Unit-I
Basic Foundations

Prepared By:
Ghanashyam BK

Contents

- Review of Set Theory, Logic, Functions, Proofs
- Automata, Computability and Complexity:
  - Complexity Theory, Computability Theory, Automata Theory
- Basic concepts of Automata Theory:
  - Alphabets,
  - Power of Alphabet,
  - Kleen Closure Alphabet,
  - Positive Closure of Alphabet,
  - Strings,
  - Empty String,
  - Substring of a string,
  - Concatenation of strings,
  - Languages,
  - Empty Language

Review of Set Theory

- **Sets**
  - a collection of well defined objects.
  - the element of a set has common properties.
  - e.g. all the student who enroll for a course “theory of computation” make up a set.
- **Examples**
  - The set of even positive integer less than 20 can be expressed by
    - E = {2,4,6,8,10,12,14,16,18}

      Or
    - E = {x|x is even and 0<x<20}

Review of Set Theory

- **Finite and Infinite Sets**
  - A set is finite if it contains finite number of elements.
  - And, infinite otherwise
  - The empty set has no element and is denoted by ϕ
- **Cardinality of set**
  - It is a number of element in a set.
  - The cardinality of set E is

    $$|E|=9$$
- **Subset**
  - A set A is subset of a set B if each element of A is also element of B and is $A\subseteq B$ ted by

Review of Set Theory

<u>Set operations</u>

- **Union:**
  - The union of two set has elements, the elements of one of the two sets and possibly both.
  - Union is denoted by +
- **Intersection**
  - The intersection of two sets is the collection of all elements of the two sets which are common
  - is denoted by .

Review of Set Theory

Set operations

- **Differences**
  - The difference of two sets A and B.
  - denoted by A-B
  - e set of all elements that are in the set A but not in the set B.

Review of Set Theory

- **Sequences and Tuples**
  - sequence of objects is a list of objects in some order.
  - For example, the sequence 7,4,17 would be written as (4,7,17)
  - In set the order does not matter but in sequence it does.
  - repetition is not permitted in a set but is allowed in a sequence
- **Relations**
  - A binary relation on two sets A and B is a subset of A×B
  - for example, if A={1,3,9}, B={x,y}, then {(1,x),(3,y),(9,x)} is a binary relation on 2- sets.

Logic

- Computers represent information using bits
- A bit is a symbol with two possible values, 0 and 1.
- By convention, 1 represents T (true) and 0 represents F (false)
- A variable is called a Boolean variable if its value is either true or false.
- Bit operation – replace true by 1 and false by 0 in logical operations.

Prepositional Logic

- A proposition is a collection of declarative statements that has either a truth value "true” or a truth value "false“
- consists of propositional variables and connectives.
- denote the propositional variables by capital letters (A, B, etc).
- connectives connect the propositional variables.
- examples of Propositions are given below:
  - "Man is Mortal", it returns truth value “TRUE”
  - "12 + 9 = 3 – 2", it returns truth value “FALSE”

Prepositional Logic

- uses five connectives which are:
  - OR (∨)
    - is true if at least any of the propositional variable A or B is true.
  - AND (∧)
    - is true if both the propositional variable A and B is true
  - Negation/ NOT (¬)
    - is false when A is true and is true when A is false.
  - Implication / if-then (→)
    - It is false if A is true and B is false. The rest cases are true.
  - If and only if (⇔)
    - is true when p and q are same, i.e. both are false or both are true.

Predicate Logic

- an expression of one or more variables defined on some specific domain
- The following are some examples of predicates –
  - Let E(x, y) denote "x = y"
  - Let X(a, b, c) denote "a + b + c = 0"
  - Let M(x, y) denote "x is married to y“
- **Quantifiers**
  - The variable of predicates is quantified by quantifiers.
  - Two types of quantifier in predicate logic –
    - Universal Quantifier and
    - Existential Quantifier.

Predicate Logic

- **Universal Quantifier**
  - states that the statements within its scope are true for every value of the specific variable.
  - denoted by the symbol ∀
  - $\forall x P(x)$ is read as for every value of x, P(x) is true.
- **Existential Quantifier**
  - states that the statements within its scope are true for some values of the specific variable.
  - denoted by the symbol ∃
  - $\exists x P(x)$ is read as for some values of x, P(x) is true.

Functions

- an object that setup an input- output relationship
- i.e. a function takes an input and produces the required output
- For a function f , with input x, the output y, we write $f(x)=y$.
- We also say that f maps x to y.

Method of proofs

Deductive Proof

- Consists of a sequence of statements whose truth leads us from some initial statement (hypothesis) to a conclusion statement.
- Proof must follow by some accepted logical principle.
- The hypothesis may be true or false.
- **Theorem:**
  - If $x >= 4$, then $2^x >= x^2$.

Method of proofs

Mathematical Induction

- Let A be a set of natural numbers such that :
  - $0 \epsilon A$
  - For each natural number n, if {0,1,2,3,…….n}ϵA. Then A=N.
  - In particular, induction is used to prove assertions of the form " for all nϵN, the property is valid". i.e.
    - In the basis step, one has to show that P(0) is true. i.e. the property is true for 0.
    - P holds for n will be the assumption.
    - Then one has to prove the validity of P for n+1.

Complexity Theory

- What can be computed efficiently?
- Are there problems that no program can solve in a limited amount of time or space?
- **Decidable Problem :**
  - The problems that can be solved by computer in limited time.
- **Undecidable Problem :**
  - can not predict the time of the problem in which a problem can be solved

Computability Theory

- What can be computed?
- Are there problems that no program can solve?
- Classify problems as being solvable or unsolvable.
- **Solvable Problems**
  - said to be solvable if you find a solution means there exists a potential solution
- **Unsolvable Problem :**
  - instant of time neither we are able to solve the problem nor in a position to say that the problem can not be solved

Automata Theory

- Study of abstract machine and their properties, providing a mathematical notion of “computer”
- Automata are abstract mathematical models of machines that perform computations on an input by moving through a series of states
- If the computation of an automaton reaches an accepting configuration it accepts that input

Why Study of Automata

- For software designing and checking behavior of digital circuits
- For designing software for checking large body of text as a collection of web pages, to find occurrence of words, phrases, patterns (i.e. pattern recognition, string matching, …)
- Designing “lexical analyzer” of a compiler, that breaks input text into logical units called “tokens

Brief History:

- Before 1930's, no any computer were there and Alen Turing introduced an abstract machine that had all the capabilities of today's computers. This conclusion applies to today's real machines.
- Later in 1940's and 1950's, simple kinds of machines called finite automata were introduced by a number of researchers.
- In late 1950's the linguist N. Chomsky begun the study of formal grammar which are closely related to abstract automata.
- In 1969 S. Cook extended Turing's study of what could and what couldn't be computed and classified the problem as:
  - Decidable
  - Tractable/intractable

Example 1

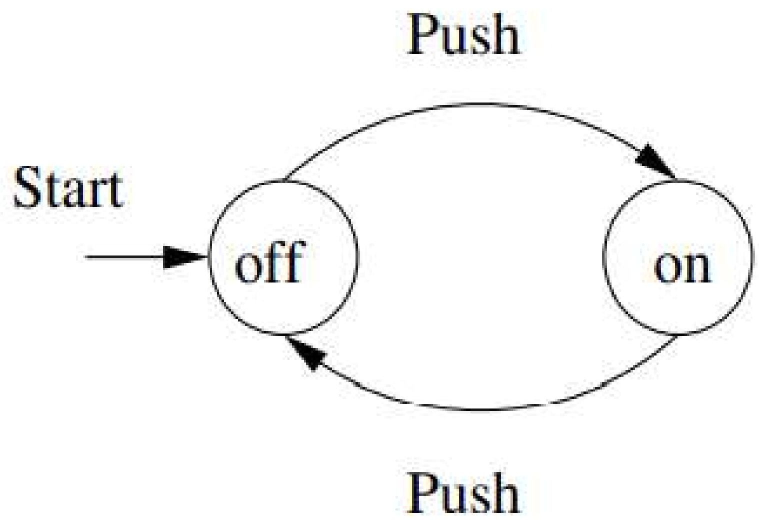

Fig: A finite automaton modeling an on/off switch

Example 1

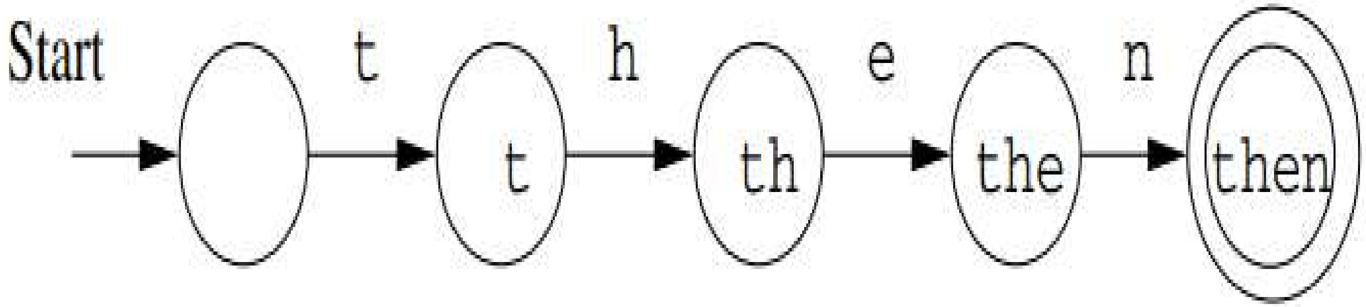

Fig: finite automaton modeling recognition of **then**

Basic concepts of Automata Theory

- **Alphabets - (Represented by 'Σ')**
  - Alphabet is a finite non-empty set of symbols.
  - The symbols can be the letters such as {a, b, c}, bits {0, 1}, digits {0, 1, 2, 3… 9}.
  - Common characters like $, #, etc.
    - {0,1} – Binary alphabets
    - {+, −, *} – Special symbol

