# DATA STRUCTURES AND ALGORITHMS

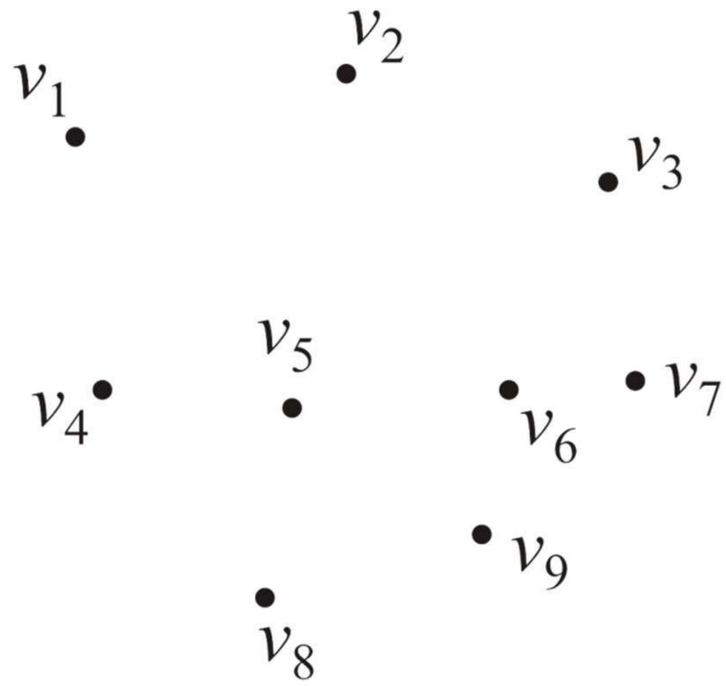
## Lecture 13: Graphs

Lecturer: Mohsin Abbas

National University of Modern Languages, Islamabad

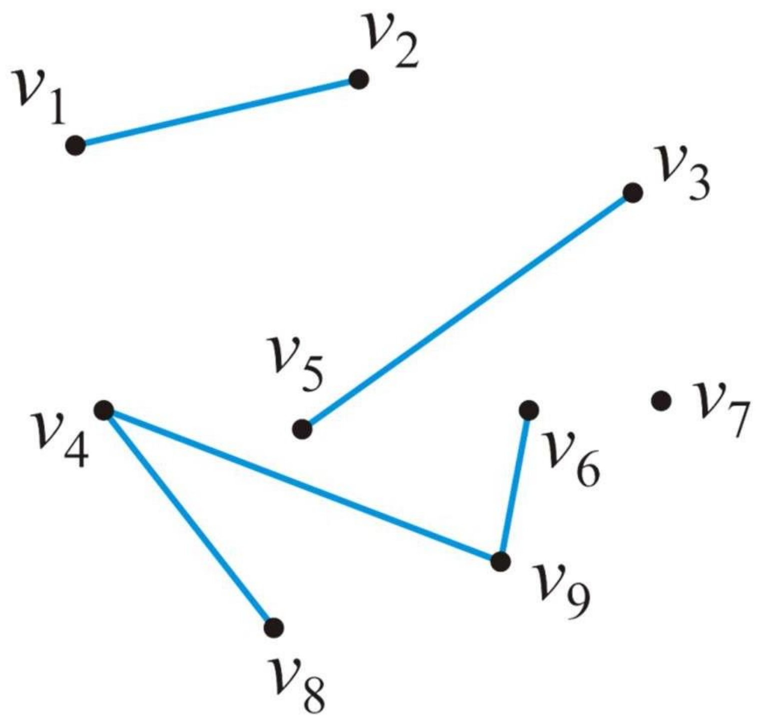
# GRAPHS

- Consider this collection of vertices
  - $V = \{v_1, v_2, ..., v_9\}$
  - Where $|V| = n$



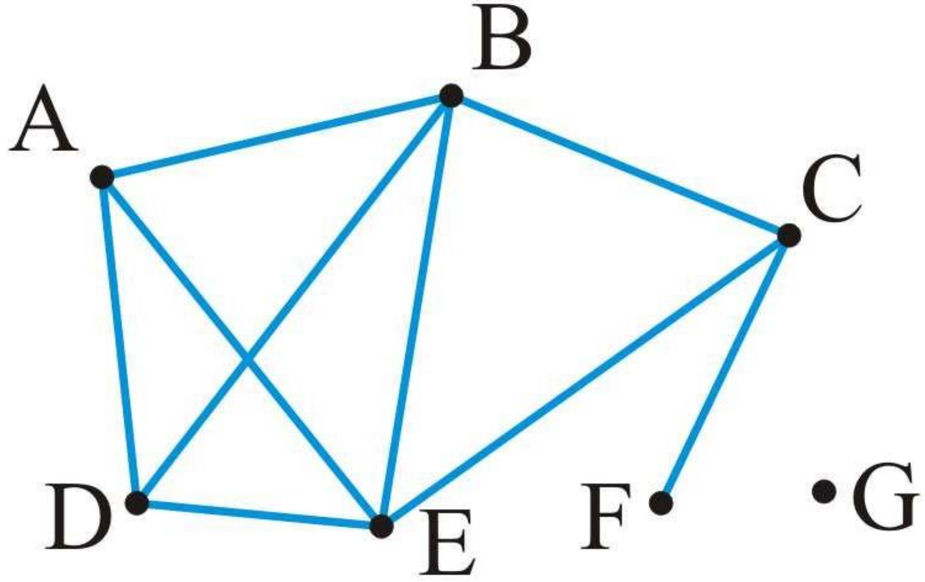
# UNDIRECTED GRAPHS

- Associated with these vertices are $|E| = 5$ edges
  - $E = \{\{v_1, v_2\}, \{v_3, v_5\}, \{v_4, v_8\}, \{v_4, v_9\}, \{v_6, v_9\}\}$
- Pair $\{v_j, v_k\}$ indicates following relations
  - Vertex $v_j$ is adjacent to vertex $v_k$
  - Vertex $v_k$ is adjacent to vertex $v_j$



# UNDIRECTED GRAPHS – EXAMPLE

- Given $|V| = 7$ vertices and $|E| = 9$ edges
  - V = {A, B, C, D, E, F, G}
  - E = {{A, B}, {A, D}, {A, E}, {B, C}, {B, D}, {B, E}, {C, E}, {C, F}, {D, E}}



# APPLICATIONS OF GRAPHS

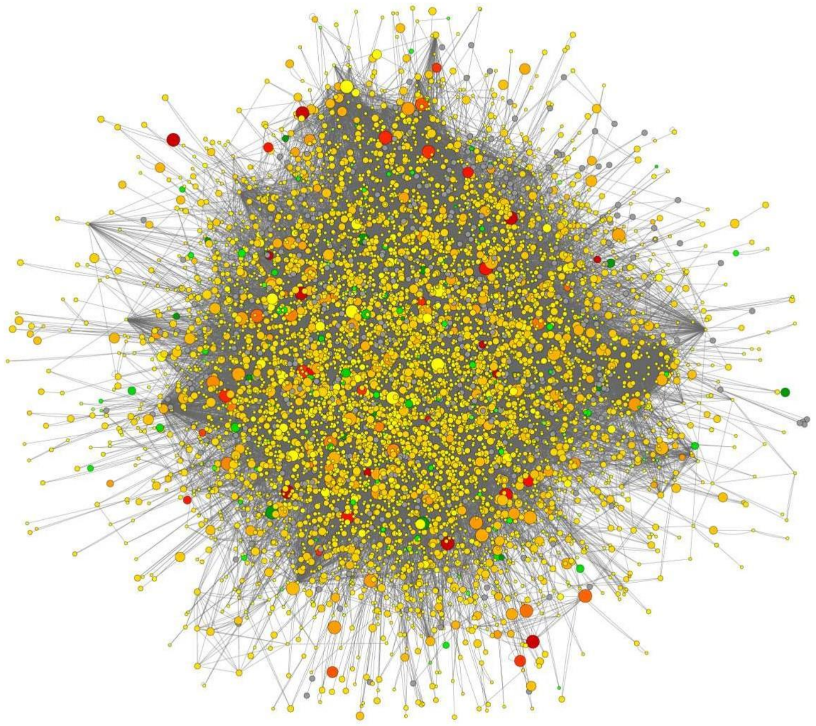
- Driving Map
  - Vertex = Intersection, destinations
  - Edge = Road
- Airline Traffic
  - Vertex = Cities serviced by the airline
  - Edge = Flight exists between two cities
- Computer networks
  - Vertex = Server nodes, end devices, routers
  - Edge = Data link

# APPLICATIONS OF GRAPHS

- Many real-world applications concern large graphs

- Web document graph - 1 trillion webpages
  - Vertex = Webpage
  - Edge = Hyperlink

- Social networks - 1.3 billion users
  - Vertex = Users
  - Edge = Friendship relation



# UNDIRECTED GRAPHS – DEFINITION

- An undirected Graph is defined as $G=(V,E)$ consisting of
  - Set V of vertices: $V = \{v_1, v_2, \ldots, v_n\}$
    - ➢ Number of vertices is denoted by $|V| = n$
  - Set E of unordered pairs $\{v_i, v_j\}$ termed edges
    - ➢ Edges connect the vertices
- Maximum number of edges in an undirected graph is $O(|V|^2)$

