Two Way Deterministic Finite Automata

Jagvir Singh
and
Pavan Kumar Akulakrishna

Indian Institute of Science

November 29, 2013
Overview

- Introduction.
- Formal Construction.
- Example.
- Configuration and Acceptance.
- 2DFA vs DFA.
2-way Deterministic Finite Automata

1. Generalised version of DFA.
2. Process the input in either direction.
   - Have read only head which can move in both direction over the input string.
   - Revisit the characters again and again.
3. Like a Turing Machine but.
   - Have read only head.
   - Have finite memory like DFA.
2DFA has finite set of states $Q$ like DFA.

Input string
- Input string is stored on finite tape.
- One character per cell.
- Input string is stored in between two extra symbol called left endmarker($\langle -$) and right endmarker($- \rangle$).

At any time instance the machine is in state $p$ and scan some symbol $a_i \in \Sigma$ or an endmarkers $\{\langle -, - \rangle\}$, based on $p$ and current symbol it will move its head one cell in direction $d \in \{L, R\}$ and enter in new state $q$.

Machine head never go outside the endmarkers.
Accept and reject states.
- 2DFA needs only single accept and single reject state.
- It will accept the input string by entering in a special accept state $t$.
- It will reject the input string by entering in a special reject state $r$.
- Accept and reject states are like sink state.

The machine action on a present state and head symbol is depend on transition function $\delta$.

Transition function take present state and head symbol as input argument and return next state and direction of movement of head.
Formal definition of 2DFA

2DFA is represented by octuple.

\[ M = \{ Q, \Sigma, \delta, s, t, r, \vdash, \dashv \} \]

where

- \( Q \) is a finite set of states.
- \( \Sigma \) is a finite set of input symbol.
- \( \delta : Q \times (\Sigma \cup \{ \vdash, \dashv \}) = Q \times \{ L, R \} \) is a transition function.
- \( s \in Q \) is a start state.
- \( t \in Q \) is a accept state.
- \( r \in Q \) is a reject state.
- \( \vdash \) is left endmarker.
- \( \dashv \) is right endmarker.
Some properties of transition function

1. Input is endmarker.
   - $\delta(p, \downarrow) = (q, R)$
   - $\delta(p, \uparrow) = (q, L)$

2. Accept and reject states are $t$, $r$ respectively and current input symbol is $a \in \Sigma \cup \{\downarrow\}$.
   - $\delta(t, a) = (t, R)$ and $\delta(t, \downarrow) = (t, L)$
   - $\delta(r, a) = (r, R)$ and $\delta(r, \downarrow) = (r, L)$

3. In general
   
   $\delta(p, a) = (q, d)$ where $p, q \in Q$ and $d \in \{L, R\}$
Example (Constructing a normal DFA)

Construct the DFA to accept the language 
\[ L = \{ x \in \Sigma^* | \#a(x) \text{ are multiple of 3, } \#b(x) \text{ are multiple of 2} \} \]

Construct a normal DFA

1. DFA \( M_1 \) accepting \( L_1 = \{ x \in \Sigma^* | \#a(x) \text{ are multiple of 3} \} \)
   \[ M_1 = \{ Q_1, \Sigma, \delta_1, s_1, F_1 \} \]

2. DFA \( M_2 \) accepting \( L_2 = \{ x \in \Sigma^* | \#b(x) \text{ are multiple of 2} \} \)
   \[ M_2 = \{ Q_2, \Sigma, \delta_2, s_2, F_2 \} \]

3. DFA \( M \) accepting \( L \) such that \( M = M_1 \times M_2 \)
   \[ M = \{ Q, \Sigma, \delta, s, F \} \]
Example (Constructing a 2DFA)

Construct a 2DFA accepting the set

\[ L = \{ x \in \Sigma^* | \#a(x) \text{ are multiple of } 3, \#b(x) \text{ are multiple of } 2 \} \]

