Formal Definition of DFA

A deterministic finite automaton (DFA) is a 5-tuple \( M = (Q, \Sigma, \delta, q_0, F) \) where

- \( Q \) is a finite set of states;
- \( \Sigma \) is the input alphabet;
- \( \delta : Q \times \Sigma \to Q \) is the transition function;
- \( q_0 \in Q \) is the start state; and
- \( F \subset Q \) is the set of final or accepting states.
The Model

A typical mental model for a DFA looks like this:

![Diagram of a DFA model]

- **Input Tape**
- **Output**
- **Read Head**
- **Finite State Control**
- **State**
Characteristics of The Model

- The symbols on the input tape are from the input alphabet $\Sigma$, one symbol per tape square.

- The read-only tape head examines one square at a time and proceeds only left to right. The computation ends when the tape head moves off the rightmost square.

- The STATE register contains the current state from $Q$. Initially contains $q_0$.

- The FINITE STATE CONTROL uses the transition function $\delta$ to implement computation steps.

- Acceptance depends on whether the STATE register contains a final state (from $F$) after the last step of the computation.
Example Using The Model

Let $M_1$ be the DFA given by:

$$M_1 = (Q_1, \Sigma, \delta_1, q_0, F_1)$$

$$Q_1 = \{q_0, q_1, q_2, q_3\}$$

$$\Sigma = \{a, b\}$$

$$F_1 = \{q_3\}$$

The transition function $\delta_1 : Q_1 \times \Sigma \rightarrow Q_1$ is given by this table:

<table>
<thead>
<tr>
<th>$q$</th>
<th>$\delta_1(q, a)$</th>
<th>$\delta_1(q, b)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_0$</td>
<td>$q_1$</td>
<td>$q_2$</td>
</tr>
<tr>
<td>$q_1$</td>
<td>$q_0$</td>
<td>$q_3$</td>
</tr>
<tr>
<td>$q_2$</td>
<td>$q_3$</td>
<td>$q_0$</td>
</tr>
<tr>
<td>$q_3$</td>
<td>$q_2$</td>
<td>$q_1$</td>
</tr>
</tbody>
</table>

Try the model on these inputs: $ababab$ and $baabba$.
Acceptance

Informally, a string $w \in \Sigma^*$ is accepted by $M$ if $M$ is in a final state after reading $w$.

To define acceptance formally, we need the notions of configuration and of the yields relation.
Configurations

A configuration of $M$ is an element of $Q \times \Sigma^*$.

For an input $w \in \Sigma^*$, the initial or start configuration is

$$(q_0, w).$$

The yields (in one step) relation $\vdash_M$ is a binary relation on $Q \times \Sigma^*$. If $q_i, q_j \in Q$, $\sigma \in \Sigma$ and $v \in \Sigma^*$, then

$$(q_i, \sigma v) \vdash_M (q_j, v)$$

if and only if

$$\delta(q_i, \sigma) = q_j.$$
Example

For the previous example \( M_1 \) and the input \( w = abbab \), the initial configuration is

\[
(q_0, abbab).
\]

The initial configuration yields what configuration in one step:

\[
(q_0, abbab) \downarrow_{M_1} ?
\]

How does the computation go when expressed using configurations and the yields relation ?
Yields in \( t \) Steps

A recursive definition for the **yields in \( t \) steps** relation \( \vdash_{M}^{t} \) is

- **Basis.** For all \( q_{i} \in Q \) and all \( v \in \Sigma^{*} \),

\[
(q_{i}, v) \vdash_{M}^{0} (q_{i}, v).
\]

- **Recursive Step.** If

\[
(q_{i}, x) \vdash_{M}^{t} (q_{j}, y).
\]

and

\[
(q_{j}, y) \vdash_{M} (q_{k}, z).
\]

then

\[
(q_{i}, x) \vdash_{M}^{t+1} (q_{k}, z).
\]
Yields in Zero or More Steps

The yields (in zero or more steps) relation $\vdash^*$ is the union

$$\vdash^* = \bigcup_{t \geq 0} \vdash^t_M.$$

In other words,

$$(q_i, x) \vdash^*_{M} (q_j, y)$$

holds if and only if there exists a $t \geq 0$ such that

$$(q_i, x) \vdash^t_{M} (q_j, y).$$
Exercise

For the previous example $M_1$, give all the configurations $(q_k, z)$ satisfying

$$(q_2, baab) \xrightarrow{*}_{M_1} (q_k, z).$$
Acceptance Again

A string $w \in \Sigma^*$ is **accepted** by the DFA $M$ if

$$(q_0, w) \xrightarrow{*}_M (q_i, \lambda),$$

where $q_i \in F$.