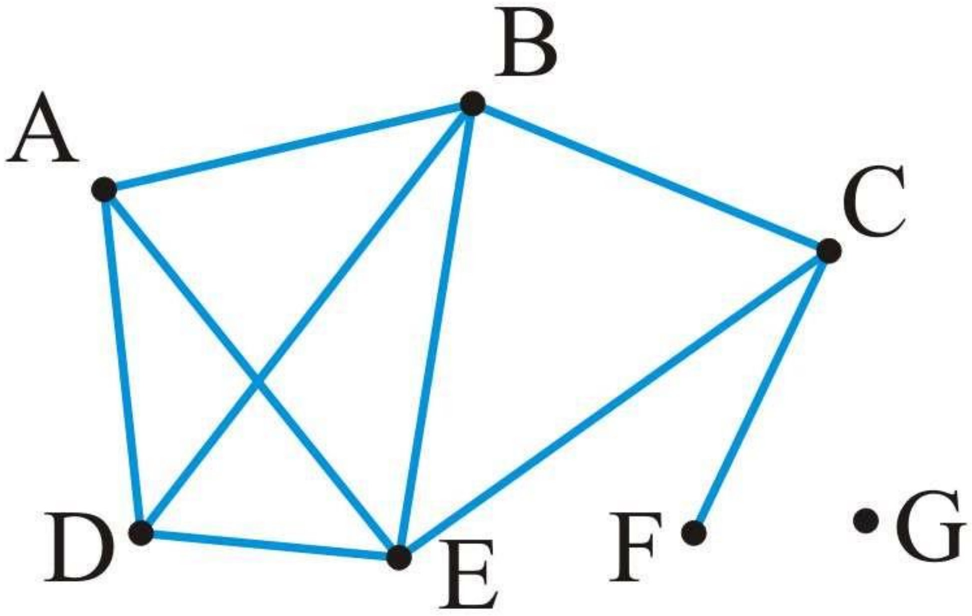
$$|E| \leq \binom{|V|}{2} = \frac{|V|\,(|V|-1)}{2} = O(|V|)^2$$

  - Assumption: A vertex is never adjacent to itself
  - For example, $\{v_1, v_1\}$ will not define an edge

- Many data structures can implement abstract undirected graphs
  - Adjacency matrices, Adjacency lists

# DEGREE

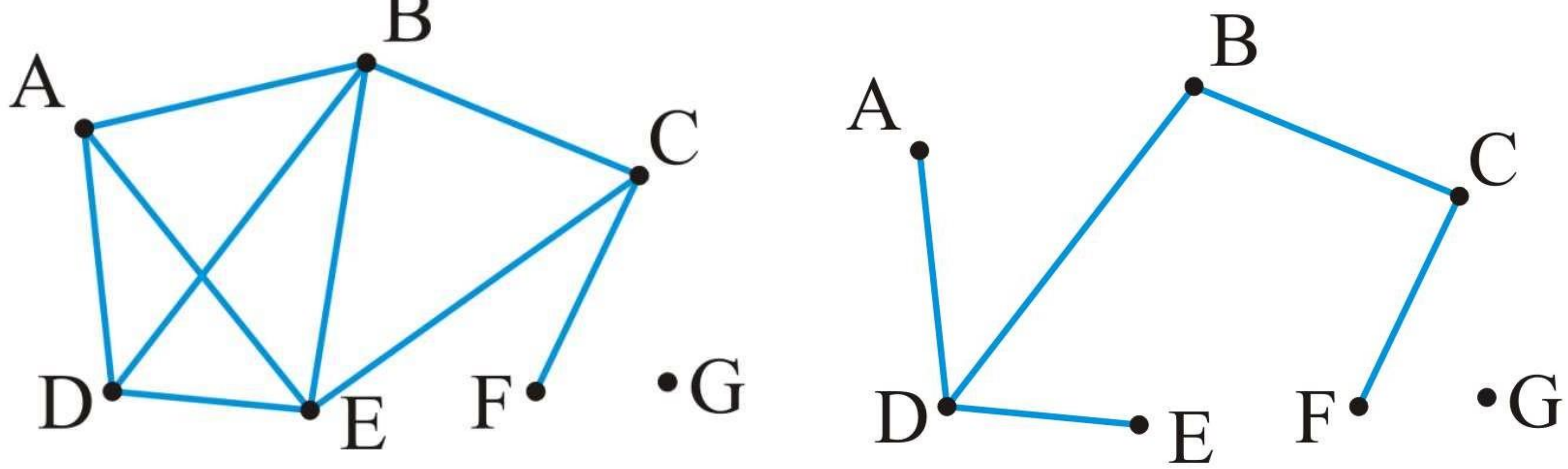
- Degree of a vertex is defined as the number of adjacent vertices
  - degree(A) = degree(D) = degree(C) = 3
  - degree(B) = degree(E) = 4
  - degree(F) = 1
  - degree(G) = 0




- Vertices adjacent to a given vertex are its neighbors

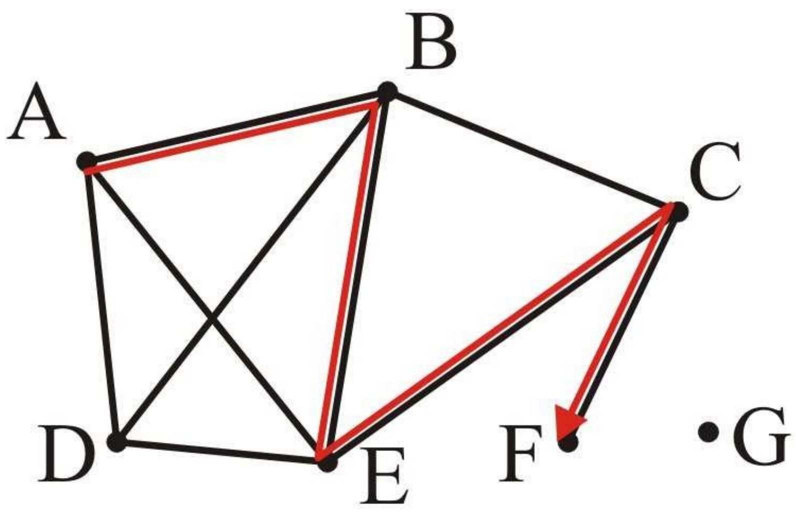
# SUBGRAPH

- A sub-graph of a graph G is defined by
  - Subset of the vertices
  - Subset of the edges that connected the subset of vertices in the original graph
- Every graph is a subgraph of itself



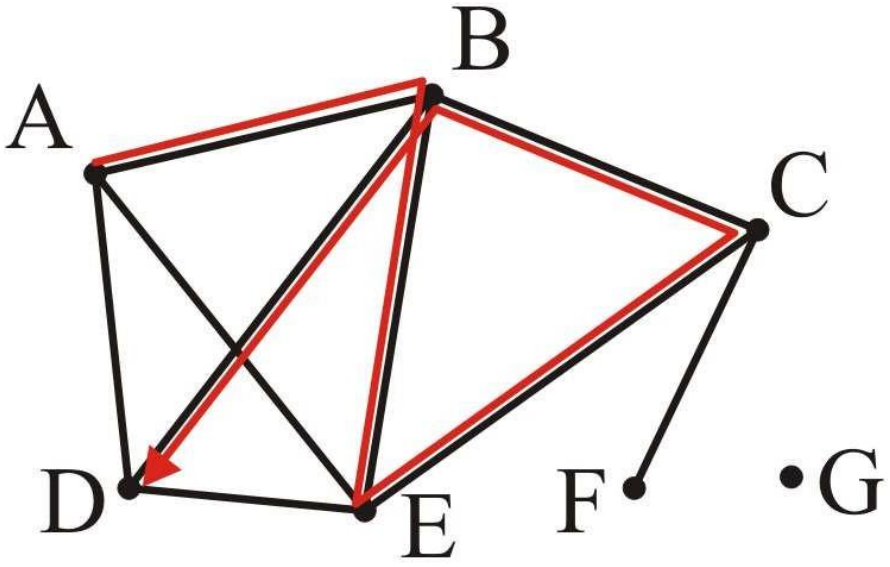
# PATH

- Path in an undirected graph is an ordered sequence of vertices
  - Consecutive vertices are connected through edges
- Path from vertex 0 to vertex k is $(v_0, v_1, v_2, ..., v_k)$
  - where $\{v_j - 1, v_j\}$ is an edge for $j = 1, ..., k$
- Length of a path is equal to the number of edges
- Example: Path from A to F
  - Path: (A, B, E, C, F)
  - Length of the path is 4



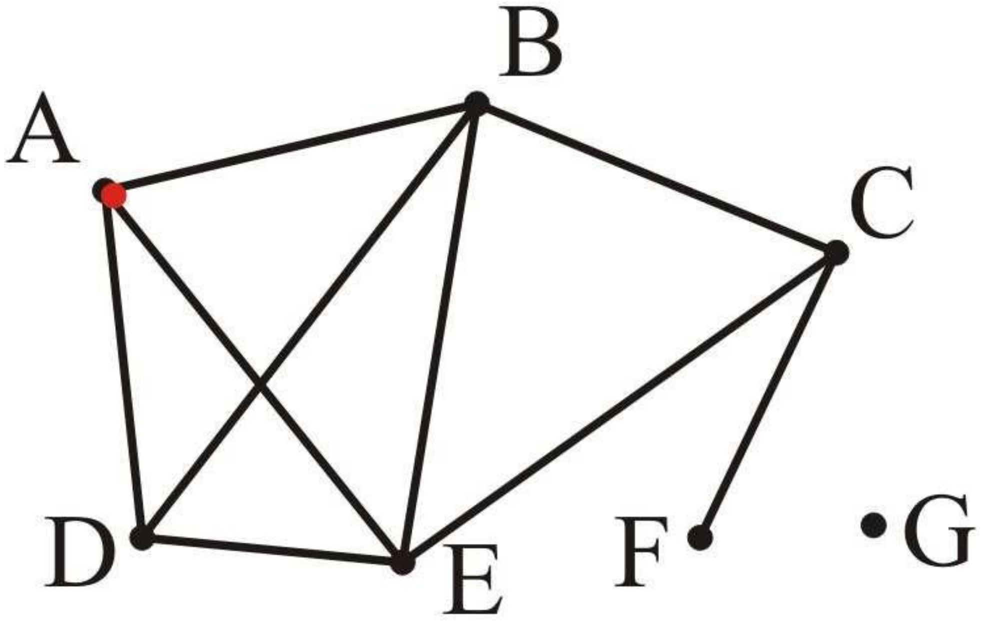
# PATH – EXAMPLE

- Path of length 5: ( A, B, E, C, B, D)
  - Repetition of vertex B



# PATH – EXAMPLE

- A trivial path of length 0: (A)



