Chapter 5 – Networks

# 5.2 Graphs

A ***graph*** is a series of points and lines that can be used to represent the connections that exist in various settings.

 

In a graph, the lines are called ***edges*** or ***arcs*** and the points are called ***vertices*** or ***nodes***, with each edge joining a pair of vertices.

When vertices are joined by an edge, they are known as ***adjacent vertices***. Note that the edges of a graph can intersect without there being a vertex.

A ***simple graph*** is one in which pairs of vertices are connected by one edge at most.



A ***complete graph*** is one where an edge connecting each vertex to all other vertices.



The number of edges of any complete graph can be determined by the use of the formula:

$E=\frac{V(V-1)}{2}$ Where V is the vertices and E is the edges

A ***connected graph*** is one where it is possible to reach every vertex of the graph by moving along the edges.



The graph below is not a connected graph, because not all vertices are connected as one or more vertex/vertices are not connected, sometimes referred as an ***isolated vertex***. This graph is also known as a ***disconnected graph***.



If there are more than one route connecting two vertices, than there are ***multiple edges***.

A route that connect itself to just the one vertex is called a ***loop***.

If it is only possible to move along the edges of a graph in one direction, the graph is called a ***directed graph*** and the edges are represented by arrows. Otherwise it is an ***undirected graph***.

 

**Undirected Graph**

**Directed Graph**

Example 1: The diagram represents a system of paths and gates in a large park. Draw a graph to represent the possible ways of travelling to each gate in the park.





The ***degree of a vertex*** is the number of edges that are directly connected to that vertex.



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Deg (A) = | 2 | Deg (C) = | 5 | Deg (E) = | 3 |
| Deg (B) = | 2 | Deg (D) = | 2 | Deg (F) = | 2 |

Example 2: For the graph in the following diagram, show that the number of edges is equal to the half the sum of the degree of the vertices.



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Deg (A) = |  | Deg (C) = |  | Deg (E) = |  |
| Deg (B) = |  | Deg (D) = |  | Deg (F) = |  |

The sum of the degrees of the vertices =

Total number of edges = \_\_\_\_\_\_\_\_\_\_

When comparing the sum of the degrees of the vertices to the total number of edges, we can see that

***Isomorphic graphs*** have the same number of vertices and edges, with corresponding vertices having identical degrees and connections.





Note, both graphs have an equal number of edges with corresponding vertex having the same number of degree.

Example 3: Confirm whether the following two graphs are isomorphic.



Identify the degree and edges of the vertices for each graph

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Graph** | **A** | **B** | **C** | **D** | **E** |  | **No. of Edges** |
| 1 |  |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |  |

Identify the vertex connections for each graph

|  |  |
| --- | --- |
| **Vertex** | **Connections** |
| A |  |  |  |
| B |  |  |  |
| C |  |  |  |
| D |  |  |  |
| E |  |  |  |

***Adjacency matrix*** is a matrix that represents the number of edges that connect the vertices of a graph. It is a ***square matrix*** with equal number of rows and columns. The adjacency matrix would always be ***symmetrical*** around the ***leading diagonal line***.

**TO**



 

**FROM**

Any non-zero value in the leading diagonal indicates the existence of a loop.



 

A row/column consisting of all zeros indicates an isolated vertex (a vertex that is not connected to any other vertex).





 

Example 4: Construct the adjacency matrix for the given graph.



|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | TO |  |  |  |  |  |  |  |  |
|  |  | A | B | C | D | E | F |  |  |  |  |  |  |  |  |
|  | A |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| F | B |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| R | C |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| O | D |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| M | E |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | F |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

# 5.3 planar graphs

***Planar graph*** is a graph that can be redrawn with no crossing edges.



Example 5: Redraw the graph so that it has no intersecting/crossing edges.



## Euler’s Formula

In all planar graphs, the edges and vertices create distinct areas referred to as ***faces*** or ***regions***.



The planar graph shown in the diagram at above has five faces including the area around the outside.

Consider the following group of planar graphs.



The number of vertices, edges and faces for each graph is summarised in the following table.

|  |  |  |  |
| --- | --- | --- | --- |
| **Graph** | **Vertices** | **Edges** | **Faces** |
| 1 |  |  |  |
| 2 |  |  |  |
| 3 |  |  |  |

For each of these graphs, we can obtain a result that is well known for any planar graph. The difference between the vertices and edges added to the number of faces will always equal 2.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Graph | V | − | E | + | F | = | 2 |
| 1 |  | − |  | + |  | = |  |
| 2 |  | − |  | + |  | = |  |
| 3 |  | − |  | + |  | = |  |

The ***Euler’s formula*** for any connected planar graphs is therefore: **V – E + F = 2**

This formula can be rearranged in the following ways:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| V = E – F + 2 | or | E = V + F − 2 | or | F = E – V + 2 |

Example 6: How many faces will there be for a connected planar graph of 7 vertices and 10 edges?

