

TERMINOLOGY AND BASIC GRAPH THEORY

Introduction

This chapter presents an overview of basic graph theory, including its association with set theory. Graphs can be shown to be quite useful, especially as a mathematical tool for studying network problems. We shall begin our study with graph theory as applied to static problems in network theory, which are those problems that are related to the structure of the network. Static problems include the assessment of the impact of the loss of one or more communicating nodes or one or more communication links. We use graph theory in an attempt to create networks that are less vulnerable to such loss.

In another chapter of these notes, we shall consider the application of graph theory to dynamic problems, such as dynamic load balancing. We shall show that certain algorithms become unstable under dynamic conditions, in that they present alternating optimal solutions: try this, no try that, etc. This observation should serve as a caution not to trust results from static graph theory to work in all dynamic problem areas.

But first we must get started with the basic graph theory. We begin with the definition of sets and develop the idea of a graph as a set of vertices and a set of edges.

Terminology and notations used

A **graph** G is a finite non-empty set of objects called **vertices** together with a (possibly empty) finite set of unordered pairs of distinct vertices of G called **edges**. The **vertex set** of G is commonly denoted by $V(G)$, and the **edge set** commonly denoted by $E(G)$. The cardinality of the vertex set of a graph G is called the **order** of G , and the cardinality of the edge set is called the **size** of G . An **(n, m) -graph** G is a graph with n vertices and m edges; $|V(G)| = n$ and $|E(G)| = m$. Although graphs are formally defined in terms of sets, they are commonly depicted by figures in which the nodes are depicted as circles (or ellipses) and the edges as lines between the circles.

The formal definition of a graph is based on set theory and utilizes the Cartesian product of sets, for which we present the standard definition. We begin by recalling that a **set** is an unordered collection of elements. For a set A , we write $a \in A$ if element a is a member of set A and $a \notin A$ if it is not. We sometimes define sets by a complete listing of the members of the set and sometimes by a description of the form $\{x \mid p(x)\}$, to be read as the set of all x such that $p(x)$ is true. Although it would be a bit strange, one can define the set of all odd integers as $\{x \mid (x \text{ is an integer}) \text{ and } (x \text{ is an odd number})\}$.

Let A and B be two arbitrary sets, defined over the same type of elements. We say that A is a **subset** of B , denoted as $A \subseteq B$, if every element that is in A is also in B ; more formally: $A \subseteq B$ if and only if $a \in A$ implies $a \in B$. Two sets A and B are equal if and only if both $A \subseteq B$ and $B \subseteq A$. We say that A is a **proper subset** of B , denoted $A \subset B$, if $A \subseteq B$, but $A \neq B$.

Definition: For any two sets A and B , the **Cartesian product** of A and B , denoted by $A \times B$, is the set of pairs of elements defined by $A \times B = \{ (a, b) \mid a \in A, b \in B \}$; thus it is the set of pairs of elements (a, b) for which the first element is a member of set A and the second element is a member of set B .

As we shall soon see, one may take the Cartesian product of a set with itself. Thus we have $A \times A = \{ (a_1, a_2) \mid a_1 \in A, a_2 \in A \}$. This work will use the Cartesian product sets for which the elements of the pair are distinct and unordered; thus $a_1 \neq a_2$ and (a_1, a_2) is considered the same element as (a_2, a_1) . For example, consider $A = \{1, 2, 3, 4\}$. The set $A \times A$ has 16 elements; our work focuses on a subset of $A \times A$, $E \subset A \times A$, that can be listed as

$$E = \{ (1, 2), (1, 3), (1, 4), (2, 3), (2, 4), (3, 4) \}.$$

Let X be an arbitrary set with a finite number of elements. The **cardinality** of X , denoted by $|X|$, is the number of elements in the set. If $|X| = 0$, the set is said to be the **empty set**, denoted by Φ . If $|X| > 0$, the set is said to be **non-empty**. We now define the basic set operations. For two sets A and B :

- the **set intersection**, denoted $A \cap B$, is $A \cap B = \{ x \mid x \in A \text{ and } x \in B \}$,
- the **set union**, denoted $A \cup B$, is $A \cup B = \{ x \mid x \in A \text{ or } x \in B \}$, and
- the **set difference**, denoted $A - B$, is $A - B = \{ x \mid x \in A \text{ and } x \notin B \}$, and
- the **set symmetric difference**, denoted $A \oplus B$, is $A \oplus B = (A \cup B) - (A \cap B)$; that is, the set of elements either in set A or in set B , but not in both sets.

As an example, consider the following two sets, each a subset of the integers.

$$A = \{2, 4, 6, 8, 10, 12, 14, 16, 18\}$$

$$B = \{3, 6, 9, 12, 15, 18\}$$

Then $A \cap B = \{6, 12, 18\}$

$$A \cup B = \{2, 3, 4, 6, 8, 9, 10, 12, 14, 15, 16, 18\}$$

$$A - B = \{2, 4, 8, 10, 14, 16\}$$

$$A \oplus B = \{2, 3, 4, 8, 9, 10, 14, 15, 16\}.$$

These set operations are often illustrated using Venn diagrams, as shown below.

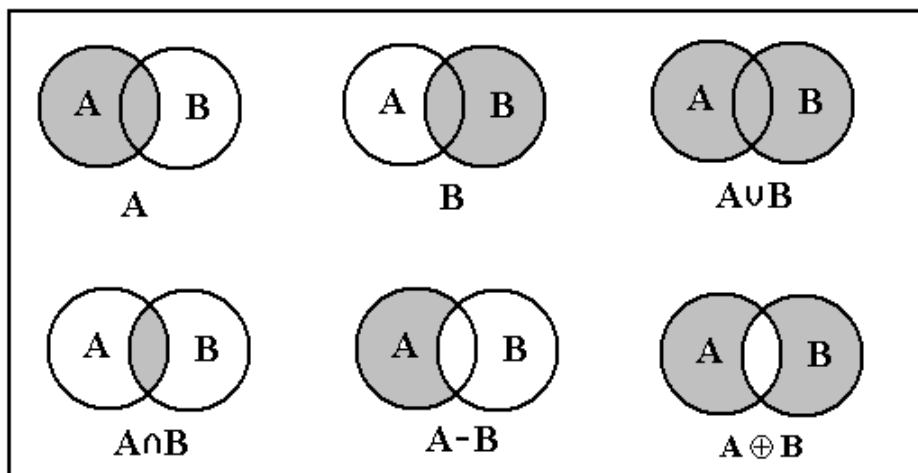


Figure 1: Venn Diagrams for Common Set Operations

Set Elements and Singleton Sets

At this point, it is important to make the distinction between elements of sets and the sets themselves. A set element can never be equal to a set. A **singleton set** is a set with one element. Singleton sets, unlike the empty set Φ , generally have no special significance in set theory and are mentioned only to clarify the notations used in the theory.

Consider the following set: $F = \{0, 1, 2, 3\}$, the set of integers modulo 4. The number 1 is an element of that set, so we can write $1 \in F$. Note that the element 1 is distinct from $\{1\}$, which is the set consisting of the single element 1. The following are true statements.

$1 \in F$	the element 1 is a member of the set F.
$1 \in \{1\}$	the element 1 is a member of this set also.
$\{1\} \subset F$	the set $\{1\}$ is a proper subset of the set F.
$\{1\} \subseteq F$	the set $\{1\}$ is a subset of the set F.
$\Phi \subseteq F$	the empty set is a subset of every set. In order to falsify this claim, one would have to show an element $x \in \Phi$, such that $x \notin F$. But the empty set has no members, so one cannot find such an element.

Note that we normally write subset inclusion as $X \subseteq Y$, unless it is important to state that X is a proper subset of set Y. We say $X \subset Y$ if either $X \subset Y$ or $X = Y$ is an acceptable condition.

The following statements are not correct and in many cases violate the conventions of set theory.

$1 \subset F$	an element cannot be a subset of any set. Elements are members of sets.
$1 = \{1\}$	an element can never be equal to a set; the two are different object types.
$\{1\} \in F$	F is a set of elements, so another set cannot be a member of F.
$\{1\} = F$	Obviously we have $\{1\} \subseteq F$, but $F \subseteq \{1\}$ is shown to be false by noting that $2 \in F$, but $2 \notin \{1\}$.

Sets of Sets

Just so the student knows it can be done, we can define a set containing other sets as members. Thus we can define $G = \{ \{0\}, \{1\}, \{2\}, \{3\} \}$. Note that G is not equal to F, as F has four integers as members and G has four singleton sets as members. In this case, we can properly write that $\{1\} \in G$, as the set G contains the element $\{1\}$.

Power Sets

We shall normally avoid sets that contain other sets as members. There is one important set of sets that we should discuss – the power set. We define the term and give an example.

For a given set X, the **power set of X**, denoted $P(X)$, is the set of all subsets of X.

If $A = \{0, 1, 2, 3\}$, as above, then

$$P(A) = \{ \Phi, \{0\}, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{1, 4\}, \{2, 3\}, \{2, 4\}, \{3, 4\}, \{1, 2, 3\}, \{1, 2, 4\}, \{1, 3, 4\}, \{2, 3, 4\}, \{1, 2, 3, 4\} \}$$

It can be proven that if $|X| = K$, then $|P(X)| = 2^K$; here $|A| = 4$ and $|P(A)| = 16 = 2^4$.

We now restate the definition of a graph, using the more precise terminology.

Definition: A graph G is a finite non-empty set of vertices, denoted $V(G)$, together with a (possibly empty) finite set $E \subseteq V(G) \times V(G)$ of unordered pairs. As before, we let $|V(G)| = n$ and $|E(G)| = m$ and speak of an (n, m) -graph, usually called G .

The definition above is a bit too general for use in association with graph theory, so we immediately restrict it a bit. We introduce the idea of simple graphs, which is the type of graphs normally implied by the term “graph”. A **simple graph** is a graph without edges connecting any vertex to itself. The graph in the next figure is not simple, as it has edges connecting vertex 1 to itself and vertex 4 to itself.

The graph at right is also described as follows

$$V(G) = \{1, 2, 3, 4\}$$

$$E(G) = \{ (1, 1), (1, 4), (2, 3), (2, 4), (4, 4) \}$$

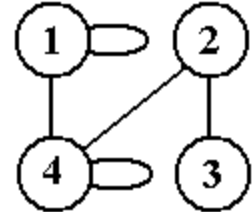


Figure 2: Two Representations of a Not-Simple Graph

We shall restrict our study of graphs to simple graphs. While graphs with loops are valuable in a number of studies, they are not useful in the analysis of networks and do present a number of difficulties that we would like to avoid. So – only simple graphs.

For a set of n objects, there are $n \cdot (n - 1)$ ordered pairs of distinct elements; that is pairs (i, j) , with $i \neq j$, in which the element (i, j) is different from the element (j, i) . For a set of n objects,

there are $\binom{n}{2} = \frac{n \cdot (n - 1)}{2}$ unordered pairs of distinct objects, in which the element (i, j) is

considered to be the same as the element (j, i) . We have two options, depending on whether the edge set contains ordered or unordered pairs.

Directed graphs correspond to edge sets that contain ordered pairs.

Undirected graphs correspond to edge sets that contain unordered pairs.

Consider the figure below, which shows an undirected graph and a directed graph.

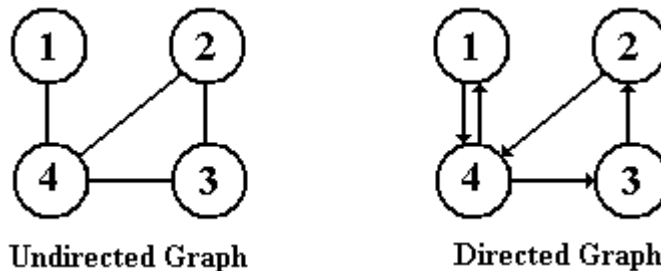


Figure 3: Two Sample Graphs On the Same Vertex Set

Each of the graphs in the figure has the vertex set $V = \{1, 2, 3, 4\}$. The edge set of the directed graph is $E = \{ (1, 4), (2, 4), (3, 2), (4, 1), (4, 3) \}$. The edge set of the undirected graph is $E = \{ (1, 4), (2, 3), (2, 4), (3, 4) \}$. Note that in the undirected graph, the following edges are implicitly listed: $(3, 2), (4, 1), (4, 2), (4, 3)$, as the pairs representing the edges are unordered. Thus, the pairs $(1, 4)$ and $(4, 1)$ are considered equivalent in an undirected graph and each represents the same edge. In a directed graph each of the ordered pairs $(1, 4)$ and $(4, 1)$ represents a distinct edge.

The student should note that it is always possible to create a directed graph that is equivalent to an undirected graph; one need only “double up” each edge in the undirected graph. The following figure shows an undirected graph and its equivalent directed graph.

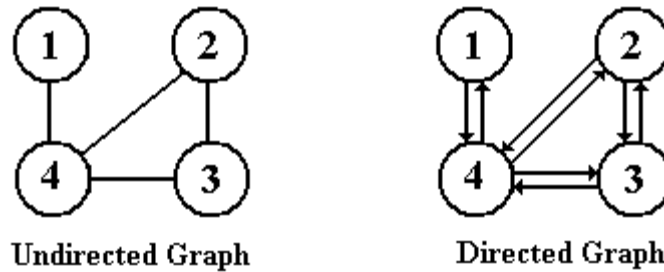


Figure 4: An Undirected Graph and the Equivalent Directed Graph

For the moment we shall restrict our discussions to undirected simple graphs, which we shall call “graphs” with no further distinction. As noted above, the edge set for an undirected graph with vertex set given by $V(G) = \{1, 2, \dots, n\}$. $E(G)$ is a subset of $V(G) \times V(G)$ that contains only

unordered pairs of distinct elements. For a set of n objects, there are $\binom{n}{2} = \frac{n \cdot (n-1)}{2}$ unordered

pairs of distinct objects, this is the maximum size of the edge set for an (n, m) -graph. Thus we have the following limits on the number of edges in a simple undirected graph G .

Proposition 1: Let G be a simple undirected graph, with $|V(G)| = n$.

$$\text{Then } 0 \leq |E(G)| \leq \binom{n}{2} = \frac{n \cdot (n-1)}{2}.$$

The complement of a graph G , denoted G_C , is the graph with the vertex set $V(G)$ and edge set defined by $E(G_C) = \{ (u, v) \mid u \in V(G), v \in V(G), (u, v) \notin E(G) \}$. If G is an (n, m) -graph, then G_C is an $(n, n \cdot (n-1)/2 - m)$ -graph.

In much of graph theory, the vertices of the graphs are labeled by integers, so that a four-vertex graph would have $V(G) = \{1, 2, 3, 4\}$. The set of possible edges for such a graph would be $\{ (1, 2), (1, 3), (1, 4), (2, 3), (2, 4), (3, 4) \}$. Consider two $(4, 3)$ -graphs, a graph and its complement. First we give a rather formal definition of the two graphs.

$$\begin{aligned} G &= (V(G), E(G) \mid V(G) = \{1, 2, 3, 4\}, E(G) = \{(1, 2), (1, 3), (1, 4)\}) \\ G_C &= (V(G), E(G) \mid V(G) = \{1, 2, 3, 4\}, E(G) = \{(2, 3), (2, 4), (3, 4)\}) \end{aligned}$$

While this is a sufficient definition of the two graphs, most people prefer the visual representation of the graphs. Here are standard representations of G and G_C .

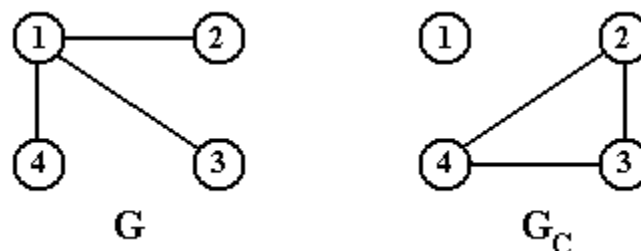


Figure 5: A Graph and Its Complement

Two graphs often have the same structure, differing only in the way their vertices and edges are labeled or in the way they are drawn. To make this idea more exact and to develop a way to focus on the essential structure of graphs, we introduce the concept of **graph isomorphism**. Two graphs G_1 and G_2 are said to be **isomorphic**, denoted by $G_1 \cong G_2$, if there exists a one-to-one mapping Φ from $V(G_1)$ onto $V(G_2)$ such that the mapping preserves the adjacency, that is to say that $(u, v) \in E(G_1)$ if and only if $(\Phi(u), \Phi(v)) \in E(G_2)$. Were we to push a point, we would note that graph isomorphism forms equivalence classes on graphs: if $G_1 \cong G_2$ and $G_2 \cong G_3$, then $G_1 \cong G_3$. The next figure shows two graphs that are labeled and drawn differently, but are isomorphic.



Figure 6: Two Isomorphic Graphs

The two graphs in Figure 6 are isomorphic under the following transformation: $\Phi(1) = A$, $\Phi(2) = B$, $\Phi(3) = C$, and $\Phi(4) = D$. The edge lists of the two graphs show this.

Graph on left: (1, 2), (1, 4), (2, 3), and (3, 4)

Graph on right: (A, B), (A, D), (B, C), and (C, D).

If we take the graph on left and apply the transformation to its vertex labels, we arrive at the edge list (A, B), (A, D), (B, C), and (C, D). This is precisely the edge list of the graph on the right, as expected. Thus, the two graphs are isomorphic.

The basic use of the idea of graph isomorphism is that we can view isomorphic graphs as identical and ask questions only about graphs that are not isomorphic. We use isomorphism to define a method of classifying graphs. For integers $n > 0$ and $m \geq 0$, let $\Gamma(n)$ denote the collection of all non-isomorphic graphs with n vertices and $\Gamma(n, m)$ denote the collection of all non-isomorphic graphs on n vertices and m edges. The next figure shows the sets $\Gamma(n)$ for $1 \leq n \leq 4$. Note that the set $\Gamma(1)$ has only one member – a single graph with one vertex and no edges incident on it; an isolated vertex.

