Outline

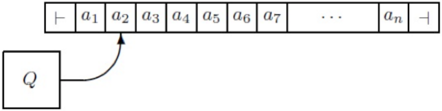
What is 2DFA?

2DFA

A Two-Way Deterministic Finite Automaton (2DFA) is a type of finite state machine that extends the capabilities of a regular Deterministic Finite Automaton (DFA) by allowing its read head to move bidirectionally along the input tape.

- 2DFAs were introduced in 1959 by Rabin and Scott.

How 2DFA works?



- Two-way Finite Automata has a read head, which can move left or right over the input string .
- Read head can revisit the input symbols any number of times.
- The input string is enclosed between left and right endmarkers $\vdash$ and $\dashv$, which are not elements of the input alphabet Σ.
- The read head will not move outside of the endmarkers.
- A 2DFA needs only a single accept state and a single reject state.

Turing Machine vs 2DFA

Turing Machine:

- Contains a read/write head that moves left or right along the tape
- Has unbounded memory.

2DFA:

- Has read only head that moves left or right along the tape.
- Has finite memeory like DFA.

Formal representation of 2DFA

A *2DFA* is of the form

$$(Q, \Sigma, \vdash, \dashv, \delta, s, t, r)$$

where,

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

Properties of transition function

For all states p,

- $\delta(p, \vdash) = (q, R)$, for some $q \in Q$
- $\delta(p, \dashv) = (q, L)$, for some $q \in Q$

Current input symbol is $a \in \Sigma \cup \{\vdash\}$, $t =$ accept state, $r =$ reject state.

- $\delta(t, a) = (t, R)$ and $\delta(t, \dashv) = (t, L)$
- $\delta(r, a) = (r, R)$ and $\delta(r, \dashv) = (r, L)$

In general, $\delta(p, a) = (q, d)$ where $p, q \in Q$ and $d \in \{L, R\}$

Example 2DFA

2DFA for a^*

$(Q, \Sigma, \vdash, \dashv, \delta, s, t, r)$ where,

- $Q = \{q_0, q_1, q_2\}$
- $\Sigma = \{a, b\}$
- s = start state = q_0
- t = accept state = q_1
- r = reject state = q_2

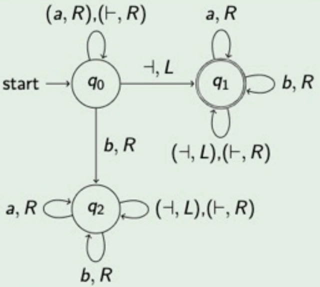


Table: Transition function δ

$\delta(,)$	$\vdash$	a	b	$\dashv$
q_0	(q_0, R)	(q_0, R)	(q_2, R)	(q_1, L)
q_1	(q_1, R)	(q_1, R)	(q_1, R)	(q_1, L)
q_2	(q_2, R)	(q_2, R)	(q_2, R)	(q_2, L)

Configurations

Fix an input $x \in \Sigma^*$. $x = a_1 a_2 a_3 \ldots a_n$. Let $a_0 = \vdash$ and $a_{n+1} = \dashv$. $a_0 a_1 a_2 a_3 \ldots a_n a_{n+1} = \vdash x \dashv$.

Configuration

A configuration of the machine on input x is a pair (q, i) such that $q \in Q$ and $0 \leq i \leq n+1$. Informally, the pair (q, i) gives a current state and current position of the read head.
The start configuration is $(s, 0)$, meaning that the machine is in its start state s and scanning the left endmarker.

A binary relation $\xrightarrow[x]{1}$, the next configuration relation, is defined on configurations as follows:

$$\delta(p, a_i) = (q, L) \Rightarrow (p, i) \xrightarrow[x]{1} (q, i-1),$$

$$\delta(p, a_i) = (q, R) \Rightarrow (p, i) \xrightarrow[x]{1} (q, i+1).$$

Configurations

The relation $\xrightarrow[x]{1}$ describes one step of the machine on input x. We define the relations $\xrightarrow[x]{n}$ inductively, $n \geq 0$:

- $(p, i) \xrightarrow[x]{0} (p, i)$
- if $(p, i) \xrightarrow[x]{n} (q, j)$ and $(q, j) \xrightarrow[x]{1} (u, k)$, then $(p, i) \xrightarrow[x]{n+1} (u, k)$.

For any configuration (p, i), there is exactly one configuration (q, j) such that $(p, i) \xrightarrow[x]{n} (q, j)$.

Acceptance and Rejection

$$(p, i) \xrightarrow[x]{*} (q, j) \text{ iff } \exists n \geq 0 \text{ such that } (p, i) \xrightarrow[x]{n} (q, j).$$

Acceptance

The input x is accepted by the machine iff $(s, 0) \xrightarrow[x]{*} (t, k)$ for some k.

