Introduction and Background

CS60001: Foundations of Computing Science

Pallab Dasgupta

Professor, Dept. of Computer Sc. & Engg.,

Indian Institute of Technology Kharagpur

Set Theory

- A *set* is a group of objects represented as a unit
  - Example: set of odd positive integers less than 50 and divisible by 5

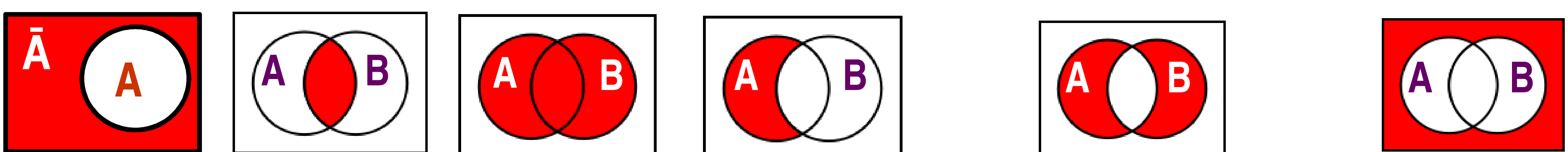
    { 5, 15, 25, 35, 45 }

- Let A and B are two sets. A is a *subset* of B ($A \subseteq B$) if every element of A is also an element of B, i.e., $x \in A \Rightarrow x \in B$
  - A is a proper subset of B ($A \subset B$) if A is a subset of B and $A \neq B$
  - Example: $A \subset B$, where

    B = set of odd positive integers less than 50 & divisible by 5 ≡ {5, 15, 25, 35, 45}

    A = set of odd positive integers less than 50 & divisible by 15 ≡ {15, 45}

- Set Operations



Complement ($\bar{A}$) | Intersection ($A \cap B$) | Union ($A \cup B$) | Difference ($(A - B) = (A \cap B')$) | Symmetric Difference ($A \Delta B$) | DeMorgan's Law ($(A \cup B)' = A' \cap B'$)

Set Theory (contd…)

- **Notations**
  - $\mathcal{N}$ = Set of natural number
  - $\mathcal{Z}$ = Set of integers [$\mathcal{Z}^+$ = Set of positive integers]
  - $\mathcal{R}$ = Set of real numbers [$\mathcal{R}^+$ = Set of positive real numbers]
  - $\mathcal{Q}$ = Set of rational numbers
  - $\mathcal{C}$ = Set of complex numbers

- *Power Set* of A, P(A) is the set of all subsets of A
  - A = { 1, 2 } then P(A) ={Φ, {1}, {2}, {1, 2} }, Here Φ is *Null Set*

- *Cartesian Product of* A and B (written as A × B) is the set of all pairs where the first element is a member of A and the second element is a member of B
  - Example: A = {1, 2} and B = {x, y, z},

    Then, A × B = {(1, x), (1, y), (1, z), (2, x), (2, y), (2, z)}

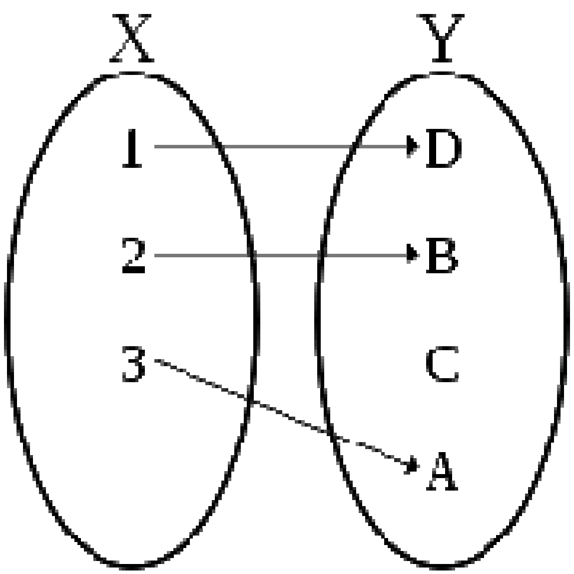
Function

- A *function (or mapping)* is an object that sets up an input-output relationship
  - If *f* is a function whose output value is *b* when the input value is *a*, we write *f(a) = b*
  - Let $f(x_1) = y_1$ and $f(x_2) = y_2$. If $y_1 \neq y_2$, then $x_1 \neq x_2$.
- The set of possible input to the function is called *domain*
- The outputs of a function come from a set is called *range*
  - *f* is a function with domain D and range R is represented as, $f : D \rightarrow R$
- Example:

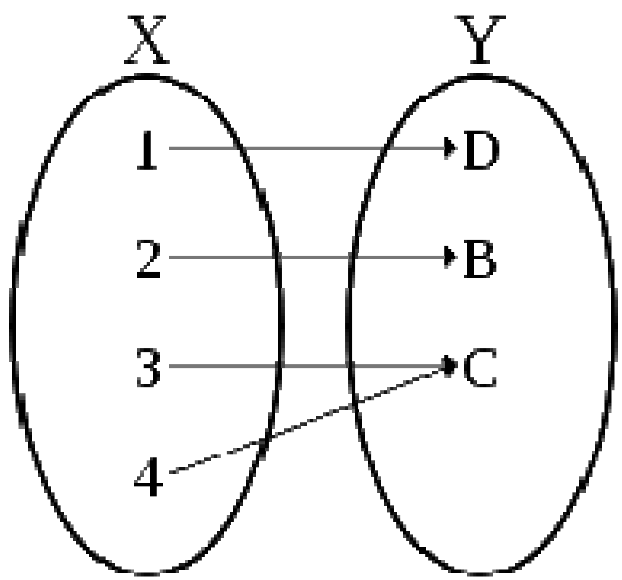
  Consider the function, $f : \{1, 2, 3, 4, 5, 6\} \rightarrow \{0, 1, 2\}$
  - The function *f* takes positive integers less than 7 and outputs the result modulo 3; i.e., f(n) = n%3
  - Domain of *f* is, D = {1, 2, 3, 4, 5, 6}
  - Range of *f* is, R = {0, 1, 2}

Function (contd….)

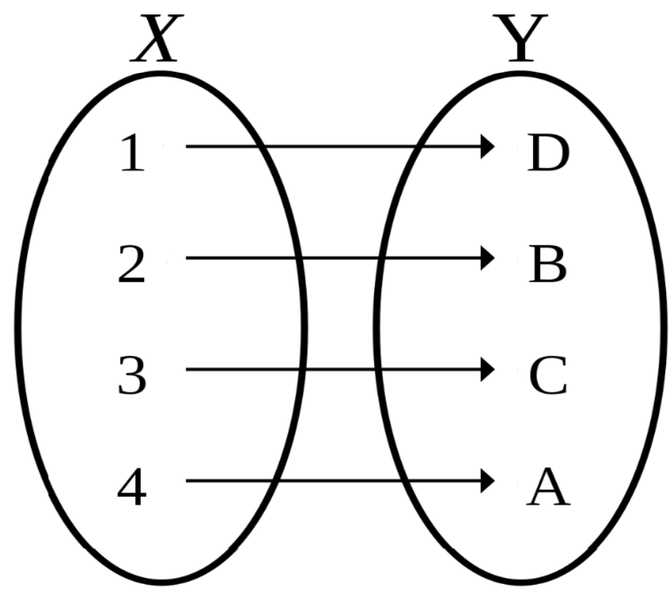
- **Mapping are of 3 types:**
  - Injective Mapping – Into Mapping, i.e., $\forall x_1, x_2 \in X$, if $f(x_1) = f(x_2)$ then $x_1 = x_2$ and equivalently, if $x_1 \neq x_2$ then $f(x_1) \neq f(x_2)$
  - Surjective Mapping – Onto Mapping, i.e., $Y = f(X)$
  - Bijective Mapping – Injective + Surjective (One-to-one Onto Mapping)



Injective Mapping



Surjective Mapping



Bijective Mapping

Relation

- A property whose domain is a set of *k*-tuples (A × A × … × A) is called *relation*
  - If K = 2 the relation is called *binary relation*
    - Example: *less than (<)* is a binary relation

k number of *A*s

- A binary relation *R* is an *equivalence relation* if R satisfies following conditions:
  - R is reflexive i.e., ∀x, x*R*x
  - R is symmetric i.e., ∀x ∀y, (x*R*y => y*R*x)
  - R is transitive i.e., ∀x ∀y ∀z, (x*R*y and y*R*z => x*R*z)

- A binary relation *R* is an *partial-order relation* if R satisfies following conditions:
  - R is reflexive i.e., ∀x, x*R*x
  - R is anti-symmetric i.e., ∀x ∀y, (x*R*y and yRx => x ≡ y)
  - R is transitive i.e., ∀x ∀y ∀z, (x*R*y and y*R*z => x*R*z)

Graph

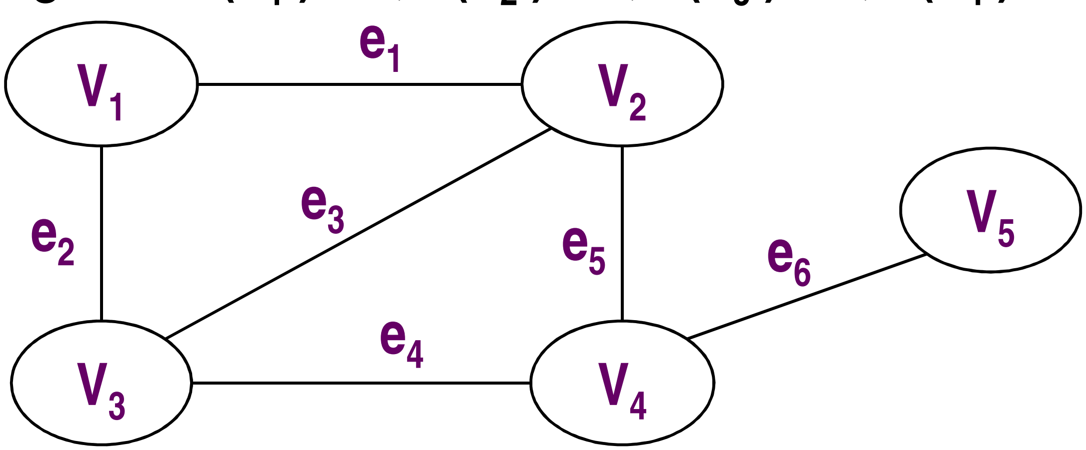
- An *undirected graph* is a set of points with lines connecting some of the points
  - G = (V, E) where V is the set of vertices and E is the set of edges
- Number of edges incident at a particular node (v) is the *degree [d(v)]* of the node

- Example: G = (V, E), where

Set of Vertices: V ={ V_1, V_2, V_3, V_4, V_5 }

Set of Edges: E ={ e_1, e_2, e_3, e_4, e_5, e_6 }

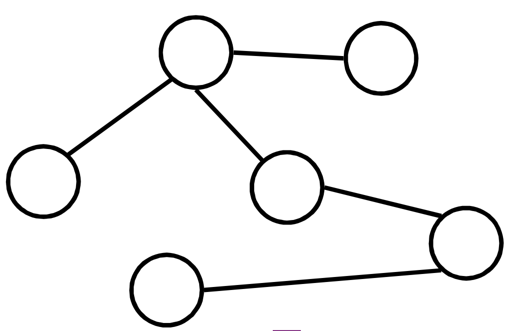
Degrees: $d(V_1) = 2$, $d(V_2) = 3$, $d(V_3) = 3$, $d(V_4) = 3$ and $d(V_5) = 1$

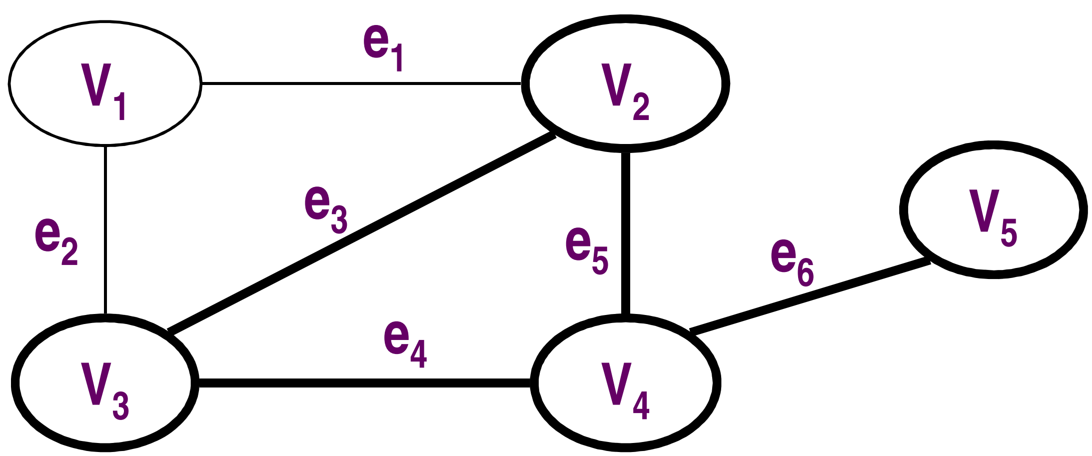


Graph(contd…)