# 5.4 Connected Graphs

## Traversing connected graphs

Traversing refers to the movement across the network with a starting and finishing vertices.

The definitions of the main terms used when describing movement across a network are as follows:

***Walk***: Any route taken through a network, including routes that repeat edges and vertices



***Trail***: A walk in which no edges are repeated



***Path***: A walk in which no vertices are repeated, except possibly the start and finish



***Cycle***: A path beginning and ending at the same vertex



***Circuit***: A trail beginning and ending at the same vertex.



Example 7: In the following network, identify two different routes: one cycle and one circuit.



For a cycle, identify a route that doesn’t repeat a vertex apart from the start/finish.

Cycle:

For a circuit, identify a route that doesn’t repeat an edge and ends at the same vertex.

Circuit:

## Euler trails and circuits

In some practical situations, it is most efficient if a route travels along each edge only once. Examples include parcel deliveries and council garbage collections. If it is possible to travel a network using each edge only once, the route is known as an ***Euler trail*** or ***Euler circuit***.

An ***Euler trail*** is a trail in which every edge is used once.



An ***Euler circuit*** is a circuit in which every edge is used once.



If the degree of all the vertices of a connected graph are even, then an ***Euler circuit*** exists.

If there are exactly 2 vertices of odd degree of a connected graph, then an ***Euler trail*** exists.

## Hamiltonian paths and cycles

A ***Hamiltonian path*** is a path that reaches all vertices of a network without necessarily using all of the available edges.



A ***Hamiltonian cycle*** is a cycle that reaches all vertices of a network without necessarily using all of the available edges.



Example 8: Identify an Euler trail and a Hamiltonian path in the following graph.



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Deg (A) = |  | Deg (C) = |  | Deg (E) = |  | Deg (G) = |  |
| Deg (B) = |  | Deg (D) = |  | Deg (F) = |  | Deg (H) = |  |

Euler trial involves identifying a route that uses each edge once.

Euler trail:

Hamiltonian path involves identifying a route that uses each vertex once.

Hamiltonian path:

## Terminologies Summary

***Walk*** – a route through a network.

***Trail*** – walk through the network with no edges repeat.

A trail will form when exactly 2 vertices with odd degree.

***Eulerian*** (**E**dge)

***Circuit*** – walk that starts and Finishes at the same vertex with no edges repeat.

A circuit will form when all vertices have even degree.

***Path*** – walk through the network with no vertices repeat.

***Halmiltonian*** (Vertex)

***Cycle*** – walk that Starts and Finishes at the same vertices with no vertices repeat.

# 5.5 Weighted graphs and trees

## Weighted graphs

In many applications using graphs, it is useful to attach a value to the edges. These values could represent the length of the edge in terms of time or distance, or the costs involved with moving along that section of the path. Such graphs are known as ***weighted graphs***.



Weighted graphs can be particularly useful as analysis tools. For example, they can help determine how to travel through a network in the shortest possible time.

Example 9: The graph represents the distances in kilometres between eight locations. Identify the shortest distance to travel from A to D that goes to all vertices.



|  |  |
| --- | --- |
| Possible paths | Distance |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Therefore the shortest distance from A to D that travels to all vertices is

## Trees

A ***tree*** is a simple connected graph with no circuits, no loops or multiple edges. A tree is therefore must contain only one face. The number of edges in a tree is always 1 less than the number of vertices.



***Spanning trees*** are sub-graphs (graphs that are formed from part of a larger graph) that include all of the vertices of the original graph. In practical settings, they can be very useful in analysing network connections. For example a ***minimum spanning tree*** for a weighted graph can identify the lowest-cost connections. Spanning trees can be obtained by systematically removing any edges that form a circuit, one at a time.



## Prim’s algorithm

Prim’s algorithm is a set of logical steps that can be used to identify the minimum spanning tree for a weighted connected graph.

Steps for Prim’s algorithm:

Step 1: Begin at a vertex with low weighted edges.

Step 2: Progressively select edges with the lowest weighting, unless they form a circuit.

Step 3: Continue until all vertices are connected.

Example 10: Use Prim’s algorithm to identify the minimum spanning tree of the graph shown.

 