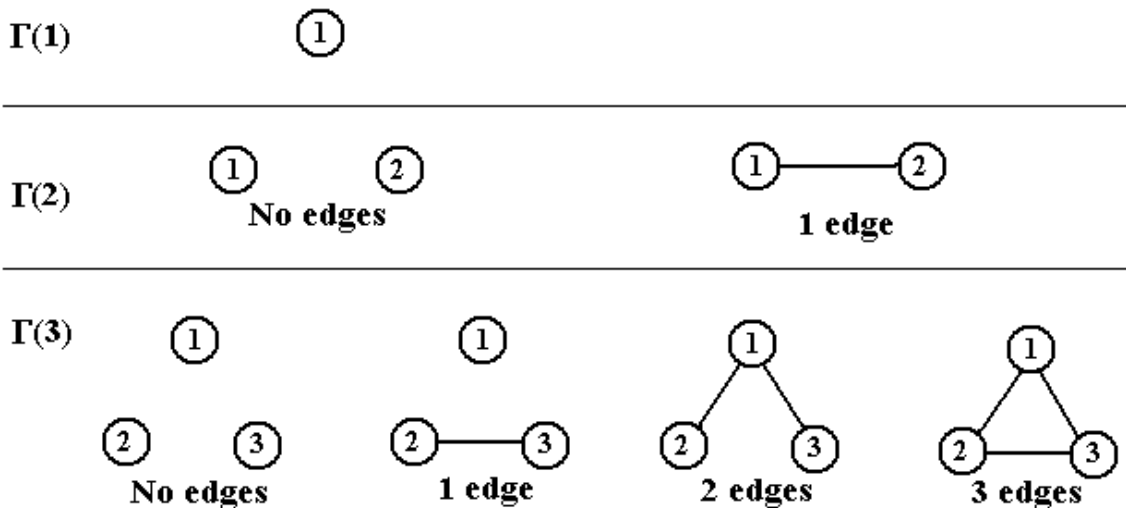


Figure 7: The sets $\Gamma(1)$, $\Gamma(2)$, and $\Gamma(3)$

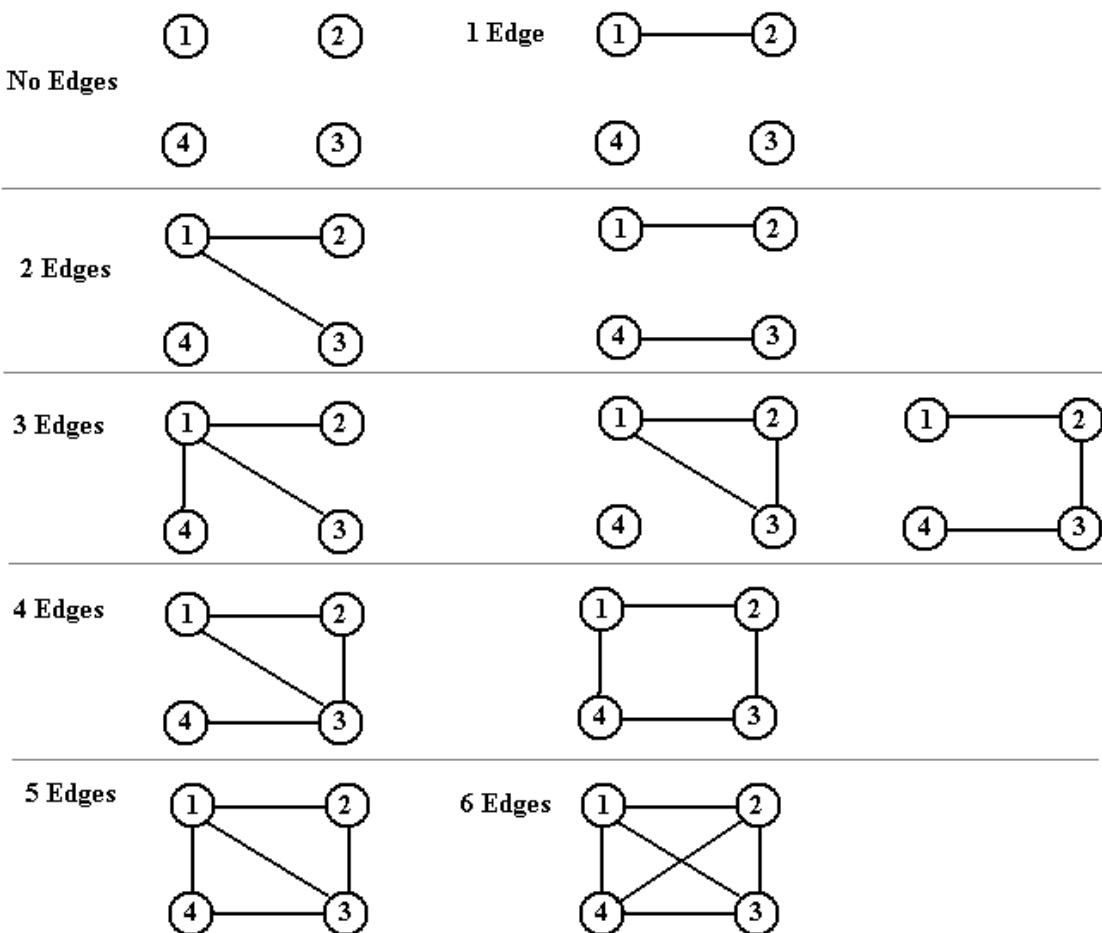


Figure 8: The Eleven Members of $\Gamma(4)$

Notation: At this point, we make a change in the way we refer to vertices. In our previous discussions, we used the “pure mathematics” approach to describing graphs, in which vertices were denoted by an integer in the range 1 to $|V(G)|$ inclusive and edges were denoted by unordered pairs of integers. In our studies, we normally use different labels for graphs, normally labeling the vertices as v_1, v_2, \dots, v_n for a graph with n vertices. When discussing a few vertices, we might give them labels, such as u, v , and w . Similarly, in discussing edges, we might use notation such as (u, v) or (v_i, v_j) . The student should note that there is no theoretical significance to this; it is just one of many conventional notations used in describing graphs.

An edge (u, v) is said to join the vertices u and v . If $(u, v) \in E(G)$, then vertices u and v are said to be **adjacent**; u is adjacent to v , and v is adjacent to u . The edge (u, v) is said to be **incident** on its end vertices u and v . Again, in simple graphs we assume $u \neq v$.

The **degree** of a vertex v in G , denoted either as d_v or $d(v)$, is the number of edges incident on the vertex v . Since each edge incident on the vertex v causes another vertex to be adjacent to v , we might say that the degree of the vertex is the number of vertices adjacent to it. The two definitions are entirely equivalent. Occasionally, when a vertex is a part of two or more different graphs, we use the full notation $d_G(v)$ to indicate the degree of a vertex v in the graph G . Normally such precision is not required.

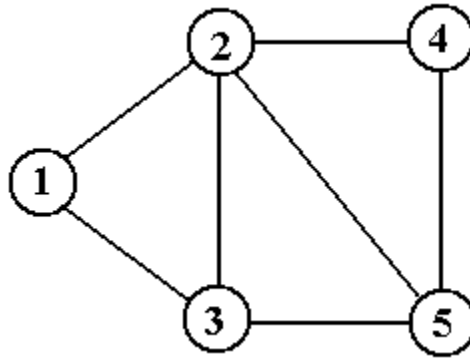


Figure 9: Illustration of Vertex Degrees

In this figure, $d_1 = d(v_1) = 2$, $d_2 = d(v_2) = 4$, $d_3 = d(v_3) = 3$, $d_4 = d(v_4) = 2$, and $d_5 = d(v_5) = 3$. Note that $\sum_{j=1}^n d_j = 2 + 4 + 3 + 2 + 3 = 14 = 2 \bullet m$. This is not a coincidence.

A vertex of zero degree is called an **isolated vertex** in that it has no edges incident on it and thus is not adjacent to any other vertex. At this point it will be convenient to state a few lemmas and theorems related to vertex degree.

Lemma 2: Let G be an (n, m) -graph and let $v \in V(G)$. Then $0 \leq d(v) \leq (n - 1)$.

Proof: The assertion that $d(v) \geq 0$ comes from the fact that $d(v)$ is a counting number.

If $v \in V(G)$, there are only $(n - 1)$ other vertices in $V(G)$ to which v may be adjacent, thus it follows that $d(v) \leq (n - 1)$.

Theorem 3: Let G be an (n, m) -graph with $V(G) = \{v_1, v_2, \dots, v_n\}$. Let the degree of vertex v_j be given by $d_j = d(v_j)$. Then $\sum_{j=1}^n d_j = 2 \bullet m$.

Proof: Every edge in G is incident on two vertices; hence, when the degrees of the vertices are summed, each edge is counted twice. This completes the proof.

Theorem 4: Let G be an (n, m) -graph, with $m \geq n$. Then G has at least two vertices of degree $d(v) \geq 2$.

Proof: Assume that G has only one vertex with degree $d(v) \geq 2$. By Lemma 2, we have $d(v) \leq (n - 1)$, so we let the one vertex of degree greater than 1 have degree $(n - 1)$. The

maximum value of $\sum_{j=1}^n d_j$ is then $1 \bullet (n - 1) + (n - 1) \bullet 1 = 2 \bullet (n - 1)$, being generated by the one

vertex of degree $(n - 1)$ and the $(n - 1)$ vertices of degree 1. As a result of theorem 3, we have $m \leq (n - 1)$, which contradicts the assumption that $m \geq n$.

