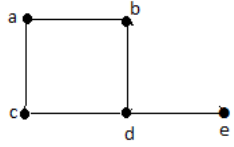
**Lecture 9**

**Graph Theory**

**What is a Graph?**

A graph is a pictorial representation of a set of objects where some pairs of objects are connected by links. The interconnected objects are represented by points termed as vertices, and the links that connect the vertices are called edges.

Formally, a graph is a pair of sets **(V, E)**, where V is the set of vertices and **E** is the set of edges, connecting the pairs of vertices. Take a look at the following graph:



In the above graph,

V = {a, b, c, d, e}

E = {ab, ac, bd, cd, de}

**Applications of Graph Theory**

Graph theory has its applications in diverse fields of engineering:

**• Electrical Engineering** – The concepts of graph theory is used extensively in designing circuit connections. The types or organization of connections are named as topologies. Some examples for topologies are star, bridge, series, and parallel topologies.

**• Computer Science** – Graph theory is used for the study of algorithms. For example,

o Kruskal's Algorithm

o Prim's Algorithm

o Dijkstra's Algorithm

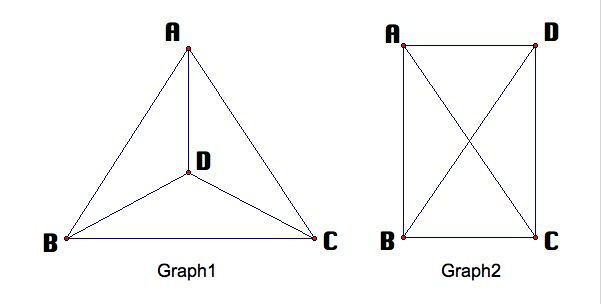
• Computer Network – The relationships among interconnected computers in the network follows the principles of graph theory.

• Science – The molecular structure and chemical structure of a substance, the DNA structure of an organism, etc., are represented by graphs.

• Linguistics – The parsing tree of a language and grammar of a language uses graphs.

• General – Routes between the cities can be represented using graphs. Depicting hierarchical ordered information such as family tree can be used as a special type of graph called tree.

Graph theory concerns the relationship among lines and points. It consists of some points and some lines between them. No attention is paid to the position of points and the length of the lines. Thus, the two graphs below are the same graph.



Graph is formed by vertices and edges connecting the vertices.

Mathematically, graph is a tuple ***G = (V, E)*** where ***V*** is a (ﬁnite) set of vertices and ***E*** is a ﬁnite collection of edges. The set ***E*** contains elements from the union of the one and two element subsets of ***V***. That is, each edge is either a one or two element subset of ***V***.

Therefore we can say, a graph ***G*** consists of two things:

1. A *set* ***V = V(G)*** whose elements are called vertices, points, or nodes of ***G***.

2. A set ***E = E(G)*** of unordered pairs of distinct vertices called edges of ***G***.

Often, we label the vertices with letters (for example: ***a, b, c, ...* or *v1 ,v2, ...)*** or numbers ***1, 2, ,...***

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Graph 1 | Graph 2 | Graph 3 |

**Fig. 1**

We have here

***V = {v1,...,v5}***for the vertices and

***E******= {(v1,v2), (v2,v5), (v5,v5), (v5,v4), (v5,v4)}***for the edges.

Labeling the edges in graph 3 as like

***E = {e1,..., e5}.***

So we can say,***E*** is a multi-set, in other words, its elements can occur more than once so that every element has a multiplicity.

The two edges ***(u, v)*** and ***(v, u)*** are the same. In other words, the pair is not ordered.

**Example 1:**

In Fig. 2(a):

(i) The set of vertices V ***= {1, 2, 3, 4}***.

(ii) The set of edges ***E = {{1, 2}, {2, 3}, {3, 4}, {4, 1}}*** , then

(ii) The graph ***G = (V, E)*** has four vertices and four edges.