# SIMPLE PATH

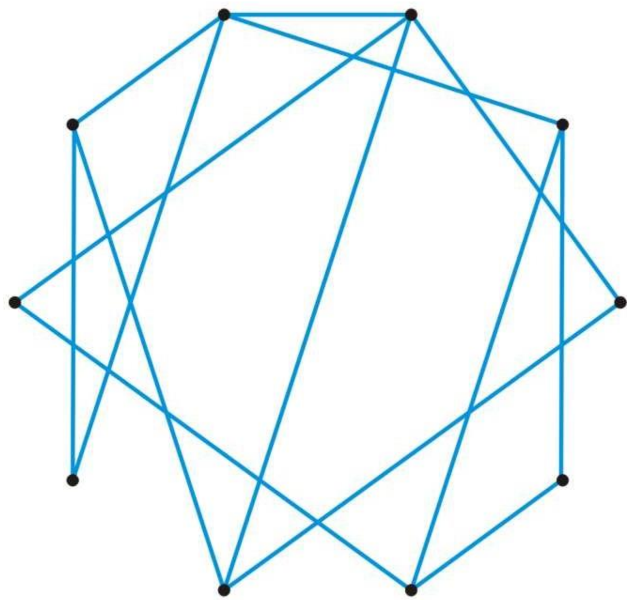
- A simple path has no repetitions other than perhaps the first and last vertices

- A simple cycle is a simple path of at least two vertices with the first and last vertices equal
  - Note: these definitions are not universal

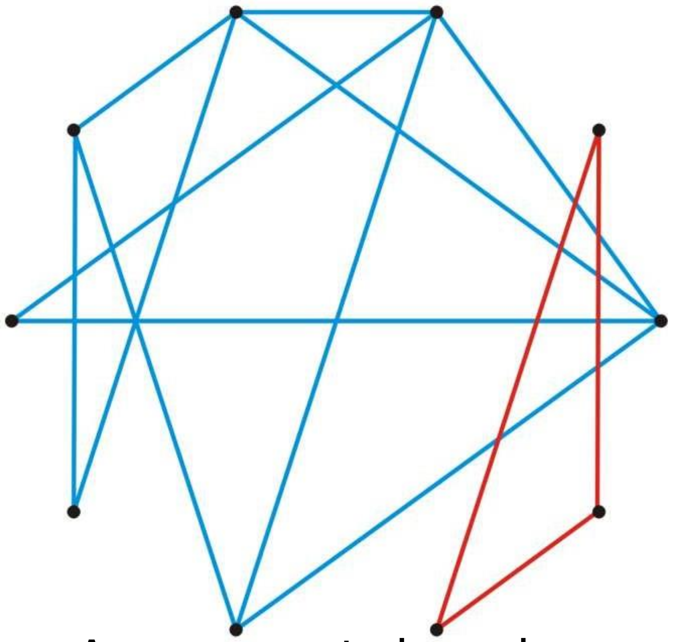
- A loop is an edge from a vertex onto itself

# CONNECTEDNESS

- Two vertices $v_i$, $v_j$ are said to be connected if there exists a path from $v_i$ to $v_j$
- A graph is connected if there exists a path from every vertex to every other vertex



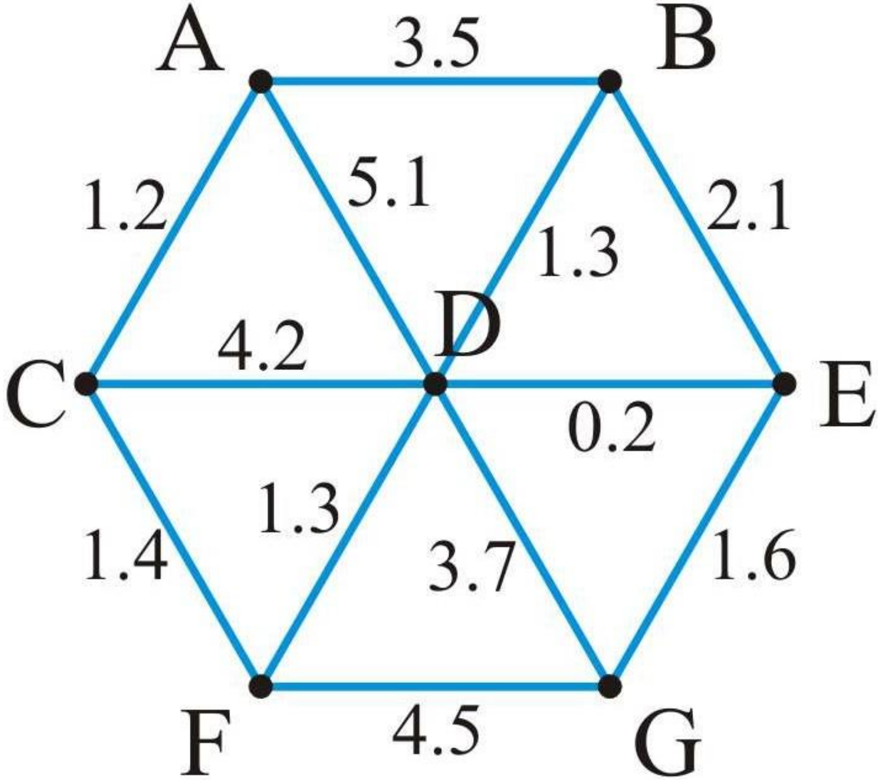
A connected graph



An unconnected graph

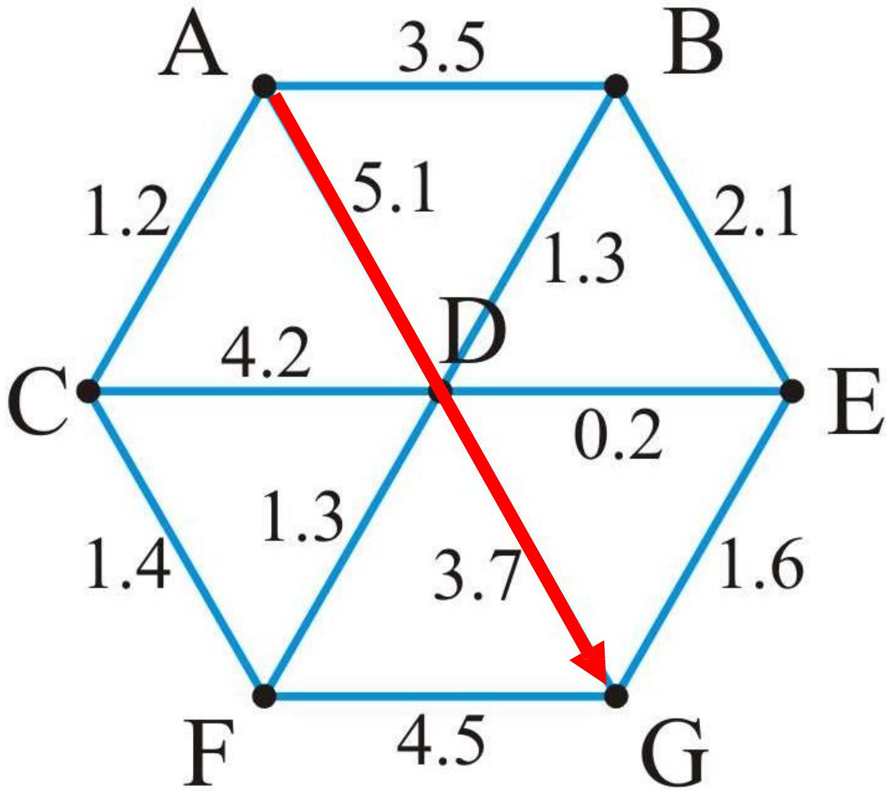
# WEIGHTED GRAPHS

- A weight may be associated with each edge in a graph
  - This could represent distance, energy consumption, cost, etc.
  - Such a graph is called a weighted graph
- Pictorially, we will represent weights by numbers next to the edges



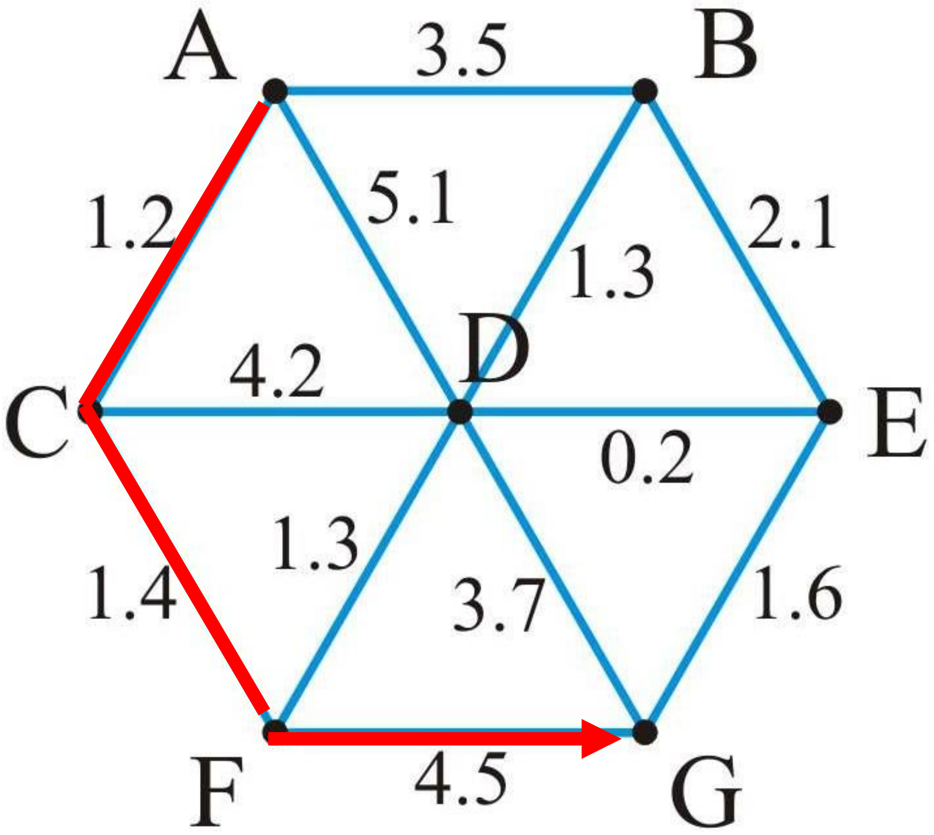
# WEIGHTED GRAPHS

- Length of a path within a weighted graph is the sum of all of the edges which make up the path
- The length of the path ( A, D, G) in the following graph is 8. 8



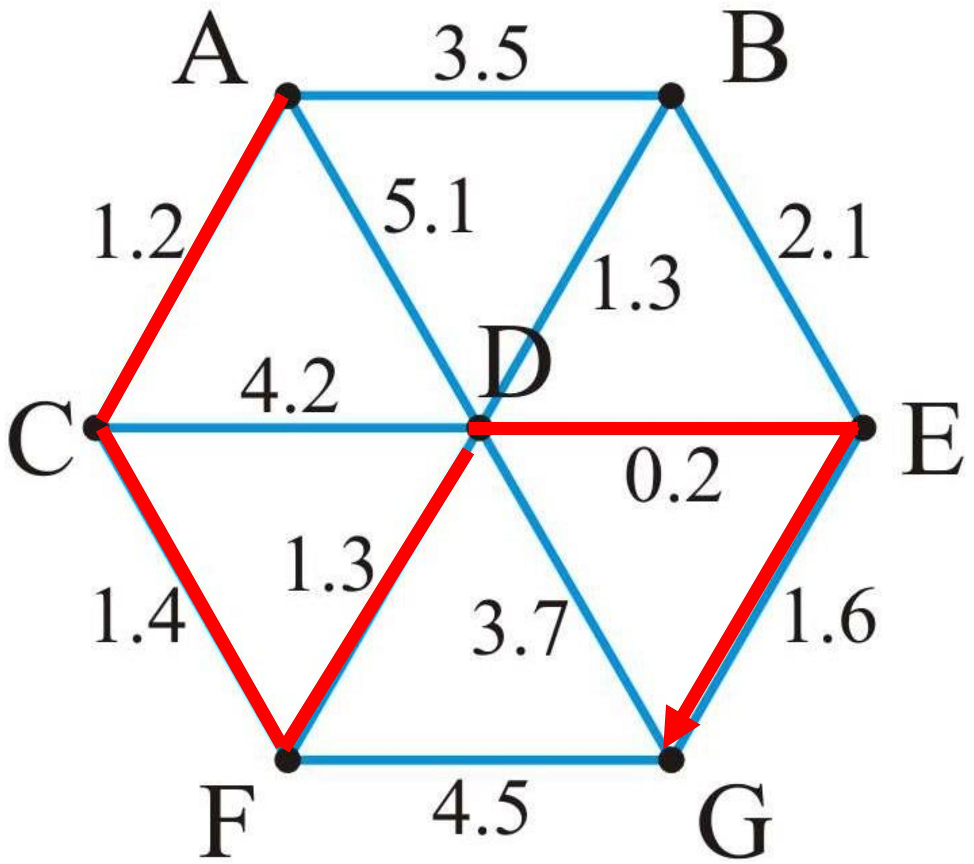
# WEIGHTED GRAPHS – EXAMPLE

- Different paths may have different weights
  - Another path is (A, C, F, G) with length 1.2 + 1.4 + 4.5 = 7.1



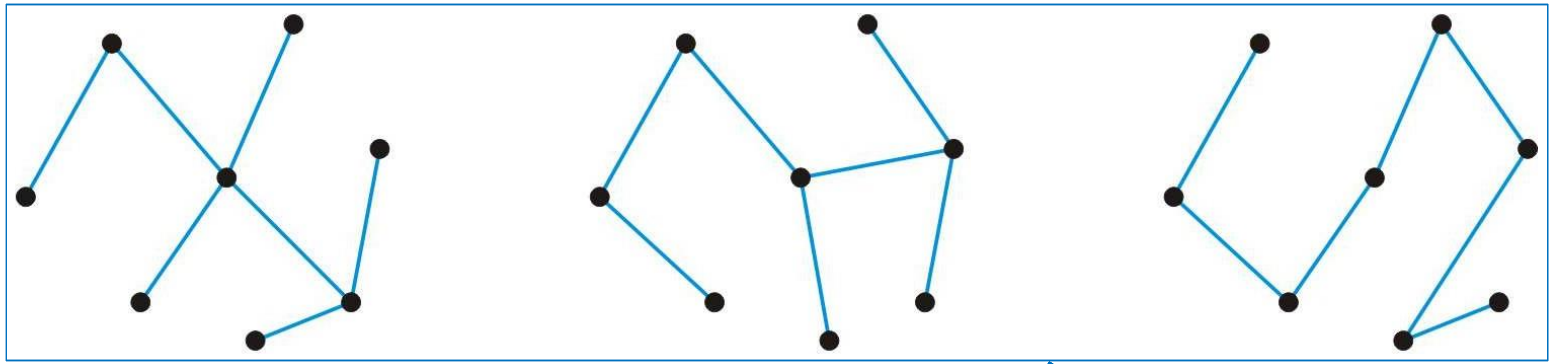
# WEIGHTED GRAPHS – EXAMPLE

- Find the shortest path between two vertices A and G
- Shortest path is (A, C, F, D, E, G) with length 5.7



# TREES

- A graph is a tree if it satisfies the following two conditions
  - Graph is connected
  - There is a unique path between any two vertices

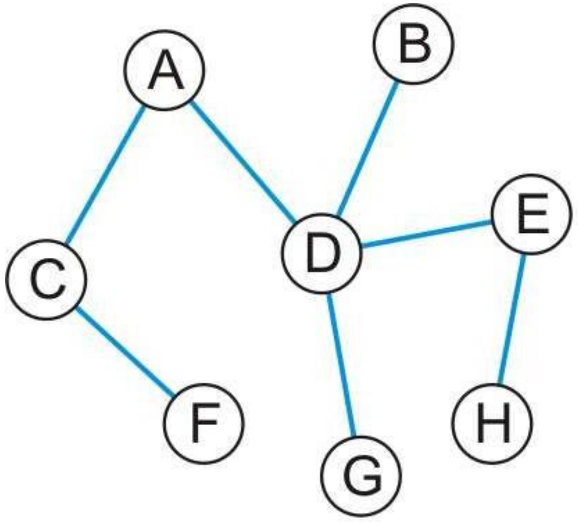


Three trees on same 8 vertices

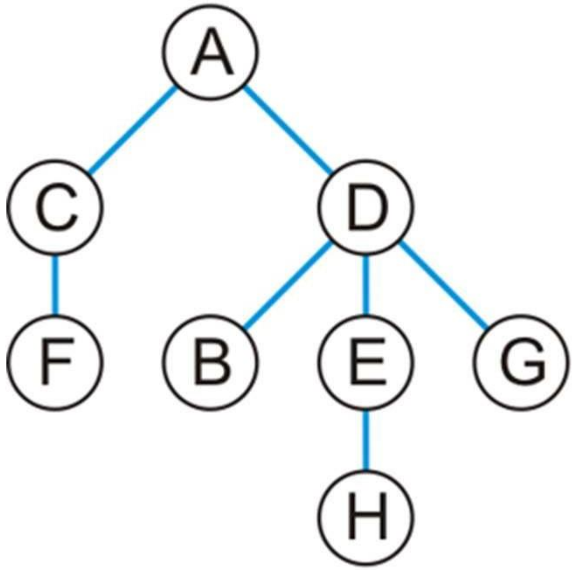
- Consequences
  - The number of edges is $|E| = |V| - 1$
  - The graph is acyclic, that is, it does not contain any cycles
  - Adding one more edge must create a cycle
  - Removing any one edge creates two disjoint non-empty sub-graphs

# TREES

- Any tree can be converted into a rooted tree by
  - Choosing any vertex to be the root
  - Defining its neighboring vertices as its children
- Recursively defining
  - All neighboring vertices other than that one designated as parent to be that vertex children