A graph G is called **regular** if all of its vertices have the same degree and is called **pseudoregular** if the degrees of its vertices differ by at most one. For a vertex v , define $N(v)$, the **open neighborhood** of v , as the set of vertices adjacent to v . As each edge incident on a vertex v connects it to an adjacent vertex, it follows immediately that $|N(v)| = d(v)$. The **closed neighborhood** of a vertex, denoted $N[v]$, adds the vertex itself to its open neighborhood; $N[v] = N(v) \cup \{v\}$. Note that $\{v\}$ is the set containing only one element – the vertex v .

A regular graph in which all vertices have degree k is called **k -regular**. The 3-regular graphs are called **cubic** and have been studied extensively.

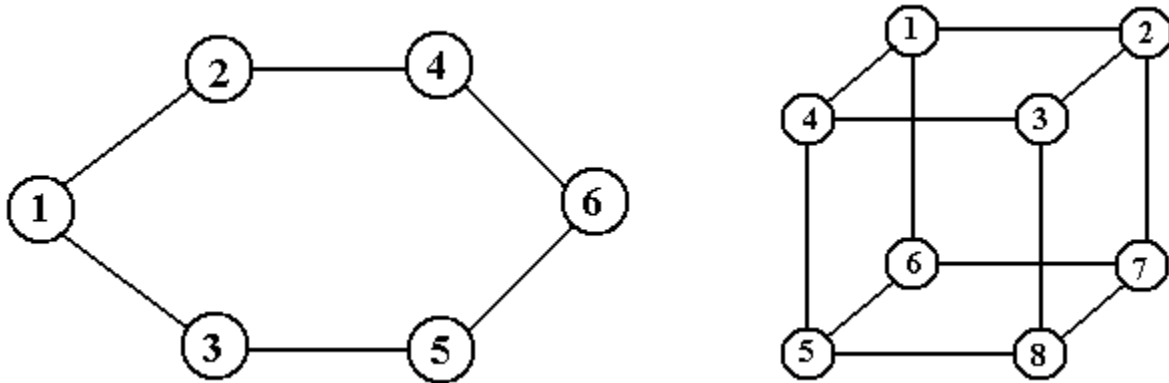


Figure 10: A 2-regular and a 3-regular graph

Note that in the above figure, that the 3-regular graph has been drawn so that its vertices appear at the corner of a cube. This is one of the reasons for the name “cubic”. The next topic to be discussed considers the number of vertices that are adjacent to each of two distinct vertices.

Let u and v be two distinct vertices in (n, m) -graph G . The **codegree** of the two vertices, denoted by $\text{codeg}(u, v)$, is the number of vertices adjacent to both u and v . In set notation, we can write $\text{codeg}(u, v) = |N(u) \cap N(v)|$. We now give an upper limit on the codegree of two vertices.

Lemma 5: Let u and v be two distinct vertices in (n, m) -graph G .
Then $0 \leq \text{codeg}(u, v) \leq (n - 2)$.

Proof: The assertion that $\text{codeg}(u, v) \geq 0$ comes from the observation that it is a counting number. The upper limit comes from the observation that there are only $(n - 2)$ other vertices in G , so that $|N(u) \cap N(v)| \leq (n - 2)$.

We now place a lower limit on the codegree of two adjacent vertices.

Lemma 6: Let x and y be two adjacent vertices in an (n, m) -graph G . Then
 $d(x) + d(y) \leq \text{Codeg}(x, y) + n$.

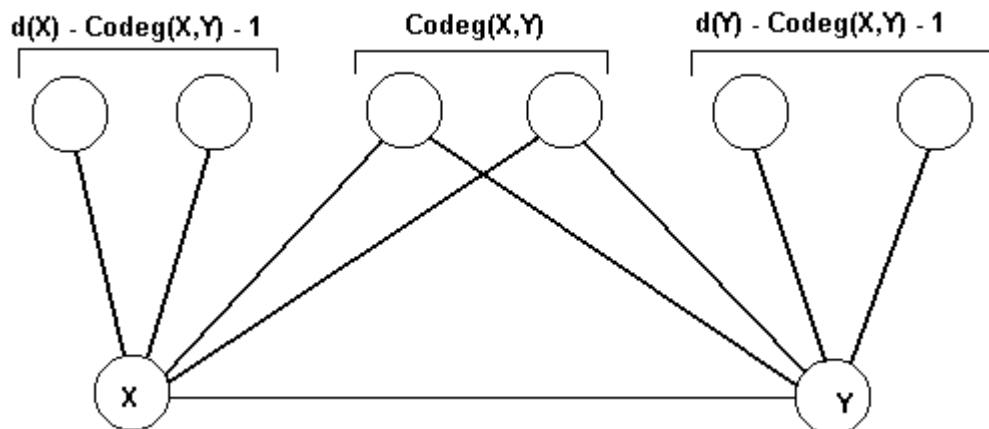


Figure 11: The Degrees and Codegree of Two Adjacent Vertices

Proof: Consider the two adjacent vertices x and y in the above diagram. Other than vertex y , there are $d(x) - 1$ vertices adjacent to vertex x , of which $d(x) - \text{Codeg}(x, y) - 1$ are adjacent to vertex x but not vertex y and $\text{Codeg}(x, y)$ are adjacent to both x and y . The number of vertices (other than x or y) that are adjacent to either vertex x or vertex y or both is given by $d(x) - \text{Codeg}(x, y) - 1 + \text{Codeg}(x, y) + d(y) - \text{Codeg}(x, y) - 1 = d(x) + d(y) - \text{Codeg}(x, y) - 2$. But other than vertices x and y there are only $(n - 2)$ other vertices in the graph G , so we have $d(x) + d(y) - \text{Codeg}(x, y) - 2 \leq (n - 2)$ or $d(x) + d(y) \leq \text{Codeg}(x, y) + n$.

A **subgraph** of a graph G is a graph having all of its nodes and edges in G . Thus, H is a subgraph of G if and only if $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$. A subgraph of G is a **spanning subgraph** if it contains all of the nodes of G . Thus H is a spanning subgraph of G if $V(H) = V(G)$ and $E(H) \subseteq E(G)$. For any sets U of nodes in G , $U \subseteq V(G)$, the **induced subgraph** $\langle U \rangle$ is the maximal subgraph with vertex set U . Put another way, the induced subgraph $\langle U \rangle$ is the graph with vertex set $U \subseteq V(G)$, with any two vertices being adjacent in $\langle U \rangle$ if and only if they are adjacent in G . We say more on induced subgraphs later in this chapter.

Let u and v be vertices in a graph G , with u and v not necessarily distinct. A **u - v walk** of G is a finite, alternating sequence of vertices and edges starting with u and ending with v : thought of as $u = u_0, e_1, u_1, e_2, \dots, u_{s-1}, e_s, u_s = v$, such that $e_i = (u_{i-1}, u_i)$. The number s , the number of edges in the sequence, is called the **length** of the walk. A **u - v path** is a u - v walk in which no vertex is repeated. A **cycle** is a u - v walk in which all vertices are distinct with the sole exception that $u = v$. Paths and cycles, as special cases of walks, have obvious definitions for their lengths. A graph G is said to be **connected** if there exists a path between every pair of distinct vertices in the graph, otherwise it is **disconnected**. For a connected graph G , we define the **distance** $d(u, v)$ as the minimum of the lengths of all u - v paths connecting the 2 vertices u and v . A path of minimum length between two vertices is sometimes called a **geodesic**.

A **connected component**, or simply a **component**, of a graph G is a maximal connected subgraph of G . If a graph is connected, it has only one component; otherwise it has two or more components. For any vertex $v \in V(G)$, the component containing v is formed by adding v to the set of all vertices reachable by a path from v .

The following figure shows a graph with three components.

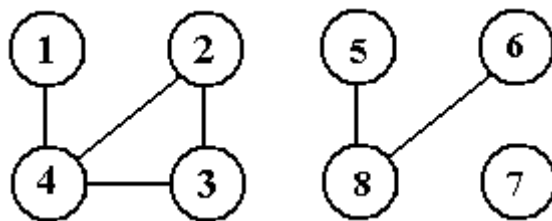


Figure 12: A Disconnected Graph With Three Components

The three components of the graph in this example are $\{1, 2, 3, 4\}$, $\{5, 6, 8\}$, and $\{7\}$. Note that there is a path from vertex 1 to vertex 3, so the two vertices are in the same component of the graph. Since there is no path from vertex 1 to vertex 5, they are in different components.

Recalling that a cycle is a path through a graph beginning and ending on the same vertex, we give the following definition of a graph without cycles.

Definition: An **acyclic graph** is a graph that does not contain a cycle.

Definition: A **tree** is a connected acyclic graph.

Definition: A **rooted tree** is a tree in which one vertex has been distinguished and called the **root**. Most trees of interest in computer science are rooted trees.

Trees play only a small part in the analysis of networks. The one tree of greatest importance for networks is the **star graph**, also called $K_{1,n-1}$ (see below). The next figure shows two of the smaller star graphs $K_{1,2}$ (also called a P_3 , see below) and $K_{1,4}$. Note that we, as computer science people, see each tree as a rooted tree with the root vertex (or root node) being vertex 1.

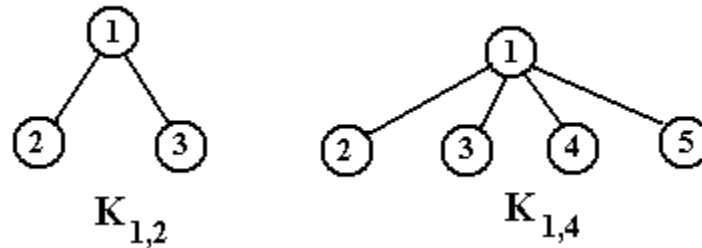


Figure 13: Two Star Graphs

Recalling that a subgraph H of graph G is a spanning subgraph if $V(H) = V(G)$ and $E(H) \subseteq E(G)$. If H is a spanning subgraph of G and H happens to be a tree, then H is said to be a spanning tree of the graph G .

Upon reflection, one should realize that a graph G may have many distinct spanning trees; indeed it is almost obvious that a graph G has a unique spanning tree if and only if G is itself a tree. The next figure shows a graph and its three spanning trees.

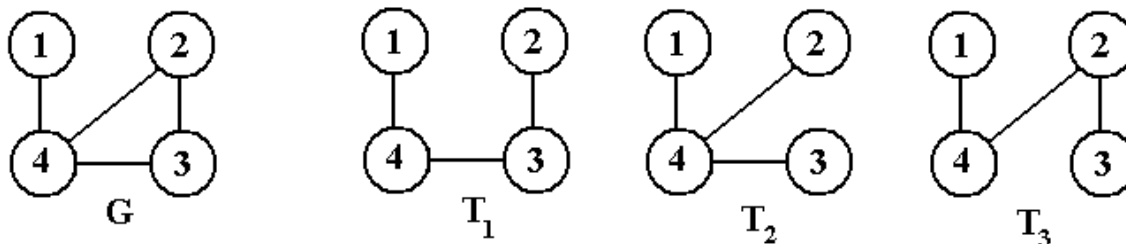


Figure 14: A Graph G and Its Spanning Trees

In the example above, we see a $(4, 4)$ -graph G (4 vertices and 4 edges) and its 3 spanning trees T_1 , T_2 , and T_3 . As we shall prove soon, a tree on four vertices must have exactly three edges. In the above example there are three edges that can be removed to yield a tree; removal of edge $(1, 4)$ will cause the graph to be disconnected. Note that each of T_1 and T_3 is isomorphic to the graph P_4 – a path on four vertices, while T_2 is isomorphic to $K_{1,2}$ – the star graph on three vertices.

We will soon quote one of the basic theorems regarding trees, but need to begin with a definition and a simple lemma. The definition serves to eliminate trivial exceptions from our theorems on trees.

Definition: A **nontrivial tree** is a tree with at least two vertices.

Lemma 7: Every nontrivial tree has at least two vertices of degree 1.

Proof: Let P be a longest path in a nontrivial tree T and let u and v be the end-nodes of the path P . Since T is acyclic, u and v each have only one neighbor in P , and since P is a longest path each has no neighbors in $T - P$ (else the path could be extended). Thus there must be at least two vertices of degree one in a nontrivial tree.

Before we quote the “big theorem” we must explain a new bit of terminology. Let G be an (n, m) -graph on at least two vertices, and let u and v be vertices that are not adjacent in G . Then by $G + (u, v)$ we denote the $(n, m + 1)$ -graph formed from G by adding making the two vertices u and v to be adjacent by adding the edge (u, v) . Another term often used is $G + e$, denoting the addition of a new edge to a graph G .

Theorem 8: The following statements are equivalent.

1. G is a tree with n vertices and m edges.
2. Every two distinct vertices of G are connected by a unique path.
3. G is connected and $m = n - 1$.
4. G is acyclic and $m = n - 1$.
5. G is acyclic and if any two nonadjacent vertices of G are joined by an edge e , then $G + e$, the graph with one edge added, has exactly one cycle.

Proof: This is a well-known result. The theorem as stated is a slight rewording of Theorem 1.2 in reference [R01]. We shall adapt the proof from that reference and show the proof as an example of how graph theorists think. If the statements are all equivalent, then they must either be all true or all false for a given graph. The strategy for a proof of equivalence is quite simple; we just prove that any one statement implies all of the other statements. In this case we shall show a circular equivalence; thus $1 \Rightarrow 2$, $2 \Rightarrow 3$, $3 \Rightarrow 4$, $4 \Rightarrow 5$, and $5 \Rightarrow 1$.

Proof that $1 \Rightarrow 2$

Since G is a tree, it is a connected graph without cycles. Since G is connected, every two distinct nodes in G are connected by a path. Suppose two distinct nodes u and v in G that are connected by two distinct paths P and P^* . Let w be the first node of path P (as we traverse from u to v) such that w is on both P and P^* , but its successor on P is not on P^* . Note that $w = v$ is allowed in this proof. We then follow path P from u to w and path P^* backwards from w to u to form a cycle. Thus the assumption of two distinct paths between any two vertices implies that the graph contains cycles and cannot be a tree.

Proof that $2 \Rightarrow 3$

If every distinct pair of nodes in G is connected by a unique path, then G is connected by definition. We prove that $n = m + 1$ by induction. Reference to figures 7 and 8 of this work will show that the statement is true for $n = 2, 3$, and 4. It is also vacuously true for $n = 1$. Now assume that the result is true for all graphs with fewer than n vertices.

Suppose that G is a graph with n nodes ($n \geq 2$), m edges, and let v be one of the nodes of degree one in G (see Lemma 6). Then $G - v$, the graph obtained by removing the vertex v and the edge incident on it from G , has $(n - 1)$ vertices, one less than G , and still satisfies property 2. By the inductive hypothesis $G - v$ has $m = (n - 1) - 1$, thus the number of edges in G is $m = (n - 1)$.

Proof that 3 \Rightarrow 4

Assume that G has a cycle of length p . Then there are p vertices and p edges on the cycle, and for each of the $(n - p)$ vertices not on the cycle there is an incident edge on a geodesic from that vertex to a vertex in the cycle. Each such edge is different, so $(n - p) + p = n \geq m$, which is a contradiction.

Proof that 4 \Rightarrow 5

Since G is acyclic, each component of G is a tree. If there are k components, then each component has one more vertex than edge and $n = m + k$, so the assumption that $n = m + 1$ implies that $k = 1$ and that G is connected. Thus G is a tree and there is exactly one path connecting any two nodes in G . If we add an edge (u, v) to G , that edge together with the unique path in G joining u and v forms a cycle. The cycle is unique because the path is unique.

Proof that 5 \Rightarrow 1

For this proof, we need a new notation. Let u and v be two non-adjacent vertices in a graph G . The graph $G + (u, v)$ is the graph created by adding the edge (u, v) to G . If G is an (n, m) -graph, then $G + (u, v)$ is an $(n, m + 1)$ -graph. In general, we use the notation $G + e$ to indicate the graph generated from G by adding some new edge to G .

The graph G must be connected, for otherwise an edge e could be added joining two nodes in different components, and the graph $G + e$ would be acyclic. Thus G is connected and acyclic, thus G is a tree.

We use the above theorem on trees to derive a result of importance to this work.

Lemma 9: Let G be an (n, m) -graph with $m < (n - 1)$. Then G is disconnected.

Proof: Let G be an (n, m) -graph with $m = (n - k)$, with $k \geq 2$. If G is connected, then there is a path between any two distinct vertices $u, v \in V(G)$. Select u and v as non-adjacent vertices and add the edge $e = (u, v)$. We now have an $(n, (n - k + 1))$ -graph that contains a cycle, beginning at u , going to v , and returning to u by the existing path. If $k > 2$, repeat the above step $(k - 2)$ times, noting only that the addition of new edges does not remove the first cycle created. We then have an $(n, (n - 1))$ -graph that is connected but contains a cycle. This contradicts Theorem 8 and thus we conclude that the original graph G could not have been a connected graph. The interested reader will find another proof of this lemma in the discussion of Theorem 1.3.1 in [R02]. It is important to note that $m < (n - 1)$ does not require that the graph have isolated vertices. The reader should examine the two $(4, 2)$ -graphs shown in Figure 8, only one of which has an isolated vertex. A graph with an isolated vertex must be disconnected, but there are very many disconnected graphs that have no isolated vertices.

For a connected graph G , we define $e(v)$, the **eccentricity** of vertex v , as the maximum of the distances from v to the other vertices in the graph. The **radius** of a connected graph G , denoted $\text{rad}(G)$, is the minimum value of the eccentricity of its vertices, while the **diameter** of the graph, denoted $\text{diam}(G)$, is the maximum value of the eccentricity of its vertices. Before we quote a familiar theorem relating the radius and diameter of a graph, we give an example.

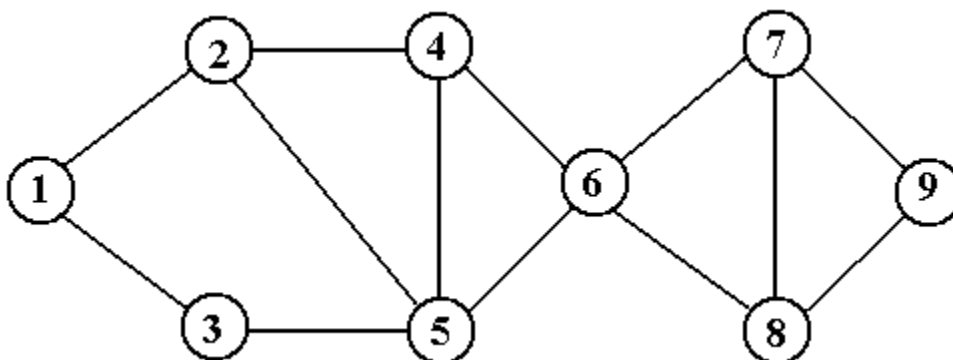


Figure 15: A Graph with Radius 3 and Diameter 5

In order to compute the radius and diameter of the graph, we first compute the eccentricity of each vertex. We construct the distance matrix for the graph.

Vertex	$d(v, 1)$	$d(v, 2)$	$d(v, 3)$	$d(v, 4)$	$d(v, 5)$	$d(v, 6)$	$d(v, 7)$	$d(v, 8)$	$d(v, 9)$	$e(v)$
1	0	1	1	2	2	3	4	4	5	5
2	1	0	2	1	1	2	3	3	4	4
3	1	2	0	2	1	2	3	3	4	4
4	2	1	2	0	1	1	2	2	3	3
5	2	1	1	1	0	1	2	2	3	3
6	3	2	2	1	1	0	1	1	2	3
7	4	3	3	2	2	1	0	1	1	4
8	4	3	3	2	2	1	1	0	1	4
9	5	4	4	3	3	2	1	1	0	5

Note that the maximum of the vertex eccentricities is 5; this is the diameter of the graph. The minimum of the vertex eccentricities is 3; this is the radius of the graph. A **central vertex** is a vertex with eccentricity equal to the radius of the graph. The **center of a graph**, denoted $Z(G)$, is the set of all central vertices; here $Z(G) = \{4, 5, 6\}$. Since the radius of the graph is defined to be the minimum of the eccentricities of the vertices, it should be obvious that there is at least one vertex of minimum eccentricity, and thus the center $Z(G)$ has at least one element.

Theorem 10: For every connected graph G , $\text{rad}(G) \leq \text{diam}(G) \leq 2 \cdot \text{rad}(G)$.

Proof: The inequality $\text{rad}(G) \leq \text{diam}(G)$ arises from the definition that the radius is the minimum of a set of numbers while the diameter is the maximum of the same set of numbers. In order to verify the second inequality, select vertices u and v in G such that $d(u, v) = \text{diam}(G)$. Let w be any vertex in $Z(G)$, the center of G . Then $d(u, w) \leq e(w)$ and $d(v, w) \leq e(w)$, where $e(w) = \text{rad}(G)$. It can be shown that $d(u, v) \leq d(u, w) + d(w, v)$ for any three vertices u, v , and w , so we have $d(u, v) \leq d(u, w) + d(w, v) = 2 \cdot e(w) = 2 \cdot \text{rad}(G)$.

An (n, m) -graph G is called **k -partite**, $1 < k \leq n$, if the vertices of G can be partitioned into k vertex sets V_1, V_2, \dots, V_k such that no two vertices in the same set are connected by an edge in G . The vertex sets are called **parts** of $V(G)$. For $k = 2$, we have a 2-partite graph, more commonly called a **bipartite graph**. A 3-partite graph is also called a **tripartite graph**.

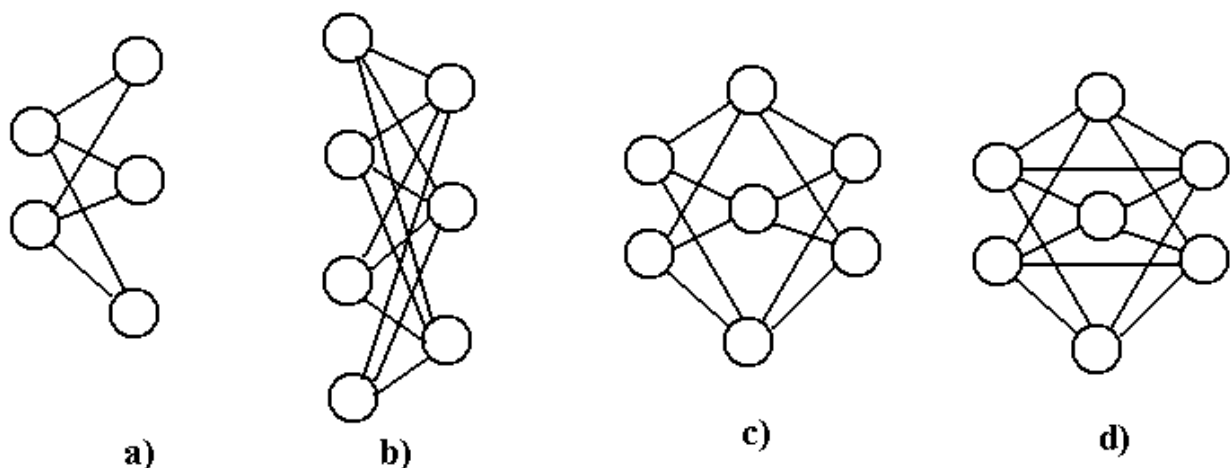


Figure 16: Some Bipartite Graphs and a Tripartite Graph

Fig 16a is $K_{2,3}$. Both fig 16b and 16c are $K_{3,4}$ – fig 16c is just drawn funny. Note that the vertices on the left and right side in fig 16c are not adjacent. Fig 16d is not a complete graph.

Before continuing, we note that any tree is also a bipartite graph. We show this fact by constructing the two vertex parts of the graph. Let T be a tree on n vertices and hence $(n - 1)$ edges. Select one vertex, call it u , and place it in the vertex part called V_1 . Place every vertex at distance 1 from u into vertex part V_2 , every vertex distance 2 from u into vertex part V_1 , and in general every vertex at odd distance from u into V_2 and at even distance from u into V_1 . Since Theorem 8 assures us that the path from any vertex to u is unique, we do not try to place any vertex into both V_1 and V_2 . We now show that no two vertices in a vertex part can be adjacent. Suppose that v and w are two vertices in a vertex part that are adjacent. We have paths from u to both v and w , thus creating the cycle from the path from u to w , the edge (w, v) and the path from v to u . But the tree T is acyclic, so that vertices in the same vertex part cannot be adjacent and T is bipartite. In a rooted tree, think of one vertex part as all vertices at an odd distance from the root vertex and the other vertex part as the rest of the vertices.

A graph G is called **complete** if every pair of its vertices is connected by an edge. By K_n we denote a complete graph on n vertices. C_n denotes the **cycle on n vertices**, and P_n denotes the **path on n vertices**. Note that for $n \geq 2$, K_n has $n \bullet (n - 1)/2$ edges, C_n has n edges and P_n has $(n - 1)$ edges. K_3 , which is isomorphic to C_3 , is called the **triangle graph**, or **triangle**. K_1 denotes the empty graph on one vertex, it is a graph with one isolated vertex and no edges.

G is called a **complete k -partite** graph if it is k -partite and whenever two vertices are in different parts of the graph they are connected by an edge in $E(G)$. A complete 2-partite graph is called **complete bipartite** and is denoted by $K_{a,b}$, where the number of vertices in the two vertex parts is a and b respectively. $K_{1,n-1}$ denotes a **star graph** on n vertices with $(n - 1)$ edges. $K_{1,n-1}$ is a complete bipartite graph; it is also a tree. Note that $K_{1,2}$ is isomorphic to P_3 . Note that there is only one connected graph on two vertices; it can be called either $K_{1,1}$ or K_2 or P_2 , but not C_2 as a cycle must have at least three vertices.

For an arbitrary graph G , nG denotes n copies of the graph G . Let nK_1 denote the empty graph on n vertices. Then we have $m(nK_1) = 0$; $nK_1 \in \Gamma(n,0)$. The maximum number of edges in a graph in $\Gamma(n)$ equals $m(K_n) = n \bullet (n - 1)/2$.

There are many ways to combine graphs to produce new graphs. We shall consider only one binary operator – the **union**. This is defined as follows.

Definition: The union of two graphs G_1 and G_2 , denoted $G = G_1 \cup G_2$, is that graph with $V(G) = V(G_1) \cup V(G_2)$ and $E(G) = E(G_1) \cup E(G_2)$.

Note that one can use this union operator as an alternate definition of the graph nG , based on a recursive definition: $2G = G \cup G$, and $nG = (n - 1)G \cup G$. One important graph for our consideration will be $G = K_{1,n-j} \cup (j - 1)K_1$. For $n = 6$ and $j = 3$, we have the graph in the following figure. The graph has two isolated vertices.