In Fig. 2(b):

(i) ***V*** consists of vertices *A, B, C, D.*

(ii) ***E*** consists of edges *e1 = {A, B}, e2 = {B, C}, e3 = {C, D}, e4 = {A, C}, e5 = {B, D}.*

In the Fig. (c):

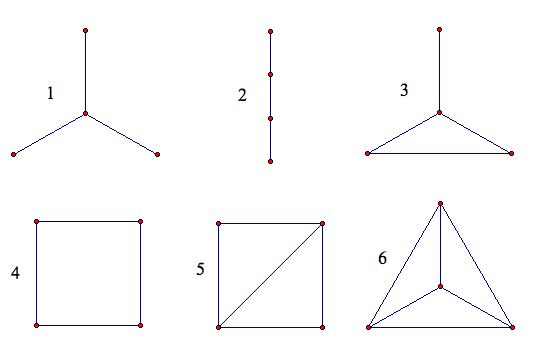
(i) The edges e4 and e5 are called multiple edges since they connect the same endpoints, and

(ii) The edges e6 is called a loop since its endpoints are the same vertex.

Such a diagram is called a **Multigraph.**

|  |  |  |
| --- | --- | --- |
| (a) | (b) | (c) M*ultigraph* |
| Fig. 2 | | |
| (Vertices may Circles, points/dots, nodes) | | |

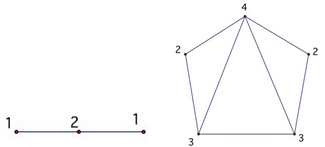
**Problem -1(a).** Find all possible graphs with 4 vertices.

****

**Degree**

The degree of a vertex ***v*** of a graph is the number of edges incident with ***v***.

The degree of each vertex in this graph below is represented by the number.



Let ***G = (V, E)*** be a graph and let ***v ∈ V***.

The degree of vertex ***v*** in a graph ***G***, written ***deg(v)***, is equal to the number of edges in ***G*** which contain ***v***, (number of non-self-loop edges adjacent to *v* plus two times the number of self-loops) that is, which are incident on ***v***. That is each edge is counted twice in counting the degrees of the vertices of ***G***, we have the following simple by important result.

The sum of the degrees of the vertices of a graph ***G*** is equal to twice the number of edges in ***G***.

Therefore in the **Fig. 2(c),** we have ***deg(A)*** = 2, ***deg(B)*** = 3, ***deg(C)*** = 3, ***deg(D)*** = 2.

The sum of the degrees equals 10 which, as expected, is twice the number of edges.

A vertex is said to be ***even*** or ***odd*** according as its degree is an ***even*** or ***odd*** number. Thus ***A*** and ***D*** are even vertices whereas ***B*** and ***C*** are odd vertices.

A vertex of degree zero is called an **isolated** vertex.

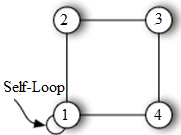
In multigraph, loop is counted twice toward the degree of its endpoint. For example, in **Fig. 2(c),** we have ***deg(D) = 4***, since the edge ***e6*** is counted twice, hence ***D*** is an even vertex.

Mathematically: deg(***v) = |{e ∈ E : ∃u ∈ V (e = {u, v})}|+ 2|{e ∈ E : e = {v}}|***

Here if ***S*** is a set, then ***|S|*** is the cardinality of that set.

Note that each vertex in the graph in **Fig. (a)** has degree 2.

If we replace the edge set in example 1 with: ***E = {{1, 2}, {2, 3}, {3, 4}, {4, 1}, { 1}}*** then the visual representation of the graph includes a loop that starts and ends at Vertex 1. This is illustrated in **Fig. 3.**



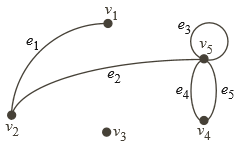
**Fig. 3**

In this example the degree of Vertex 1 is now 4.

We obtain this by counting the number of non self-loop edges adjacent to Vertex 1 (there are 2) and adding two times the number of self-loops at Vertex 1 (there is 1) to obtain 2 + 2×1 = 4.

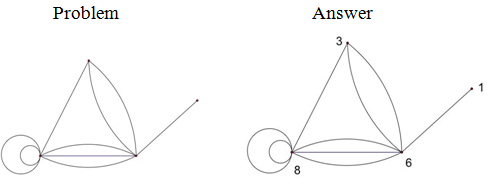
The minimum degree of the vertices in a graph ***G*** is denoted ***δ(G)*** (= 0 if there is an isolated vertex in ***G***). Similarly, we write ***∆(G)*** as the maximum degree of vertices in ***G***.

In the figure below ***δ(G)*** = 0 and ∆(G) = ***5***.



**Problem 2**

Find the degree of each vertex of the graph below:

****

|  |  |
| --- | --- |
| **Problem 3 (Ex. 8.1p211)** Consider the graph:  (a) Describe formally the graph *G* in the diagram, that is, find the set  *V(G)* of vertices of *G* and the set *E(G)* of edges of *G*.  (b) Find the degree of each vertex and verify its result. |  |

**Ans.**

(a) There are five vertices so *V(G) = {A, B, C, D, E}.*

There are seven pairs of vertices:

*E(G) =* *{{A, B},* *{A,C},* *{A, D},* *{ B, C},* *{ B, E},* *{C, D},* *{C, E} .*

(b) The degree of a vertex is equal to the number of edges to which it belongs:

e. g. *deg(A) = 3* since A belongs to the three edges *{A, B}, {A,C}, {A, D}.*

Similarly,

*deg(B) = 3* since B belongs to the three edges *{ A, B}, {B,E}, {B, C}.*

*deg(C) = 4*since C belongs to the four edges *{A, C}, {C, B}, {C, E}, {C, D}.*

*deg(D) = 2*since D belongs to the four edges *{A, D}, {C, D}.*

and

*deg(E) = 2*since E belongs to the four edges  *{B, E}, {C, E.*

**Verifying**

Sum of the degrees of the vertices is: 3+3+4+2+2 = 14,

which does equal twice the number of edges.

**Problem 3:**

Consider, the following graph G

|  |  |
| --- | --- |
| http://www.personal.kent.edu/~rmuhamma/GraphTheory/MyGraphTheory/Diagrams/g8.gif | The graph *G* has  (i)*. deg*(*u*) = 2,  (ii)*. deg*(*v*) = 3,  (iii). *deg*(w) = 4 and  (iv). *deg*(*z*) = 1.  ∴ Sum of degree of vertices : 2+3+4+1= 10  (equal to twice of edges = 5x2= 10) |

**Loop and Multiple Edges**

|  |  |
| --- | --- |
| A loop is an edge whose endpoints are equal i.e., an edge joining a vertex to it self is called a loop. We say that the graph has multiple edges if in the graph two or more edges joining the same pair of vertices. | http://www.personal.kent.edu/~rmuhamma/GraphTheory/MyGraphTheory/Diagrams/g1.gif |

**Vertex Adjacency**

Let ***G = (V,E)*** be a graph. Two vertices ***v1*** and ***v2*** are said to be adjacent if there exists an edge ***e ∈ E*** so that ***e = {v1,v2}****.* A vertex ***v*** is self-adjacent if ***e = {v}*** is an element of ***E*.**

**Edge Adjacency**

Let ***G = (V,E)*** be a graph. Two edges ***e1***and ***e2*** are said to be adjacent if there exists a vertex ***v*** so that ***v*** is an element of both ***e1*** and ***e2*** (as sets). An edge ***e*** is said to be adjacent to a vertex ***v*** if ***v*** is an element of ***e*** as a set.

**Incident**

Two vertices, ***u*** and ***v***, of a graph are **ADJACENT**if there is an edge, vx, joining them; the vertices are then considered **INCIDENT** to the edge, vx.

**Neighborhood**

Let ***G = (V, E)*** be a graph and ***v ∈ V*** .

The neighbors of ***v*** are the set of vertices that are adjacent to ***v***.

Formally, ***N(v) = {u ∈ V : ∃e ∈ E (e = {u, v}*** or ***u = v*** and ***e = {v})}***

In some texts, ***N(v)*** is called the open neighborhood of v while N***[v] = N(v)∪{v}*** is called the closed neighborhood of ***v***.

When ***v*** is an element of more than one graph, we write ***NG(v)*** as the neighborhood of ***v*** in graph ***G***.