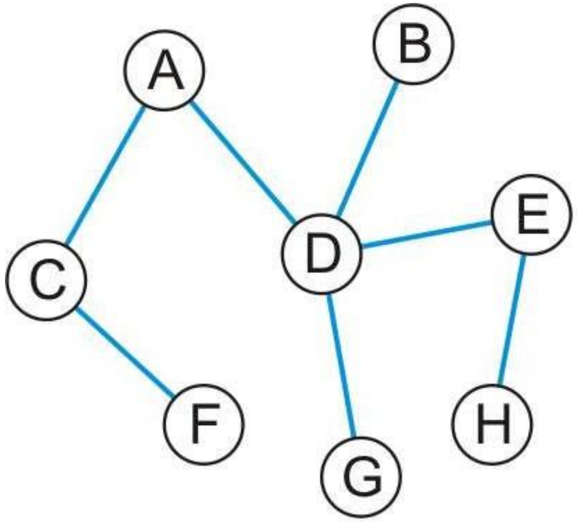



Unrooted Tree

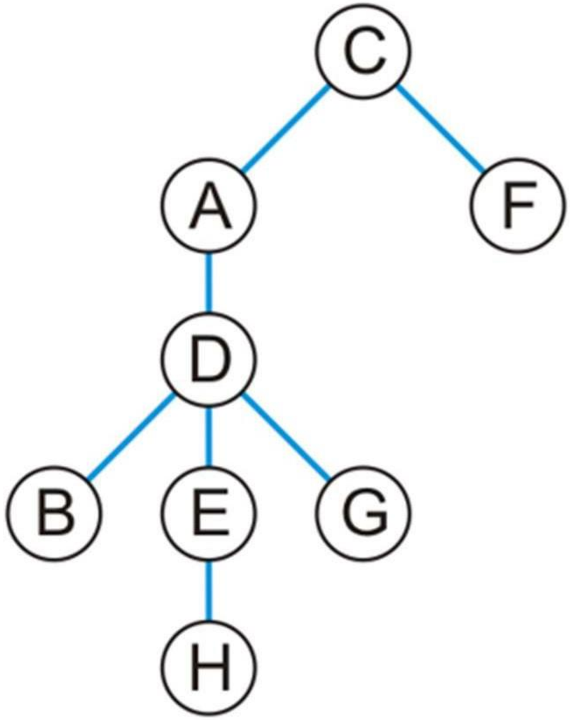



Tree rooted at A

# TREES – EXAMPLE

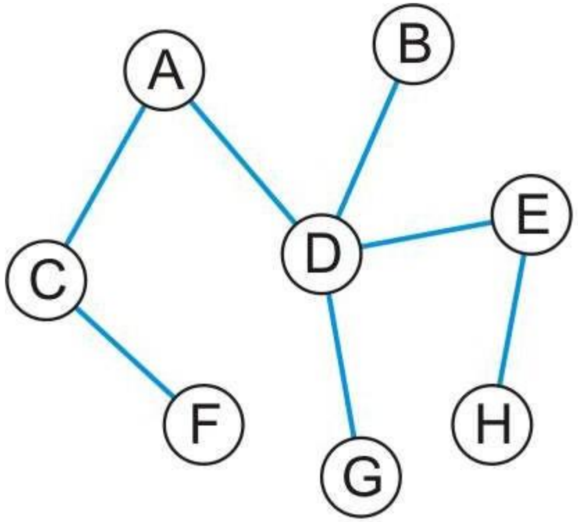



Unrooted Tree

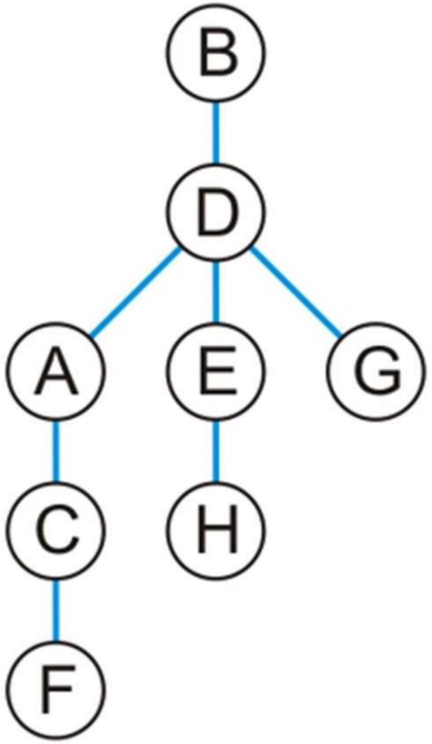



Tree rooted at C

# TREES – EXAMPLE




Unrooted Tree

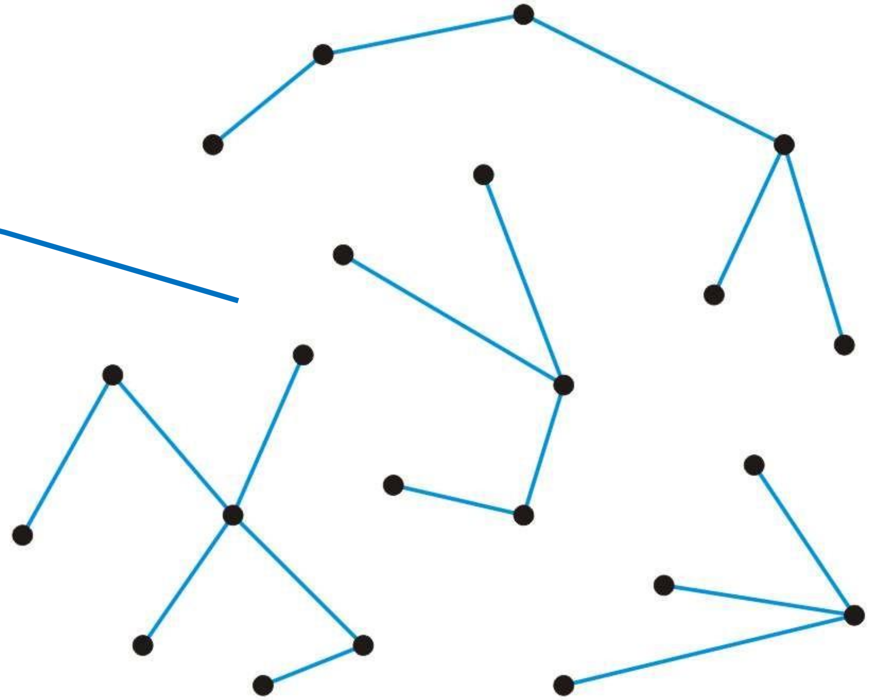



Tree rooted at B

# FOREST

- A forest is any graph that has no cycles
- Consequences
  - The number of edges is $|E| < |V|$
  - The number of trees is $|V| - |E|$
  - Removing any one edge adds one more tree to the forest

- Forest with 22 vertices and 18 edges
- Four trees

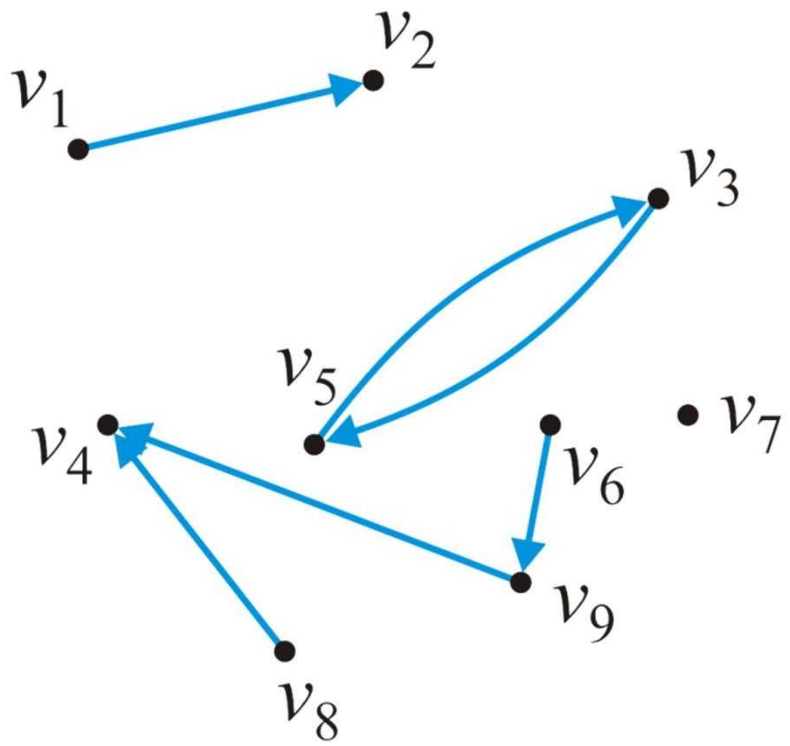


# DIRECTED GRAPHS

- In a directed graph, the edges on a graph are associated with a direction
  - Edges are ordered pairs $(v_j, v_k)$ denoting a connection from $v_j$ to $v_k$
  - The edge $(v_j, v_k)$ is different from the edge $(v_k, v_j)$
- Streets are directed graphs
  - In most cases, you can go two ways unless it is a one-way street

# DIRECTED GRAPHS

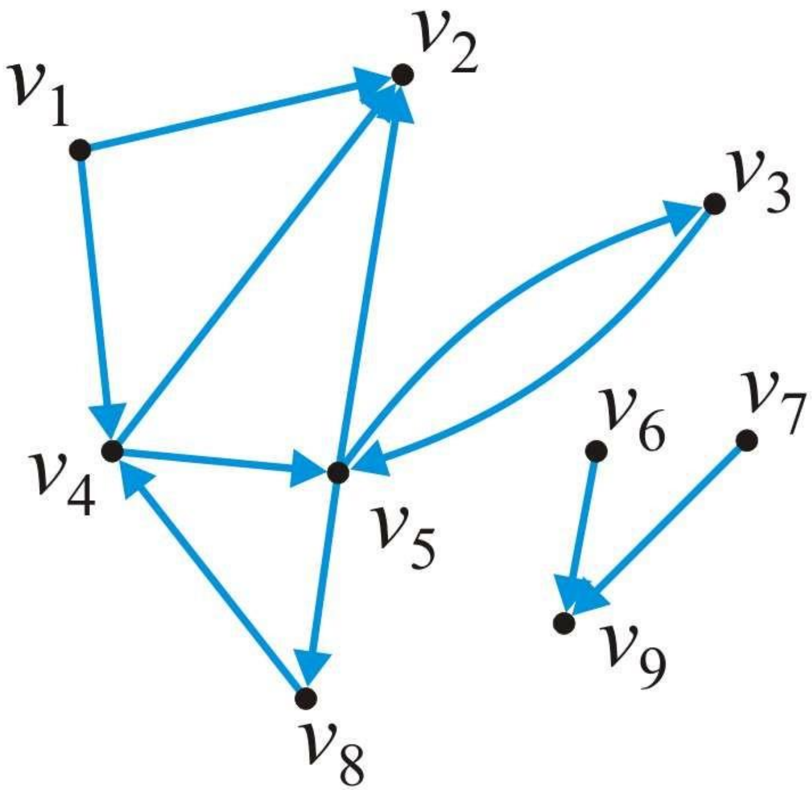
- Given our graph of nine vertices $V = \{v_1, v_2, \dots v_9\}$
  - These six pairs $(v_j, v_k)$ are directed edges
  - $E = \{(v_1, v_2), (v_3, v_5), (v_5, v_3), (v_6, v_9), (v_8, v_4), (v_9, v_4)\}$



# IN AND OUT DEGREE

- Degree of a vertex must be modified to consider both cases:
  - Out-degree of a vertex is the number of vertices which are adjacent to the given vertex
    - ➢ Number of outgoing edges
  - In-degree of a vertex is the number of vertices which this vertex is adjacent to
    - ➢ Number of incoming edges

- In this graph:
  - In-degree($v_1$) = 0 out-degree($v_1$) = 2
  - In-degree($v_5$) = 2 out-degree($v_5$) = 3

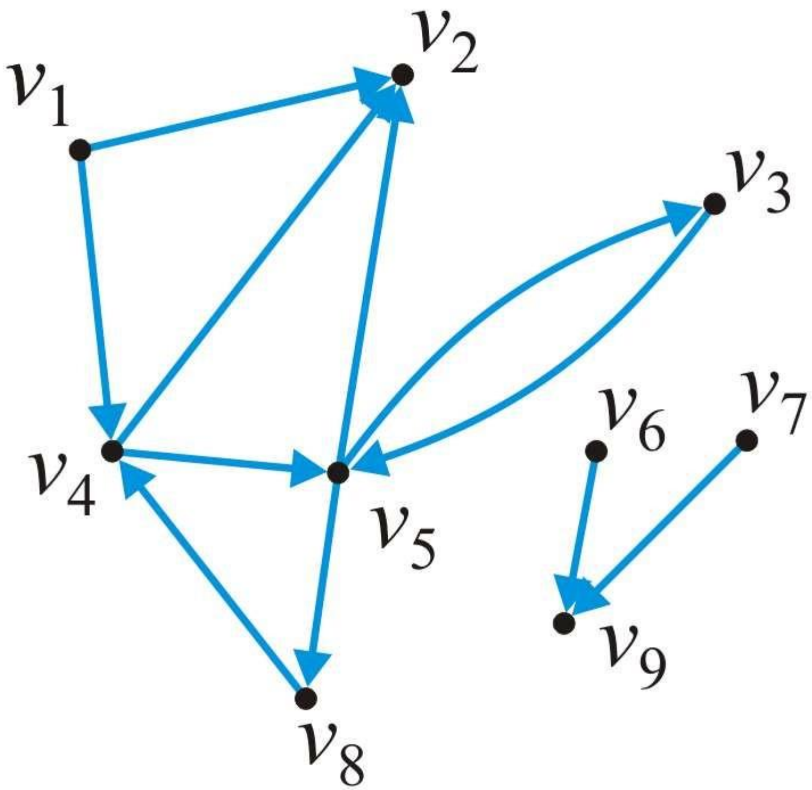


# PATH

- A path in a directed graph is an ordered sequence of vertices
  - $(v_0, v_1, v_2, ..., v_k)$
  - where $(v_j - 1, v_j)$ is an edge for $j = 1, ..., k$

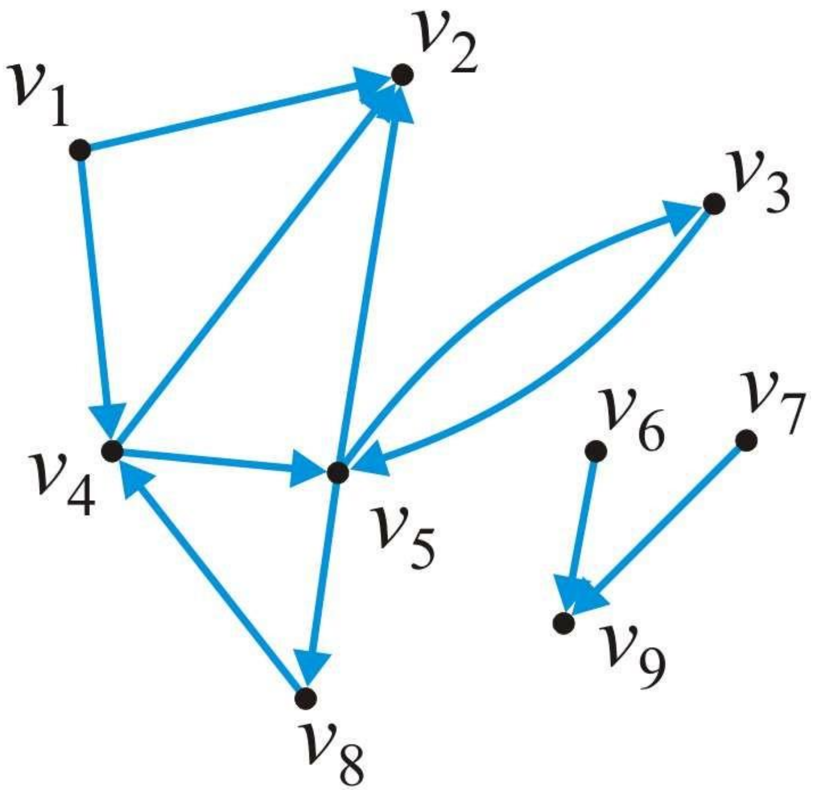
- A path of length 5 in this graph is
  - $(v_1, v_4, v_5, v_3, v_5, v_2)$

- A simple cycle of length 3 is
  - $(v_8, v_4, v_5, v_8)$



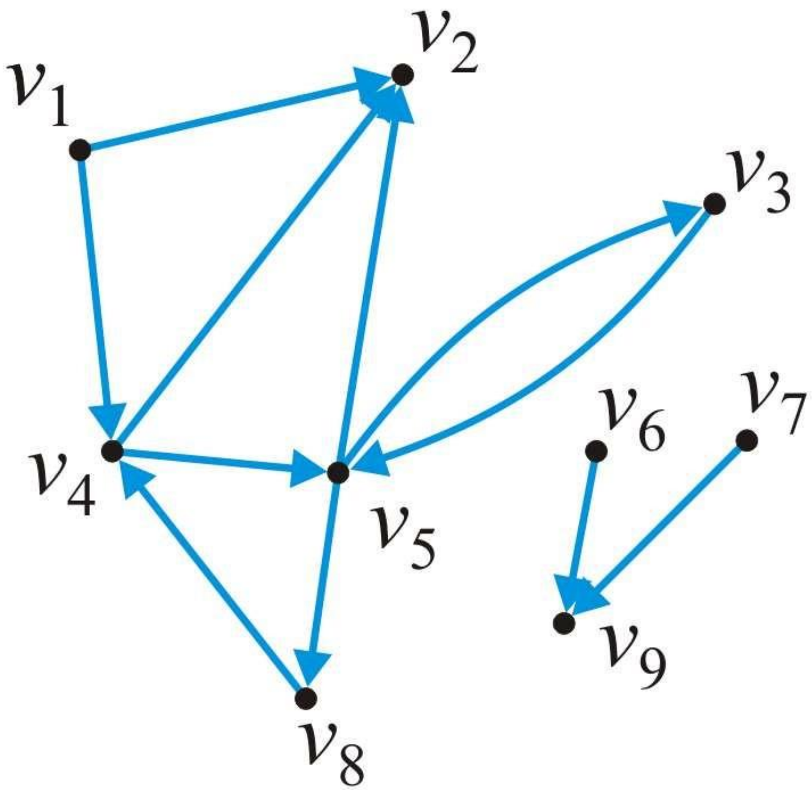
# CONNECTEDNESS

- Two vertices $v_j$, $v_k$ are said to be connected if there exists a path from $v_j$ to $v_k$
  - A graph is strongly connected if there exists a directed path between any two vertices
  - A graph is weakly connected if there exists a path between any two vertices that ignores the direction



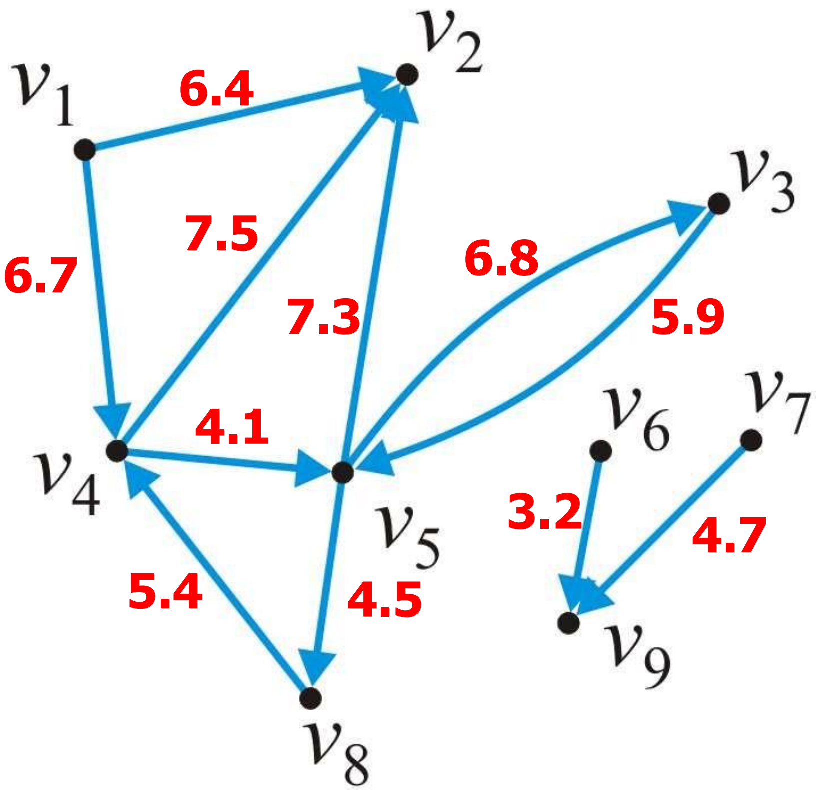
# CONNECTEDNESS – EXAMPLE

- The sub-graph $\{v_3, v_4, v_5, v_8\}$ is strongly connected
- The sub-graph $\{v_1, v_2, v_3, v_4, v_5, v_8\}$ is weakly connected



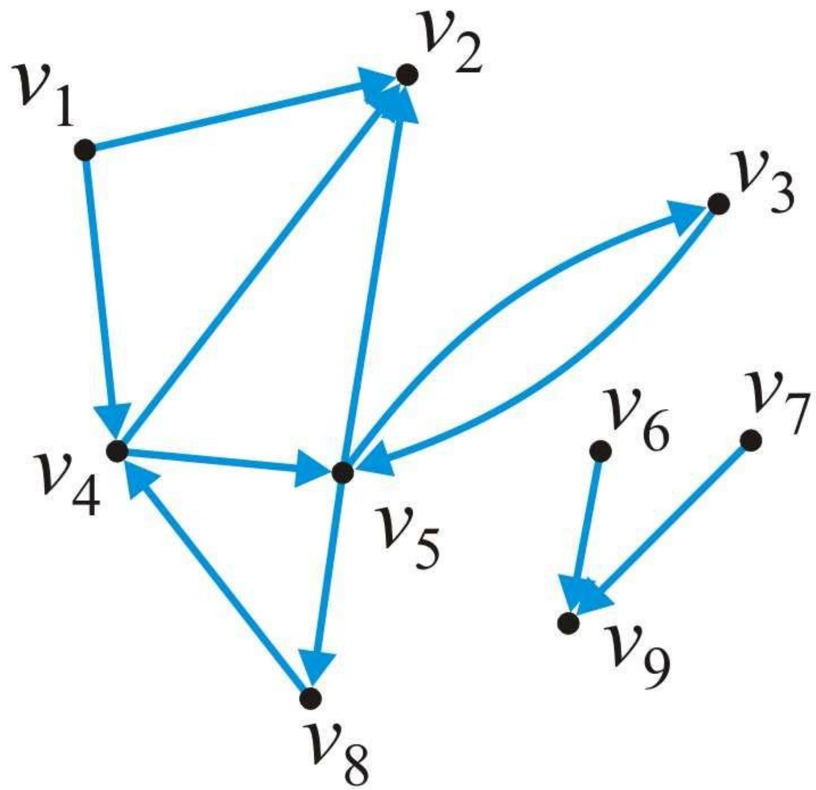
# WEIGHTED DIRECTED GRAPHS

- Each edge is associated with a value
- If both $(v_j, v_k)$ and $(v_k, v_j)$ are edges
  - It is not required that they have the same weight



# REPRESENTATION

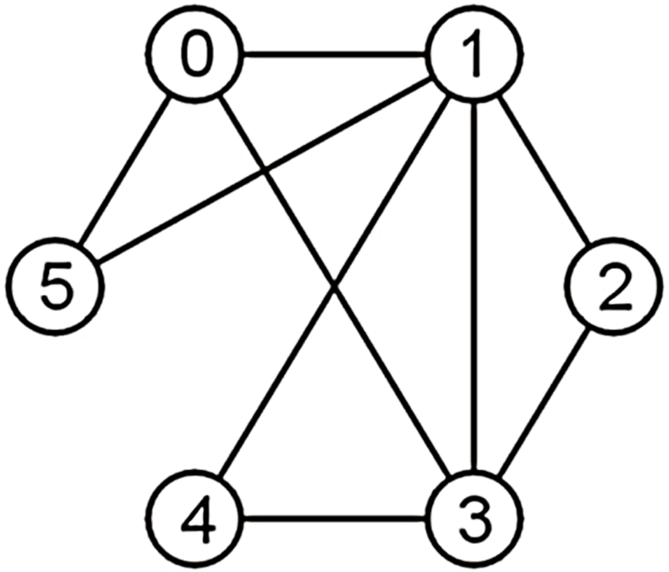
- How do we store the adjacency relations?
  - Adjacency matrix
  - Adjacency list



# ADJACENCY MATRIX

- Two dimensional matrix of size n x n where n = |V|
- a[i, j] = 0 (F) if there is no edge between vertices $v_i$
- a[i, j] = 1 (T) if there is an edge between vertices $v_i$
- Adjacency matrix of undirected graphs is symmetric
  - a[i, j] = a[j, i]

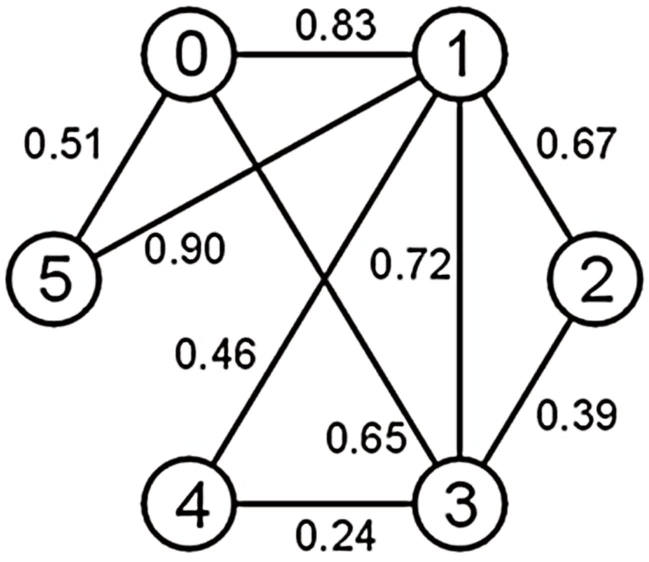
| | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | | T | F | T | F | T |
| 1 | T | | T | T | T | T |
| 2 | F | T | | T | F | F |
| 3 | T | T | T | | T | F |
| 4 | F | T | F | T | | F |
| 5 | T | T | F | F | F | |



# ADJACENCY MATRIX – WEIGHTED GRAPH

- The matrix entry [j, k] is set to the weight of the edge $(v_j, v_k)$
- How to indicate absence of an edge in the graph?

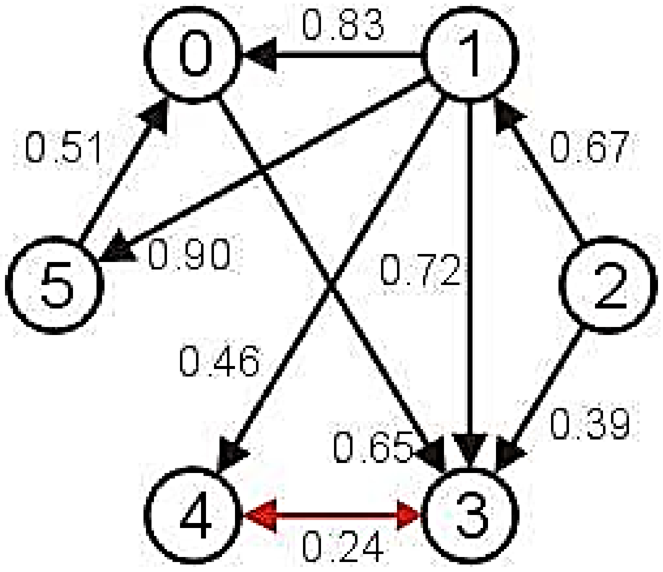
| | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | | 0.83 | | 0.65 | | 0.51 |
| 1 | 0.83 | | 0.67 | 0.72 | 0.46 | 0.90 |
| 2 | | 0.67 | | 0.39 | | |
| 3 | 0.65 | 0.72 | 0.39 | | 0.24 | |
| 4 | | 0.46 | | 0.24 | | |
| 5 | 0.51 | 0.90 | | | | |



# ADJACENCY MATRIX – DIRECTED GRAPH

- For directed graph the matrix would not necessarily be symmetric

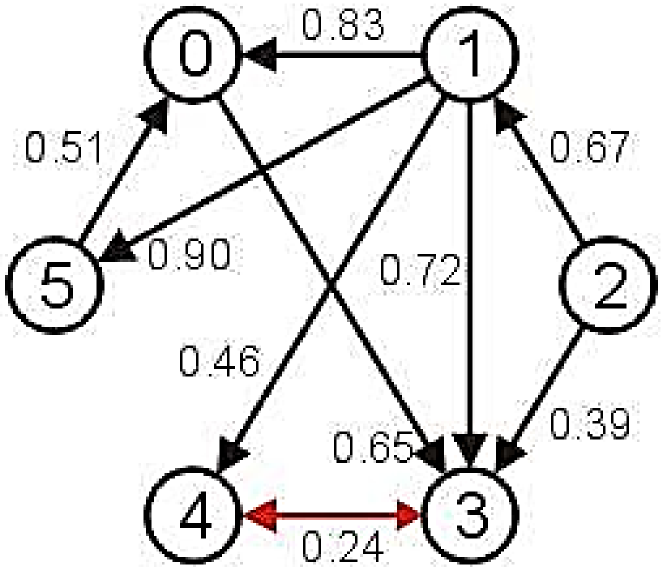
| | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | | | | 0.65 | | |
| 1 | 0.83 | | | 0.72 | 0.46 | 0.90 |
| 2 | | 0.67 | | 0.39 | | |
| 3 | | | | | 0.24 | |
| 4 | | | | 0.24 | | |
| 5 | 0.51 | | | | | |