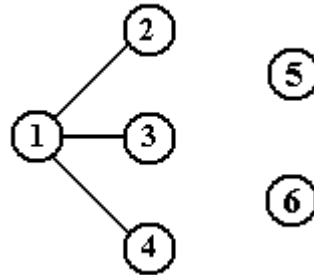


Figure 17: The (6, 3)-Graph $K_{1,3} \cup 2K_1$.

The **vertex connectivity** or simply **connectivity** of a connected graph G , denoted $\kappa(G)$ is the minimum number of vertices the removal of which from G yields either an isolated vertex or a disconnected graph. If $\kappa(G) \geq r$, then the graph G is said to be **r-connected**. The **edge connectivity** of a graph G , denoted $\lambda(G)$ is the minimum number of edges the removal of which results in a disconnected graph.

Let G be an (n, m) -graph with vertices v_1, v_2, \dots, v_n having degrees d_1, d_2, \dots, d_n , where $d_i = d(v_i)$. We label the vertices so that $d_1 \geq d_2 \geq \dots \geq d_n$ to get a sequence called the **degree sequence** of G , denoted by $D(G) = (d_1, d_2, \dots, d_n)$. By $\Delta(G)$ we denote the maximum degree in G , and by $\delta(G)$ we denote the minimum degree in G . If the degree sequence $D(G)$ is presented in the as above, then $\Delta(G) = d_1$ and $\delta(G) = d_n$. By **DSS(G)** we denote the sum of the squares of the vertex degrees of a graph G . In other words, for an (n, m) -graph G with degree sequence

given by $D(G) = (d_1, d_2, \dots, d_n)$, we have $DSS(G) = \sum_1^n (d_j)^2$.

We now consider two degree sequences, both for (n, m) -graphs and define a useful concept, called **degree sequence dominance**.

Definition: Let $D(G) = (d_1, d_2, \dots, d_n)$ and $D(H) = (d'_1, d'_2, \dots, d'_n)$ denote the degree sequences of (n, m) -graphs G and H , respectively. $D(G)$ is said to **dominate** $D(H)$ if $\sum_{i=1}^j d_i \geq \sum_{i=1}^j d'_i$ for all $j = 1, 2, \dots, n$ with strict inequality for at least one value of j .

The importance of degree sequence dominance arises from its relation to the sum of the squares of the degrees of the vertices, as seen in the following proposition.

Proposition 11: Let G and H be two (n, m) -graphs such that the degree sequence of G dominates the degree sequence of H . Then $DSS(G) > DSS(H)$.

Proof: Let $D(G) = (d_1, d_2, \dots, d_n)$ and $D(H) = (d'_1, d'_2, \dots, d'_n)$ denote the degree sequences of (n, m) -graphs G and H , respectively. Since the degree sequence of G dominates that of H , there must be at least one index k , $1 \leq k \leq n$, such that $d_k > d'_k$. Let k be the smallest index for which $d_k > d'_k$ and let $d_k = d'_k + d$, with $d > 0$.

Since the degree sequence is ordered, we have $d'_k \geq d'_j$ for all $j > k$. Since the two degree sequences add to the same sum, we must have

$$\sum_{i=k+1}^n d_i = \sum_{i=k+1}^n d'_i + d.$$

We alter the degree sequence of H by adding the value to d to d'_k , increasing $DSS(H)$ by $2d \bullet d'_k + d^2$. Decreasing d'_{k+1} to balance the sum then decreases the new value of $DSS(H)$ by $d^2 - 2d \bullet d'_{k+1}$, yielding a net change of $2d \bullet d'_k + 2d \bullet d'_{k+1}$, which is a positive number. Thus by modifying the degree sequence of H to make it look more like that of G , we have increased the value of $DSS(H)$. One can easily see that these changes to make the degree sequence of H identical to that of G continually increase $DSS(H)$. Thus we must have started with $DSS(H) < DSS(G)$.

A **sparse graph** is an (n, m) -graph for which $m \leq \lfloor n^2/4 \rfloor$, and a **dense graph** is an (n, m) -graph for which $m > \lfloor n^2/4 \rfloor$, where $\lfloor x \rfloor$ is the largest integer not greater than the real number x . $\lfloor n^2/4 \rfloor = (n^2/4)$ if and only if n is an even integer.

For graphs G and H , let $*H(G)$ denote the number of induced subgraphs of G which are isomorphic to H and $\#H(G)$ denote the number of (not necessarily induced) subgraphs of G which are isomorphic to H . Thus $\#P_3(G)$ and $*P_3(G)$ denote the number of subgraphs and the number of induced subgraphs, respectively, of G isomorphic to P_3 , the path on three vertices. $\#K_3(G)$ denotes the number of subgraphs isomorphic to K_3 , the triangle. Because all triangles as subgraphs are induced, $\#K_3(G) = *K_3(G)$; we use $*K_3(G)$ to denote the number of triangles in a graph G . Any graph for which $*K_3(G) = 0$ is said to be **triangle-free** or **K_3 -free**. We shall see that bipartite graphs are K_3 -free.

Recall that a **bipartite graph** is an (n, m) -graph G with the property that $V(G)$ can be broken into two disjoint sets V_1 and V_2 , such that a vertex in V_1 is adjacent only to vertices in V_2 and a vertex in V_2 is adjacent only to vertices in V_1 . We now prove one result of major importance to our work: that bipartite graphs do not contain a K_3 . We do this by first proving the more general result and then applying an obvious definition.

Theorem 12: A graph G is bipartite if and only if all of its cycles are even.

Proof: This proof is quite important, so is quoted almost verbatim from Theorem 1.3 in Distances in Graphs [R01]. If G is bipartite, then its vertex set V can be partitioned into two sets V_1 and V_2 so that every edge of G joins a vertex in V_1 with a vertex in V_2 . Thus every cycle [of length k] $v_1, v_2, \dots, v_k, v_1$ in G necessarily has its oddly subscripted vertices in V_1 , say, and the others in V_2 , and so its length is even. [Otherwise, we would have the edge (v_k, v_1) connecting two vertices in V_1 , contradicting our hypothesis.]

For the converse, we assume without loss of generality that G is connected (for otherwise we can consider the components of G separately). Take any vertex $v_1 \in V(G)$ and let [vertex set] V_1 consist of v_1 and all vertices at even distance from v_1 , while [vertex set] $V_2 = V - V_1$. Since all cycles of G are even, every edge of G joins a vertex of V_1 with a vertex of V_2 . For suppose there is an edge (u, v) joining two vertices of V_1 . Then the union of geodesics [shortest paths] from v_1 to v and from v_1 to u together with the edge (u, v) contains an odd cycle, a contradiction.

We now present the important result as a corollary to the above theorem.

Corollary 13: A bipartite graph does not contain a K_3 (triangle).

Proof: We have just shown that a bipartite graph does not contain any cycle of odd length. Specifically, it does not contain a C_3 (a cycle on three vertices), which is isomorphic to a K_3 (complete graph on three vertices).

In terms that we shall use later, we have just shown that if G is a bipartite graph then $*K_3(G) = \#K_3(G) = 0$.

We now link sparse and dense graph to graphs containing triangles by use of a famous theorem due to Turan. Turan's work is considered the first theorem in an important area of graph theory, called extremal graph theory, which we now discuss briefly.

Extremal Graph Theory

The study of extremal graphs is generally the study of the largest or smallest graphs that have certain properties. The best reference on the topic is the book Extremal Graph Theory by Bollobas [R03], a book that is rare and hard to find. The book contains references to many of the original papers in the subject; unfortunately many of them were written in Hungarian and have yet to be translated.

For these notes, we focus on extremal graph theory of complete subgraphs; that is subgraphs that are isomorphic to a complete graph K_n . We quote from chapter VI of Bollobas [R03] to introduce the subject.

Given a graph F_1 , what is $ex(n; F_1)$, the maximum number of edges of a graph of order n [having n vertices] not containing F_1 as a subgraph. ... [The] best known extremal result of graph theory [is] Turan's theorem. This result, proved in 1940 and always considered to be the first extremal theorem, answers this question above in the case $F_1 = K_r$.

Turan's theorem is based on specific complete q -partite graphs, denoted $T_q(n)$.

Definition: Given natural numbers n and q , denote by $T_q(n)$ the complete q -partite graph with $\lfloor n/q \rfloor, \lfloor (n+1)/q \rfloor, \dots, \lfloor (n+q-1)/q \rfloor$ vertices in each of the vertex sets. Note that $T_q(n)$ is the unique complete q -partite graph of order n whose vertex sets have size as equal as possible. For convenience, we number the vertex parts beginning with 0, so that vertex part k has size $\lfloor (n+k)/q \rfloor, 0 \leq k \leq (q-1)$.

It is a standard result that a q -partite graph of order n having n_0, n_1, \dots, n_{q-1} vertices in its vertex parts has at most $\binom{n}{2} - \sum_0^{q-1} \binom{n_k}{2}$ edges. $T_q(n)$ is the unique q -partite graph of order n , denoted by $t_q(n)$. Turan also proved in 1941 that every other graph of order n and size $t_{r-1}(n)$ contains a K_r as a subgraph.

For this research the most important Turan graph will be $T_2(n)$, the complete bipartite graph with vertex parts of size $\lfloor n/2 \rfloor$ and $\lfloor (n+1)/2 \rfloor$.

