Deterministic Finite Accepter (or Automata)

A DFA = (Q, Σ, δ, q₀, F)

where

Q is finite set of states
Σ is tape (input) alphabet
q₀ is initial state
F ⊆ Q is set of final states.
δ: Q × Σ → Q

Example: Create a DFA that accepts even binary numbers.

Transition Diagram:

$$M = (Q, Σ, δ, q₀, F) =$$

Tabular Format

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

Example of a move: δ(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 = δ(q, s)
  s = next symbol to the right on tape
if q ∈ F then accept

Example of a trace: 11010
Pictorial Example of a trace for 100:

Definition:
\[ \delta^*(q, \lambda) = q \]
\[ \delta^*(q, wa) = \delta(\delta^*(q, w), a) \]

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

Example: \( L(M) = \)

\[
\begin{array}{c}
\text{q0} & \xrightarrow{b} & \text{q1} & \xrightarrow{a} & \text{q2} \\
\text{trap} & \xrightarrow{a,b} & \text{q0} & \xrightarrow{b} & \text{q1} & \xrightarrow{a} & \text{q2} \\
\end{array}
\]

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

**Example:**

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

**Example:** Create a DFA that accepts even binary numbers that have an even number of 1’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, Σ, δ, q₀, F)

where

- Q is a finite set of states
- Σ is the tape (input) alphabet
- q₀ is the initial state
- F ⊆ Q is the set of final states.
- δ : Q × (Σ ∪ {λ}) → 2^Q

Example

\[\begin{align*}
q_0 & \rightarrow q_1 \\
q_0 & \rightarrow q_2 \\
q_1 & \rightarrow q_2 \\
q_1 & \rightarrow q_3 \\
q_2 & \rightarrow q_2 \\
q_3 & \rightarrow q_3
\end{align*}\]

Note: In this example \(δ(q₀, a) = \)

L=

Example

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

Definition

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

Example

From previous example:

\(δ^*(q₀, ab) = \)

\(δ^*(q₀, aba) = \)

Definition: For an NFA M, \(L(M) = \{w \in Σ^* \mid δ^*(q₀, w) \cap F \neq \emptyset\}\)

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:

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 $M_N$.

$Q_D =$

$F_D =$

$\delta_D =$

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 $A=\{q_i, q_j, ... q_k\}$ with missing edge for $a \in \Sigma$
   (b) Compute $B = \delta^*(q_i, a) \cup \delta^*(q_j, a) \cup ... \cup \delta^*(q_k, a)$
   (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.
Properties and Proving - Problem 1

Consider the property Replace one a with b or R1awb for short. If \( L \) is a regular, prove \( R1awb(L) \) is regular.

The property \( R1awb \) applied to a language \( L \) replaces one \( a \) in each string with a \( b \). If a string does not have an \( a \), then the string is not in \( R1awb(L) \).

Example 1: Consider \( L = \{ aaab, bbaa \} \)

\( R1awb(L) = \)

Example 2: Consider \( \Sigma = \{ a, b \}, L = \{ w \in \Sigma^* \mid w \) has an even number of a’s and an even number of b’s\} \)

\( R1awb(L) = \)

Proof:
Properties and Proving - Problem 2

Consider the property Truncate_all_preceeding_b's or TruncPreb for short. If L is a regular, prove TruncPreb(L) is regular.

The property TruncPreb applied to a language L removes all preceding b’s in each string. If a string does not have an preceding b, then the string is the same in TruncPreb(L).

Example 1: Consider \( L = \{aaab, bbaa\} \)

\( \text{TruncPreb}(L) = \)

Example 2: Consider \( L = \{(bba)^n \mid n > 0\} \)

\( \text{TruncPreb}(L) = \)

Proof:
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: