Lecture one

Topics that must be covered in this lecture:

- ***Introduction***
- ***Sets, operations of set and Properties of set.***
- ***Alphabets and Strings***
- ***Language.***

Introduction

What is Computation?

Computation is simply a sequence of steps that can be performed by a computer.

Theory of computation is the theoretical study of capabilities and limitations of Computers (Develop formal mathematical models of computation that reflect real-world computers).

As a student, you must study the following:

1- Automata and formal language.

Which answers - What are computers (Or what are models of computers)

2- Compatibility.

Which answers - What can be computed by computers?

3- Complexity.

Which answers - What can be efficiently computed?

In automata we will simulates parts of computers. Or we will make mathematical models of computers. Automata are more powerful than any real computer because we can design any machine on papers that can do everything we want.

Some applications of computation theory:

1- Analysis Of Algorithm
2- Complexity Theory
3- Cryptography
4- Compilers

Sets

A set is a collection of "objects" called the elements or members of the set.

Common forms of describing sets are:

- List all the elements, e.g. {a, b, c, d}
- Form new sets by combining sets through operators.

Examples in Set Representations:

- C = { a, b, c, d, e, f, g, h, i, j, k }
- C = { a, b, …, k } finite set
- S = { 2, 4, 6, … } infinite set
- S = { j : j > 0, and j = 2k for k>0 }
- S = { j : j is nonnegative and even }
- let A = { 1, 2, 3, 4, 5 } and U means Universal Set: all possible elements
  U = { 1 , … , 10 }

Terminology and Notation:

• To indicate that x is a member of set S, we write x∈S.
• We denote the empty set (the set with no members) as {} or ∅.
• If every element of set A is also an element of set B, we say that A is a subset of B, and write A⊆ B
• If every element of set A is also an element of set B, but B also has some elements not contained in A, we say that A is a proper subset of B, and write A⊂B
• We may also use the inverse notation: B⊇A and B⊃A for B is a (proper) superset of A.

Operations on Sets:

- The union of sets A and B, written A∪B
  is a set that contains everything that is in A, or in B, or in both.
- The intersection of sets A and B, written A∩B
  is a set that contains exactly those elements that are in both A and B.
- The set difference of set A and set B, written A – B is a set that contains everything that is in A but not in B.
- The complement of a set A, written as ^{-}A or A^{-} (A bar) is the set containing everything that is not in A. We assume for this definition some universal set U that contains "everything" (meaning "everything we are interested in at the moment"). Then
  A^{-} = U - A. ex: U={1,2,3,4,5,6} and A={3,5,6} then A^{-}={1,2,4}
- The length of a set A, written |A|, is the number of elements in a set A .
  ex: A={a,b,c,d} then length of A is 4

- A power set is a set of sets Power : E.g. let S = { a, b, c }
  2^{S}= { ∅, {a}, {b}, {c}, {a, b}, {a, c}, {b, c}, {a, b, c} }
  Observation:$|2^{S}| = 2^{|S|}$($8 = 2^3$)

- Cartesian Product:

  Let A = { 2, 4 } and B = { 2, 3, 5 }

  A X B = { (2, 2), (2, 3), (2, 5), (4, 2), (4, 3), (4, 5) }

  |A X B| = |A| |B|

  Generalizes to more than two sets

  A X B X … X Z

Properties of sets:

Let A, B, and C be subsets of the universal set U:

- Distributive properties

  $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$

  $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$
- Idempotent properties

  $A \cap A = A$,

  $A \cup A = A$.
- Double Complement property

  $(A^-)^- = A$.
- De Morgan's laws

  $(A \cup B)^- = A^- \cap B^-$

  $(A \cap B)^- = A^- \cup B^-$
- Commutative properties

  $A \cap B = B \cap A$

  $A \cup B = B \cup A$.
- Associative laws

  $A \cap (B \cap C) = (A \cap B) \cap C$

  $A \cup (B \cup C) = (A \cup B) \cup C$
- Identity properties

  $A \cup \emptyset = A$

  $A \cap U = A$
- Complement properties

  $A \cup A^- = U$,

  $A \cap A^- = \emptyset$.

