Data Structure

تركيب بيانات
الفرقة الثالثة علوم إحصاء وعلوم الحاسب

By
Dr. Reda Elbarougy
د/ رضا الباروجى
Lecturer of computer sciences
In Mathematics Department
Faculty of Science
Damietta University

الاحد ٢٩-٤-٢٠١٧

رقم المحاضرة

م	التاريخ	الموضوع	الحالة	
١	٢٠١٧-٠٢-١٢	الفصل الأول مقدمة	تم	
٢	٢٠١٧-٠٢-١٩	الفصل الأول مقدمة	تم	
٣	٢٠١٧-٠٣-٠٥	الفصل الثانى اساسيات	تم	
٤	٢٠١٧-٠٣-١٢	الفصل الرابع المصفوفات	تم	
٥	٢٠١٧-٠٣-١٩	الفصل الرابع المصفوفات	تم	
٦	٢٠١٧-٠٣-٢٦	الفصل الخامس		
٧	٢٠١٧-٠٤-٠٢	الفصل الخامس		
٨	٢٠١٧-٠٤-٠٤	الفصل السادس	أضافى	Stacks
٩	٢٠١٧-٠٤-٠٩	الفصل السادس		
	٢٠١٧-٠٤-١٦	أجازة		
١٠	٢٠١٧-٠٤-١٩	لا		
١١	٢٠١٧-٠٤-٢٣	الفصل السابع		
١٢	٢٠١٧-٠٤-٣٠			
	٢٠١٧-٠٥-٠٧			

Chapter 8:
Graphs and their application

Objectives

Objectives

After completing this chapter, you will be able to:

- Use the relevant terminology to describe the difference between graphs and other types of collections
- Recognize applications for which graphs are appropriate
- Explain the structural differences between an adjacency matrix representation of a graph and the adjacency list representation of a graph
- Analyze the performance of basic graph operations using the two representations of graphs
- Describe the differences between a depth-first traversal and a breadth-first traversal of a graph

Outline

8.1 Introduction

8.2 Graph Theory Terminology

- Graph and Multigraph
- Proposition 8.1
- Directed Graphs

8.3 Sequential Representation Of Graphs

- Adjacency matrix;
- Proposition 8.2
- Path matrix
- Proposition 8.3

8.5 Linked representation of a graph

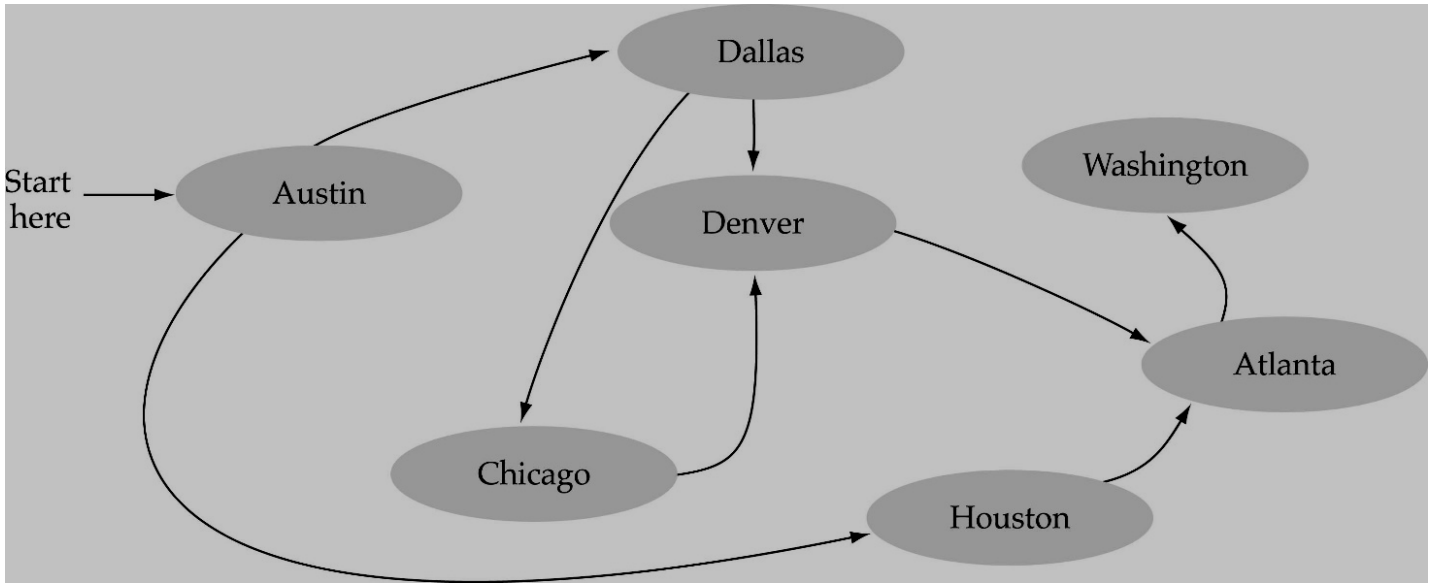
Introduction

- A Graph is a nonlinear data structure, which is having point to point relationship among the nodes. Each node of the graph is called as a vertex and link or line drawn between them is called and edge

8.2 Graph Theory Terminology

What is a graph?

- A data structure that consists of a set of nodes (*vertices*) and a set of edges that relate the nodes to each other
- The set of edges describes relationships among the vertices

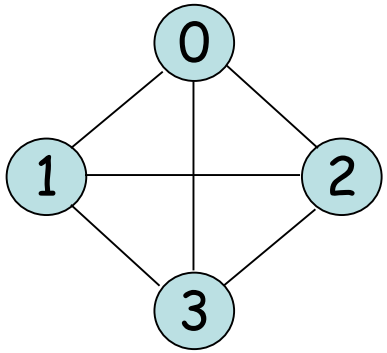


Graphs

- A graph G consists of two things:
  1) A set V of elements called nodes (or points or vertices)
  2) A set E of edges such that each edge e in E is identified with a unique (unordered) pair [u, v] of nodes in V, denoted by e = [u, v]
- Sometimes we indicate the parts of a graph by writing G = (V, E).

V(G): a finite, nonempty set of vertices

E(G): a set of edges (pairs of vertices)



V(G) = {0, 1, 2, 3}

E(G) = {(0, 1), (0, 2), (0, 3), (1, 2), (1, 3), (2, 3)}

Graphs and Multigraphs

- Suppose edge **e=[u,v]**, then the nodes u and v are called **end points** of the edge e.
- The node u is called **source** node and node v is called **destination** node,
- The nodes u and v are called **adjacent** nodes or **neighbors**.
- The line drawn between to adjacent nodes is called an **edge**.
- If an edge is having direction, then the source node is called adjacent to the destination and destination node is adjacent from source.
- The **degree of a node** u, written **deg(u)**, is the number of edges containing u.
- Isolated node: If degree of a node is zero i.e. if the node is not having any edges, then the node is called isolated node.
- If deg(u) = 0 — that is, if u does not belong to any edge—then u is called an **isolated node**.

Path and Cycle

- Path: A path is a sequence of consecutive edges between a source and a destination through different nodes.
- A path, said to be closed if source is equal to destination.
- Simple path: The path is said to be simple if all nodes are distinct.
- Length of a path: Number of edges on the path.
- A path P of length n from a node u to a node v is defined as a sequence of n + 1 nodes.

$$\mathbf{P = (v_0, v_1, v_2, \ldots, v_n)}$$

such that $u = v_0$; v_{i-1} is adjacent to v_i for $i = 1,2, \ldots, n$ and $v_n = v$.

- Cycle: A cycle is closed path with length 3 or more. A cycle of length k is called a k-cycle.
- **Loops:** An edge e is called a loop if it has identical endpoints, that is, if e = [u, u].

Path and Cycle

Types of Path

i. Simple Path

ii. Cycle Path

i. Simple Path: Simple path is a path in which first and last vertex are different ($V_0 \neq V_n$)

ii. Cycle Path: Cycle path is a path in which first and last vertex are same ($V_0 = V_n$). It is also called as Closed path.

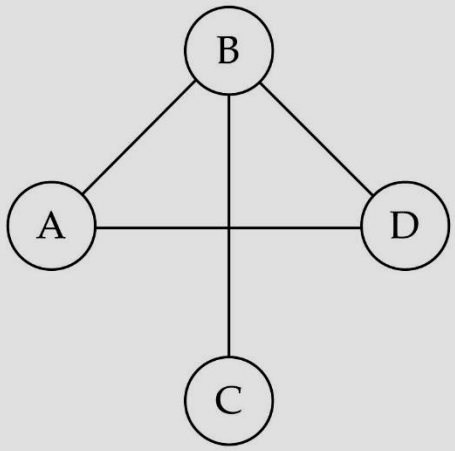
Graphs and Multigraphs

- **Connected Graph:**
  A graph G is said to be connected if there is a path between any two of its nodes.
- **Complete Graph :** A graph is called complete if all the nodes of the graph are adjacent to each other. A complete graph with n nodes will have n*(n-1)/2 edges.
- **Tree:**
  A connected graph T without any cycles is called a tree graph or free tree or, simply, a tree.

Directed vs. undirected graphs

- When the edges in a graph have no direction, the graph is called *undirected*

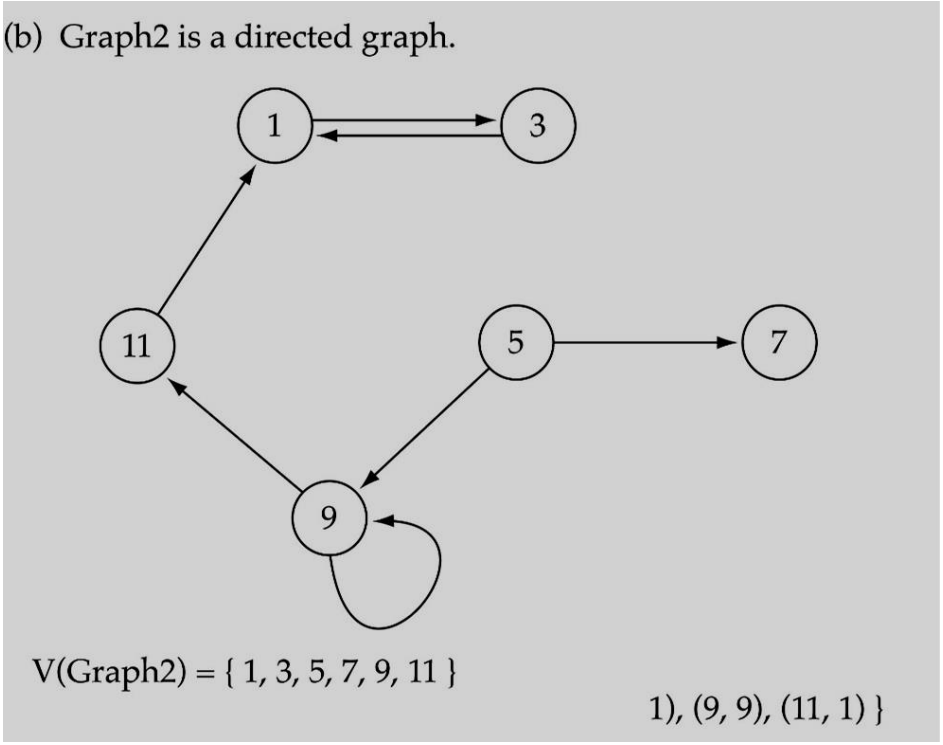
ected graph.



V(Graph1) = { A, B, C, D }
E(Graph1) = { (A, B), (A, D), (B, C), (B, D) }

Directed vs. undirected graphs

- When the edges in a graph have a direction, the graph is called *directed* (or *digraph*)



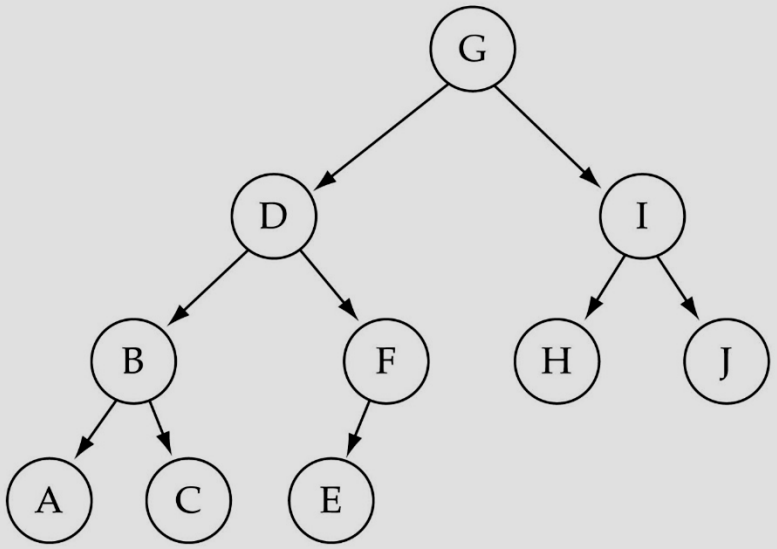
E(Graph2) = {(1,3) (3,1) (5,9) (9,11) (5,7)

Warning: if the graph is directed, the order of the vertices in each edge is important !!

Trees vs graphs

- Trees are special cases of graphs!!

(c) Graph3 is a directed graph.



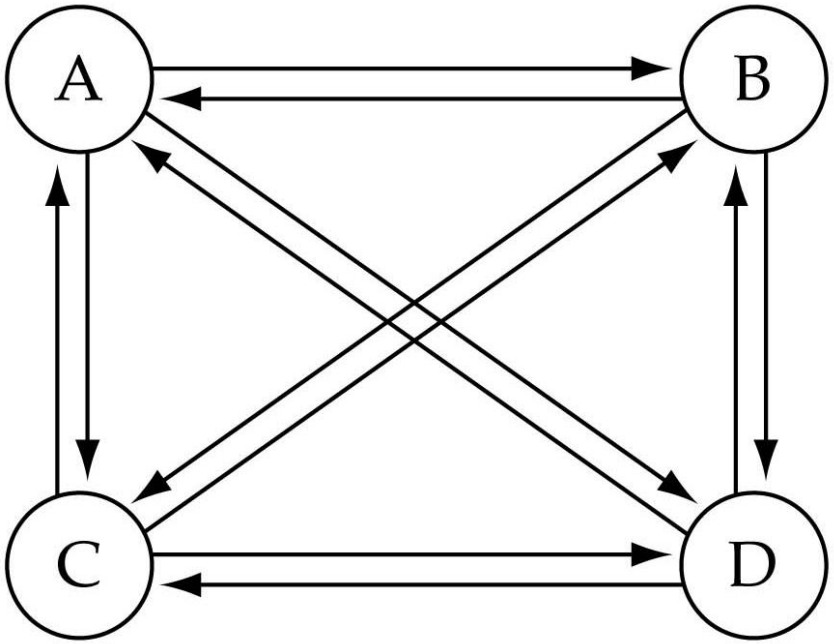
V(Graph3) = { A, B, C, D, E, F, G, H, I, J }
E(Graph3) = { (G, D), (G, J), (D, B), (D, F) (I, H), (I, J), (B, A), (B, C), (F, E) }

Graph terminology

- What is the number of edges in a complete directed graph with N vertices?

*N * (N-1)*

$O(N^2)$



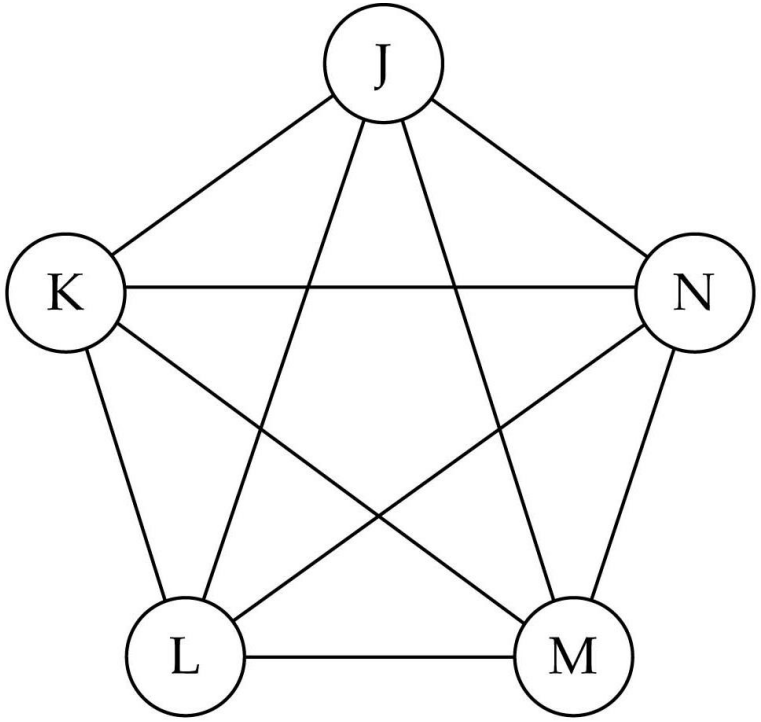
(a) Complete directed graph.

Graph terminology

- What is the number of edges in a complete undirected graph with N vertices?

*N * (N-1) / 2*

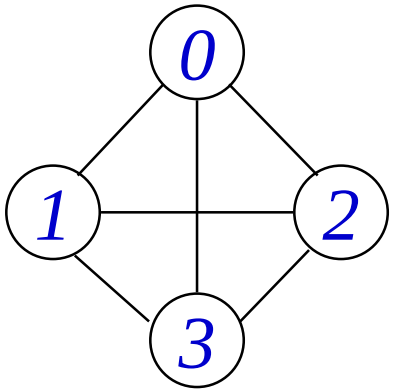
$O(N^2)$



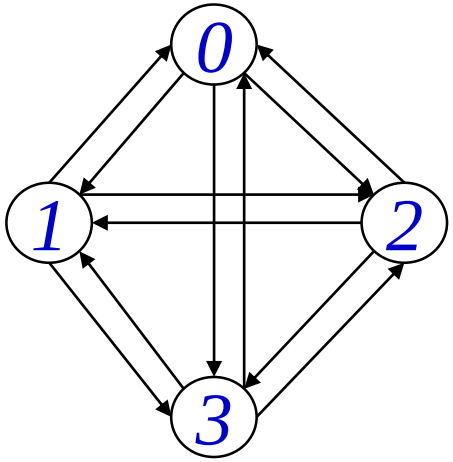
(b) Complete undirected graph.

Graphs and Multigraphs

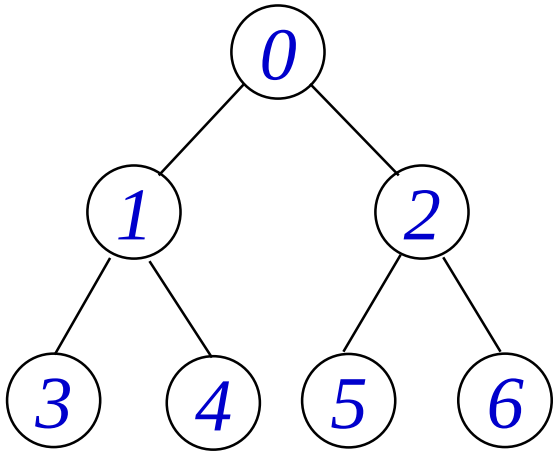
- Examples for Graph
  - complete undirected graph: n(n-1)/2 edges
  - complete directed graph: n(n-1) edges



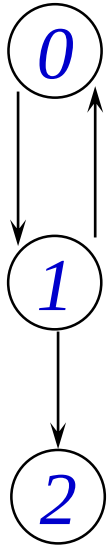
G_1



complete graph



G_2



G_3

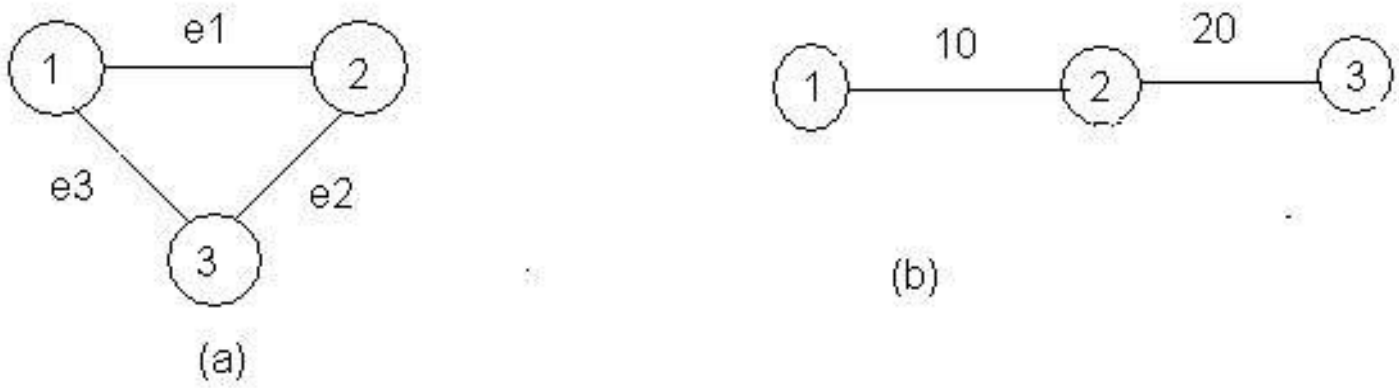
incomplete graph

$V(G_1)=\{0,1,2,3\}$
$V(G_2)=\{0,1,2,3,4,5,6\}$
$V(G_3)=\{0,1,2\}$

$E(G_1)=\{(0,1),(0,2),(0,3),(1,2),(1,3),(2,3)\}$
$E(G_2)=\{(0,1),(0,2),(1,3),(1,4),(2,5),(2,6)\}$
$E(G_3)=\{<0,1>,<1,0>,<1,2>\}$

Graphs and Multigraphs

- **Labeled Graph :** A graph G is said to be labeled if its edges are assigned data.
- **Weighted Graph :** A graph is said to be weighted if each edge e in the graph G is assigned a non-negative numerical value W(e) called the weight or cost of the edge. If an edge does not have any weight then the weight is considered as 1.



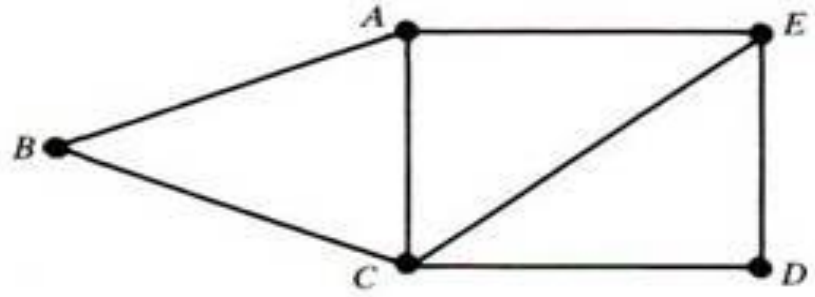
weighted or Labeled Graph

Graphs and Multigraphs

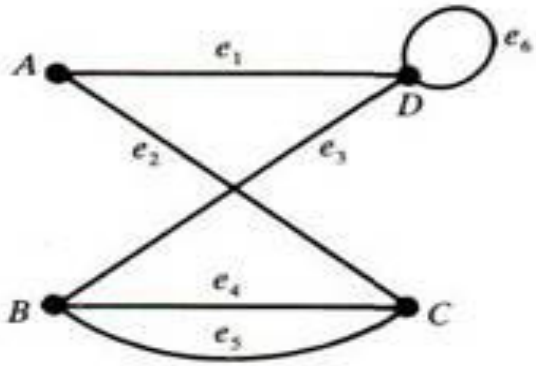
The definition of a graph may be generalized by permitting the following:

- **Multiple edges:** Distinct edges e and e' are called multiple edges if they connect the same endpoints, that is, if e = [u, v] and e' = [u, v].
- **Finite Graph:** A **multigraph** M is said to be finite if it has a finite number of nodes and a finite number of edges.
- **Multigraph :** If a graph has two parallel path to an edge or multiple edges along with a loop is said to be multigraph.

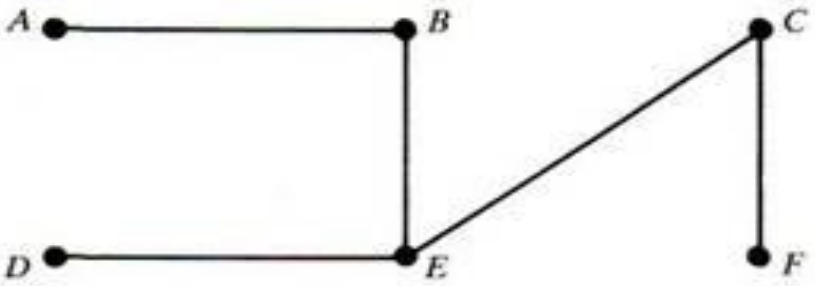
Example 8.1



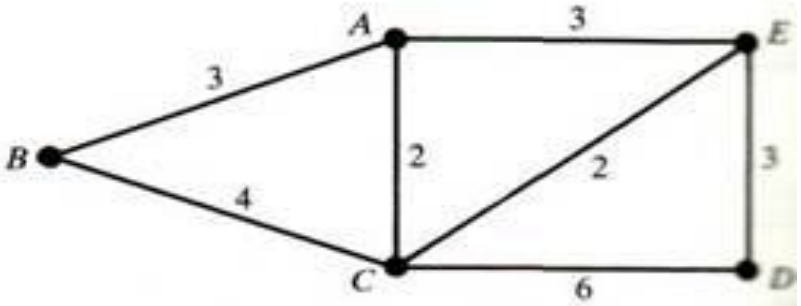
(a) Graph.



(b) Multigraph.

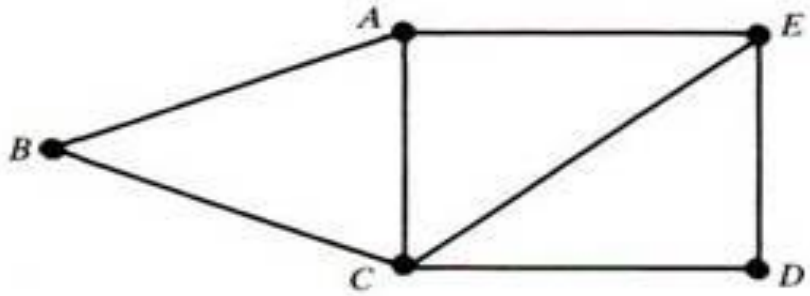


(c) Tree.



(d) Weighted graph.

Example 8.1



(*a*) Graph.

(*a*) Figure 8-1(*a*) is a picture of a connected graph with 5 nodes—A, B, C, D and E—and 7 edges:

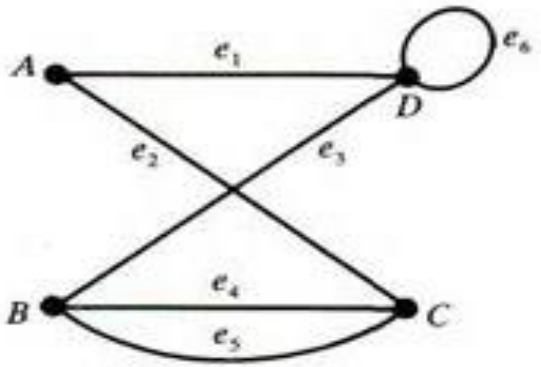
$$[A, B], \quad [B, C], \quad [C, D], \quad [D, E], \quad [A, E], \quad [C, E] \quad [A, C]$$

There are two simple paths of length 2 from B to E: (B, A, E) and (B, C, E). There is only one simple path of length 2 from B to D: (B, C, D). We note that (B, A, D) is not a path, since $[A, D]$ is not an edge. There are two 4-cycles in the graph:

$$[A, B, C, E, A] \quad \text{and} \quad [A, C, D, E, A].$$

Note that $\deg(A) = 3$, since A belongs to 3 edges. Similarly, $\deg(C) = 4$ and $\deg(D) = 2$.

Example 8.1

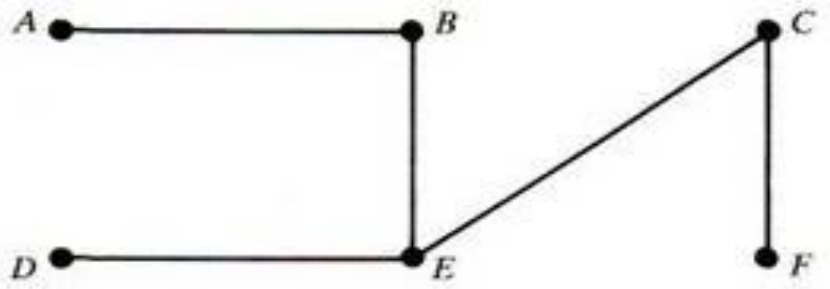


(*b*) Multigraph.

(*b*) Figure 8-1(*b*) is not a graph but a multigraph. The reason is that it has multiple edges—$e_4 = [B, C]$ and $e_5 = [B, C]$—and it has a loop, $e_6 = [D, D]$. The definition of a graph usually does not allow either multiple edges or loops.

(*c*) Figure 8-1(*c*) is a tree graph with $m = 6$ nodes and, consequently, $m - 1 = 5$ edges. The reader can verify that there is a unique simple path between any two nodes of the tree graph.

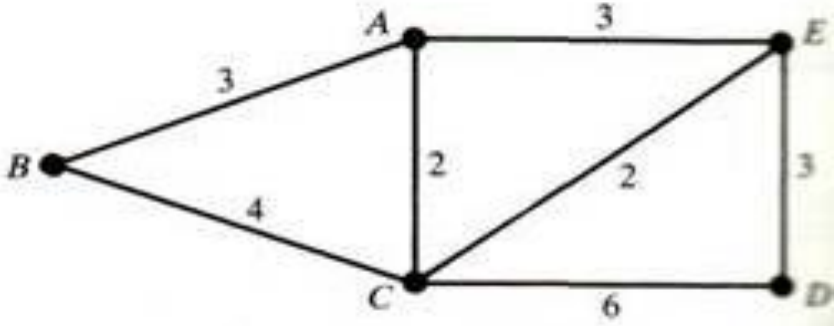
Example 8.1



(c) Tree.

(c) Figure 8-1(c) is a tree graph with $m = 6$ nodes and, consequently, $m - 1 = 5$ edges. The reader can verify that there is a unique simple path between any two nodes of the tree graph.

Example 8.1



(*d*) Weighted graph.

(*d*) Figure 8-1(*d*) is the same graph as in Fig. 8-1(*a*), except that now the graph is weighted. Observe that $P_1 = (B, C, D)$ and $P_2 = (B, A, E, D)$ are both paths from node B to node D. Although P_2 contains more edges than P_1, the weight $w(P_2) = 9$ is less than the weight $w(P_1) = 10$.

Directed Graphs

A graph in which the edges are having direction is called directed graph or digraph, otherwise the graph is called undirected graph.

Directed Graphs

A directed graph G, also called a digraph or graph is the same as a multigraph except that each edge e in G is assigned a direction, or in other words, each edge e is identified with an ordered pair (u, v) of nodes in G.

Suppose G is a directed graph with a directed edge $e = (u, v)$. Then e is also called an *arc*. Moreover, the following terminology is used:

(1) e *begins* at u and *ends* at v.

(2) u is the *origin* or *initial point* of e, and v is the *destination* or *terminal point* of e.

(3) u is a *predecessor* of v, and v is a *successor* or *neighbor* of u.

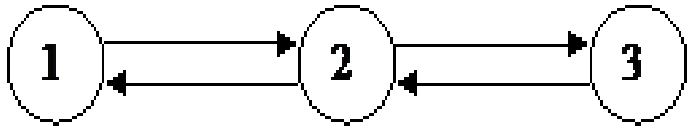
(4) u is *adjacent to* v, and v is *adjacent to* u.

Outdegree and Indegree

Degree/order: A degree of a node is the number of edges containing that node. The number edges pointing towards the node are called in-degree/in-order. The number edges pointing away from the node are called out-degree/out-order.

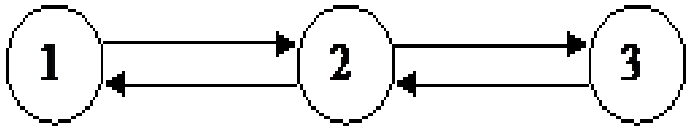
Outdegree and Indegree

Indegree: The indegree of a node u in G, written indeg(u), is the number of edges ending at u.



Indegree of 1 = 1 **Indegree of 2 = 2**

Outdegree: The outdegree of a node u in G, written outdeg(u), is the number of edges beginning at u.



Outdegree of 1 =1 **Outdegree of 2 =2**

Graphs and Multigraphs

- **Source:** A node u is called a source if it has a positive outdegree but zero indegree.
- **Sink:** A node u is called a sink if it has a zero outdegree but a positive indegree

Simple Directed Graph

Simple Directed Graph

A directed graph G is said to be simple if G has no parallel edges. A simple graph G may have loops, but it cannot have more than one loop at a given node.

The notions of *path*, *simple path* and *cycle* carry over from undirected graphs to directed graphs except that now the direction of each edge in a path (cycle) must agree with the direction of the path (cycle). A node v is said to be *reachable* from a node u if there is a (directed) path from u to v.

A directed graph G is said to be *connected*, or *strongly connected*, if for each pair u, v of nodes in G there is a path from u to v and there is also a path from v to u. On the other hand, G is said to be *unilaterally connected* if for any pair u, v of nodes in G there is a path from u to v or a path from v to u.

Example 8.2

Figure 8-2 shows a directed graph G with 4 nodes and 7 (directed) edges. The edges e_2 and e_3 are said to be *parallel*, since each begins at B and ends at A. The edge e_7 is a *loop*, since it begins and ends at the same point, B. The sequence $P_1 = (D, C, B, A)$ is not a path, since (C, B) is not an edge—that is, the direction of the edge $e_5 = (B, C)$ does not agree with the direction of the path P_1. On the other hand, $P_2 = (D, B, A)$ is a path from D to A, since (D, B) and (B, A) are edges. Thus A is reachable from D. There is no path from C to any other node, so G is not strongly connected. However, G is unilaterally connected. Note that indeg$(D) = 1$ and outdeg$(D) = 2$. Node C is a sink, since indeg$(C) = 2$ but outdeg$(C) = 0$. No node in G is a source.

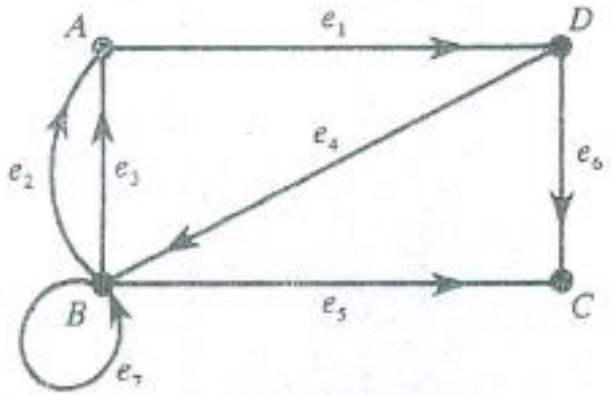


Fig. 8-2

Tree

Let T be any nonempty tree graph. Suppose we choose any node R in T. Then T with this designated node R is called a *rooted tree* and R is called its *root*. Recall that there is a unique simple path from the root R to any other node in T. This defines a direction to the edges in T, so the rooted tree T may be viewed as a directed graph. Furthermore, suppose we also order the successors of each node v in T. Then T is called an *ordered rooted tree*. Ordered rooted trees are nothing more than the general trees discussed in Chap. 7.

simple directed Graph

- *A directed graph G is said to be simple if G has no parallel edges.*
- *A simple graph G may have loops, but it cannot have more than one loop at a given node.*
- *Our study will focus on simple directed Graph edge is a directed edge*

Representation of graph

There are two standard ways of maintaining a graph G in the memory of a computer.

1. The sequential representation
2. The linked representation

Sequential Representation Of Graphs

There are two different sequential representations of a graph. They are

1. Adjacency Matrix representation
2. Path Matrix representation

Adjacency Matrix Representation

- Suppose G is a simple directed graph with m nodes, and suppose the nodes of G have been ordered and are called $v_1, v_2, \ldots, v_m$. Then the adjacency matrix $A = (a_{ij})$ of the graph G is the $m \times m$ matrix defined as follows:

$$a_{ij} = \begin{cases} 1 \text{ if } v_i \text{ is adjacent to } V_j, \text{ that is, if there is an edge } (V_i, V_j) \\ 0 \text{ otherwise} \end{cases}$$

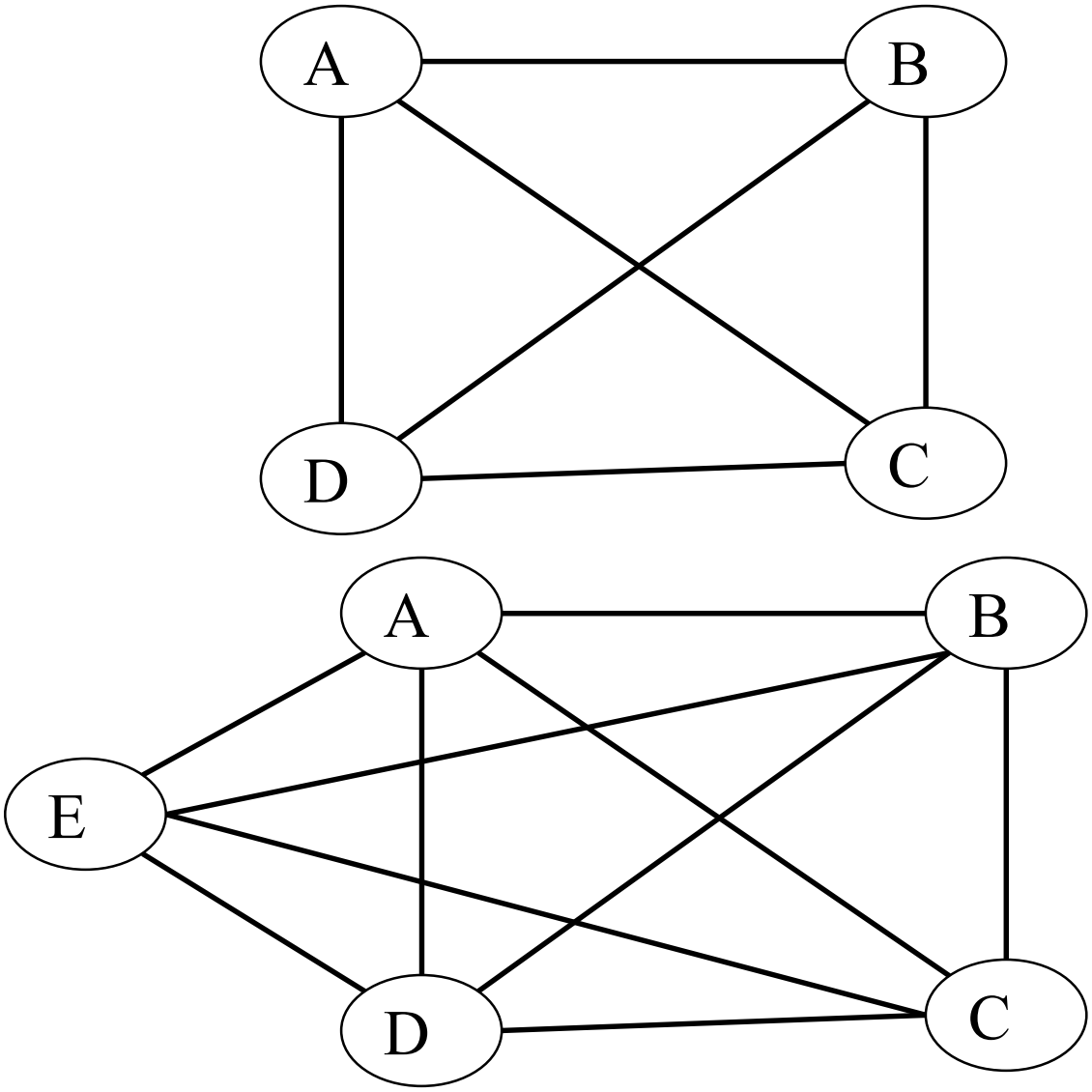
- Suppose G is an undirected graph. Then the adjacency matrix A of G will be a symmetric matrix, i.e., one in which $a_{ij} = a_{ji}$; for every i and j.

