Deterministic Finite Accepter (or Automata)

A DFA = \((Q, \Sigma, \delta, q_0, F)\)

where

- \(Q\) is finite set of states
- \(\Sigma\) is tape (input) alphabet
- \(q_0\) is initial state
- \(F \subseteq Q\) is set of final states.
- \(\delta : Q \times \Sigma \rightarrow Q\)

**Example:** Create a DFA that accepts even binary numbers.

Transition Diagram:

\[
\begin{array}{c}
\text{q0} \\
\rightarrow \quad 0 \\
\quad \quad \rightarrow \text{q1} \\
\quad 1 \quad 0 \\
\text{q1} \\
\end{array}
\]

\[M = (Q, \Sigma, \delta, q_0, F) = \]

Tabular Format

<table>
<thead>
<tr>
<th>(Q)</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>(q0)</td>
<td>(q1)</td>
<td>(q0)</td>
</tr>
<tr>
<td>(q1)</td>
<td>(q1)</td>
<td>(q0)</td>
</tr>
</tbody>
</table>

Example of a move: \(\delta(q0, 1) = \)
Algorithm for DFA:

Start in start state with input on tape
q = current state
s = current symbol on tape
while (s != blank) do
    q = \delta(q, s)
    s = next symbol to the right on tape
if q \in F then accept

Example of a trace: 11010

Pictorial Example of a trace for 100:

\begin{itemize}
  \item 1) \begin{array}{cccc}
          & 1 & 0 & 0 \\
q_0 & \rightarrow & q_1
        \end{array}
  \item 2) \begin{array}{cccc}
          & 1 & 0 & 0 \\
q_0 & \rightarrow & q_1
        \end{array}
  \item 3) \begin{array}{cccc}
          & 1 & 0 & 0 \\
q_0 & \rightarrow & q_1
        \end{array}
  \item 4) \begin{array}{cccc}
          & 1 & 0 & 0 \\
q_0 & \rightarrow & q_1
        \end{array}
\end{itemize}

Definition:

\begin{align*}
\delta^*(q, \lambda) &= q \\
\delta^*(q, wa) &= \delta(\delta^*(q, w), a)
\end{align*}

Definition The language accepted by a DFA \( M = (Q, \Sigma, \delta, q_0, F) \) is set of all strings on \( \Sigma \) accepted by \( M \). Formally,

\[ L(M) = \{ w \in \Sigma^* \mid \delta^*(q_0, w) \in F \} \]
**Trap State**

Example: \( L(M) = \{ b^n a \mid n > 0 \} \)

![Diagram of a DFA](image)

You don't need to show trap states! Any arc not shown will by default go to a trap state.

**Example:** Create a DFA that accepts even binary numbers that have an even number of 1’s.

**Example:**

\[ L = \{ w \in \Sigma^* \mid \text{w has an even number of a’s and an even number of b’s} \} \]

**Definition** A language is regular iff there exists DFA \( M \) s.t. \( L = L(M) \).
Chapter 2.2

Nondeterministic Finite Automata (or Accepter)

**Definition**

An NFA = $(Q, \Sigma, \delta, q_0, F)$

where

- $Q$ is finite set of states
- $\Sigma$ is tape (input) alphabet
- $q_0$ is initial state
- $F \subseteq Q$ is set of final states.
- $\delta : Q \times (\Sigma \cup \{\lambda\}) \rightarrow 2^Q$

**Example**

![NFA Diagram]

Note: In this example $\delta(q_0, a) = $

**Example**

$L = \{ (ab)^n \mid n > 0 \} \cup \{ a^n b \mid n > 0 \}$

**Definition** $q_j \in \delta^*(q_i, w)$ if and only if there is a walk from $q_i$ to $q_j$ labeled $w$.

**Example** From previous example:

$\delta^*(q_0, ab) =$

$\delta^*(q_0, aba) =$

**Definition:** For an NFA $M$, $L(M) = \{ w \in \Sigma^* \mid \delta^*(q_0, w) \cap F \neq \emptyset \}$

The language accepted by nfa $M$ is all strings $w$ such that there exists a walk labeled $w$ from the start state to final state.
2.3 NFA vs. DFA: Which is more powerful?

Example:

![Diagram of NFA](image)

**Theorem** Given an NFA $M_N=(Q_N, \Sigma, \delta_N, q_0, F_N)$, then there exists a DFA $M_D=(Q_D, \Sigma, \delta_D, q_0, F_D)$ such that $L(M_N) = L(M_D)$.

**Proof:**

We need to define $M_D$ based on $M_N$.

$Q_D = \ldots$

$F_D = \ldots$

$\delta_D : \ldots$

**Algorithm to construct $M_D$**

1. start state is $\{q_0\} \cup \text{closure}(q_0)$

2. While can add an edge
   
   (a) Choose a state $A=\{q_i, q_j, \ldots, q_k\}$ with missing edge for $a \in \Sigma$
   
   (b) Compute $B = \delta^*(q_i, a) \cup \delta^*(q_j, a) \cup \ldots \cup \delta^*(q_k, a)$

   (c) Add state $B$ if it doesn’t exist

   (d) add edge from $A$ to $B$ with label $a$

3. Identify final states

4. if $\lambda \in L(M_N)$ then make the start state final.
Example:

Minimizing Number of states in DFA

Why?

Algorithm

- Identify states that are indistinguishable
  These states form a new state

Definition Two states p and q are indistinguishable if for all \( w \in \Sigma^* \)

\[
\delta^*(q, w) \in F \Rightarrow \delta^*(p, w) \in F \\
\delta^*(p, w) \notin F \Rightarrow \delta^*(q, w) \notin F
\]

Definition Two states p and q are distinguishable if \( \exists \ w \in \Sigma^* \) s.t.

\[
\delta^*(q, w) \in F \Rightarrow \delta^*(p, w) \notin F \text{ OR} \\
\delta^*(q, w) \notin F \Rightarrow \delta^*(p, w) \in F
\]
Example:
Example: