

DATA STRUCTURES AND ALGORITHMS

LECTURE 19

GRAPH THEORY

IMRAN IHSAN
ASSISTANT PROFESSOR
AIR UNIVERSITY, ISLAMABAD

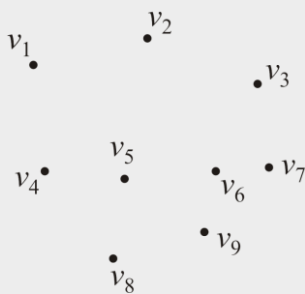
GRAPH

DEFINITION

Consider this collection of vertices

$$V = \{v_1, v_2, \dots, v_9\}$$

where $|V| = n$



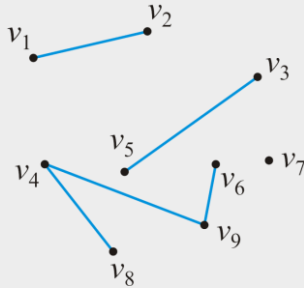
GRAPH

DEFINITION

Associated with these vertices are $|E| = 5$ edges

$$E = \{\{v_1, v_2\}, \{v_3, v_5\}, \{v_4, v_8\}, \{v_4, v_9\}, \{v_6, v_9\}\}$$

- The pair $\{v_j, v_k\}$ indicates that both vertex v_j is adjacent to vertex v_k and vertex v_k is adjacent to vertex v_j



UNDIRECTED GRAPHS

We will assume in this course that a vertex is never adjacent to itself

- For example, $\{v_1, v_1\}$ will not define an edge

The maximum number of edges in an undirected graph is

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



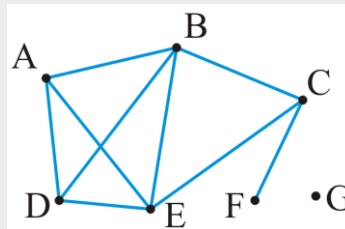
UNDIRECTED GRAPHS

Example: given the $|V| = 7$ vertices

$$V = \{A, B, C, D, E, F, G\}$$

and the $|E| = 9$ edges

$$E = \{\{A, B\}, \{A, D\}, \{A, E\}, \{B, C\}, \{B, D\}, \{B, E\}, \{C, E\}, \{C, F\}, \{D, E\}\}$$



DEGREE

UNDIRECTED GRAPHS

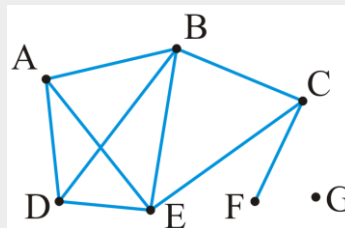
The degree of a vertex is defined as the number of adjacent vertices

$$\text{degree}(A) = \text{degree}(D) = \text{degree}(C) = 3$$

$$\text{degree}(B) = \text{degree}(E) = 4$$

$$\text{degree}(F) = 1$$

$$\text{degree}(G) = 0$$

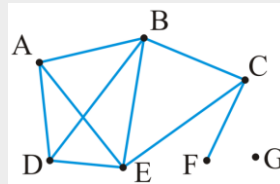
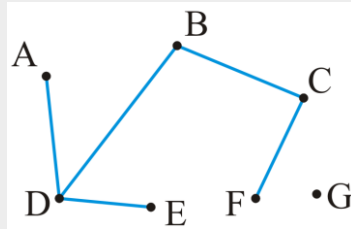


Those vertices adjacent to a given vertex are its *neighbors*



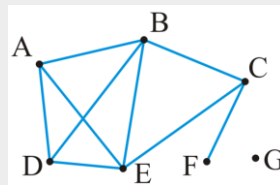
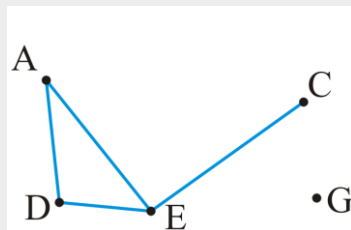
SUB-GRAPHS

- A sub-graph of a graph a subset of the vertices and a subset of the edges that connected the subset of vertices in the original graph



VERTEX-INDUCED SUB-GRAPHS

- A vertex-induced sub-graph is a subset of a the vertices where the edges are all edges in the original graph



PATHS

A path in an undirected graph is an ordered sequence of vertices

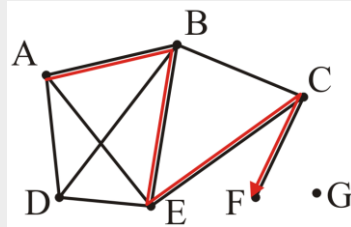
$$(v_0, v_1, v_2, \dots, v_k)$$

where $\{v_{j-1}, v_j\}$ is an edge for $j = 1, \dots, k$

- Termed a *path from* v_0 to v_k
- The length of this path is k

A path of length 4:

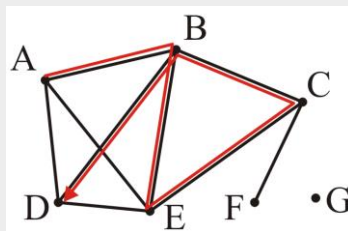
(A, B, E, C, F)



PATHS

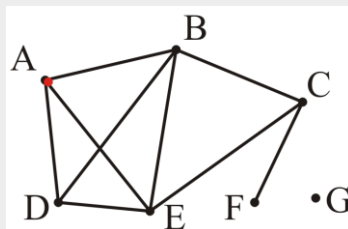
A path of length 5:

(A, B, E, C, B, D)



A *trivial* path of length 0:

(A)



SIMPLE PATHS

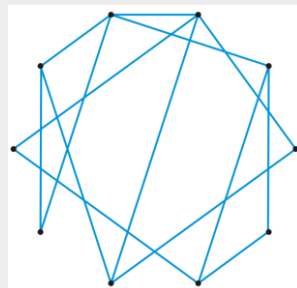
- A simple path has no repetitions other than perhaps the first and last vertices
- A simple cycle is a simple path of at least two vertices with the first and last vertices equal
 - Note: these definitions are not universal



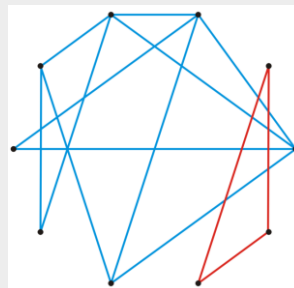
CONNECTEDNESS

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 between any two vertices



A connected graph

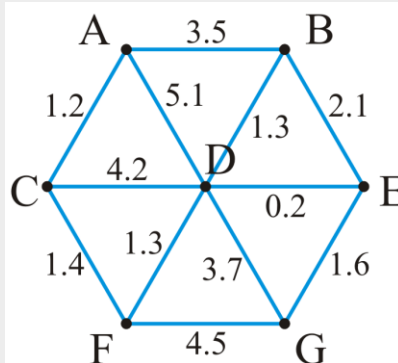


An unconnected graph



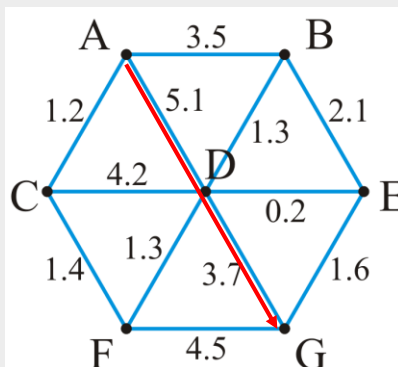
WEIGHTED GRAPHS

- 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



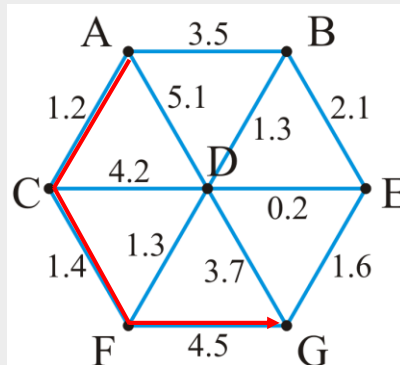
WEIGHTED GRAPHS

- The 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 $5.1 + 3.7 = 8.8$



WEIGHTED GRAPHS

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



www.au.edu.pk

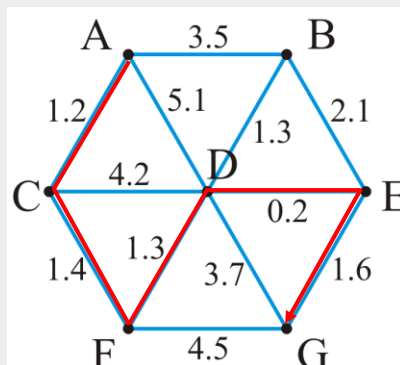
Air University, Islamabad

www.imranhsan.com

15

WEIGHTED GRAPHS

- Problem: find the shortest path between two vertices
 - Here, the shortest path from A to H is (A, C, F, D, E, G) with length 5.7



www.au.edu.pk

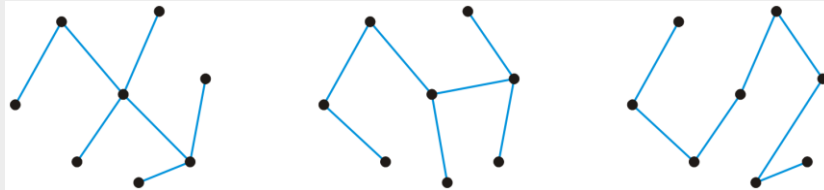
Air University, Islamabad

www.imranhsan.com

16

TREES

- A graph is a tree if it is connected and there is a unique path between any two vertices
 - Three trees on the same eight vertices

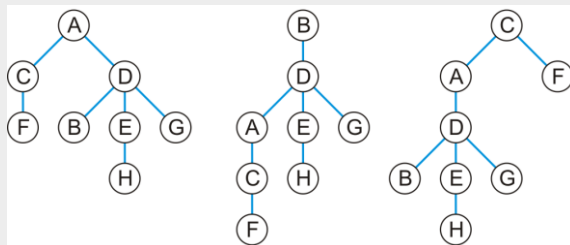
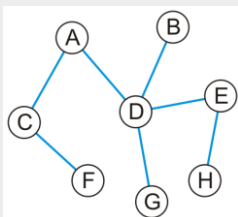


- Consequences:
 - The number of edges is $|E| = |V| - 1$
 - The graph is acyclic, that is, it does not contain any cycles
 - Adding one more edge must create a cycle
 - Removing any one edge creates two disjoint non-empty sub-graphs



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
- and then recursively defining:
 - All neighboring vertices other than that one designated its parent are now defined to be that vertices children
- Given this tree, here are three rooted trees associated with it



FORESTS

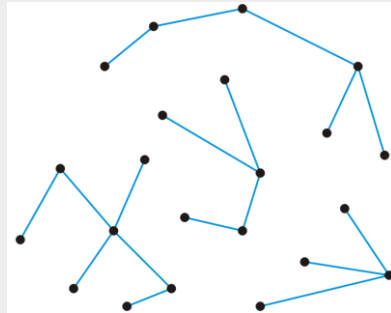
A forest is any graph that has no cycles

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

Here is a forest with 22 vertices and 18 edges

- There are four trees



DIRECTED GRAPHS

In a *directed graph*, the edges on a graph are be 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

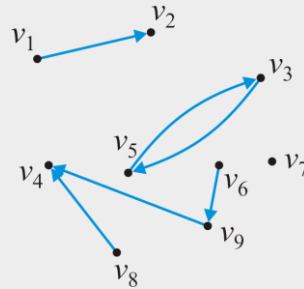


DIRECTED GRAPHS

Given our graph of nine vertices $V = \{v_1, v_2, \dots, v_9\}$

- These six pairs (v_j, v_k) are *directed edges*

$$E = \{(v_1, v_2), (v_3, v_5), (v_5, v_3), (v_6, v_9), (v_8, v_4), (v_9, v_4)\}$$



- The maximum number of directed edges in a directed graph is

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



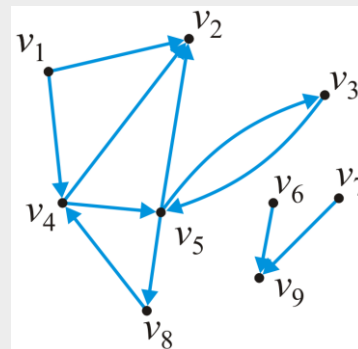
IN AND OUT DEGREES

The degree of a vertex must be modified to consider both cases:

- The *out-degree* of a vertex is the number of vertices which are adjacent to the given vertex
- The *in-degree* of a vertex is the number of vertices which this vertex is adjacent to

In this graph:

$\text{in_degree}(v_1) = 0$	$\text{out_degree}(v_1) = 2$
$\text{in_degree}(v_5) = 2$	$\text{out_degree}(v_5) = 3$



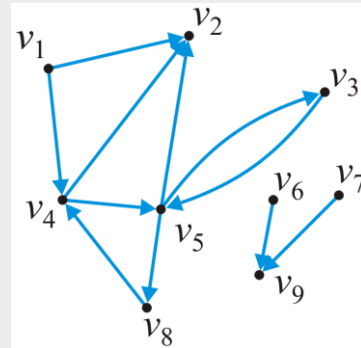
SOURCES AND SINKS

Some definitions:

- Vertices with an in-degree of zero are described as *sources*
- Vertices with an out-degree of zero are described as *sinks*

In this graph:

- Sources: v_1, v_6, v_7
- Sinks: v_2, v_9



PATHS

A path in a directed graph is an ordered sequence of vertices

$$(v_0, v_1, v_2, \dots, v_k)$$

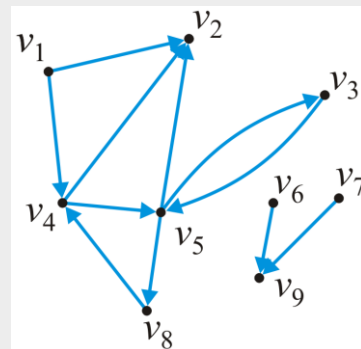
where (v_{j-1}, v_j) is an edge for $j = 1, \dots, k$

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

$$(v_8, v_4, v_5, v_8)$$



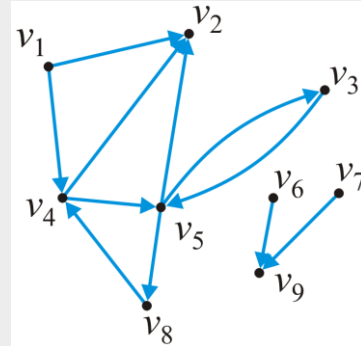
CONNECTEDNESS

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* there exists a path between any two vertices that ignores the direction

In this graph:

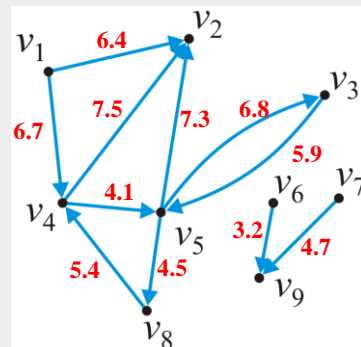
- The sub-graph $\{v_3, v_4, v_5, v_8\}$ is strongly connected
- The sub-graph $\{v_1, v_2, v_3, v_4, v_5, v_8\}$ is weakly connected



WEIGHTED DIRECTED GRAPHS

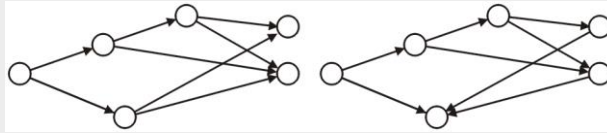
In a weighted directed graphs, each edge is associated with a value

Unlike weighted undirected graphs, if both (v_j, v_k) and (v_k, v_j) are edges, it is not required that they have the same weight

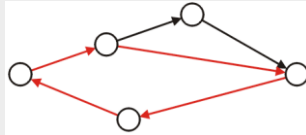


DIRECTED ACYCLIC GRAPHS

- A directed acyclic graph is a directed graph which has no cycles
 - These are commonly referred to as DAGs
 - They are graphical representations of partial orders on a finite number of elements
- These two are DAGs:



- This directed graph is not acyclic:



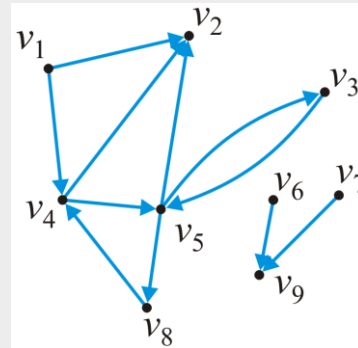
DIRECTED ACYCLIC GRAPHS

- Applications of directed acyclic graphs include:
 - The parse tree constructed by a compiler
 - A reference graph that can be garbage collected using simple reference counting
 - Dependency graphs such as those used in instruction scheduling and makefiles
 - Dependency graphs between classes formed by inheritance relationships in object-oriented programming languages
 - Information categorization systems, such as folders in a computer
 - Directed acyclic word graph data structure to memory-efficiently store a set of strings (words)
 - Reference: http://en.wikipedia.org/wiki/Directed_acyclic_graph



REPRESENTATIONS

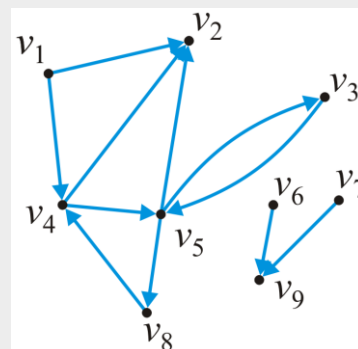
- How do we store the adjacency relations?
 - Binary-relation list
 - Adjacency matrix
 - Adjacency list



REPRESENTATIONS

BINARY-RELATION LIST

- The most inefficient is a relation list:
 - A container storing the edges
 $\{(1, 2), (1, 4), (3, 5), (4, 2), (4, 5), (5, 2), (5, 3), (5, 8), (6, 9), (7, 9), (8, 4)\}$
- Requires $\Theta(|E|)$ memory
- Determining if v_j is adjacent to v_k is $O(|E|)$
- Finding all neighbors of v_j is $\Theta(|E|)$



REPRESENTATIONS

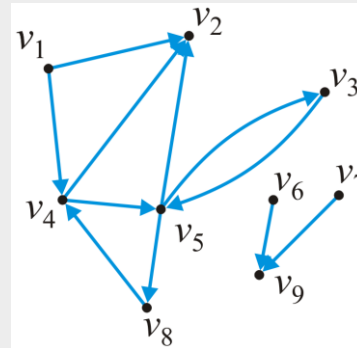
ADJACENCY MATRIX

Requiring more memory but also faster, an adjacency matrix

- The matrix entry (j, k) is set to true if there is an edge (v_j, v_k)

	1	2	3	4	5	6	7	8	9
1		T		T					
2									
3					T				
4		T			T				
5		T	T					T	
6									T
7									T
8				T					
9									

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



www.au.edu.pk

Air University, Islamabad

www.imranhsan.com

31

REPRESENTATIONS

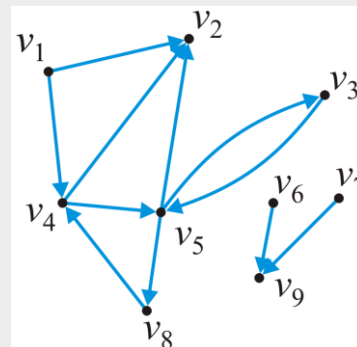
ADJACENCY LIST

Most efficient for algorithms is an adjacency list

- Each vertex is associated with a list of its neighbors

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

- Requires $\Theta(|V| + |E|)$ memory
- On average:
 - Determining if v_j is adjacent to v_k is
 - Finding all neighbors of v_j is



www.au.edu.pk

Air University, Islamabad

www.imranhsan.com

32

THE GRAPH ADT

- The Graph ADT describes a container storing an adjacency relation
- Queries include:
 - The number of vertices
 - The number of edges
 - List the vertices adjacent to a given vertex
 - Are two vertices adjacent?
 - Are two vertices connected?
- Modifications include:
 - Inserting or removing an edge
 - Inserting or removing a vertex (and all edges containing that vertex)
- The run-time of these operations will depend on the representation

