Finite Automata

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

2 states: closed, opened

input:
- neither sensor
- front but not back
- back but not front
- both

$\Sigma = \{0, 1, 2, 3\}$

in a course of a day, the signals are:
0 1 1 2 3 0 2 0 1 1 0
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

Meaning:
- \( q_0 \): haven't seen anything (E)
- \( q_1 \): starts with 1 and last symbol 1
- \( q_2 \): starts with 1 and last symbol 0
- \( q_3 \): start with 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

- $r_0 = q_0$
- $\delta(r_k, w_i) = r_{k+1}$ $\forall k \in \{0, \ldots, n-1\}$
- $r_n \in F$

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 \( L_1 \)
- 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?

\[ L = \{0^k1^k \mid k \geq 0\} \]
Let $A$ and $B$ be languages. The following three language operations are called the **regular operations**:

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

The natural numbers are closed under multiplication but not division.

What about the class of regular languages?

Is the class of regular languages closed under $\cup$?

means: if $A, B$ are regular languages, is $A \cup B$ always regular? **YES**
Thm 1.25: The class of regular languages is closed under the union operation.

Example: 

\[ A = \{ w \in \{0,1\}^* \mid \text{\#1's in } w \text{ is odd} \} \]

\[ B = \{ w \in \{0,1\}^* \mid \text{\#0s in } w \text{ is even} \} \]

**Proof:** (sketch of construction)

We describe \( M = (Q, \Sigma, \delta, q_0, F) \) for \( A \cup B \):

- \( Q = Q_A \times Q_B \)
- \( q_0 = (q_{A0}, q_{B0}) \)
- \( F = F_A \times Q_B \cup Q_A \times F_B \)
- \( \delta((r_{A}, r_{B}), \sigma) = (\delta_A(r_{A}, \sigma), \delta_B(r_{B}, \sigma)) \) where \( \sigma \in \Sigma \)

\[ \forall r_{A} \in Q_A \land \forall \sigma \in \Sigma \land \forall r_{B} \in Q_B \]

Note: if \( F = F_A \times F_B \), then \( M \) accepts \( A \cap B \) ⇒ regular languages are closed under intersection.
Thm: The class of regular languages is closed under the complement operation.

Example: \[ L = \{ w \in \{0, 1\}^* \mid |w| \text{ is divisible by 3} \} \]

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

Want to construct \( M_1 = (Q_1, \Sigma, \delta_1, q_{10}, F_1) \) for \( \overline{L} \)

let
- \( Q_1 = Q \)
- \( \delta_1 = \delta \)
- \( q_{10} = q_0 \)
- \( F_1 = Q - F \)

\( \square \) end of construction
Thm 1.26: The class of regular languages is closed under the concatenation operation.

Suppose we have

\[ A = \{ w \in \{0,1\}^* \mid \text{number of 1's in } w \text{ is odd} \} \]

\[ B = \{ w \in \{0,1\}^* \mid |w| \text{ is divisible by } 3 \} \]

want to construct an FA for \( A \cdot B \)

idea: run \( M_A \), then, when in an accepting state, magically (on \( a \)) jump to the initial state of \( M_B \) and finish the computation by \( M_B \)