The **language** $L(M)$ **accepted** by the DFA $M$ is the set of all strings accepted by $M$.

Said another way,

$$L(M) = \left\{ w \in \Sigma^* \mid (q_0, w) \xrightarrow{*}_M (q_i, \lambda) \text{ and } q_i \in F \right\}. $$
Exercises

Suppose \( M_2 = (Q_2, \Sigma, \delta_2, q_0, F_2) \) has transition function \( \delta_2 \) given by

\[
\begin{array}{c|cc}
q & \delta_2(q, a) & \delta_2(q, b) \\
\hline
q_0 & q_0 & q_1 \\
q_1 & q_0 & q_0 \\
\end{array}
\]

and that \( F_2 = \{q_0\} \). What is \( L(M_2) \)?

Said another way, what is

\[
L(M_2) = \left\{ w \in \{a, b\}^* \mid (q_0, w) \xrightarrow{*}_{M_2} (q_0, \lambda) \right\}?
\]

What is \( L(M_1) \)?

Said another way, what is

\[
L(M_1) = \left\{ w \in \{a, b\}^* \mid (q_0, w) \xrightarrow{*}_{M_1} (q_3, \lambda) \right\}?
\]
State Diagrams

A DFA $M = (Q, \Sigma, \delta, q_0, F)$ has a graph representation called a state diagram.

The state diagram $G$ for $M$ has

- Node set $Q$.

- An arc from $q_i$ to $q_j$ labeled $\sigma$ if $\delta(q_i, \sigma) = q_j$.

- The start state is designated $\rightarrow q_0$.

- A final state $q_i$ is designated $q_i$.
Example

The state diagram for $M_1$ is

![State Diagram]

- Every node has $|\Sigma|$ outgoing arcs, one for each symbol in $\Sigma$.

- A string $w \in \Sigma^*$ determines a path in $G$ from $q_0$ to the last state in the computation on input $w$.

- Follow the path for $w = abaab$. 
Extended Transition Function

The extended transition function

\[ \hat{\delta} : Q \times \Sigma^* \rightarrow Q \]

is defined recursively as follows.

- **Basis:** If \(|w| = 0\), then
  \[ \hat{\delta}(q_i, w) = q_i. \]

- **Recursive Step:** If \(|w| > 0\), then \(w = u\sigma\), where \(u \in \Sigma^*\) and \(\sigma \in \Sigma\). Define
  \[ \hat{\delta}(q_i, w) = \delta(\hat{\delta}(q_i, u), \sigma). \]
Example

Start with $M_1$ again:

Use the definition of the extended transition function to compute

$$\hat{\delta}(q_0, ba) = \quad ?$$

$$\hat{\delta}(q_2, abab) = \quad ?$$
Alternate Definition of Acceptance

A string \( w \in \Sigma^* \) is accepted by the DFA \( M \) if
\[
\hat{\delta}(q_0, w) \in F.
\]

The language \( L(M) \) accepted by the DFA \( M \) is
\[
L(M) = \left\{ w \in \Sigma^* \mid \hat{\delta}(q_0, w) \in F \right\}.
\]
Example

Recall $M_2 = (\{q_0, q_1\}, \{a, b\}, \delta_2, q_0, \{q_0\})$ with transition function $\delta_2$ given by

<table>
<thead>
<tr>
<th>$q$</th>
<th>$\delta_2(q, a)$</th>
<th>$\delta_2(q, b)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_0$</td>
<td>$q_0$</td>
<td>$q_1$</td>
</tr>
<tr>
<td>$q_1$</td>
<td>$q_0$</td>
<td>$q_0$</td>
</tr>
</tbody>
</table>

By definition,

$$L(M_2) = \left\{ w \in \{a, b\}^* \mid \hat{\delta}(q_0, w) = q_0 \right\}.$$  

From the observations

$$\hat{\delta}(q_i, wa) = q_0$$
$$\hat{\delta}(q_i, wb) = q_1 - i$$
$$\hat{\delta}(q_i, wbb) = q_i,$$

we conclude that

$$\Sigma^*\{a\} \subset L(M_2)$$
$$L(M_2)\{bb\} \subset L(M_2)$$
$$w \in L(M_2) \iff wb \notin L(M_2).$$
Constructing DFAs

Construct a DFA $M_3$ that accepts

$$L_3 = \{x \in \{a, b\}^* \mid n_b(x) \equiv 0 \mod 3\}.$$  

Need three states, to remember the number of $b$’s modulo 3.

