GRAPH THEORY

Question: What is a graph???

- How can we lay cable at minimum cost to make every telephone reachable from every other?
- What is the fastest route from the national capital to each state capital?
- How can n jobs be filled by n people with maximum total utility?
- What is the maximum flow per unit time from source to sink in a network of pipes?
- How many layers does a computer chip need so that wires in the same layer don't cross?
- How can the season of a sports league be scheduled into the minimum number of weeks?
- In what order should a traveling salesman visit cities to minimize travel time?
- Can we color the regions of every map using four colors so that neighboring regions receive different colors?

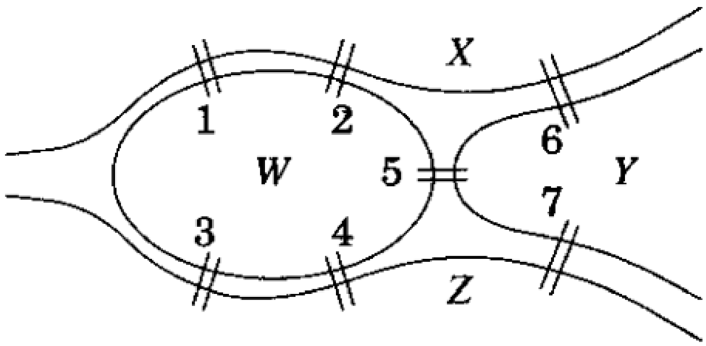
These and many other practical problems involve graph theory. In this course, we develop the theory of graphs and apply it to such problems.

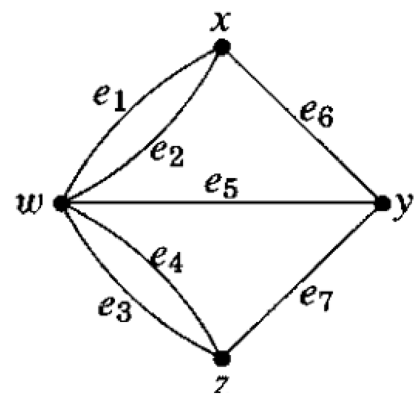
Origin of Graph Theory:

The problem that is often said to have been the birth of graph theory is **"The Konigsberg Bridge Problem".**

Problem:

The city of Konigsberg was located on the Pregel river in Prussia. The city occupied two islands plus areas on both banks. These regions were linked by seven bridges as shown on the left below. The citizens wondered whether they could leave home, cross every bridge exactly once, and return home. The problem reduces to traversing the figure on the right, with heavy dots representing land masses and curves representing bridges.





The model on the right makes it easy to argue that the desired traversal does not exist. Each time we enter and leave a land mass, we use two bridges ending at it. We can also pair the first bridge with the last bridge on the land mass where we begin and end. Thus existence of the desired traversal requires that each land mass be involved in an even number of bridges. This necessary condition did not hold in Königsberg. ∎

Introduction to Graph Models and Terminology:

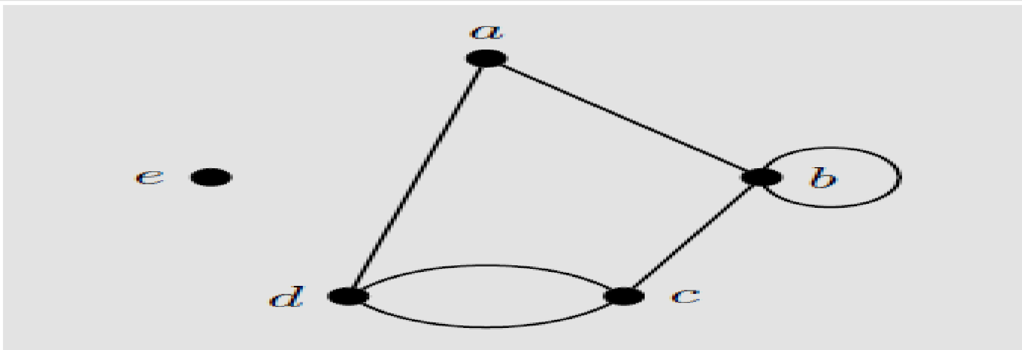
- Note that many aspects of graph theory rely on basic set theory concepts (mainly the subset relationship).

Graphs:

A graph G is a triple consisting of a vertex set V(G), an edge set E(G), and a relation that associates with each edge two vertices (not necessarily distinct) called its endpoints.

Definition 1.1 A *graph* G consists of two sets: $V(G)$, called the vertex set, and $E(G)$, called the edge set. An *edge*, denoted xy, is an unordered pair of vertices. We will often use G or $G = (V, E)$ as short-hand.

Example 1.1 Let G_4 be a graph where $V(G_4) = \{a, b, c, d, e\}$ and $E(G_4) = \{ab, cd, cd, bb, ad, bc\}$. Although G_4 is defined by these two sets, we generally use a visualization of the graph where a dot represents a vertex and an edge is a line connecting the two dots (vertices). A drawing of G_4 is given below.



Note that two lines were drawn between vertices c and d as the edge cd is listed twice in the edge set. In addition, a circle was drawn at b to indicate an edge (bb) that starts and ends at the same vertex.

Example 1.2.1. There were six people: A, B, C, D, E and F in a party and several handshakes among them took place. Suppose that

A shook hands with B, C, D, E and F;
B, in addition, shook hands with C and F;
C, in addition, shook hands with D and E;
D, in addition, shook hands with E;
E, in addition, shook hands with F.

The situation can be clearly shown by the multigraph in Fig. 1.2.3, where people are represented by vertices and two vertices are joined by an edge whenever the corresponding persons shook hands.

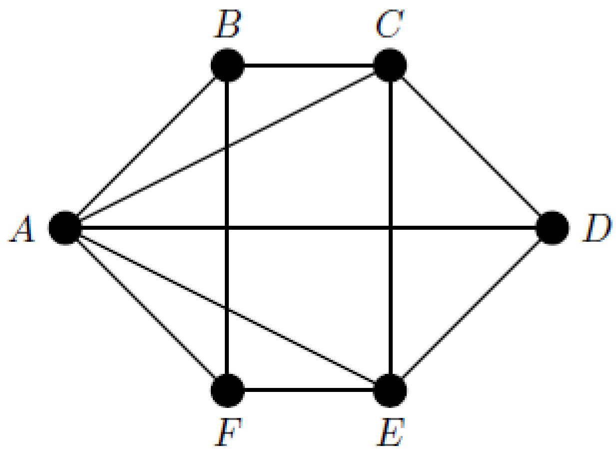
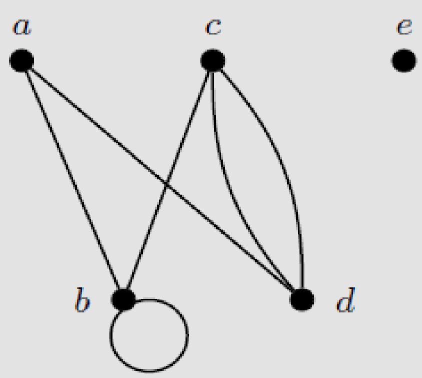


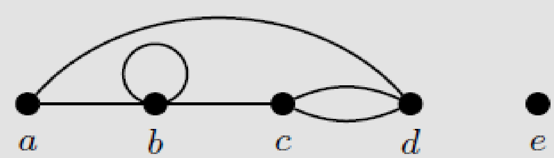
Fig. 1.2.3

Size of a Graph:

Definition 1.2 The number of vertices in a graph G is denoted $|V(G)|$, or more simply $|G|$. The number of edges is denoted $|E(G)|$ or $\|G\|$.

Example 1.2 Consider the graph G_4 from Example 1.1. Below are two different drawings of G_4.





To verify that these drawings represent the same graph from Example 1.1, we should check the relationships arising from the vertex set and edge set. For example, there are two edges between vertices c and d, a loop at b, and no edges at e. You should verify the remaining edges.

Definition 1.3 Let G be a graph.

- If xy is an edge, then x and y are the ***endpoints*** for that edge. We say x is ***incident to*** edge e if x is an endpoint of e.
- If two vertices are incident to the same edge, we say the vertices are ***adjacent***, denoted $x \sim y$. Similarly, if two edges share an endpoint, we say they are adjacent. If two vertices are adjacent, we say they are ***neighbors*** and the set of all neighbors of a vertex x is denoted $N(x)$.
  - ab and ad are adjacent edges in G_4 since they share an endpoint, namely vertex a
  - $a \sim b$, that is a and b are adjacent vertices as ab is an edge of G_4
  - $N(d) = \{a, c\}$ and $N(b) = \{a, b, c\}$

Independent & Isolated Vertices:

- If two vertices (or edges) are not adjacent then we call them ***independent***.
- If a vertex is not incident to any edge, we call it an ***isolated vertex***.
  - e is an isolated vertex of G_4

Loop & Multi-edges:

- If both endpoints of an edge are the same vertex, then we say the edge is a ***loop***.
  - bb is a loop in G_4
- If there is more than one edge with the same endpoints, we call these ***multi-edges***.
  - cd is a multi-edge of G_4

Simple Graph & Degree of a Vertex:

- If a graph has no multi-edges or loops, we call it ***simple***.
- The ***degree*** of a vertex v, denoted $\deg(v)$, is the number of edges incident to v, with a loop adding two to the degree. If the degree is even, the vertex is called ***even***; if the degree is odd, then the vertex is ***odd***.
  - $\deg(a) = 2,\ \deg(b) = 4,\ \deg(c) = 3,\ \deg(d) = 3,\ \deg(e) = 0$

K-Regular Graph:

If all vertices in a graph G have the same degree k, then G is called a **k-regular** graph. When $k = 3$, we call the graph **cubic**.

Null Graph:

The null graph is the graph whose vertex set and edge set are empty.

MATRIX REPRESENTATION:

- Very large graphs (such as those modeling the spread of an infectious disease, the connections within a terrorist organization, or the results from a season of NCAA Division 1 football) would be unwieldy without additional resources.
- One way to tackle large graphs is to represent them in such a way that a computer program can perform the required analysis.
- One method, which we will use at various times throughout this book, is to form the **adjacency matrix A(G)** of the graph G.

Adjacency Matrices:

Definition 1.19 The ***adjacency matrix*** $A(G)$ of the graph G is the $n \times n$ matrix where vertex v_i is represented by row i and column i and the entry a_{ij} denotes the number of edges between v_i and v_j.

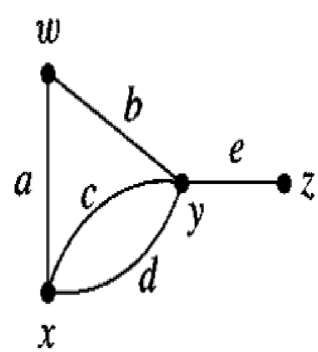
17. Definition. Let G be a loopless graph with vertex set $V(G) = \{v_1, \ldots, v_n\}$ and edge set $E(G) = \{e_1, \ldots, e_m\}$. The **adjacency matrix** of G, written $A(G)$, is the n-by-n matrix in which entry $a_{i,j}$ is the number of edges in G with endpoints $\{v_i, v_j\}$. The **incidence matrix** $M(G)$ is the n-by-m matrix in which entry $m_{i,j}$ is 1 if v_i is an endpoint of e_j and otherwise is 0.

1.1.18. Remark. An adjacency matrix is determined by a vertex ordering. Every adjacency matrix is **symmetric** ($a_{i,j} = a_{j,i}$ for all i, j). An adjacency matrix of a simple graph G has entries 0 or 1, with 0s on the diagonal. The degree of v is the sum of the entries in the row for v in either $A(G)$ or $M(G)$. ∎

1.1.19. Example. For the loopless graph G below, we exhibit the adjacency matrix and incidence matrix that result from the vertex ordering w, x, y, z and the edge ordering a, b, c, d, e. The degree of y is 4, by viewing the graph or by summing the row for y in either matrix. ∎

$$\begin{array}{c c} & \begin{array}{cccc} w & x & y & z \end{array} \\ \begin{array}{c} w \\ x \\ y \\ z \end{array} & \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 2 & 0 \\ 1 & 2 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \end{array}$$

$A(G)$



G

$$\begin{array}{c c} & \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{array}{c} w \\ x \\ y \\ z \end{array} & \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

$M(G)$

Example of Graph to Adjacency Matrix:

Example 1.16 Find the adjacency matrix for the graph G_4 from Example 1.1.

Solution:

$$\begin{array}{c} \\ a \\ b \\ c \\ d \\ e \end{array} \begin{array}{c} \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 2 & 0 \\ 1 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

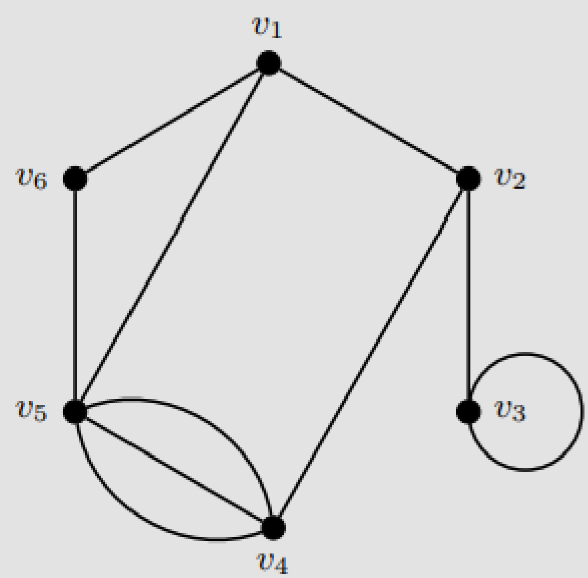
Note that the entry $(2, 2)$ represents the loop at b and the entries $(3, 4)$ and $(4, 3)$ show that there are two edges between c and d. The column for e has all 0's since e is an isolated vertex.

Example of Adjacency Matrix to Graph:

Example 1.17 Draw the graph whose adjacency matrix is shown below.

$$\begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 3 & 0 \\ 1 & 0 & 0 & 3 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Solution: Since the matrix has 6 rows and columns, we know that the graph must have 6 vertices. We will label them as $v_1, v_2, \ldots, v_6$.



Properties of Adjacency Matrix:

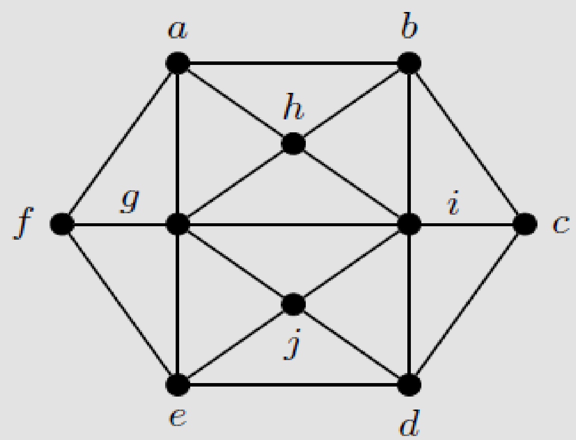
- First, the matrix is **symmetric** along the main diagonal since if there is an edge v_iv_j then it will be accounted for in both the entry (i, j) and (j, i) in the matrix.
- Second, the **main diagonal** represents all **loops** in the graph.
- Finally, the degree of a vertex can be easily calculated from the adjacency matrix by **adding the entries along the row (or column)** representing the vertex but **double** any item along the diagonal.
- In the matrix above, we would get deg(a) = 2 and deg(b) = 4, which matches the graph representation from Example 1.1.

SUBGRAPHS:

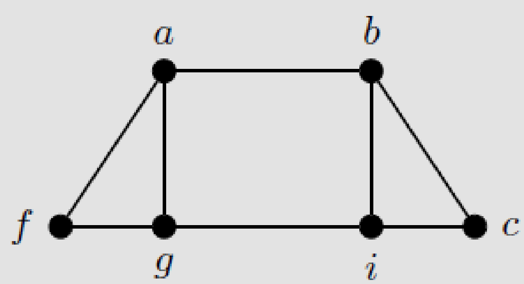
- When examining graphs, especially if they are particularly large, we may want to discuss a smaller portion of the graph, called a subgraph.

Definition 1.4 A *subgraph* H of a graph G is a graph where H contains some of the edges and vertices of G; that is, $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$.

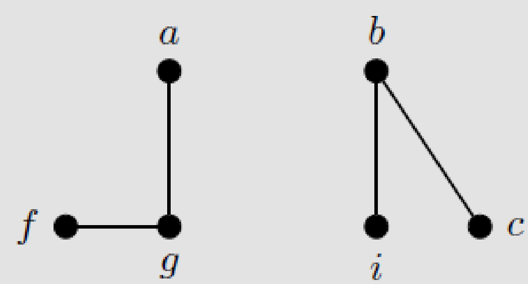
Example 1.3 Consider the graph G below. Find two subgraphs of G, both of which have vertex set $V' = \{a, b, c, f, g, i\}$.



Solution: Two possible solutions are shown below. Note that the graph H_1 on the left contains every edge from G amongst the vertices in V', whereas the graph H_2 on the right does not since some of the available edges are missing (namely, ab, af, ci, and gi).



H_1



H_2

- The graph shown on the left above is a special type, called an induced subgraph, since all the edges are present between the chosen vertices. Another special type of subgraph, called a spanning subgraph, includes all the vertices of the original graph.

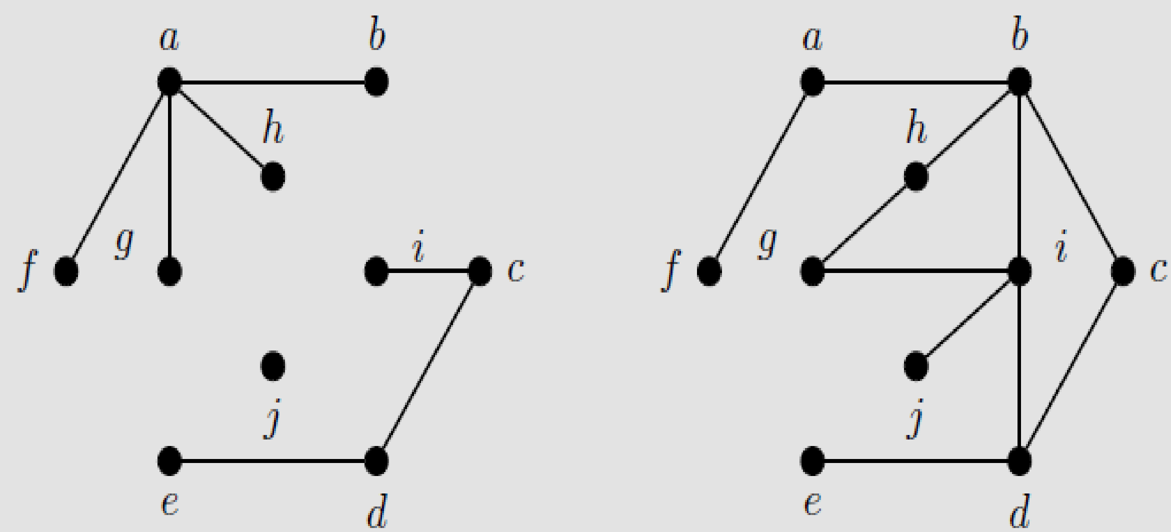
Induced & Spanning Subgraphs:

Definition 1.5 Given a graph $G = (V, E)$, an ***induced subgraph*** is a subgraph $G[V']$ where $V' \subseteq V$ and every available edge from G between the vertices in V' is included.

We say H is a ***spanning subgraph*** if it contains all the vertices but not necessarily all the edges of G; that is, $V(H) = V(G)$ and $E(H) \subseteq E(G)$.

Example 1.4 Find a spanning subgraph of the graph G from Example 1.3 above.

Solution: Two possible solutions are shown below. Note that both graphs contain all the vertices from G, but only in the graph H_4 could we move between any two vertices in the graph (which we will later call connected). Spanning subgraphs similar to H_4 will be studied in Chapter 3.



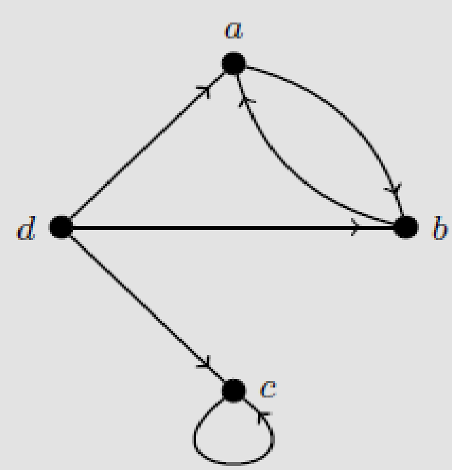
H_3 H_4

DIGRAPHS or DIRECTED GRAPHS:

Definition 1.6 A *directed graph*, or *digraph*, is a graph $G = (V, A)$ that consists of a vertex set $V(G)$ and an *arc set* $A(G)$. An *arc* is an ordered pair of vertices.

Digraphs have many similar properties to (undirected) graphs. Looking at the digraph above, we can see that the number of wins is modeled as the number of arcs coming from a team's vertex, and the number of losses is the number of arcs entering the vertex.

Example 1.6 Let G_5 be a digraph where $V(G_5) = \{a, b, c, d\}$ and $A(G_5) = \{ab, ba, cc, dc, db, da\}$. A drawing of G_5 is given below.



- Analogous definitions to those in Definition 1.3 exist for digraphs.

In-Degree & Out-Degree:

Definition 1.7 Let $G = (V, A)$ be a digraph.

- Given an arc xy, the ***head*** is the starting vertex x and the ***tail*** is the ending vertex y.
  - a is the head of arc ab and the tail of arcs da and ba from G_5

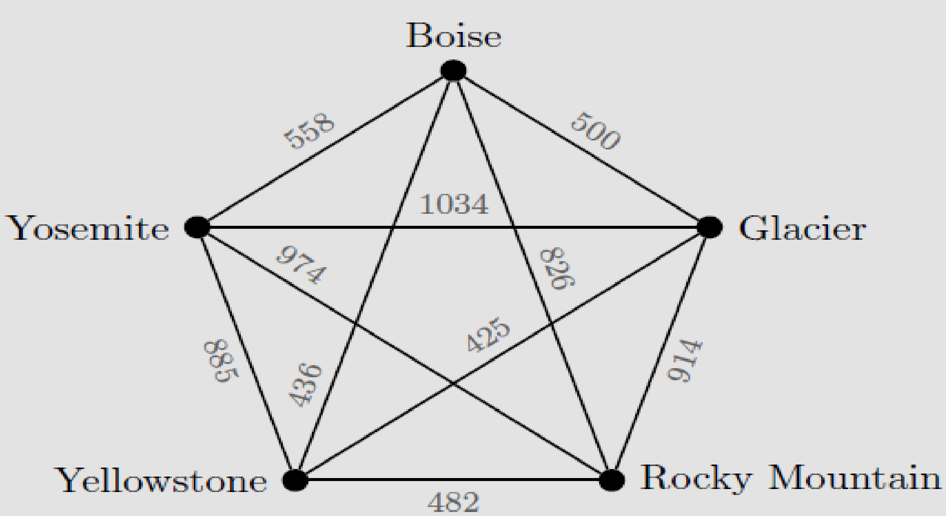
- Given a vertex x, the ***in-degree*** of x is the number of arcs for which x is a tail, denoted $\deg^-(x)$. The ***out-degree*** of x is the number of arcs for which x is the head, denoted $\deg^+(x)$.
  - $\deg^-(a) = 2, \deg^-(b) = 2, \deg^-(c) = 2, \deg^-(d) = 0$
  - $\deg^+(a) = 1, \deg^+(b) = 1, \deg^+(c) = 1, \deg^+(d) = 3$
- The ***underlying graph*** for a digraph is the graph $G' = (V, E)$ which is formed by removing the direction from each arc to form an edge.

➢ Knowing the degrees in a graph or digraph can tell you a lot of information but need not uniquely determine the underlying graph structure.

Weighted Graphs:

- Digraphs are used to model asymmetric relationships between discrete objects.
- We now consider a different edge relationship, where instead of direction we are concerned with quantity. These graphs are called weighted graphs.

Example 1.7 Sam wants to visit 4 national parks over the summer. To save money, he needs to minimize his driving distance. The graph below has weights along each edge indicating the driving distance between his home (in Boise, Idaho) and the four national parks he will visit.



Definition 1.8 A *weighted graph* $G = (V, E, w)$ is a graph where each of the edges has a real number associated with it. This number is referred to as the ***weight*** and denoted $w(xy)$ for the edge xy.

- Weighted graph can also refer to a graph in which each of the vertices is assigned a weight, and denoted w(v) for a vertex v.
- Also, the weight associated with an edge can represent more than just distance.
- For example, we may be interested in time, cost or some other measure related to the connection between two discrete objects.

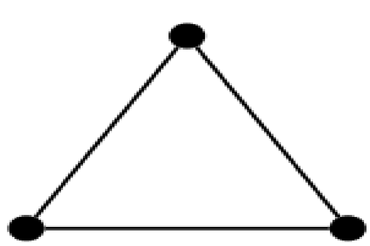
Complete Graphs:

Definition 1.9 A simple graph G is ***complete*** if every pair of distinct vertices is adjacent. The complete graph on n vertices is denoted K_n.

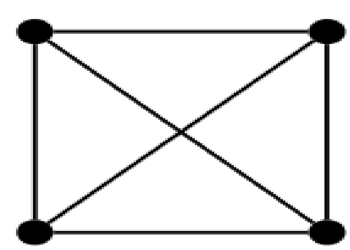
- The first six complete graphs are shown given below.

K_1

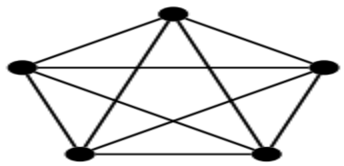
K_2



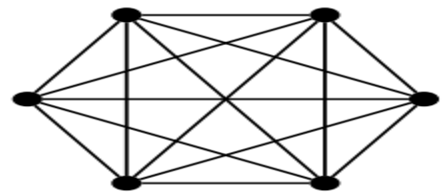
K_3



K_4



K_5



K_6

- Complete graphs are special for a number of reasons.
- In particular, if you think of an edge as describing a relationship between two objects, then a complete graph represents a scenario where every pair of vertices satisfies this relationship.
- Other useful properties of complete graphs are given below.

Properties of K_n

(1) Each vertex in K_n has degree $n-1$.

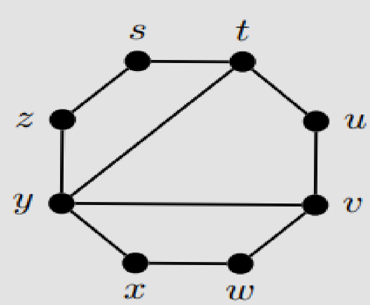
(2) K_n has $\dfrac{n(n-1)}{2}$ edges.

(3) K_n contains the most edges out of all simple graphs on n vertices.

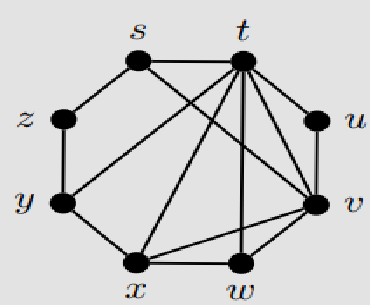
Clique Size:

Definition 1.10 The ***clique-size*** of a graph, $\omega(G)$, is the largest integer n such that K_n is a subgraph of G but K_{n+1} is not.

Example 1.9 Find $\omega(G)$ for each of the graphs shown below.

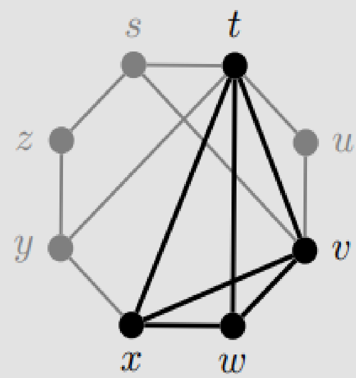


G_1



G_2

Solution: First note that G_1 does not contain any triangles (K_3) but does have an edge and so contains K_2. Thus $\omega(G_1) = 2$. Next, in G_2 the vertices t, v, w, x are all adjacent, as shown below, but we cannot find a collection of 5 vertices that are all adjacent (not enough vertices have degree at least 4). Thus $\omega(G_2) = 4$.



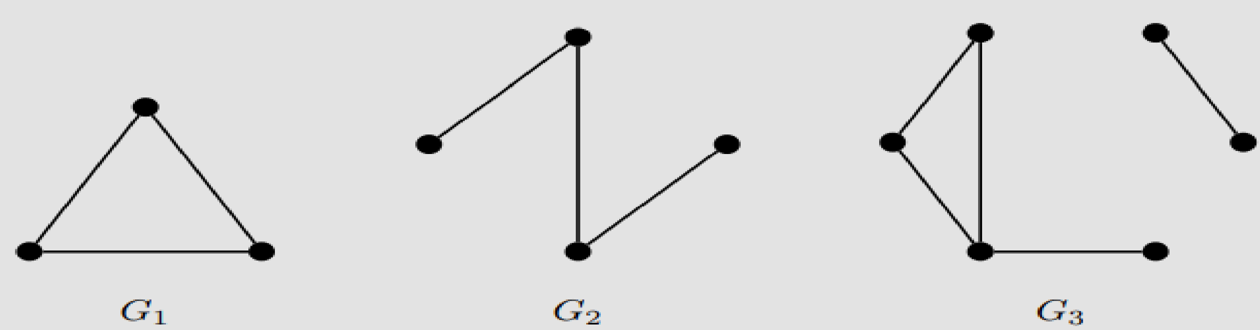
GRAPH COMPLEMENTS:

Definition 1.11 Given a simple graph $G = (V, E)$, define the ***complement*** of G as the graph $\overline{G} = (V, \overline{E})$, where an edge $xy \in \overline{E}$ if and only if $xy \notin E$.

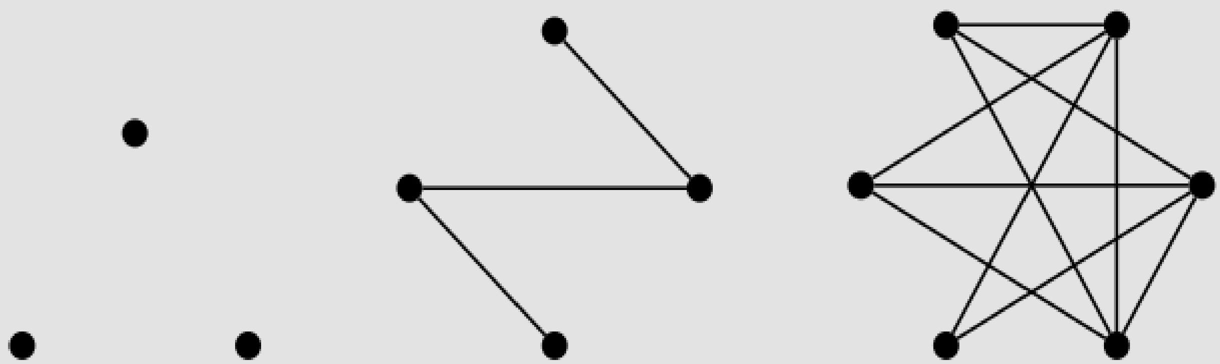
Remark:

Graph complements are only defined for simple graphs (graphs without loops and multi-edges).

Example 1.10 Find the complements of each graph shown below.



Solution: For each graph we simply add an edge where there wasn't one before and remove the current edges.



$\overline{G}_1$ $\overline{G}_2$ $\overline{G}_3$

Remark:

If we have a graph **G** on n vertices and add every edge in G to the edges of $\overline{G}$, then the resulting graph is simply K_n.

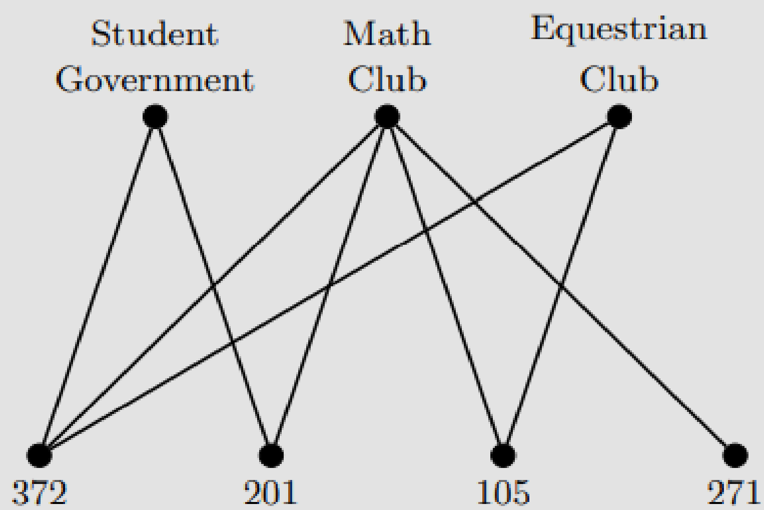
BIPARTITE GRAPHS:

- As we have already seen, problems that can be modeled by a graph need to consist of distinct objects (such as people or places) and a relationship between them.
- The proper model will allow the graph structure, or properties of the graph, to answer the question being asked.
- If we want to display the relationship between different types of objects, we would use a **bipartite graph**.

Definition 1.12 A graph G is ***bipartite*** if the vertices can be partitioned into two sets X and Y so that every edge has one endpoint in X and the other in Y.

Example 1.11 Three student organizations (Student Government, Math Club, and the Equestrian Club) are holding meetings on Thursday afternoon. The only available rooms are 105, 201, 271, and 372. Based on membership and room size, the Student Government can only use 201 or 372, Equestrian Club can use 105 or 372, and Math Club can use any of the four rooms. Draw a graph that depicts these restrictions.

Solution: Each organization and room is represented by a vertex, and an edge denotes when an organization is able to use a room.



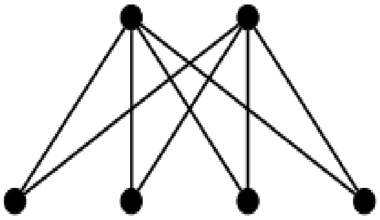
Note that edges do not occur between two organizations or between two rooms, as these would be nonsensical in the context of the problem. The graph above is a bipartite graph.

Complete Bipartite Graph:

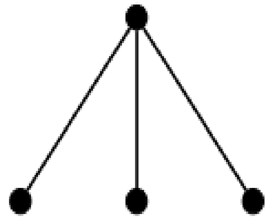
Definition 1.13 $K_{m,n}$ is the ***complete bipartite graph*** where $|X| = m$ and $|Y| = n$ and every vertex in X is adjacent to every vertex in Y.

Examples:

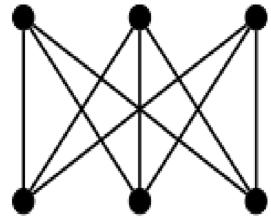
Below are a few complete bipartite graphs.



$K_{2,4}$

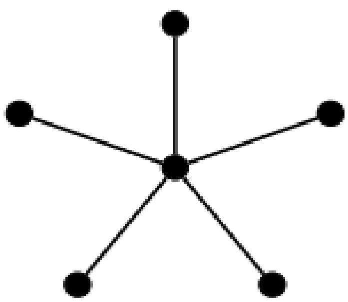


$K_{1,3}$

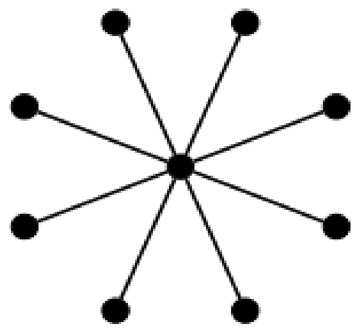


$K_{3,3}$

- When m = 1, we call $K_{1,n}$ a **star** since we could draw these with a singular vertex in the center and the remaining vertices surrounding it, as seen below with $K_{1,5}$ and $K_{1,8}$.



$K_{1,5}$

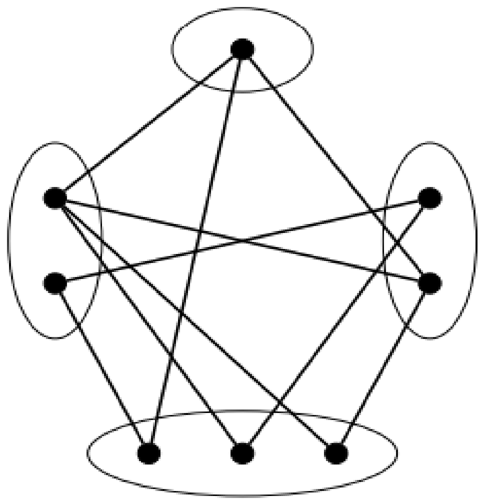


$K_{1,8}$

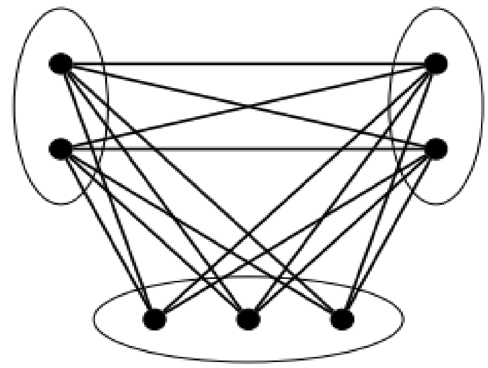
- We can also further generalize bipartite graphs where we break the vertices into more than just two sets.

k-Partite Graphs:

Definition 1.14 A graph G is ***k-partite*** if the vertices can be partitioned into k sets $X_1, X_2, \ldots X_k$ so that every edge has one endpoint in X_i and the other in X_j where $i \neq j$.



