Lecture 2 Overview

- Finite Automaton (DFA): definition, examples
- Regular languages
- Introduction to Nondeterministic Finite Automata (NFAs)

Finite Automaton (DFA)

- Simple computational model
- Models a computer with a very limited amount of memory

Ex. 1) Finite automaton that accepts only the key word “MAIN”.
Ex. 2) Finite automaton that accepts all binary strings that end in a 0.
Ex. 3) Vending Machine
Applications of DFAs

- **Recognize patterns in strings** (e.g., key words, variable names, constants in programming languages)
  This is used in
  - Text editors (e.g., Emacs)
  - UNIX tools (e.g., grep)
  - Scripting languages (e.g., Perl)
  - Compilers: Lexical analysis phase
    Strings of symbols are partitioned into tokens of the language
    \[
    \text{const MAX} = 1000 \implies \text{const } \text{MAX} \equiv 1000
    \]

Applications of DFAs

- **Keep track of limited state information** (e.g., vending machines)
  This is used in
  - Control systems (e.g., automatic door, refrigerator light control, elevator)
  - Games programming
  - Circuit design
Original Question

What problems can be solved by ... DFAs?
In order to answer this, we must define DFAs precisely.
First we will need some terminology and notation.

Terminology and Notation

- An **alphabet** is a finite set. Its members are called symbols.
  \[ \Sigma_1 = \{A, B, \ldots, Z\} \]
  \[ \Sigma_2 = \{0, 1\} \text{ (binary alphabet)} \]
  \[ \Sigma_3 = \{n, d\} \]

- A **string** is a finite sequence of symbols from an alphabet.
  \text{MAIN} is a string over \( \Sigma_1 \)
  \text{10101110} is a string over \( \Sigma_2 \)
  \text{ndddddn} is a string over \( \Sigma_3 \)
Terminology and Notation

- The length of a string $s$ is the number of symbols in $s$. It is denoted $|s|$.
  
  $|\text{MAIN}| = 4$
  
  $|10101110| = 8$
  
  $|\text{ndddd}n| = 6$

- The empty string, denoted $\varepsilon$, is a string of length 0.

- The concatenation of strings $x$, $y$ is the string $xy$.
  
  $x = \text{chalk}$, $y = \text{board} \Rightarrow xy = \text{chalkboard}$
  
  $x^2 = \text{chalkchalk}$, $xy^3 = \text{chalkboardboardboard}$
  
  $x\varepsilon = x$, $\varepsilon x = x$

7

Terminology and Notation

- A language is a set of strings.
  
  $L_1 = \{\text{MAIN}\}$
  
  $L_2 = \{w \mid w \text{ is a binary string ending in 0}\}$
  
  $L_3 = \{\text{nd, dn, dd, nnn, nnd, \ldots}\}$

- Given an alphabet $\Sigma$, $\Sigma^*$ denotes the set of all strings over $\Sigma$, including $\varepsilon$.
  
  $\{0, 1\}^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, 001, 010, 011, \ldots\}$

  $\Sigma^*$ is a language.
  
  A language $L$ over alphabet $\Sigma$ is a subset of $\Sigma^*$, i.e., $L \subseteq \Sigma^*$.  

8
Formal Definition of Finite Automaton

Def: A finite automaton (or DFA) is a 5-tuple $M = (Q, \Sigma, \delta, q_0, F)$, where

1. $Q$ is a finite set of states
2. $\Sigma$ is an alphabet
3. $\delta : Q \times \Sigma \rightarrow Q$ is the transition function
4. $q_0 \in Q$ is the start state
5. $F \subseteq Q$ is the set of accept (final) states.

Note that $Q$, $\Sigma$ and $F$ are sets, $\delta$ is a function, and $q_0$ is a state.

Examples of DFAs

Ex. 1) DFA that “accepts” only the key word “MAIN”
$M_1 = (Q, \Sigma, \delta, q_0, F)$, where

1. $Q = \{q_0, q_1, q_2, q_3, q_4, q_5\}$
2. $\Sigma = \{A, \ldots, Z\}$
3. $\delta(q_0, M) = q_1$, $\delta(q_0, s) = q_5$ for every $s \in \Sigma \setminus \{M\}$ $\delta(q_1, A) = q_2$, $\delta(q_1, s) = q_5$ for every $s \in \Sigma \setminus \{A\}$
4. $q_0$
5. $F = \{q_4\}$
Examples of DFAs

Ex. 2) DFA that “accepts” all binary strings ending in 0

\[ M_2 = (Q, \Sigma, \delta, q_0, F) \], where

1. \( Q = \{q_0, q_1\} \)
2. \( \Sigma = \{0, 1\} \)
3. \( \delta(q_0, 0) = q_1 \)
   \( \delta(q_0, 1) = q_0 \)
   \( \delta(q_1, 0) = q_1 \)
   \( \delta(q_1, 1) = q_0 \)
4. \( q_0 \)
5. \( F = \{q_1\} \)

Formal Definition of Computation for DFAs

Let \( M = (Q, \Sigma, \delta, q_0, F) \) be a DFA and \( w = w_1w_2 \cdots w_n \) a string over \( \Sigma \). \( M \) accepts \( w \) if there is a sequence of states \( r_0, r_1, \ldots, r_n \in Q \) such that

1. \( r_0 = q_0 \) (starts right)
2. \( \delta(r_i, w_{i+1}) = r_{i+1} \) for \( i = 0, \ldots, n-1 \) (moves right)
3. \( r_n \in F \) (ends right)

Note that for all \( w \in \Sigma^* \), \( M \) either accepts \( w \) or rejects \( w \).
Formal Definition of Computation for DFAs

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA. $M$ recognizes a language $L$ if $L = \{w \in \Sigma^* \mid M \text{ accepts } w\}$, i.e.,

For all $w \in \Sigma^*$:

- $w \in L \Rightarrow M \text{ accepts } w$
- $w \notin L \Rightarrow M \text{ rejects } w$

Notation: $L(M)$ denotes the language recognized by $M$.

Regular Languages

**Def:** A language $L$ is a **regular language** if there exists a DFA $M$ that recognizes $L$.

**Ex.** 1) $L_1 = \{\text{MAIN}\}$ is regular because $M_1$ recognizes it.

**Ex.** 2) $L_2 = \{w \mid w \text{ is a binary string ending in } 0\}$ is regular because $M_2$ recognizes it.

**Problem:** Let $\Sigma = \{0, 1\}$. Prove that $L = \{w \mid w \text{ contains the substring } 1010\}$ is regular.

**DIRECT PROOF – Proof by Construction**
Regular Languages

Problems

1. Prove that \( L_1 = \emptyset \) is regular.
2. Let \( \Sigma = \{0, 1\} \). Prove that 
   \( L_2 = \{ w \mid w \text{ contains at least three } 1\text{s} \} \) is regular.
3. Let \( \Sigma = \{0, 1\} \). Prove that 
   \( L_3 = \{ w \mid w \text{ contains a } 1 \text{ in the } 3\text{rd position from the end} \} \) is regular.

Nondeterministic Finite Automaton (NFA)

Nondeterminism allows several possible next states at every step.

Determinism:  \hspace{1cm} \text{Nondeterminism:}

\[ \text{one next state per } s \hspace{1cm} \text{set of next states per } s \]