- G is a *subgraph* of H if the nodes of G are a subset of the nodes H, and the edges of G are the edges of H on the corresponding nodes
  - Example: Subgraph H = (V_H, E_H) where;

    $V_H = \{V_2, V_3, V_4, V_5\}$ and $E_H = \{e_3, e_4, e_5, e_6\}$
- A *path* in a graph is a sequence of node connected by edges
  - V_1, V_2, V_3, V_4, V_5 is a path
- A path is a *cycle* if it starts and ends in the same node
  - V_1, V_2, V_4, V_3 is a cycle
- A graph is a *tree* if it is connected and has no cycle


Tree



Graph(contd…)

- If a graph has arrows instead of lines, the graph is called *directed graph*
  - Edges from vertex *i* to vertex *j* are represented as pairs *(i, j)*
  - Out-degree [$d^+(v)$]: number of arrows pointing from a particular node (v)
  - In-degree [$d^-(v)$]: number of arrows pointing to a particular node (v)

- Example: G = (V, E) where,
  - Set of vertices, V = {1, 2, 3, 4, 5}
  - Set of directed edges, E = {(1,2), (1,5), (2,1), (2,3), (2,4), (5,3), (5,4)}
  - In-degrees and Out-degrees,

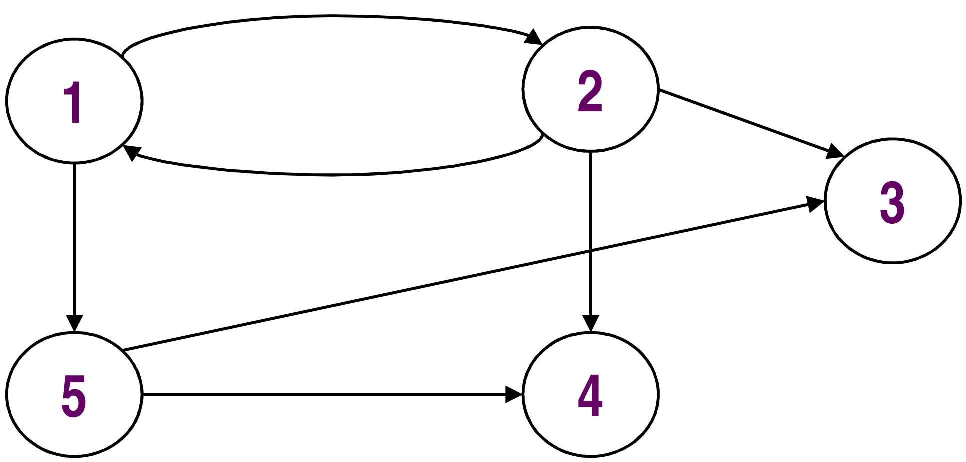
    $d^+(1) = 2; d^-(1) = 1$

    $d^+(2) = 3; d^-(2) = 1$

    $d^+(3) = 0; d^-(3) = 2$

    $d^+(4) = 0; d^-(4) = 2$

    $d^+(5) = 2; d^-(5) = 1$



Boolean Logic

- It is a mathematical system built around two values **TRUE** and **FALSE**
  - The value TRUE and FALSE are called Boolean values and are often representated by the values 1 and 0
- Basic operations are as follows:
  - Negation (~) ,Conjunction (Λ) ,Disjunction (V)
- Truth Table of Basic operations:

a	~a
0	1
1	0

a	b	a Λ b
0	0	0
0	1	0
1	0	0
1	1	1

a	b	a V b
0	0	0
0	1	1
1	0	1
1	1	1

Boolean Logic(contd…)

- Several other Boolean operations occasionally appear
  - Exclusive-Or (XOR): $P \oplus Q \equiv (\sim P \wedge Q) \vee (P \wedge \sim Q) \equiv \sim(P \Leftrightarrow Q)$
  - Implication: $P \Rightarrow Q \equiv \sim P \vee Q$
  - Equality: $P \Leftrightarrow Q \equiv (P \Rightarrow Q) \wedge (Q \Rightarrow P)$

- Distributive law:
  - $P \wedge (Q \vee R) \equiv (P \wedge Q) \vee (P \wedge R)$
  - $P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$ [Dual]

- Commutative law:
  - $P \vee Q \equiv Q \vee P$ and $Q \wedge R \equiv R \wedge Q$