Sequential Representation Of Graphs

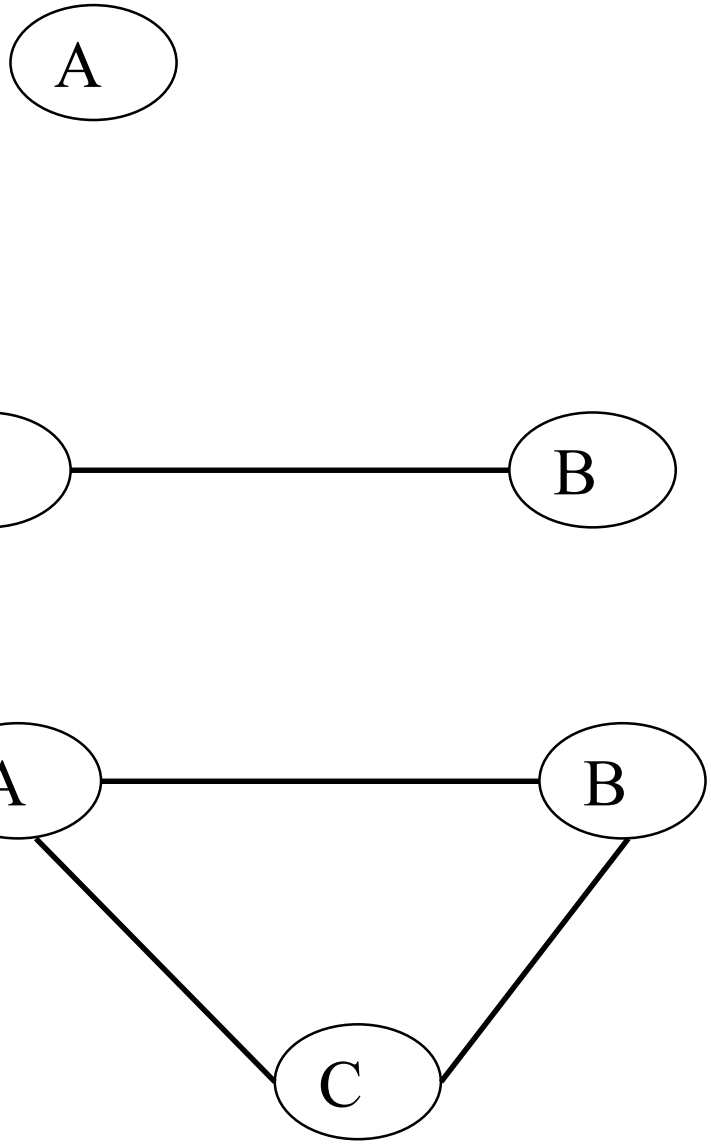
Drawbacks

1. It may be difficult to insert and delete nodes in G.
2. If the number of edges is O(m) or O(m log2 m), then the matrix A will be sparse, hence a great deal of space will be wasted.

Sequential Representation Of Graphs



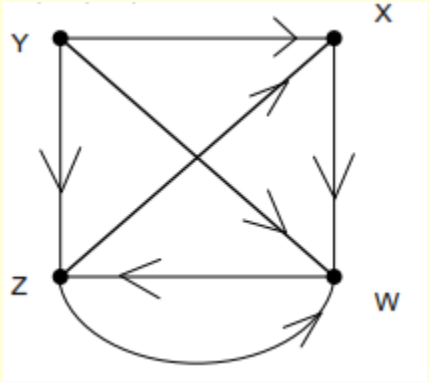
Sequential Representation Of Graphs



Example 8.3

Consider the graph G in following Fig. suppose the nodes are stored in memory in a linear array DATA as follows:

DATA : X , Y , Z , W



We assume that the ordering of the nodes in G is as follows:

V_1=X

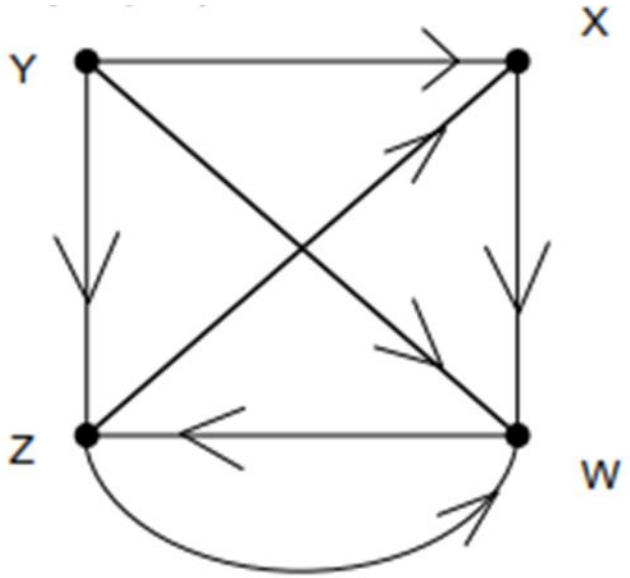
V_2= y

V_3= z

V_4=W

Example 8.3

The adjacency matrix A of G is as follows :



$$A = \begin{array}{c} \\ x \\ y \\ z \\ w \end{array} \begin{array}{c} \begin{array}{cccc} x & y & z & w \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \end{array}$$

Multigraph

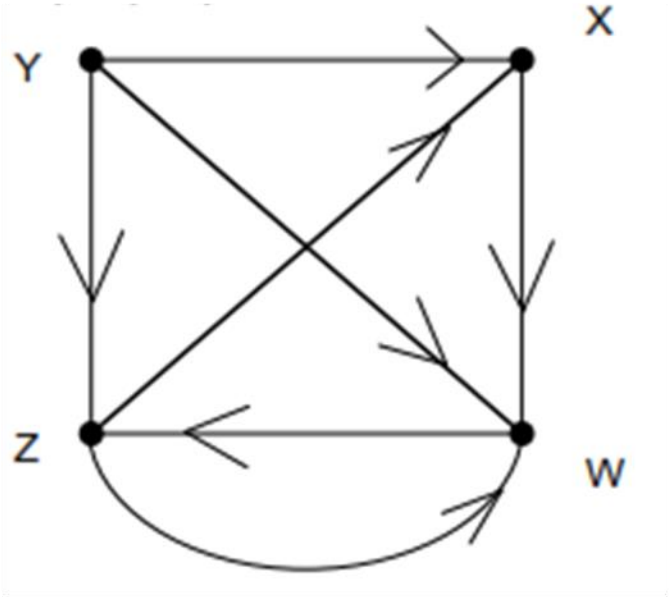
The above matrix representation of a graph may be extended to multigraph.