Rejection

- The input x is rejected by the machine if $(s, 0) \xrightarrow[x]{*} (r, k)$ for some k.

If the machine neither reaches accept state nor reject state then the machine is said to be looping on that input.
Language accepted by the machine $= \{x \in \Sigma^* | x$ is accepted by the machine$\}$.

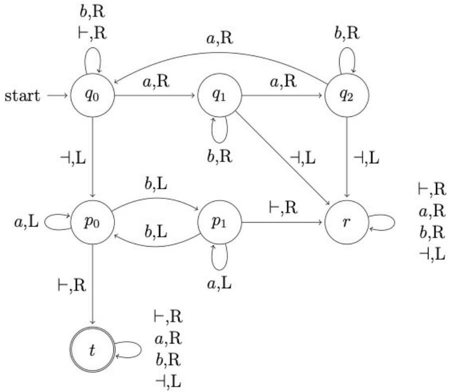
Constructing 2DFA 'M'

$$L(M) = \{x \in \Sigma^* \mid \#a(x) \text{ is multiple of 3, } \#b(x) \text{ is multiple of 2}\}$$

Machine Description

- Machine starts scanning from the left endmarker.
- Scan input string from left to right, counting only 'a's. If the count of 'a's is not a multiple of 3, reject and enter state r.
- Let q_0, q_1, q_2 be the states for counting 'a's.
  q_0: 3k; q_1: 3k+1; q_2: 3k+2.
- If the count of 'a's is a multiple of 3, start scanning from the right, counting only 'b's. If the count of 'b's is not a multiple of 2, enter state t; otherwise, enter state r.
- Let p_0, p_1 be the states for counting 'b's.
  p_0: 2k; p_1: 2k+1.

Constructing 2DFA 'M'



- input = bbaabab :

$$q_0 \xrightarrow{\vdash} q_0 \xrightarrow{b} q_0 \xrightarrow{b} q_0 \xrightarrow{a} q_1 \xrightarrow{a} q_2 \xrightarrow{b} q_2 \xrightarrow{a} q_0 \xrightarrow{b} q_0 \xrightarrow{\dashv} p_0 \xrightarrow{b}$$
$$p_1 \xrightarrow{a} p_1 \xrightarrow{b} p_0 \ldots\ldots \xrightarrow{\dashv} t$$

DFA to 2DFA conversion

Theorem

- 2DFA only accepts regular languages.
- L(2DFA) = L(DFA) = Regular languages

Proof: L(DFA) $\subseteq$ L(2DFA)

For an arbitary DFA 'X', let us construct a 2DFA 'Y' that accepts the same language as X.

- X : (Q, Σ, δ, s, F)
- Let Y : (Q $\cup$ {t, r}, Σ, $\vdash$, $\dashv$, δ', s, t, r)
- δ': $\delta_R \cup$
  $\delta'((s, \vdash)) = (s, R)$
  $\delta'((f, \dashv)) = (t, L)$ for $f \in F$; $\delta'((n, \dashv)) = (r, L)$ for $n \in$ Q-F.
  $\delta'((t, a)) = (t, R)$; $\delta'((r, a)) = (r, R)$ for $a \in \Sigma - \{\dashv\}$
  $\delta'((t,\dashv)) = (t,L)$; $\delta'((r,\dashv)) = (r,L)$

DFA to 2DFA conversion

Proof: L(X) $\subseteq$ L(Y)

- Let x $\in$ L(X), len(x)=n.
- Then, $\widehat{\delta}$(s,x) = f where f $\in$ F
- $(s,0) \xrightarrow[\vdash x \dashv]{1} (s,1) \xrightarrow[\vdash x \dashv]{n} (f,n+1) \xrightarrow[\vdash x \dashv]{1} (t,n)$.
- As $(s,0) \xrightarrow[\vdash x \dashv]{*} (t,n+1)$, x $\in$ L(Y).
- Hence, L(X) $\subseteq$ L(Y).

Recall:
A configuration of the machine on input x is the pair (q, i) , which gives the current state and current position of the read head.
$(s, 0)$ means that the machine is in its start state s and scanning the left endmarker.

DFA to 2DFA conversion

Proof: L(Y) $\subseteq$ L(X)

- Let x $\in$ L(Y), len(x)=n.
- Then (s,0) $\xrightarrow[\vdash x \dashv]{*}$ (t,m) in Y
- So (s,0) $\xrightarrow[\vdash x \dashv]{1}$ (s,1) $\xrightarrow[\vdash x \dashv]{n}$ (f,n+1) $\xrightarrow[\vdash x \dashv]{1}$ (t,n) .
- As $\widehat{\delta}$(s,x) = f where f $\in$ F, x $\in$ L(X)
- Hence, L(Y) $\subseteq$ L(X).

Therefore, L(Y) = L(X)