Basic concepts of Automata Theory

- **Power of Alphabet**
  - The set of all strings of certain length k from an alphabet is the k^{th} power of that alphabet.
  - i.e. $\Sigma_k = \{w \,/\, |w| = k\}$
  - If $\Sigma = \{0, 1\}$ then,

    $\Sigma_0 = \{\varepsilon\}$

    $\Sigma 1 = \{0, 1\}$

    $\Sigma_2 = \{00, 01, 10, 11\}$

    $\Sigma_3 = \{000, 001, 010, 011, 100, 101, 110, 111\}$

Basic concepts of Automata Theory

- **Kleen Closure Alphabet**
  - The set of all the strings over an alphabet Σ is called kleen closure of Σ
  - denoted by Σ^*
  - kleen closure is set of all the strings over alphabet Σ with length 0 or more.
  - i.e. $\Sigma^* = \Sigma 0 \cup \Sigma\ 1 \cup \Sigma\ 2 \cup \Sigma\ 3 \cup$ ....

    E.g. A = {0}

    A* = {0n / n = 0, 1, 2, ...}

Basic concepts of Automata Theory

- **Positive Closure of Alphabet**
  - The set of all the strings over an alphabet Σ, except the empty string is called positive closure
  - denoted by $\Sigma+$
  - i.e. $\Sigma + = \Sigma 1 \cup \Sigma 2 \cup \Sigma 3 \cup$..............
- **Strings**
  - String is a finite sequence of symbols taken from some alphabet.
  - E.g. 0110 is a string from binary alphabet,
  - "automata" is a string over alphabet {a, b, c ... z}.

Basic concepts of Automata Theory

- **Empty String**
  - It is a string with zero occurrences of symbols.
  - denoted by 'ε' (epsilon)
- **Substring of a string**
  - A string s is called substring of a string w if it is obtained by removing 0 or more leading or trailing symbols in w
  - It is proper substring of w if s ≠ w
  - If s is a string then Substr (s, i, j) is substring of s beginning at ith position & ending at jth position both inclusive.

Basic concepts of Automata Theory

- **Concatenation of strings**
  - Let x & y be strings then xy denotes concatenation of x & y
  - i.e. the string formed by making a copy of x & following it by a copy of y.
  - More precisely, if x is the string of i symbols as x = a1,a2,a3,…,ai & y is the string of j symbols as y = b1,b2,b3,…,bj, then xy is the string of i + j symbols as xy = a1,a2,a3,…,ai,b1,b2,b3,…,bj.
  - For example; x = 000, y = 111, xy = 000111 & yx = 111000
  - 'ε' is identity for concatenation; i.e. for any w, εw = wε = w.

Basic concepts of Automata Theory

- **Suffix of a string**
  - A string s is called a suffix of a string w if it is obtained by removing 0 or more leading symbols in w.
  - For example; w = abcd, then s = bcd is suffix of w.
    - here s is proper suffix if s ≠ w.
- **Prefix of a string**
  - A string s is called a prefix of a string w if it is obtained by removing 0 or more trailing symbols of w.
  - For example; w = abcd, then s = abc is prefix of w,
    - Here, s is proper suffix i.e. s is proper suffix if s ≠ w

Basic concepts of Automata Theory

- **Languages**
  - A language L over an alphabet Σ is subset of all the strings that can be formed out of Σ;
  - i.e. a language is subset of kleen closure over an alphabet Σ. i.e. $L \subseteq \Sigma^*$. (Set of strings chosen from Σ^* defines language)
  - For example;
    - Set of all strings over Σ = {0, 1} with equal number of 0"s & 1"s.

      L = {ε, 01, 0011, 000111, ………}
    - ϕ is an **empty language** & is a language over any alphabet.
    - {ε} is a language consisting of only empty string.
    - Set of binary numbers whose value is a prime:

      L = {10, 11, 101, 111, 1011, ……}

Exercises

1) Let A be a set with n distinct elements. How many different binary relations on A are there?
2) If ∑= {a,b,c} then find the followings
   a. ∑1 , ∑2 , ∑3
3) If ∑={0,1}. Then find the following languages
   a. The language of string of length zero.
   b. The language of strings of 0‟s and 1‟s with equal number of each.
   c. The language {0n 1 n | n≥1}
   d. The language {0i 0 j | 0≤i≤j}.
   e. The language of strings with odd number of 0‟s and even number of 1‟s.
4) Define the Kleen closure and power of alphabets.