Deterministic Finite Automata Part II

Based on slides of Aaron Deever

Deterministic Finite Automata

- Consist of:
  - A set of states
  - A start state
  - A set of accepting states
  - Input alphabet
  - Transition function
- Let's define an automaton formally

Deterministic Finite Automata

- A deterministic finite automaton (finite-state machine) is a 5-tuple $(Q, \Sigma, \delta, q_o, F)$ where
  - Q is a finite set (of states)
  - Σ is a finite alphabet of symbols
  - δ is a function from $Q \times \Sigma$ to Q (transition function)
    - $\delta : Q \times \Sigma \rightarrow Q$
  - $q_o \in Q$ is the start state
  - $F \subseteq Q$ is the set of final states

Transition Function

- The transition function
  - δ is a function from Q x Σ to Q
  - $\delta\ (q, a) = q'$ where
    - q, q' $\in Q$
    - $a \in \Sigma$
  - δ defines, given a current state q and input symbol a, to which state the DFA will move.

Language Accepted by a DFA

- Let M = $(Q, \Sigma, \delta, q_o, F)$ be a DFA
- A string $w = w_1 w_2 \ldots w_n$ is <u>accepted</u> by M if:
  - A sequence of states $r_0 r_1 \ldots r_n \in Q$ exists with the following conditions:
    - $r_0 = q_0$
    - $\delta (r_i, w_{i+1}) = r_{i+1}$ for i = 0, …, n–1
    - $r_n \in F$
  - If a string w is not accepted by M it is said to be <u>rejected</u> by M.

Language Accepted by a DFA

- In other words:
- A string $w \in \Sigma^*$ is accepted by M if:
  - Starting in the start state q_0
  - Running the machine with input w
  - The machine ends up in a final (accept) state
  - If a string w is not accepted by M it is said to be rejected by M.

Language Accepted by a DFA

- The language <u>accepted</u> or <u>recognized</u> by M is:
  - $L(M) = \{ w \in \Sigma^* \mid w \text{ is accepted by } M \}$
- If A is a language over Σ, A is accepted by M if and only if $A = L(M)$.
  - For all $w \in A$, w is accepted by M.
  - For all $w \notin A$, w is rejected by M.

Regular Languages

- Definition:
  - Language A is regular if and only if there exists a finite automaton M such that:
    - A = L(M)
  - All regular languages are accepted by some finite automaton
  - Are all languages regular?
    - No – (we'll get to examples later of languages that are not regular – there is no way to represent them with a finite automaton)

Example using Formal Definition

- Sometimes you can't use a state diagram to represent a finite automaton
  - Number of states might be impractically large
  - The automaton might depend on an unspecified parameter
- Example:
  - Let $\Sigma = \{0, 1, 2\}$
  - For each $i > 0$, let A_i be the language of all strings where the sum of the numbers is a multiple of i.
    - (for simplicity, define empty string to have sum 0 so it is accepted)
  - How to describe the machine M_i that recognizes A_i ?

A Non-Regular Language

- $L = \{0^n 1^n \mid n \geq 0\}$

- How might we prove that L is not regular?
  - Proof by contradiction
  - If L is regular, there is some finite state machine that recognizes L
  - Because there are only finite states, as more and more 0's are read, eventually a state must repeat
    - Suppose reading either 0^j or 0^k ends in a common state
    - Should the machine end in an accepting state if it subsequently reads 1^k?
      - YES and NO – this is a contradiction

Summary

- What we've learned so far:
  - A DFA can be expressed by $(Q, \Sigma, \delta, q_o, F)$
  - Transition function: δ: $Q \times \Sigma \rightarrow Q$
  - DFA accepting a string
  - The language accepted by a DFA
  - All languages accepted by some DFA are regular
  - Not all languages are regular