![State Diagram](image.png)

Construct a DFA $M_5$ that accepts

$$L_5 = \{x \in \{a, b\}^* \mid n_b(x) \equiv 0 \mod 5\}$$

(Use states $p_0, p_1, p_2, p_3, p_4$.)
Union of DFA Languages

EXERCISE.

Construct a DFA $M_{3,5}$ that accepts

$L_{3,5} = L_3 \cup L_5$

\[= \{ x \in \{a, b\}^* \mid n_b(x) \equiv 0 \mod 3 \text{ or } n_b(x) \equiv 0 \mod 5 \}$

\[\square\]
Union of DFA Languages

Now generalize the previous construction.

**Theorem.** If \( L_1 \subseteq \Sigma^* \) is accepted by a DFA \( M_1 = (Q_1, \Sigma, \delta_1, q_0, F_1) \) and \( L_2 \subseteq \Sigma^* \) is accepted by a DFA \( M_2 = (Q_2, \Sigma, \delta_2, p_0, F_2) \), then there is a DFA that accepts \( L_1 \cup L_2 \).

**Proof:** Define the DFA

\[
M = (Q_1 \times Q_2, \Sigma, \delta', (q_0, p_0), F'),
\]

where

\[
F' = (F_1 \times Q_2) \cup (Q_1 \times F_2)
\]

\[
\delta'((q, p), \sigma) = (\delta_1(q, \sigma), \delta_2(p, \sigma)).
\]

Intuitively, \( M \) runs \( M_1 \) and \( M_2 \) in parallel on the same input.

Use the equation

\[
L(M) = \{ w \in \Sigma^* \mid \delta'((q_0, p_0), w) \in F' \}
\]

to show that

\[
L(M) = L_1 \cup L_2.
\]
Partitioning

Start with a DFA $M = (Q, \Sigma, \delta, q_0, F)$.

Suppose that

$$Q = \{q_0, q_1, \ldots, q_{n-1}\}.$$ 

For each state $q_i$, define the associated language

$$L_{q_i} = \{w \in \Sigma^* \mid \tilde{\delta}(q_0, w) = q_i\}.$$ 

Then

$$L_{q_0}, L_{q_1}, \ldots, L_{q_{n-1}}$$

is a partition of $\Sigma^*$.

Why?
Example

Consider the language

\[ L_6 = \{ w \in \{a, b\}^* \mid n_a(w) = 2 \}. \]

One DFA \( M_6 \) that accepts \( L_6 \) is given by

![DFA Diagram]

State \( q_3 \) is a dead or error state.

Find the partition:

\[
L_{q_0} = \text{?} \quad L_{q_1} = \text{?} \\
L_{q_2} = \text{?} \quad L_{q_3} = \text{?}
\]

Observe that

\[ L_6 = L_{q_2}. \]
Example

Another DFA $M'_6$ that accepts $L_6$ is given by

Again, find the partition $\square$?

Express $L_6$ using the partition $\square$?
Complement of Languages

Theorem. For every DFA $M = (Q, \Sigma, \delta, q_0, F)$, there is a DFA that accepts the complement $\Sigma^* - L(M)$.

Proof: The following DFA works:

$$M' = (Q, \Sigma, \delta, q_0, Q - F)$$

Intuitively, $M'$ computes exactly as $M$ does, but rejects when $M$ accepts and accepts when $M$ rejects.

Formally, we have that

$$L(M') = \{ w \in \Sigma^* \mid \hat{\delta}(q_0, w) \in Q - F \}$$

$$= \bigcup_{q_i \in Q - F} L_{q_i}$$

$$= \Sigma^* - \bigcup_{q_i \in F} L_{q_i}$$

$$= \Sigma^* - \{ w \in \Sigma^* \mid \hat{\delta}(q_0, w) \in F \}$$

$$= \Sigma^* - L(M).$$
Exercise

Give a DFA that accepts the following language:

\[ L_8 = \{ w \in \{a, b\}^* \mid n_a(w) \neq 1 \text{ and } n_b(w) \neq 1 \} \).

What is the partition corresponding to your DFA?

Express \( L_8 \) using the partition.