Deterministic Finite Accepter (or Automata) (Read Ch. 2.1-2.2)

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

where

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

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

Transition Diagram:

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

Tabular Format

<table>
<thead>
<tr>
<th>State</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>( q_0 )</td>
<td>( q_1 )</td>
<td>( q_0 )</td>
</tr>
<tr>
<td>( q_1 )</td>
<td>( q_1 )</td>
<td>( q_0 )</td>
</tr>
</tbody>
</table>

Example of a move: \( \delta(q_0, 1) = \)
**Algorithm for DFA:**

Start in start state with input on tape  
$q = $ current state  
$s = $ current symbol on tape  

while \( s \neq \text{blank} \) do  
\[ q = \delta(q, s) \]  
\[ s = \text{next symbol to the right on tape} \]  
if $q \in F$ then accept

Example of a trace: 11010

Pictorial Example of a trace:

1)  
\[ \begin{array}{c|c|c} 1 & 0 & 0 \\ \hline q_0 & q_1 \\ \hline \end{array} \]

2)  
\[ \begin{array}{c|c|c} 1 & 0 & 0 \\ \hline q_0 & q_1 \\ \hline \end{array} \]

Definition:

Configuration: element of $K \times \Sigma^*$

Move between configurations: $\vdash$

Move between several configurations: $\vdash^*$

Examples (from prev FA):

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

$L(M)=\{w \in \Sigma^* \mid (q_0, w) \vdash^*(p, e), p \in F\}$
**Trap State**

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

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.

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

Definition
An NFA = (K, Σ, Δ, q₀, F)
where
- K is finite set of states
- Σ is tape (input) alphabet
- q₀ is initial state
- F ⊆ K is set of final states.
- Δ: subset of K × (Σ ∪ {ε}) × K

Example

```
q₀ ---a--> q₁
 |   |   |   |
|   |   |   |
q₀ ---b--> q₂
 |   |   |   |
|   |   |   |
q₁ ---b--> q₃
```

Note: In this example with state q₀ and input a,

Notation: Δ(q, a) = set of states reachable from q on a

Δ(q₀, a) =

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

Definition (qᵢ, w) ⊢ *(qⱼ, ε) if and only if there is a walk from qᵢ to qⱼ labeled w.

Example From previous example:
What is qⱼ in (q₀, ab) ⊢ *(qⱼ, ε)?

What is qⱼ in (q₁, aba) ⊢ *(qⱼ, ε)?

Definition: For an NFA M, L(M) = \{w ∈ Σ* \mid ∃p ∈ F s.t. (q₀, w) ⊢ *(p, ε)\}

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.
NFA vs. DFA: Which is more powerful?

Example:

![Diagram](image)

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

**Proof:**

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

$K_D = \ldots$  
$F_D = \ldots$  
$\delta_D : \ldots$

**Definition:** $E(q)$ is the closure of the set \{q\}

$E(q) = \{ p \in K \mid (q,e) \vdash *(p,e) \}$

**Algorithm to construct $M_D$**

1. start state is $E(q_0)$
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) apply closure to $B$, $B = E(B)$
   (d) Add state $B$ if it doesn't exist
   (e) add edge from $A$ to $B$ with label $a$
3. Identify final states

Note this proof is different than the proof in the book. In the book instead of starting with the start state, it takes the closure of the start state, including all states reachable on $\epsilon$
Example: