Finite Automata

A finite *automaton* $A = (S, I, \Sigma, T, F)$ consists of

- a set of states S
- a set of initial states $I \subseteq S$
- an input alphabet Σ
- a transition relation $T \subseteq S \times \Sigma \times S$
- a set of final states $F \subseteq S$

The *language* $L(A)$ of A is the set of words accepted by A.

An automaton is called *complete* iff

- it has at least one initial state
- every state has at least one outgoing transition for every $e \in \Sigma$.

An automaton is called *deterministic* iff

- it has at most one initial state
- every state has at most one outgoing transition for every $e \in \Sigma$.

Power Automaton $\mathbb{P}(A) = (\mathbb{P}(S), I_p, \ldots)$ is deterministic and complete with $L(A) = L(\mathbb{P}(A))$.

Complement Automaton $C(A) = (\ldots, S \backslash F)$ is the result of flipping the final states of A. $L(C(A)) = \overline{L(A)}$ iff A is complete and deterministic.

Oracle Automaton $\text{Oracle}(A) = (S, I, \Sigma \times S, T_O, F)$ where transitions are now pairs $(e \in \Sigma, s \in S)$ with e being the previous alphabet value and s being the destination state. It is deterministic iff $|I| \leq 1$ and usually not complete.

Optimized Oracle Automaton is a modification of $\text{Oracle}(A)$ which is deterministic *and* complete iff $|I| \leq 1$. This is done by replacing destination states in the transition pairs with numbers and adding transitions where necessary.

Product Automaton $A_1 \times A_2$ of A_1 and A_2 accepts $L(A) = L(A_1) \cap L(A_2)$