Specifically , if G is a multigraph then the adjacency matrix of G is the m×m matrix $a=a_{ij}$ defined by setting a_{ij} equal to the number of edges from V_i to V_j

Proposition 8.2: Let A be the adjacency matrix of a graph G. Then $a_k(i, j)$, the ij entry in the matrix A^k, gives the number of paths of length K from v_i to v_j

Example 8.3

Ex. :- Consider the following Fig. & calculate A, A^2, A^3 & A^4 .



$$A = \begin{array}{c} \\ x \\ y \\ z \\ w \end{array} \begin{array}{c} \begin{array}{cccc} x & y & z & w \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \end{array}$$

Example 8.3

$$A^2 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 2 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix} \qquad A^3 = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 1 & 0 & 2 & 2 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix} \qquad A^4 = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 2 & 0 & 2 & 3 \\ 1 & 0 & 1 & 2 \\ 1 & 0 & 1 & 1 \end{pmatrix}$$

Accordingly, in particular, there is a path of length 2 from v_4 to v_1, there are two paths of length 3 from v_2 to v_3, and there are three paths of length 4 from v_2 to v_4. (Here, $v_1 = \text{X}$, $v_2 = \text{Y}$, $v_3 = \text{Z}$ and $v_4 = \text{W}$.)

Suppose we now define the matrix B_r as follows:

$$B_r = A + A^2 + A^3 + \cdots + A^r$$

Then the ij entry of the matrix B_r gives the number of paths of length r or less from node v_i to v_j.

Path Matrix Representation

Let G be a simple directed graph with m nodes, $v_1, v_2, \ldots, v_m$. The path matrix of G is the m-square matrix $P = (p_{ij})$ defined as follows:

$$Pij = \begin{cases} 1 & \text{if there is a path from } V_i \text{ to } V_j \\ 0 & \text{otherwise} \end{cases}$$

Multigraph

The above matrix representation of a graph may be extended to multigraph.

Specifically , if G is a multigraph then the adjacency matrix of G is the m×m matrix $a=a_{ij}$ defined by setting a_{ij} equal to the number of edges from V_i to V_j

Proposition 8.2: Let A be the adjacency matrix of a graph G. Then $a_k(i, j)$, the ij entry in the matrix A^k, gives the number of paths of length K from v_i to v_j

Example

Proposition 8.3: Let A be the adjacency matrix & let P= (p_{ij}) be the path matrix of a digraph G then

p_{ij} = 1 if and only if there is a nonzero number in the ij entry of the matrix .

$$\mathbf{B_m = A + A^2 + A^3 + \ldots\ldots + A^M}$$

Example

Consider the graph G with m= 4 nodes in following Fig. Adding the matrices A, A^2, A^3 and A^4

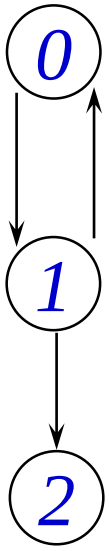
We obtain the following matrix $\mathbf{B_4=A+A^2+A^3+A^4}$

And, replacing the nonzero entries in B_4 by 1 we obtain the path matrix P of the graph G as

$$B_4 = \begin{pmatrix} 1 & 0 & 2 & 3 \\ 5 & 0 & 6 & 8 \\ 3 & 0 & 3 & 5 \\ 2 & 0 & 3 & 3 \end{pmatrix} \quad \text{and} \quad P = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix}$$

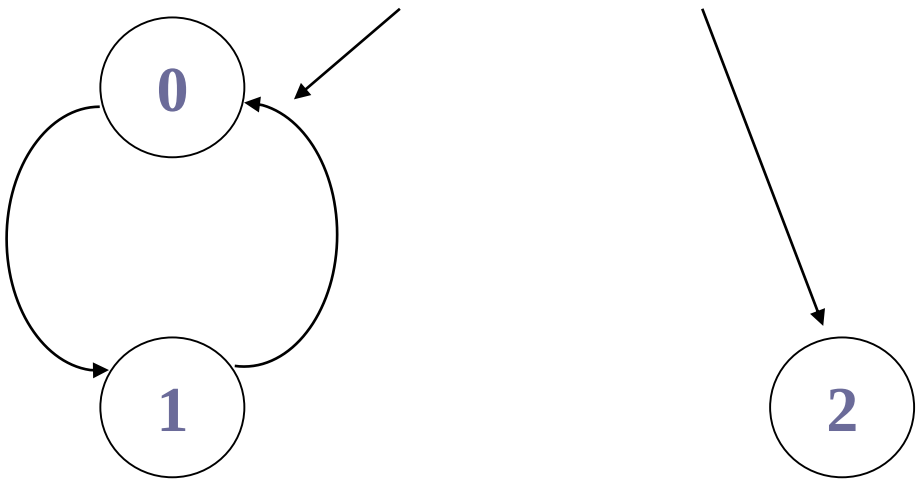
Strongly Connected Graph

- Recall that
  - A directed graph is strongly connected if, for any pair of nodes *u* and *v* *in G,* there are both a path from *u* to *v* and also a path from *v* to *u.*



G_3 *not strongly connected*

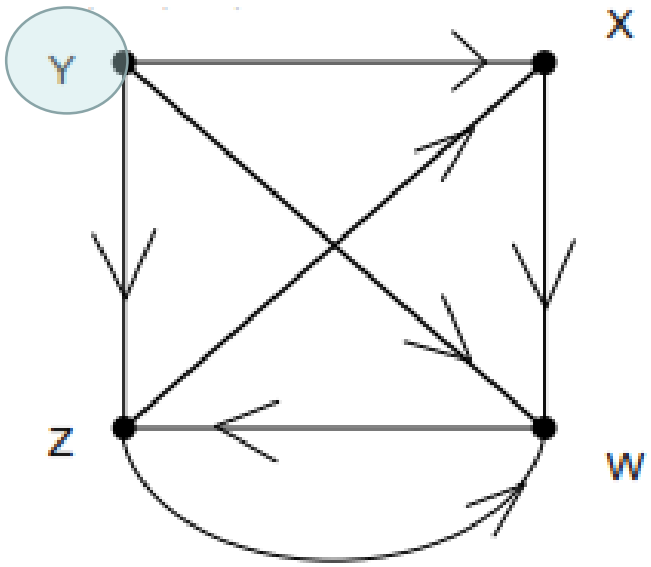
strongly connected component



Strongly Connected Graph

Examining the above path matrix P, we found that the node v_2 is not reachable from any of the other node.

Thus the graph G is not strongly connected graph.



V_1=X
V_2= y
V_3= z
V_4=W

تم الإنتهاء من المحاضرة