CS 4510: Automata and Complexity 08/23/2022

Lecture 1: Introduction

Lecturer: Zvi Galil *Author: Austin Peng*

Contents

1 Topics To Be Familiar With:

1.1 Sets

- sets are elements drawn from some universe:
  - $\mathbb{Z}$ (integers)
  - $\mathbb{N}$ (natural numbers)
  - $\mathbb{R}$ (real numbers)
  - $\mathbb{C}$ (complex numbers)
- set operations:
  - union: $A \cup B$
  - intersection: $A \cap B$
  - complement: $\overline{A}$
  - Cartesian product: $A \times b$ (pairs)
  - power: A^k (k-tuples)

1.2 Functions

- functions are mappings from one set to another:
  - $f : D \to R$
    $f(x) = x^2$
  - $f : Z \times Z \to Z$
    $f(x, y) = x + y$

1.3 Relations

- relation R can be written as xRy
  - **reflexive**: xRx
  - **symmetric**: $xRy \implies yRx$
  - **transitive**: xRy and $yRz \implies xRz$
  - **equivalence**: if relation R is reflexive, symmetric, and transitive

1.4 Graphs

- a graph $G = (V, E)$ has vertices V and edges E
- a graph may be directed or undirected

1.5 Strings

- ”abc” is a string
- a string is a sequence of symbols from an alphabet Σ
- Σ^* is the set of all strings over Σ
- Σ^* contains the empty string ε, all strings of length 1, all strings of length 2, etc.

1.6 Boolean Logic

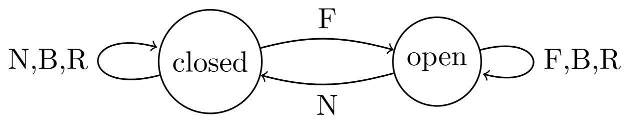
- 3 main operations: ”and” ($\wedge$), ”or” ($\vee$), ”not” ($\neg$)
- these operations are performed on variables and constants (true and false)

1.7 Proofs

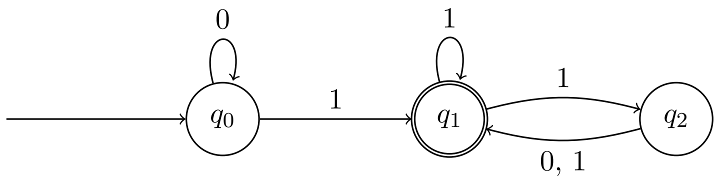
- mathematics consists of definitions, statements/lemmas/theorems, and proofs
- 3 common types of proofs:
  - proof by construction
  - proof by contradiction
  - proof by induction
- these operations are performed on variables and constants (true and false)

2 Finite Automata and (Regular Languages)

Consider an automatic door with the inputs $\{\mathrm{F(ront)}, \mathrm{N(either)}, \mathrm{B(oth)}, \mathrm{R(ear)}\}$. The door has 2 states: open and closed. The door will only open if there is someone in front and not in the rear (door swings inward). If the door is open, but nobody is there, the door will close.



Example 1. M_1



- q_0 is the starting state
- q_1 is the accepting statements
- all transitions ($\delta = 0, 1$) are defined for all states
- let x be a binary string 1101, M_1 will accept x because it ends at q_1
- $L(M_1)$ = all x such s.t M_1 accepts x if x has a 1 AND either ends with a 1 or with an even number of 0s

2.1 Formal Definition Of A Deterministic Finite Automaton

Definition 1. A deterministic finite automaton is a 5-tuple $M = (Q, \Sigma, \delta, q_0, F)$ where:

- Q is a finite set called the states
- Σ is a finite set called the alphabet
- $\delta : Q \times \Sigma \to Q$ is the transition function
- q_0 is the initial state
- $F \subseteq Q$ is the accepting states

Note: $\delta(q, a) = p$ means that if DFA is in state q and sees a, it goes to state p

2.2 Formal Definition Of Computation

Definition 2. A **computation** of a deterministic finite automaton (DFA) M on an input string $x = x_1, ..., x_n \in \Sigma^*$ is a sequence of states $q_0, ..., q_n \in Q$ s.t. for $i = 0, ..., n-1$ $\delta(q_i, x_{i+1}) = q_{i+1}$. We say that M accepts x if $q_n \in F$. If $q_n \notin F$, then we say M rejects x.

2.3 Formal Definition Of A Language

Definition 3. The **language** of M is defined as the set $\{x \in \Sigma^* : M \text{ accepts } x\}$. A language is a set of strings.

Example 2. $\emptyset, \{\varepsilon\}$ are both langauges.

Definition 4. A language L is called **regular** if there is a DFA accepting L.

Example 3. $L_w = \{w\}$ is regular. $L'_w = \{\text{all strings that end in } w\}$ is also regular.