Alphabets and Strings

Theory of computation is entirely based on symbols. These symbols are generally letters and digits. We will use small alphabets: Σ = {a, b}

Example1: suppose u=ab, v=bbbaaa, w=abba strings over Σ

Example2 : Σ 1 = {a, . . . , z} and Σ 2 = {0, . . . , 9} are alphabets. abb is a string over Σ1, and 123 is a string over Σ2. ba12 is not a string over Σ1, because it contains symbols that are not in Σ1. Similarly, 314 .. . is not a string over Σ2, because it is not a finite sequence. On the other hand, λ is a string over any alphabet.

operations of string

let w=a1a2…an   w=abba

v=b1b2….bm   v=bbbaaa

1- Concatenation

wv=a1a2…anb1b2…bm   abbabbbaaa

2- Reverse: the string spelled backward

w=ababaaabbb

W^R=an…a2a1   bbbaaababa

Example: $(\text{reverse})^R$ = esrever .

Note: $(wx)^R = x^R w^R$, for any strings w and x.

3- String Length

w=1a2…an

Length: |w|=n

Examples: |abba|=4

|aa|=2

|a|=1

4- Length of Concatenation

|uv|=|u|+|v|

Example: u=aab , |u|=3

v=abaab, |v|=5

|uv|=|aababaab|=8

|uv|=|u|+|v|=3+5=8

Empty String

A string with no letters: λ

Observations: $|\ \lambda| = 0$

$\lambda w = w\lambda = w$

$\lambda abba = abba\lambda = abba$

Substring

Substring of string: a subsequence of consecutive characters e.g.:

String	Substring
abbab	*ab*
abbab	abba
abbab	a
abbab	bbab

Prefix and Suffix

String: abbab

w=uv where u=prefix and v=suffix

Prefixes	Suffixes
λ	abbab
a	bbab
ab	bab
abb	ab
abba	b
abbab	λ

Another Operation

$$w^n = \underbrace{ww\ldots w}_{n}$$

Example: $(abba)^2$=abbaabba

Definition: $w^0 = \lambda$

$(abba)^0 = \lambda$

The * Operation (closure)

$\Sigma *$: is the set of all strings obtained by concatenating zero or more strings from Σ

Let $\Sigma = \{a, b\}$

$\Sigma * = \{\lambda, a, b, aa, ab, ba, , bb, aab, \ldots\}$

The $^+$ Operation:

$\Sigma +$: the set of all possible strings from alphabet Σ except λ

$\Sigma = \{a, b\}$

$\Sigma * = \{\lambda, a, b, aa, ab, ba, , bb, aaa, aab, \ldots\}$

$\Sigma + = \Sigma * - \lambda$

$\Sigma + = \{a, b, aa, ab, ba, bb, aaa, aab, \ldots\}$

Languages

People define the language as a way of understanding between the same group of beings, between human beings, animals, and even the tiny beings, this definition includes all kinds of understanding, talking, special signals and voices.
This definition works till the mathematician called Chomsky said that: the language is a set of strings and can be defined mathematically as:
1. A finite set of letters which called **Alphabet**. This can be seen in any natural language, for example the alphabet of English can be defined as:
$\Sigma = \{a, b, \ldots, z\}$
2. By concatenate letters from alphabet we get **words (string)**.
3. All words from the alphabet make **language**.
Example:
$\Sigma = \{a, b\}$
$\Sigma * = \{\lambda, a, b, aa, ab, ba, bb, aaa, aab, \ldots\}$
Languages:
$\{\lambda\}$
$\{a,aa,aab\}$
$\{\lambda, abba, abab, aa, ab, aaaaaa\}$

Kinds of languages:

