Lecture 6
DFA vs. NFA
Instructor: Hossein Hojjat
February 3, 2017
Recap: Finite State Automaton

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

- \( \Sigma \) alphabet
- \( Q \) states (nodes in the graph)
- \( q_0 \in Q \) initial state (with \( \rightarrow \) sign in drawing)
- \( \delta \subseteq Q \times \Sigma \times Q \) transitions (labeled edges in the graph)
- \( F \subseteq Q \) final states (double circles)

\[ \delta = \{(q_0, a, q_0), (q_0, b, q_1), (q_1, a, q_1), (q_1, b, q_1)\} \]
Question

- Design an automaton that recognizes strings over $\Sigma = \{0, 1\}$ that do not contain the substring 010
Question

- Design an automaton that recognizes strings over $\Sigma = \{0, 1\}$ that do not contain the substring $010$

Answer
Question

• Design an automaton that recognizes strings over $\Sigma = \{0, 1\}$ that end in the substring 01
Question

- Design an automaton that recognizes strings over $\Sigma = \{0, 1\}$ that end in the substring 01

Answer
Question

- Design an automaton that recognizes all numbers written in binary that are divisible by 2.
  For example, the automaton should accept the words 0, 10, 100, 110, …
  (leading zeros are ok)
Finite State Automaton (Exercise)

Question

- Design an automaton that recognizes all numbers written in binary that are divisible by 2.
  For example, the automaton should accept the words 0, 10, 100, 110, …
  (leading zeros are ok)

Answer
### Types of Finite State Automata

- **Deterministic Finite Automata (DFA)**
  - $\delta$ is a function $(Q, \Sigma) \mapsto Q$
  - One transition per input per state
  - All examples so far

- **Nondeterministic Finite Automata (NFA)**
  - $\delta$ is a function $(Q, \Sigma) \mapsto 2^Q$
  - Can have multiple transitions for one input in a given state

![Diagram of a simple DFA and NFA]
Computations of a DFA

- For each input string there is exactly one path in a DFA
For an input string there are multiple possible computation paths in an NFA.

Word is accepted if there is a path in the computation tree that leads to an accepting state.
Undefined transitions go to a trap state where no input can be accepted
Epsilon transition allows an NFA to change its state spontaneously without consuming any symbol from input.

Example
NFA that accepts all strings of the form $0^k$ where $k$ is a multiple of 2 or 3.
DFA vs. NFA

DFA:

- NFA for a language can be smaller and easier to construct than DFA
- An implementation of an NFA normally has backtracking
- An implementation of a DFA normally requires only as many steps as the input length

NFA:
Exercise

Question

- Construct an NFA that recognizes all strings over $\Sigma = \{a, b, c\}$ that do not contain all the alphabets $a$, $b$ and $c$. 
Question

- Construct an NFA that recognizes all strings over $\Sigma = \{a, b, c\}$ that do not contain all the alphabets $a$, $b$ and $c$.

Answer

- Let’s start with a regular expression

$$ (a|b)^* \mid (b|c)^* \mid (a|c)^* $$

![NFA Diagram](image-url)