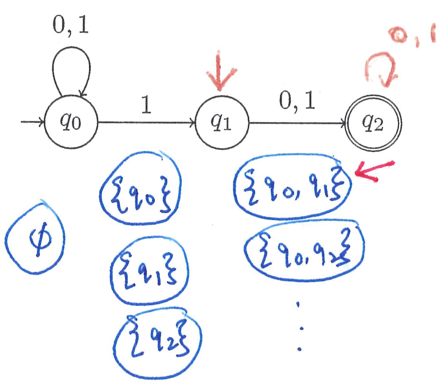
Proposition 1. *For any DFA M, there is a DFA $\overline{M}$ such that* $\mathbf{L}(\overline{M}) = \overline{\mathbf{L}(M)}$.

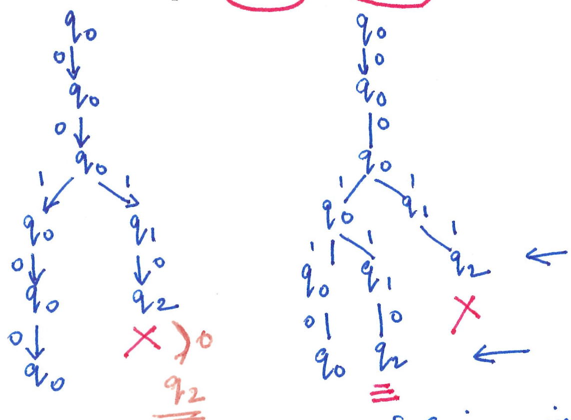
$M = (Q, \Sigma, \delta, s, F)$

$\overline{M} = (Q, \Sigma, \delta, s, \overline{F})$ $\qquad L(\overline{M}) = \overline{L(M)}$

Problem 1. Describe all the runs of N shown below on inputs 00100 and 00110.

No $\qquad$ Yes





Theorem 2. *For any NFA N, there is a DFA M such that* $\mathbf{L}(M) = \mathbf{L}(N)$.

[Given input x. Determine if N accepts x.

$N = (Q, \Sigma, \Delta, S, F)$

$M = 2^N = (2^Q, \Sigma, \delta, S, F')$

$\delta(A, a) = \bigcup_{q \in A} \Delta(q, a)$

$F' = \{A \mid A \cap F \neq \phi\}$.

Theorem 3. *For any UFA U, there is a DFA M such that* $\mathbf{L}(M) = \mathbf{L}_\forall(U)$.

$U = (Q, \Sigma, \Delta, S, F)$

$M = (2^Q, \Sigma, \delta, S, F')$

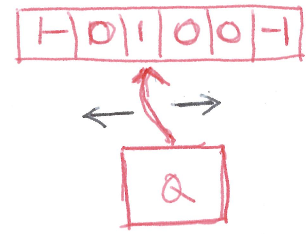
$\delta(A, a) = \bigcup_{q \in A} \Delta(q, a)$ $\qquad L(M) = L_\forall(U)$.

$F' = 2^F$.

$\overline{U} = (Q, \Sigma, \Delta, S, \overline{F})$

$\underbrace{L_\forall(U)}_{\text{Universal}} = \underbrace{\overline{L_\exists(\overline{U})}}_{\text{Nondet}}$

2-way Deterministic Finite Automaton (2DFA) is a tuple $M = (Q, \Sigma, \vdash, \dashv, \delta, s, t, r)$ where



- Q is the (finite) set of states,
- Σ is the input alphabet,
- $\vdash$ is the left endmarker, with $\vdash \notin \Sigma$,
- $\dashv$ is the right endmarker, with $\dashv \notin \Sigma$,
- $\delta : (Q \setminus \{t, r\}) \times (\Sigma \cup \{\vdash, \dashv\}) \to (Q \times \{\mathsf{L}, \mathsf{R}\})$ is the transition function,
- $s \in Q$ is the start state,
- $t \in Q$ is the accept state, and
- $r \in Q$ is the reject state, with $t \neq r$.

We also assume that the transition function satisfies the condition that for every $q \in Q \setminus \{t, r\}$,

$$\begin{aligned} \delta(q, \vdash) &= (p, \mathsf{R}) \quad \text{for some } p \in Q, \\ \delta(q, \dashv) &= (p, \mathsf{L}) \quad \text{for some } p \in Q. \end{aligned}$$

Fix input $x = a_1 a_2 \cdots a_n$. The tape has the string $\vdash x \dashv$. A **configuration** of M on input x is a pair (q, i) where $q \in Q$ and $0 \leq i \leq n + 1$. The **initial configuration** is $(s, 0)$. The **next configuration relation**, $\xrightarrow[x]{1}$, which describes step of the machine on input x, is defined as follows.

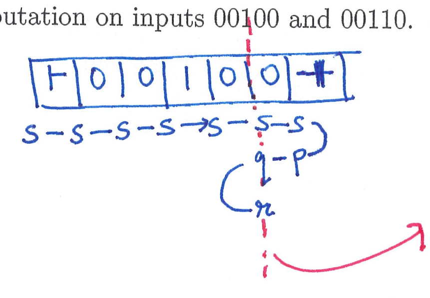
$$\begin{aligned} \delta(p, a_i) = (q, \mathsf{L}) &\Rightarrow (p, i) \xrightarrow[x]{1} (q, i - 1), \\ \delta(p, a_i) = (q, \mathsf{R}) &\Rightarrow (p, i) \xrightarrow[x]{1} (q, i + 1). \end{aligned}$$

A **run** of M on input x from configuration c is finite or infinite sequence of configurations $c_0, c_1, \ldots$ such that (a) $c_0 = (s, 0)$, (b) $c_i \xrightarrow[x]{1} c_{i+1}$ for every $i \geq 0$, and (c) if the sequence is finite, then the last configuration is either (t, j) or (r, j) for some j. The input is **accepted** if M's run on x reaches a configuration where the state is t.

Problem 2. Consider 2DFA $M_* = (\{s, p, q, t, r\}, \{0, 1\}, \vdash, \dashv, \delta, s, t, r)$ where δ is defined as follows.

$$\begin{array}{llll} \delta(s, a) = (s, \mathsf{R}) \text{ for any } a \in \{\vdash, 0, 1\} & \delta(s, \dashv) = (p, \mathsf{L}) & \delta(p, \vdash) = (q, \mathsf{R}) & \delta(q, \vdash) = (q, \mathsf{R}) \\ \delta(p, a) = (q, \mathsf{L}) \text{ for any } a \in \{0, 1, \dashv\} & \delta(q, 0) = (r, \mathsf{R}) & \delta(q, 1) = (t, \mathsf{R}) & \delta(q, \dashv) = (q, \mathsf{L}). \end{array}$$

Describe its computation on inputs 00100 and 00110.



$\{s, q, r\}$.

Crossing set : Set of state

Theorem 4. *For any 2DFA M, there is an NFA N such that $\mathbf{L}(N) = \mathbf{L}(M)$.*

Crossing set is the set of a states of M when it crosses a particular boundary.

Prove: $\exists$ NFA N s.t. $L(N) = \overline{L(M)}$

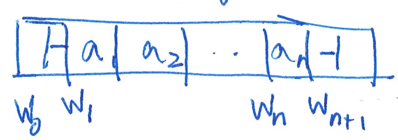
Design a NFA that solves.

Given input x

Answers "yes" if M does not accept x.

Consider computation of M on $x = a_1 a_2 \ldots a_n$.

Let W_i to be the crossing set for M on the left boundary of cell i.



If M does not accept x, $t \notin W_i \ \forall i$.

$\forall q \in W_i$