1- **Talking language**: (e.g.: English, Arabic): It has an alphabet: $\Sigma = \{a,b,c,\ldots.z\}$
From these alphabetic we make sentences that belong to the language. Now we want to know is this sentence is true or false so - We need a grammar.
Ali is a clever student. (It is a sentence ∈ English language.)
2- **Programming language**: (e.g.: c++, java):It has alphabetic: Σ ={a,b,c,.z , A,B,C,..Z , ?, /, - ,\.} From these alphabetic we make sentences that belong to programming language.
Now we want to know if this sentence is true or false so we need a compiler to make sure that syntax is true.
3- **Formal language**: (any language we want.) It has strings from these strings we make sentences that belong to this formal language. Now we want to know is this sentence is true or false so we need rules.

Another Example about formal language :

An infinite language $L=\{a^n b^n : n \geq 0\}$

$$\left.\begin{array}{l} \lambda \\ ab \\ aabb \\ aaaaabbbbb \end{array}\right\} \in L \text{ where } abb \notin L$$

Note that:

- Sets $\emptyset=\{\} \neq \{\lambda\}$
- Set size $|\{\}|=|\emptyset|=0$
- Set size $|\{\lambda\}| = 1$
- String length $|\lambda| = 0$

Operations on Languages

1- The usual set operations

$\{a,ab,aaaa\} \cup \{bb, ab\} = \{a, ab, bb, aaaa\}$

$\{a,ab,aaaa\} \cap \{bb, ab\} = \{ab\}$

$\{a,ab,aaaa\}-\{bb,ab\}=\{a,aaaa\}$

2- Complement: $\bar{L}=\Sigma^*-L$

$\Sigma^*=\{$ $\lambda,a,b,aa,ba,ab,bb,aaa,\ldots\}$

$\overline{\{a,ba\}}=\{\lambda, b, aa, ab, bb, aaa, \ldots\}$

3- Reverse

Definition: $L^R=\{w^R : w \in L\}$

Examples: $\{ab,aab,baba\}^R=\{ba,baa,abab\}$

$L=\{a^n b^n : n \geq 0\}$

$L^R=\{b^n a^n : n \geq 0\}$

4- Concatenation

Definition: $L1L2=\{xy: x \in L1, y \in L2\}$

Example: $\{a,ab,ba\}\{b,aa\}=\{ab,aaa,abb,abaa,bab,baaa\}$

5- Another operation

Definition: $L^n = \underbrace{LL\ldots L}_{n}$

$\{a,b\}^3=\{a,b\}\{a,b\}\{a,b\}=\{aaa,aab,aba,abb,bab,baa,bba,bbb\}$

Special case: $L^0=\{\lambda\}$

$\{a,bba,aaa\}^0=\{\lambda\}$

6- Star-closure (Kleen*)

Definition: $L^*=L^0 \cup L^1 \cup L^2\ldots$

Example:

$$\{a,bb\}^* = \begin{Bmatrix} \lambda, \\ a,bb, \\ aa,abb,bba,bbbb, \\ aaa,aabb,abba,abbbb,\ldots \end{Bmatrix}$$

7- positive-closure

Definition: $L^+= L^1 \cup L^2\ldots$

$= L^*-\{\lambda\}$

$$\{a,bb\}^+ = \begin{Bmatrix} a,bb, \\ aa,abb,bba,bbbb, \\ aaa,aabb,abba,abbbb,\ldots \end{Bmatrix}$$

Example: Consider the languages L1 = { λ , 0, 1} and L2 = { λ, 01, 11}.

- The union of these languages is L1∪ L2 = { λ , 0, 1, 01, 11}
- Their intersection is L1 ∩ L2 = { λ }
- And the complementation of :

  $L1^- = U - L1$

  $=\{\lambda,0,1,00,01,10,11,000,001,\ldots\}-\{\lambda,0,1\}$

  $=\{00, 01, 10, 11, 000, 001, \ldots\}$.