4-partite



$K_{2,2,3}$

- When k = 3, we call the graph **tripartite** rather than a **3-partite**.
- Above are drawings of a **4-partite graph** and the **complete tripartite graph $K_{2,2,3}$.**

GRAPH COMBINATIONS:

- As graphs are built from sets of vertices and edges, some operations on sets have natural translations onto graphs.

Union & Sum of Graphs:

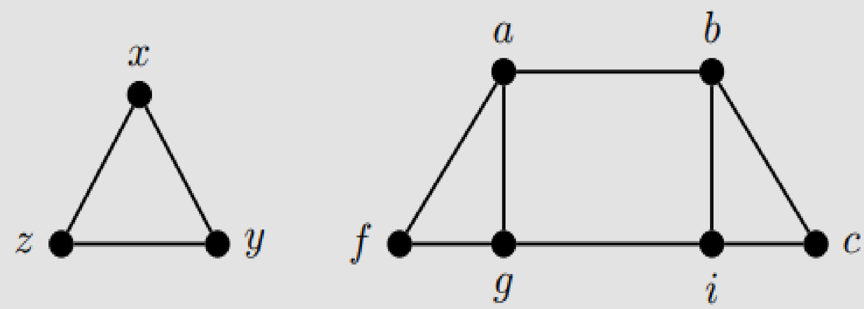
Definition 1.15 Given two graphs G and G the ***union*** $G \cup H$ is the graph with vertex-set $V(G) \cup V(H)$ and edge-set $E(G) \cup E(H)$.

If the vertex-sets are disjoint (that is $V(G) \cap V(H) = \emptyset$) then we call the disjoint union the ***sum***, denoted $G + H$.

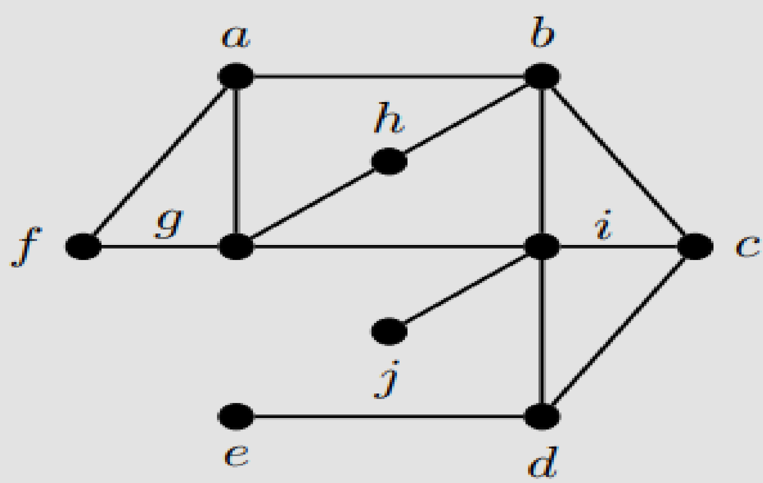
- Note that **G + H** is just a special type of union, and so unless we want to explicitly use or note that the vertex sets are disjoint, it is customary to use the union notation.

Example 1.12 Find the sum $K_3 + H_1$ and the union $H_1 \cup H_4$ using the graphs from Examples 1.3 and 1.4.

Solution: First note that, since we are finding the sum $K_3 + H_1$, we are assuming the vertex sets are disjoint. Thus the resulting graph is simply the graph below.



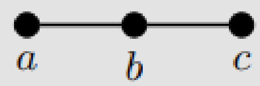
Next, since H_1 and H_4 are subgraphs of the same graph and have some edges in common, their union will consist of all the edges in at least one of H_1 and H_4, where we do not draw (or list) an edge twice if it appears in both graphs, as shown in the following graph.



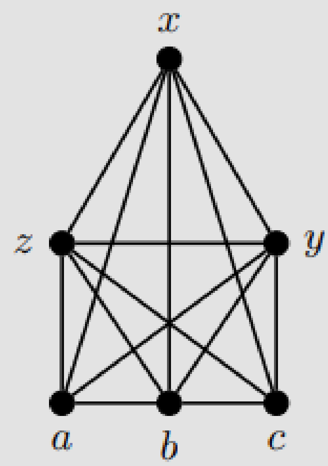
Join of Graphs:

Definition 1.16 The ***join*** of two graphs G and H, denoted $G \vee H$, is the sum $G + H$ together with all edges of the form xy where $x \in V(G)$ and $y \in V(H)$.

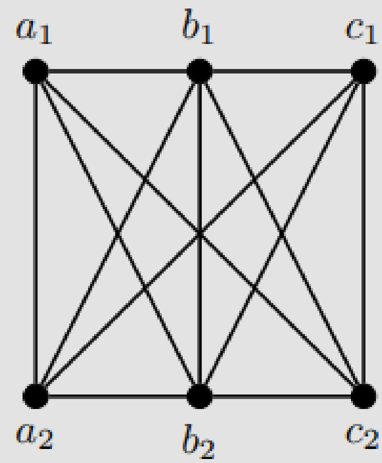
Example 1.13 Find the join of K_3 and the graph G below consisting of three vertices and two edges, as well as the join $G \vee G$.



Solution: The join $K_3 \vee G$ is shown below on the left. Note that every vertex from K_3 is adjacent to all those from G, but this is not K_6 since the edge ac is missing. The join $G \vee G$ is on the right below.



$K_3 \vee G$



$G \vee G$