FORMAL LANGUAGES AND AUTOMATA 22AIE302

Ayswarya R Kurup

10 July 2024

About the Course

References

1. Formal Language and Automata', Peter Linz, Fifth edition, 2012.
2. Introduction to Automata Theory, Languages and Computation', J.E.Hopcroft, R.MotwaniandandJ.D.Ullman, Pearson, 2001.
3. Elements of the Theory of Computation', H.R.Lewis and C.H.Papadimitriou, Prentice Hall, 1997/Pearson 1998.

Course Outcomes

1. Analyse formalisms and write formal proofs for properties
2. Use grammatical notations to represent sequence manipulation problems
3. Apply various formal grammars to the problem-solving avenues
4. Identify limitations of some computational models and possible methods of proving them

Evaluation Pattern

Assessment	Weightage(%)
Assignments	30
Quiz with equal credits	20
Midterm Examination	20
End Semester Examination	30

Pre-requisites

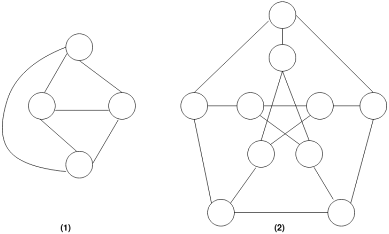
- Data Structures
- Discrete Mathematics

Syllabus I

1. Introduction to Automata and formal language: Finite State machines – Deterministic finite state machine, Non-Deterministic finite state machine, Equivalence of NFA and DFA, Minimization of Finite State Machine, Regular Expression, Regular Language, Properties of Regular Languages.
2. Context Free Grammar: Pushdown Automata, Variants of Pushdown automata, Derivations Using a Grammar, Leftmost and Rightmost Derivations, the Language of a Grammar, Sentential Forms, Parse Tress Equivalence between PDA and CFG, Context Free Language, Properties of CFL, Normal Forms.
3. Context Sensitive Language: Linear Bound Automata, Turing Machine, Variants of Turing Machine, Decidability, Post correspondence problem, Introduction to undecidable problems.

Why Formal Languages and Automata?

1. **3**-coloring problem (Graph coloring):
   Determine whether it is possible to color using three colors $\{1, 2, 3\}$provided no adjacent nodes have same color



(1) (2)

2. A computer with input X: do the program halt on X ?
3. Shortest path/ route problem:
   " I want to drive from Ettimadai to Coimbatore", which is the shortest route?

- Every computational problem can be represented as a string input to a computer / device
- String will be over an alphabet denoted by Σ
- Problem can be viewed as a decision problem

Alphabets

- Set of symbols: a finite set, Σ
- Examples:
  - Binary: $\Sigma = \{0, 1\}$
  - Decimal: $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$
  - All lower case letters: $\Sigma = \{a, b, c, ..., x, y, z\}$
  - Alphanumeric: $\Sigma = \{a - z, A - Z, 0 - 9\}$
  - DNA molecule letters: $\Sigma = \{a, c, g, t\}$
- $N = \{0, 1, 2, 3, ...\}$ cannot be an alphabet since it is infinite

Strings

- Finite collection of symbols from chosen from alphabet set Σ
- A string of length n over an alphabet Σ is an ordered n-tuple of elements of Σ
- $\Sigma = \{a, b\}$ then $\{\epsilon, ba, bab, aab\}$ are examples of strings over Σ
- If $\Sigma = \{a\}$ then $\Sigma^* = \{\epsilon, a, aa, aaa, ...\}$
- Length of a string w: $|w|$ - number of symbols
- Example: $w = 010100$ $|w| = 6$
- If $|w| = 0$, ϵ, empty string (epsilon) or λ (lambda)
- $\epsilon w = w\epsilon = w$

String Operations

1. Concatenation $(x = ab, y = bc, xy = abbc)$
2. Reversal $(x = ab, x^R = ba)$
3. ϵ denotes empty string $|\epsilon| = 0$
4. $x^K = xxx....$: x is repeated or concatenated k times
   $$x^0 = \epsilon$$
5. **Kleene star** of x, denoted by x^*
   $x^* =$ set of all x^k $= \{x^k | k \geq 0\}$
6. Σ^*: Set of all strings over Σ of finite length
7. Substring: V is a substring of W, if there exist strings X and Y
   $$W = XVY$$
   Example: "get" is a substring of "together"
8. A language over Σ is a set of strings over Σ
   $$A \subseteq \Sigma^*$$

String Operations..

" A language over Σ is a set of strings over Σ"

Examples: set of all binary strings with an odd number of 1's is a language over $\{0, 1\}$

Set of all dictionary words is a language over the English alphabet

Note: Σ^* is also a language

ϵ is also a language

Describing languages

1. Brute Force Listing: $\{a, ab, abb, ..\}$
2. Language Operations: ab^*
3. Other set theoretic operations

Set Theory

- $A = \{a, b, c\}$
  - A is the set whose elements a, b, and c
- $W = \{x : x \in N\}$
  - **W** equals the set of all **x** such that **x** is a natural number
- Empty set: $\emptyset = \{\}$

Set Operations

- Union: $A \cup B$
- Intersection: $A \cap B$
- Elements that are in A but not in B: $A - B$ (Complement of B relative to A)
- Complement of A: A^c
- A is a subset of B: $A \subset B$
- Cartesian product: $A \times B$ (set of all ordered pairs in the form (a, b))
- Function: $f : A \to B$
  (* Function from A to B is a subset of $A \times B$)

Set Theory..

