Finite Automata

- basic computational model: limited amount of memory
- example: controller for an automatic door
Finite Automata

Formal definition:
A finite automaton (FA) is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$, where

- $Q$ is a finite set of states
- $\Sigma$ is a (finite) alphabet
- $\delta : Q \times \Sigma \rightarrow Q$ is the transition function
- $q_0 \in Q$ is the start state
- $F \subseteq Q$ is the set of accept states

Pictorial representation: state diagram
Finite Automata

Formal definition:

A **finite automaton** (FA) is a 5-tuple \((Q, \Sigma, \delta, q_0, F)\), where

- \(Q\) is a finite set of **states** \(Q = \{q_0, q_1\}\)
- \(\Sigma\) is a (finite) alphabet \(\Sigma = \{0, 1, 2, 3\}\)
- \(\delta: Q \times \Sigma \to Q\) is the **transition function**
  \[
  \begin{array}{c|cccc}
  \delta(q_0, 0) & \delta(q_0, 1) & \delta(q_0, 2) & \delta(q_0, 3) \\
  \hline
  q_1 & q_0 & q_0 & q_0 \\
  \end{array}
  \]
- \(q_0 \in Q\) is the **start state**
- \(F \subseteq Q\) is the set of **accept states** \(F = \{q_1\}\)

Pictorial representation: **state diagram**
Another (more abstract) example:
- accept all strings over \{0,1\} that start with 1 and end with 0

\begin{align*}
\delta(q_0, 1) &= r_1 \\
\delta(r_1, 1) &= r_2 \\
\delta(r_2, 0) &= r_3 \\
\delta(r_3, 1) &= r_4 \\
\end{align*}

$q_0$ - have not seen anything (E)
$q_1$ - started with 1 and did not end with 0
$q_2$ - started with 1 and end w. 0
$q_3$ - started w. 0

$w = 1101$

$10 \checkmark$

$101010 \times$

$1010100 \checkmark$
Let \( M=(Q, \Sigma, \delta, q_0, F) \) be a FA. The language of \( M \) (accepted / recognized by \( M \)) is \( L(M) \). Formally:

Formally: need the definition of computation:

M accepts \( w=w_1w_2...w_n \) if there exist states \( r_0, r_1,..., r_n \) in \( Q \) such that

- \( r_0 = q_0 \)
- \( \delta(r_i, w_{i+1}) = r_{i+1} \quad \forall i \in \{0,...,n-1\} \)
- \( r_n \in F \)

A language is regular if there exists a FA that recognizes it.
Designing FAs

Examples - languages over \{0,1\} consisting of strings:
- with odd number of 1's \(L_1\)
- that contain 001 as a substring \(L_2\)
- that are even length and do not contain 00 as a substring \(L_3\)

A language that cannot be accepted by a FA?

-\(\{0^k1^\ell \mid \ell \geq 0\}\)
-\(\{0^k \mid k \geq 0 \text{ and } k \text{ is a prime}\}\)
A language that cannot be accepted by a FA?

We will show that \( \{0^{k+1^k} \mid k \geq 0\} \) is not regular.

Suppose there is a FA for

\[
0, 00, 000, 0000, \ldots
\]

since the FA has a finite number of states, there exist \( 0^k, 0^k \) s.t. \( 1^k \) that end up in the same state \( q \).

Then, \( 0^k1^k \) should be accepted, i.e. leads to a state \( r \in F \).

but then \( 0^k1^k \) leads to the same state \( r \) (because from \( q \) on \( 1^k \) we get to \( r \)).

Thus, \( 0^k1^k \) will be accepted as well.

\[ \checkmark \text{ contradiction} \]
Let $A$ and $B$ be languages. The following three language operations are called the **regular operations**:

- **union**: $A \cup B$
- **concatenation**: $A.B$
- **star**: $A^*$

The natural numbers are closed under multiplication but not division.

What about the class of regular languages?
Thm 1.25: The class of regular languages is closed under the union operation.

For example:

\[ L_1 = \{ \text{w} \in (a,b)^* \mid \text{w contains even # of a's} \} \]

\[ L_2 = \{ \text{w} \in (a,b)^* \mid \text{w contains odd # of a's} \} \]

\[ L_1 \cup L_2 = \{ \text{w} \in (a,b)^* \mid \text{w contains even # of a's or odd # of a's} \} \]
Thm 1.25: The class of regular languages is closed under the union operation.

**Proof:**

Let $M_1 = (Q_1, \Sigma, \delta_1, q_{1}, F_1)$

$M_2 = (Q_2, \Sigma, \delta_2, q_{2}, F_2)$

Construct

$M = (Q, \Sigma, \delta, q_{0}, F)$

where

$Q = \{ (q_1, q_2) \mid q_1 \in Q_1, q_2 \in Q_2 \} = Q_1 \times Q_2$

$q_0 = (q_{1}, q_{2})$

$\delta((q_1, q_2), a) = (\delta_1(q_1, a), \delta_2(q_2, a)) \quad \forall (q_1, q_2) \in Q$

$F = F_1 \times Q_2 \cup Q_1 \times F_2$

Note: if $F = F_1 \times F_2$ then we get $\cap$ → i.e. regular languages are closed under
Thm 1.26: The class of regular languages is closed under the concatenation operation.