1. Machine start scanning from the left endmarker.
2. Scan input string from left to right consider only a and ignore b.
   - if \#a(x) are not multiple of 3 then rejects x and enters in state \( r \).
3. if \#a(x) are multiple of 3 then start scanning from right consider only b and ignore a.
   - if \#b(x) are not multiple of 2 then enters in \( t \) otherwise enters in state \( r \).
Example (Formal construction of 2DFA)

\[ M = \{ Q, \Sigma, \delta, s, t, r, \l, \r \} \]

where \( \Sigma = \{ a, b \} \), \( Q = \{ q_0, q_1, q_2, p_0, p_1, t, r \} \) and the transition function \( \delta \) is given by following table.

<table>
<thead>
<tr>
<th>states</th>
<th></th>
<th>a</th>
<th></th>
<th>b</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>q_0</td>
<td></td>
<td>(q_0, R)</td>
<td></td>
<td>(q_1, R)</td>
<td></td>
</tr>
<tr>
<td>q_1</td>
<td>-</td>
<td>(q_2, R)</td>
<td></td>
<td>(q_1, R)</td>
<td></td>
</tr>
<tr>
<td>q_2</td>
<td>-</td>
<td>(q_0, R)</td>
<td></td>
<td>(q_2, R)</td>
<td></td>
</tr>
<tr>
<td>p_0</td>
<td></td>
<td>(t, R)</td>
<td></td>
<td>(p_0, L)</td>
<td></td>
</tr>
<tr>
<td>p_1</td>
<td></td>
<td>(r, R)</td>
<td></td>
<td>(p_1, L)</td>
<td></td>
</tr>
<tr>
<td>t</td>
<td></td>
<td>(t, R)</td>
<td></td>
<td>(t, R)</td>
<td></td>
</tr>
<tr>
<td>r</td>
<td></td>
<td>(r, R)</td>
<td></td>
<td>(r, R)</td>
<td></td>
</tr>
</tbody>
</table>

Jagvir Singh and Pavan Kumar Akulakrishna (IISC)
Configuration and Acceptance

Let we have input string \( x \in \Sigma^* \) such that \( x = a_1 a_2 \cdots a_{n-1} a_n \), \( |x| = n \) and let \( a_0 = \leftarrow \), \( a_{n+1} = \rightarrow \) then machine head will scan \( \leftarrow x \rightarrow \).

1. Configuration for the input \( x \) is pair \( (p, j) \) such that \( p \in Q \) and \( 0 \leq j \leq n + 1 \).

2. In pair \( (p, j) \) \( p \) is current state and \( j \) is current position of head.

3. Initial configuration of machine is \( (s, 0) \) this mean initially machine is in state \( s \) and scanning left endmarker.

4. The relation \( \xrightarrow{1} \) describes one step of the machine on input \( x \).

   - \( \delta(p, a_j) = (q, L) \Rightarrow (p, j) \xrightarrow{x} (q, j - 1) \)
   - \( \delta(p, a_j) = (q, R) \Rightarrow (p, j) \xrightarrow{x} (q, j + 1) \)
   - \( (p, j) \xrightarrow{0} (p, j) \)
   - \( (p, i) \xrightarrow{n} (q, j) \) and \( (q, j) \xrightarrow{x} (u, k) \) then \( (p, i) \xrightarrow{n+1} (u, k) \)
5. \( (p, j) \xrightarrow{x} (q, k) \overset{\text{def}}{\iff} \exists n \geq 0 \text{ such that } (p, j) \xrightarrow{n}{x} (q, k) \)

6. Machine accept input string \( x \) if \( (s, 0) \xrightarrow{x} (t, k) \) for some \( k \).

7. Machine reject input string \( x \) if \( (s, 0) \xrightarrow{x} (r, k) \) for some \( k \).

8. It is possible that machine neither accept nor reject input string \( x \) then machine go in loop.

9. Language accepted by machine \( M \) is \( L(M) = \{ x \in \Sigma^* | (s, 0) \xrightarrow{x} (t, k) \} \)