# ADJACENCY MATRIX – DIRECTED GRAPH

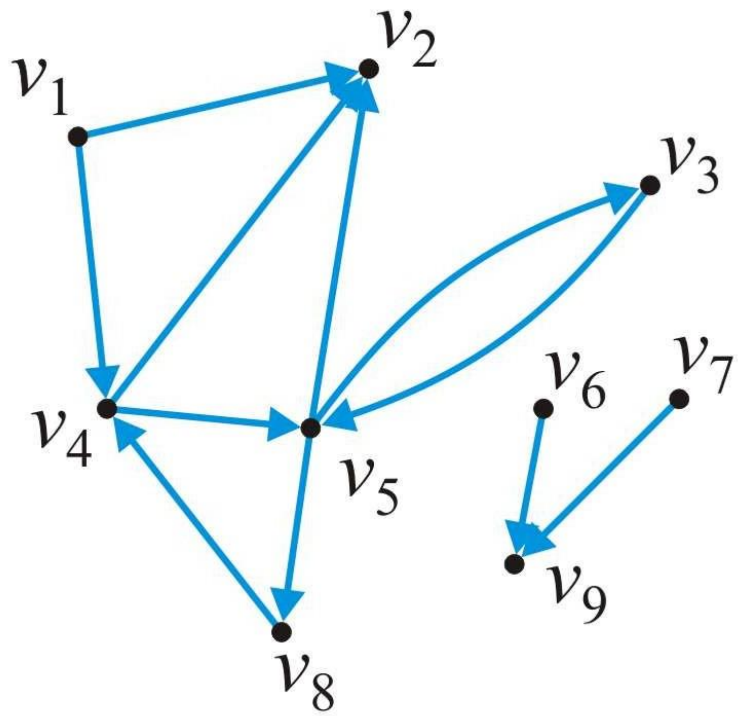
- For directed graph the matrix would not necessarily be symmetric

| | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | | | | 0.65 | | |
| 1 | 0.83 | | | 0.72 | 0.46 | 0.90 |
| 2 | | 0.67 | | 0.39 | | |
| 3 | | | | | 0.24 | |
| 4 | | | | 0.24 | | |
| 5 | 0.51 | | | | | |



# ADJACENCY MATRIX – ANALYSIS

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | | 1 | | 1 | | | | | |
| 2 | | | | | | | | | |
| 3 | | | | | 1 | | | | |
| 4 | | 1 | | | 1 | | | | |
| 5 | | 1 | 1 | | | | | 1 | |
| 6 | | | | | | | | | 1 |
| 7 | | | | | | | | | 1 |
| 8 | | | | 1 | | | | | |
| 9 | | | | | | | | | |

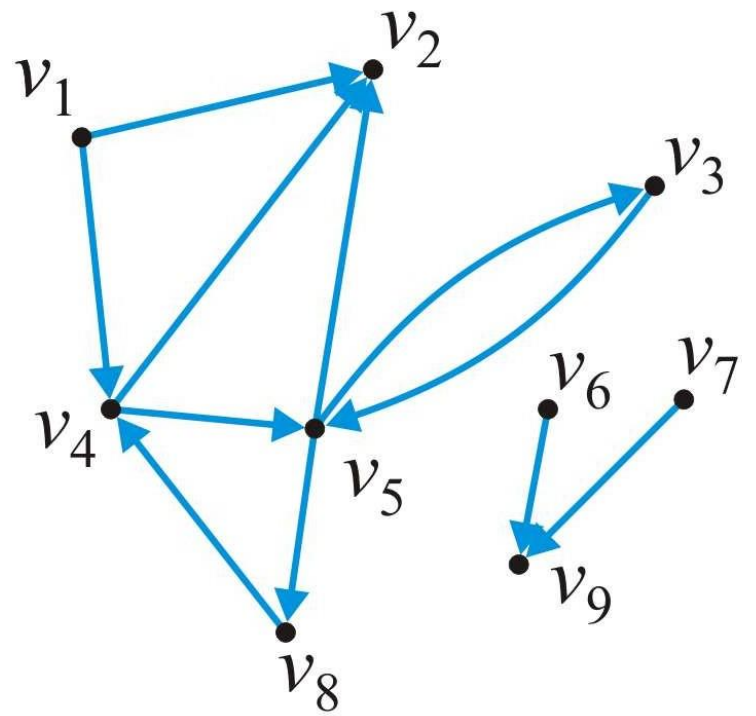



- Requires memory : $O(|V|^2)$
- Determining if $v_j$ is adjacent to $v_k$ : $O(1)$
- Finding all neighbors of $v_j$ : $O(|V|)$

# ADJACENCY MATRIX – PROBLEM

- Very sparsely populated
  - Out of 81 cells only 11 are 1 (or T)

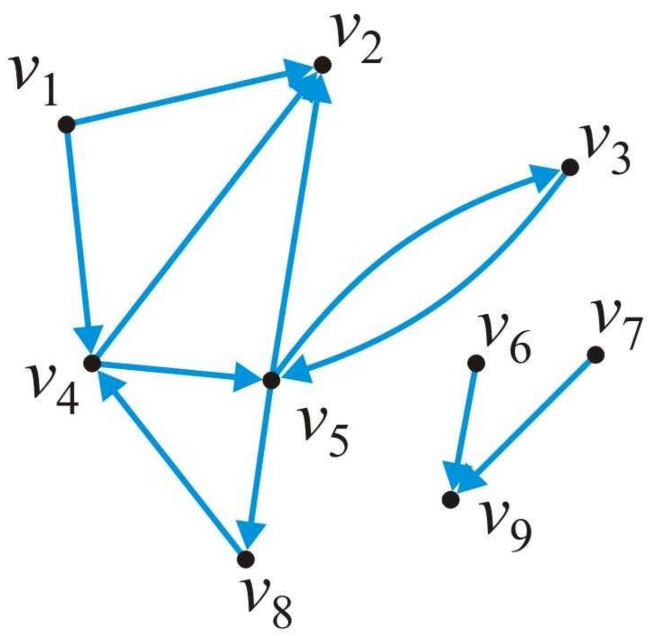
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | | 1 | | 1 | | | | | |
| 2 | | | | | | | | | |
| 3 | | | | | 1 | | | | |
| 4 | | 1 | | | 1 | | | | |
| 5 | | 1 | 1 | | | | | 1 | |
| 6 | | | | | | | | | 1 |
| 7 | | | | | | | | | 1 |
| 8 | | | | 1 | | | | | |
| 9 | | | | | | | | | |



# ADJACENCY LIST

- Each vertex is associated with a list of its neighbors
  - A vertex w is inserted in the list for vertex v if edge ( v, w) exists

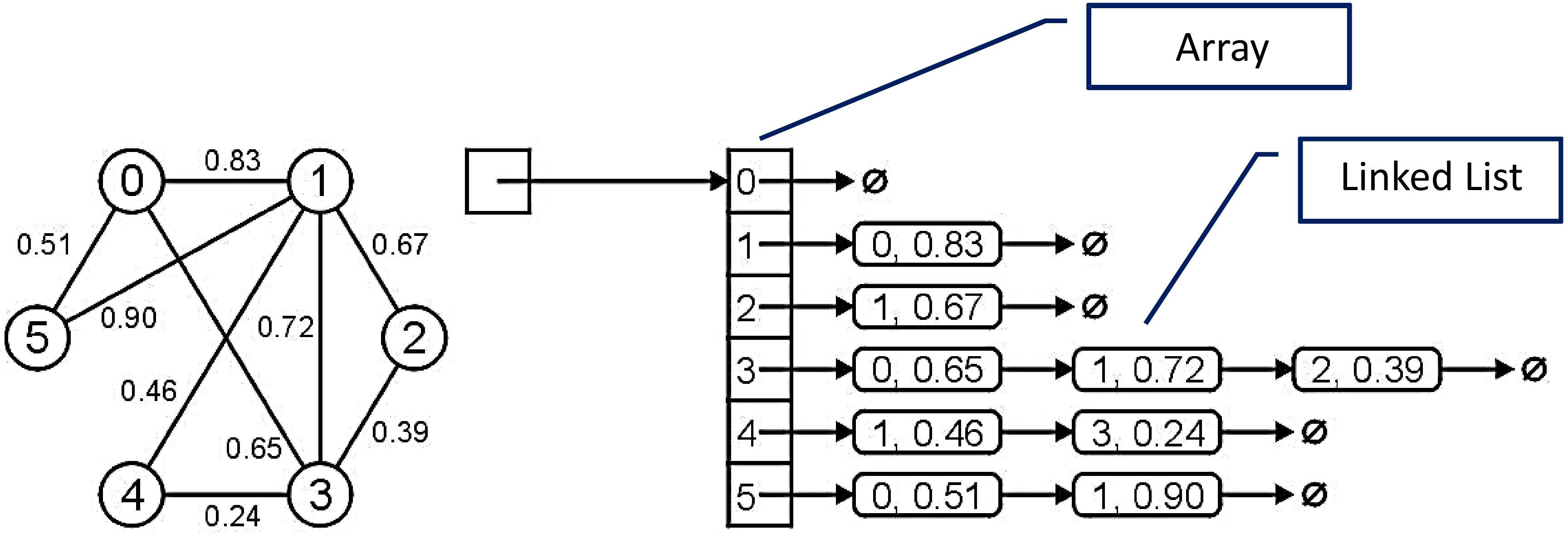
1 • → 2 → 4
2 •
3 • → 5
4 • → 2 → 5
5 • → 2 → 3 → 8
6 • → 9
7 • → 9
8 • → 4
9 •




- Requires memory : O(|V| + |E|)

# ADJACENCY LIST – WEIGHTED GRAPHS

- An adjacency list for a weighted graph contains two elements
  - First element for the vertex
  - Second element for the weight of that edge




- When the vertices are identified by a name (i.e., string)
  - Hash-table of lists is used to implement the adjacency list

# CONCLUSION

- In this lecture we have studied:
  - Graph
  - Undirected and Directed Graph
  - Weighted Graph
  - Adjacency Matrix and Adjacency List

# Question?