Power Set

- $P(X)$ is the power set of X
  - Collection of all subsets of X
- $|X|$ is the number of elements in the set X
  - $|P(X)| = 2^{|X|}$
    - $A = \{x, y, z\}$
    - $P(A) = \{\{x\}, \{y\}, \{z\}, \{x, y\}, \{y, z\}, \{x, z\}, \{x, y, z\}, \{\}\}$

AUTOMATA

- Automata Theory: Study of abstract computing devices
- Abstract devices are (simplified) models of real computations
- Computations happen everywhere: On your laptop, on your cell phone, in nature, ...
- Used in model-checking

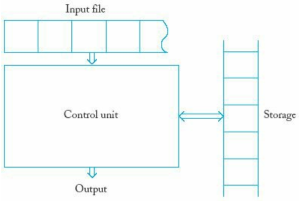
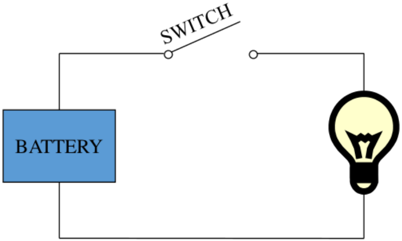


Figure: General model of an automata

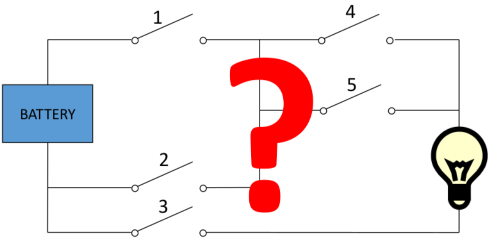
- Accepter: Automaton whose output response is limited to a simple "yes" or "no"
- Transducer: Automaton, capable of producing strings of symbols as output

A simple circuit



- **Input:** switch
- **Output:** light bulb
- **Actions / transitions :** flip switch
- **States:** on, off

A DESIGN PROBLEM



Can you design a circuit where the light is on if and only if all the switches were flipped exactly the same number of times?

A DESIGN PROBLEM...

- Such devices are difficult to reason about, because they can be designed in an infinite number of ways
- By representing them as abstract computational devices, or automata, we will learn how to answer such questions

Various types of Automata

Finite automata	Devices with a finite amount of memory. Used to model "small" computers.
Push-down automata	Devices with infinite memory that can be accessed in a restricted way. Used to model parsers, compiler for programming languges etc.
Turing Machines	Devices with infinite memory. Used to model any computer.
Time-bounded Turing Machines	Infinite memory, but bounded running time. Used to model any computer program that runs in a "reasonable" amount of time.

Finite Automata

- Computers with a limited amount of memory
- State based devices
- Examples: Timers, Door open / close, thermostat
- States acts as the memory

WHY STUDY FINITE AUTOMATA ?

- Used for design and verification of circuits and communication protocols
- Used for text-processing applications like text editors, t
- An important component of compilers - lexical analysers in programming languages
- Network Protocol Analysis
  - Ex: variable checking
- Identify simple patterns of events - DNA sequencing

Deterministic Finite Automata (DFA)

- For every combination of current state and an input symbol, there is precisely one next state
- Definition: A deterministic finite automata is defined by a 5-tuple
  $$(Q, \Sigma, \delta, q_0, F)$$
  - Q is a finite set of states
  - Σ is a finite alphabet
  - $\delta : Q \times \Sigma \rightarrow Q$ is the transition function
  - $q_0 \in Q$ is the start state
  - $F \subseteq Q$ is the set of accepting states

DFA..

$M = (Q, \Sigma, \delta, q_0, F)$ accepts a string $w = w_1, w_2, ..w_n$ if there is a sequence of states $r_0, r_1, ...r_n$ iff

1. $r_0 = q_0$
2. $\delta(r_i, w_{i+1}) = r_{i+1}, \forall\ 0 \leq i \leq n-1$
3. $r_n \in F$

M recognizes the language A if

$$A = \{w|\ M \text{ accepts } w\}$$

denoted by $L(M) = A$

Languages

- The language is a collection of appropriate strings, denoted by L
- L is a language over alphabet set Σ, only if $L \subseteq \Sigma^*$
- Examples:
  - If L takes all possible strings consisting of n 0's followed by n 1's over $\Sigma = \{0, 1\}$:
  - S $= \{\epsilon, 01, 00011, 0011, 0101, 000111, ...\}$
  - L $= \{\epsilon, 01, 0011, 000111, ...\}$ $\quad L \subseteq \Sigma^*$
- If L takes all possible strings of with equal number of 0's and 1's over $\Sigma = \{0, 1\}$:
- $L = \{\epsilon, 01, 10, 0011, 1100, 0101, 1010, 1001, ...\}$ $L = \{\}$,
- $\emptyset$ denotes the Empty language

Regular Language

- A language is regular if it is recognized by some finite automaton
- The collection of all strings that are recognized by a finite automata