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

\[ q_1 \xrightarrow{1} q_2 \xrightarrow{0} q_3 \]

- \( q_0 \) : \( \epsilon \)
- \( q_1 \): all strings starting w/ 1 and ending w/ 1
- \( q_2 \): all strings starting w/ 0
- \( q_3 \): all strings starting w/ 1 and ending w/ 0
Let $M = (Q, \Sigma, \delta, q_0, F)$ be a FA. The language of $M$ (accepted / recognized by $M$) is $L(M)$.

Formally: need the definition of computation:

**$M$ accepts** $w = w_1 w_2 \ldots w_n$ if there exist states $r_0, r_1, \ldots, r_n$ in $Q$ such that

- $r_0 = \lambda q_0$
- $\delta(r_i, w_{i+1}) = r_{i+1}$ for all $i \in \{0, 1, 2, \ldots, n-1\}$
- $r_n \in F$

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
- that contain 001 as a substring
- that are even length and do not contain 00 as a substring

A language that cannot be accepted by a FA?

E.g. the language of all palindromes over \{0,1\}:
\{a^k b^k \mid k \geq 0\}
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.

$$x, y \in \mathbb{N} \quad x \cdot y \in \mathbb{N}$$

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

 Means: for any two FA $M_A = (Q_A, \Sigma, \delta_A, q_0A, F_A)$ and $M_B = (Q_B, \Sigma, \delta_B, q_0B, F_B)$, we will construct $M = (Q, \Sigma, \delta, q_0, F)$ such that $L(M) = L(M_A) \cup L(M_B)$.

Example:

In general, let $M$ be:

\[
Q = Q_A \times Q_B \quad \text{and} \quad q_0 = (q_0A, q_0B) \quad \text{and} \quad \delta((p,q), \sigma) = (\delta_A(p,\sigma), \delta_B(q,\sigma)) \quad \forall \sigma \in \Sigma \quad \forall p \in Q_A \quad \forall q \in Q_B
\]

\[
F = \{ (p,q) \mid p \in F_A \text{ or } q \in F_B, p \in Q_A, q \in Q_B \} = Q_A \times F_B \cup F_A \times Q_B
\]
Regular languages are closed under ... [Section 1.1]

Thm 1.25: The class of regular languages is closed under the intersection operation.

Proof: the same construction as for the union, except

\[ F = F_A \times F_B \]

Think: is the class of regular languages closed under difference?

\[ L(M_A) - L(M_B) \]
Thm 1.26: The class of regular languages is closed under the concatenation operation.

Want: given $M_A, M_B$, construct $M$ s.t. $L(M) = L(M_A) \cdot L(M_B)$

Example:

$A = \{ w \in \{0, 1\}^* \mid w \text{ contains an even \# 0's} \}$

$B = \{ v \in \{0, 1\}^* \mid v \text{ starts with 1} \}$

$0001010$