***N(v)*** is the set of vertices u in (the set) ***V*** such that there exists an edge e in (the set) ***E*** so that ***e = {u,v}*** or ***u = v*** and ***e = {v}.***

The logical expression ***∃x(R(x))*** is always read in this way; that is, there exists ***x*** so that some statement ***R(x)*** holds.

Similarly, the logical expression ***∀y (R(y))*** is read: For all y the statement ***R(y)*** holds.

In figure 2(a), the neighborhood of Vertex **1** is Vertices **2** and **4** and Vertex **1** is adjacent to these vertices.

**Finite/Infinite Graph**

A **finite graph** is a [graph](https://proofwiki.org/wiki/Definition:Graph_(Graph_Theory)) with a [finite](https://proofwiki.org/wiki/Definition:Finite) number of [edges](https://proofwiki.org/wiki/Definition:Edge_(Graph_Theory)) and a [finite](https://proofwiki.org/wiki/Definition:Finite) number of [vertices](https://proofwiki.org/wiki/Definition:Vertex_(Graph_Theory)). So we may say a multigraph is said to be finite if it has a finite number of vertices and a finite number of edges.

Note that in a [simple graph](https://proofwiki.org/wiki/Definition:Simple_Graph), a [finite](https://proofwiki.org/wiki/Definition:Finite) number of [edges](https://proofwiki.org/wiki/Definition:Edge_(Graph_Theory)) follows directly from a [finite](https://proofwiki.org/wiki/Definition:Finite) number of [vertices](https://proofwiki.org/wiki/Definition:Vertex_(Graph_Theory)).

In the case of a [multigraph](https://proofwiki.org/wiki/Definition:Multigraph) this may not apply, as there may be an [infinite](https://proofwiki.org/wiki/Definition:Infinite) number of [edges](https://proofwiki.org/wiki/Definition:Edge_(Graph_Theory)) between two given [vertices](https://proofwiki.org/wiki/Definition:Vertex_(Graph_Theory)).

A graph which has either an [infinite number](https://proofwiki.org/wiki/Definition:Infinite) of [edges](https://proofwiki.org/wiki/Definition:Edge_(Graph_Theory)) or [vertices](https://proofwiki.org/wiki/Definition:Vertex_(Graph_Theory)) is an **infinite graph**

**Trivial Graph**

The finite graph with one vertex and no edges, i.e., a single point, is called the trivial graph.

**Walk, Path and Trail**

**Walk**

A walk is a graph G is an alternating sequence of n+1 vetices and n edges of the form :

v1, e1, v2, e2, v3, … .. … , en-1, vn, en, vn-1

in which v1and vn+1 are end points of the edges ei for i= 1, 2, ……, n.

**Path**

A walk in which all vertices are distinct is called a path.

A path in a multigraph G consists of an alternating sequence of vertices of the from *v0, e1, v1, e2, v2, … …. .. , en-1, vv-1, en, vn*.

Where each edge e1 contains the vertices *vi-1*and *vi* (which appear on the sides of *ei*n the sequence). The number *n* of edges is called the length of the path. When there is o ambiguity, we denote a path by its sequence of vertices (*v0, v1, … , vn*).

**Closed Path**

The path said to be closed if *v0 = vn*. Otherwise, we say the path is from *v0* to *vn* or between

*v0* and *vn* or connects *v0* to *vn.*

**Simple Path**

A simple path is a path in which all vertices are distinct.

**Trail**

A trail is path in which all edges are distinct.

**Cycle**

A cycle is a closed path in which all vertices are distinct except *v0 = vn*. A cycle of length k is called a k-cycle. In a graph, any cycle must have length 3 or more.

For let *v1*, *v2 v1, v1* , …….., *vn*  are vertices and making graphs are called cycles.

*K3, ….., K7* are cycles (denoted by *Cn*).

|  |  |
| --- | --- |
| https://upload.wikimedia.org/wikipedia/commons/thumb/e/e7/Graph_cycle.gif/220px-Graph_cycle.gif | A graph with edges colored to illustrate **path** H-A-B (green),  **Closed p1ath or** **walk** with a repeated vertex B-D-E-F-D-C-B (blue), and  A **cycle** with no repeated edge or vertex H-D-G-H (red). |

|  |  |
| --- | --- |
| Consider the following sequences:  (i). A = (P4, P1, P2, P5, P1, P2, P3, P6)  (ii). B = (P4, P1, P5, P2, P6)  (iii). C = (P4, P1, P5, P2, P3, P5, P6)  (iv). D = (P4, P1, P5, P3, P6) |  |

(i). The sequence A is path from P4 to P6 ; but it is not a trail since the edge {P1, P2}is used twice.

(ii). The sequence B is not a path since there is no edge {P2, P6}.

(iii). The sequence C is a trail since no edge is used twice; but it is not a single path since he vertex P5, is used twice.

(iv). The sequence D is simple path from P4 to P6 ; but it is not the shortest path (with respect to length) from P4 to P6. The shortest path from P4 to P6 is the simple path (P4, P5, P6) which has length 2.

**Another Example**

|  |  |
| --- | --- |
| If all the edges (but no necessarily all the vertices) of a walk are different/distinct, then  The walk is called a **Trail**.  **Trail:** The walk ***vzzywxy***is a trail since the vertices ***y*** and ***z*** both **occur twice**.  If, in addition, all the vertices are difficult, then  The trail is called **Path**.  **Path:** The walk ***vwxyz***is a path since the walk has **no repeated vertices**. | **http://www.personal.kent.edu/~rmuhamma/GraphTheory/MyGraphTheory/Diagrams/g15.gif** |

|  |  |
| --- | --- |
| A walk of length k in a graph *G* is a succession of *k* edges of G of the form ***uv*, *vw*, *wx*, . . . , *yz*.**   We denote this walk by ***uvwxyz*** and refer to it as a walk between ***u*** and ***z*.** | **http://www.personal.kent.edu/~rmuhamma/GraphTheory/MyGraphTheory/Diagrams/g14.gif** |

**Connectivity, Connected Components**

A graph G is connected if there is a path between any two of its vertices. The graph 4(a) is connected =, but the graph 4(b) is not connected since there is no path between vertices D and E.

Suppose *G* is a graph.

A connected subgraph *H* of *G* is called a connected component of *G* if H is not contained in any larger connected subgraph. It is intuitively clear that any graph *G* can be partitioned into its connected components. For example, the graph *G* in (b) has three connected components, the subgraphs induced by the vertex sets *{A, C, D}*, *{E, F}* and *{B}.*

|  |  |  |
| --- | --- | --- |
| 4(a) |  | 4(b) |

The vertex *B* in (b) is called an isolated vertex since *B* does not belong to any edge or, in other words, *deg(B) = 0*. Therefore, as noted, *B* itself forms a connected component of the graph.

**Distance and Diameter**

Consider a connected graph *G*.

**Distance**

The distance between vertices *u* and *v* in *G*, written *d(u, v)*, is the length of the shortest path between *u* and *v*.

For example, Fig.5(a), *d(A, F)* = 2 and in Fig. 5(b), *d(A, F)* = 3.

**Diameter**

The diameter of *G*, written *diam(G)*, is the maximum distance between any two points in *G*.

For example, in Fig.5(a), *diam(G)* = 3, whereas in Fig.5(b), *diam(G)* = 4.

|  |  |
| --- | --- |
| 5(a) | 5(b) |

**Cutpoints and Bridge**

Let G be a connected graph.

A vertex *v* in *G* is called a cutpoint if *G – v* is disconnected.

(we know that *G – v* is the graph obtained from *G* by deleting *v* and all edges containing *v*).

An edge e of *G* is called a bridge if *G – e* is disconnected.

(we know that *G – v* is the graph obtained from *G* by simply deleting the edge *e*).

In Fig.5(a), the vertex D is a cutpoint and there are no bridges.

In Fig.5(b), the edge e = {D, F} is a bridge. (Its endpoints D and F are necessarily cutpoints).

**Null Graphs**

A null graphs is a graph containing no edges. The null graph with n vertices is denoted by N*n*.

The following are the examples of null graphs.



Note that N*n*is regular of degree 0.

**Regular Graph**

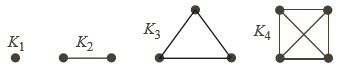
A graph is regular if all the vertices of *G* have the same degree. In particular, if the degree of each vertex is *r*, the *G* is regular of degree *r*.

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  | |  |

**Complete Graph**

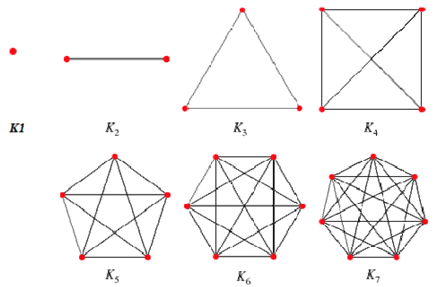
A simple graph that contains every possible edge between all the vertices is called a complete graph. A complete graph with *n* vertices is denoted as *Kn*.

The ﬁrst four complete graphs are:



The simple graph that contains exactly one edge between each pair of distinct vertices is called a complete graph. It is denoted by *Kn*, where *n* is the number of vertices.

For n = 1, 2, 3 4,5,6, 7 the complete graphs are:



If a graph has n vertices and every pair of vertices arises as the endpoints of some edge, then the graph is said to be the complete graph on the set of vertices.

A complete graph on two vertices is just a closed line segment,

A complete graph on three vertices is just the triangle with the given set of vertices;

it is a fairly straightforward exercise to prove that if X and Y are finite sets of vertices with the same number of points, then the complete graphs on X and Y are homeomorphic spaces.

**Some Basic terminologies:**

1. The two vertices u and v are end vertices of the edge ***(u,v).***

2. Edges that have the same end vertices are parallel.

3. An edge of the form ***(v,v)*** is a loop.

4. A graph is simple if it has no parallel edges or loops.

5. A graph with no edges (i.e. ***E*** is empty) is empty.

6. A graph with no vertices (i.e. ***V*** and ***E*** are empty) is a null graph.

7. A graph with only one vertex is trivial.

8. Edges are adjacent if they share a common end vertex.

9. Two vertices ***u*** and ***v*** are adjacent if they are connected by an edge, in other words, ***(u,v)*** is an edge.

10. The degree of the vertex ***v***, written as ***deg(v),*** is the number of edges with ***v*** as an end vertex. By convention, we count a loop twice and parallel edges contribute separately.

11. A pendant vertex is a vertex whose degree is **1**.

12. An edge that has a pendant vertex as an end vertex is a pendant edge.

13. An isolated vertex is a vertex whose degree is **0.**

**Notes**

|  |  |  |
| --- | --- | --- |
| • ***v1*** and ***v2*** are adjacent.  • The degree of ***v1*** is **1** so it is a pendant vertex.  • ***e1*** is a pendant edge.  • The degree of ***v5*** is 5.  • The degree of ***v4*** is 2.  • The degree of ***v3*** is **0** so it is an isolated vertex. |  | • ***v4*** and ***v5*** are end vertices of ***e5***.  • ***e4*** and ***e5*** are parallel.  • ***e3*** is a loop.  • The graph is not simple.  • ***e1*** and ***e2*** are adjacent. |

**Exercise**

|  |  |
| --- | --- |
| **Exercise-1:** Give an example of a graph such that every vertex is adjacent to two vertices and every edge is adjacent to two edges. |  |
| **Exercise-2:** Give an example of a graph such that every vertex is adjacent to three vertices and every edge is adjacent to three edges. |  |
| **Exercise-3:** Give an example of a graph such that two vertices/edges are adjacent to three vertices/edges and rest are adjacent to two vertices/edges. |  |
| **Exercise-4:** Give an example of five vertices v1,v2, v2,v3, v4, v5 such that deg v1=3, v2 is an odd vertex, v3=2 and v4 and v5 are adjacent. |  |
| **Exercise-5:** Draw a graph with five vertices and as many edges as possible.  How many edges contain in the graph?  What is the name of this graph and how is it denoted?  **Ans.** The graph contains 5x2 = 10 edges  The name of the graph is complete graph.  It is denoted by K5. |  |