Theorem 14: Let r and n be natural numbers, $r \geq 2$. Then every graph of order n and size greater than $t_{r-1}(n)$ contains a K_r , a complete graph of order r . Furthermore, $T_{r-1}(n)$ is the only graph of order n and size $t_{r-1}(n)$ that does not contain a K_r .

Proof: See the proof of theorem 1.1 in chapter VI of reference [R03].

Of special interest to much research is the largest graph that contains no K_3 .

Lemma 15: The largest graph with n vertices that contains no triangle is the complete bipartite graph $K_{a,b}$, with $n = a + b$ and $|a - b| \leq 1$.

Proof: See Theorem 4.1.2 in the book Pearls in Graph Theory [R02]. This is also a special case of Theorem 14, just above.

Remark: The complete bipartite graph $K_{a,b}$ has $m = a \bullet b$ edges. If $a > b$, then we have two possibilities for graphs satisfying theorem 15: $a = b$ and $a = b + 1$. If $a = b$, then $n = 2 \bullet b$ and $m = b^2 = n^2/4$. If $a = b + 1$, then $n = 2 \bullet b + 1$ and $m = b \bullet (b + 1) = b^2 + b$. Also we have $n^2 = (2 \bullet b + 1)^2 = 4 \bullet b^2 + 4 \bullet b + 1$, so $m = \lfloor n^2/4 \rfloor$. The graph $K_{a,b}$, as described above is the largest sparse graph and we conclude that all dense graphs must contain triangles.

Corollary 16: If G is an acyclic graph, it must be a sparse graph.

Proof: If G is not a sparse graph, it is a dense graph that must contain a triangle or $K_3 \approx C_3$, which is a cycle. Hence G is not acyclic.

We add another interesting result that might be of use in later work.

Theorem 17: If $n \geq (r + 1)$ then every (n, m) -graph with $m = t_{r-1}(n) + 1$ contains a K_{r+1} from which an edge has been omitted.

Proof: See the proof of theorem 1.2 in Chapter VI of reference [R03].

Another Count of Subgraphs

Another important count is $S_3(G)$, the number of induced three-vertex connected subgraphs of G . P_3 , the path on three vertices, and K_3 , the triangle, are the only connected graphs on three vertices, so $S_3(G) = *P_3(G) + *K_3(G)$ for any graph G .

An (n, m) -graph is said to be $\#P_3$ -optimal if it maximizes $\#P_3(G)$ for all $G \in \Gamma(n, m)$, the set of all (n, m) -graphs. $*P_3$ -optimal and S_3 -optimal graphs are those graphs which maximize the counts $*P_3(G)$ and $S_3(G)$ respectively.

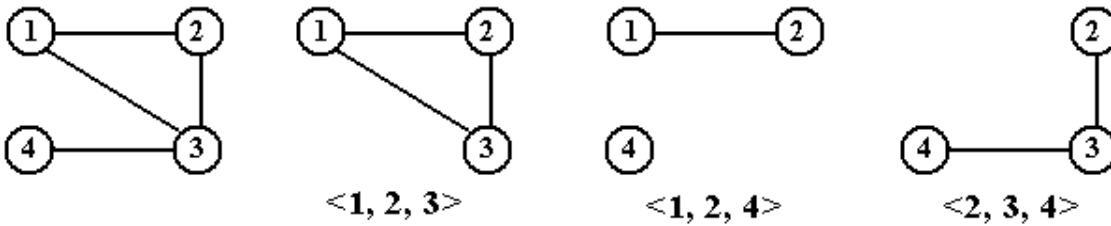


Figure 18: A Graph and Its 3-Vertex Induced Subgraphs

In the example above, we see a (4, 4)-graph and the subgraphs induced on the three distinct three-vertex subsets of $\{1, 2, 3, 4\}$. The subgraph $\langle 1, 2, 3 \rangle$ is a K_3 , which contains three non-induced P_3 's, one centered on each of its vertices. The graph $\langle 1, 2, 4 \rangle$ is $K_1 \cup K_2$, also called a K_1K_2 . The subgraph $\langle 2, 3, 4 \rangle$ is an induced P_3 .

As was mentioned above, there are three non-induced P_3 's in the above graph – one centered at vertex 1, one centered at vertex 2, and one centered at vertex 3. As a result we have one induced P_3 and three non-induced P_3 's, for a total of four. Thus, for this graph we have $*P_3(G) = 1$, $*K_3(G) = 1$, $S_3(G) = 2$, and $\#P_3(G) = 4$.

Proposition 18: For any graph G , $\#P_3(G) = *P_3(G) + 3 \cdot *K_3(G)$.

Proof: Let u, v , and w be the vertices of a triangle in G . There is a P_3 centered on each of the vertices u, v , and w . Since none of these is an induced P_3 , each triangle contributes 3 to the count $\#P_3(G)$ but 0 to the count $*P_3(G)$. The conclusion then follows by noting that each induced P_3 contributes 1 to the count of $\#P_3(G)$ and 1 to the count of $*P_3(G)$. In order to drive this point home, let's take another look at figure 18, presented above, focusing on the triangle induced by vertices 1, 2, and 3. Note that none of the P_3 's defined on these 3 vertices is induced, as each lacks one edge incident only on vertices in the set $\{1, 2, 3\}$.

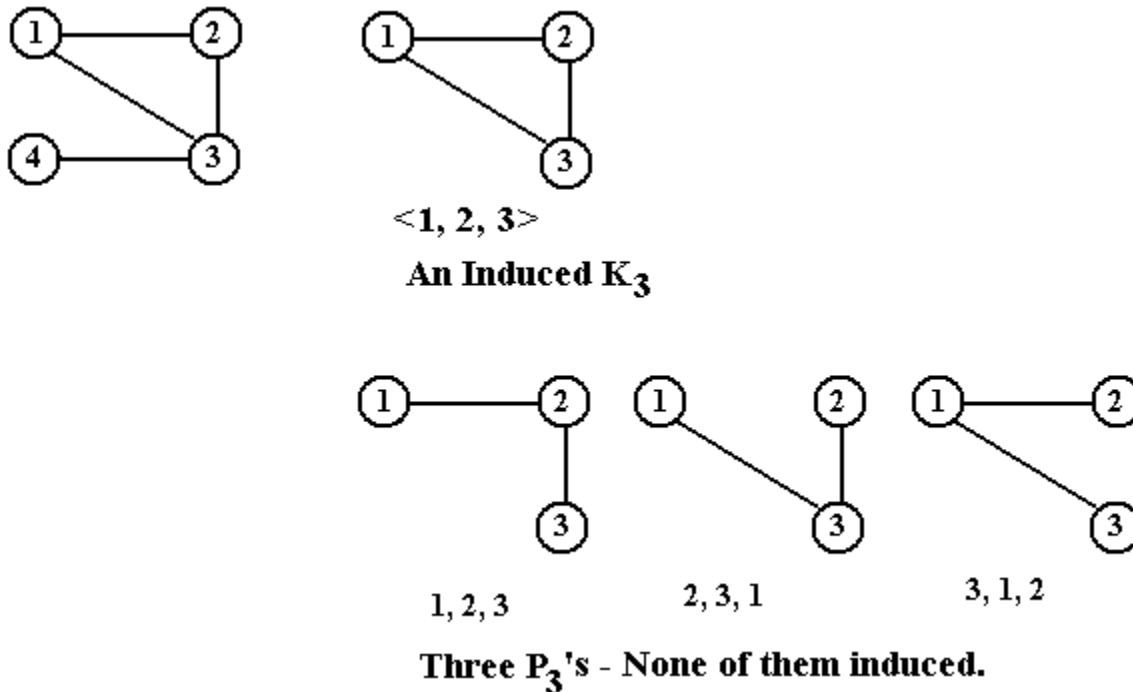


Figure 19: A Graph and Some of Its 3-Vertex Subgraphs

WEIGHTED GRAPHS

We now introduce the concept of a weighted graph – a graph in which there are weights associated with the edges. These weights can represent distances, costs, capacities, or any other measure that is associated with an edge and that can be quantified as a real number. For most weighted graphs, the weights are represented as non-negative integers, although negative edge weights appear to be used for some applications. To this author's knowledge, no work has been done on graphs with edge weights represented as complex numbers.

We begin with a formal definition of a weighted graph, and then move on to a more natural discussion of the concept in terms of drawings and adjacency matrices.

Definition: A *weighted graph* G is a triple (V, E, W) in which V is a non-empty set of vertices, $E \subseteq V \times V$ is a set of edges (the graph can be directed or undirected), and W is a function from the edge set E into \mathbf{R} , the set of real numbers. For any edge $e \in E$, $w(e)$ is the weight of e . In networks, the edge weights often represent the link transmission capacities.

We shall immediately revert to the standard practice of representing all edge weights as non-negative integers, most commonly using only positive integers. It can be proven that for most cases, this restriction does not present any difficulties. We shall begin with a simple undirected graph, in which all edges can be said to have a weight of one and then develop an example of the same graph with weighted edges. This example is taken, almost verbatim, from an excellent textbook [R04] by Sara Baase and Alan Van Gelder.

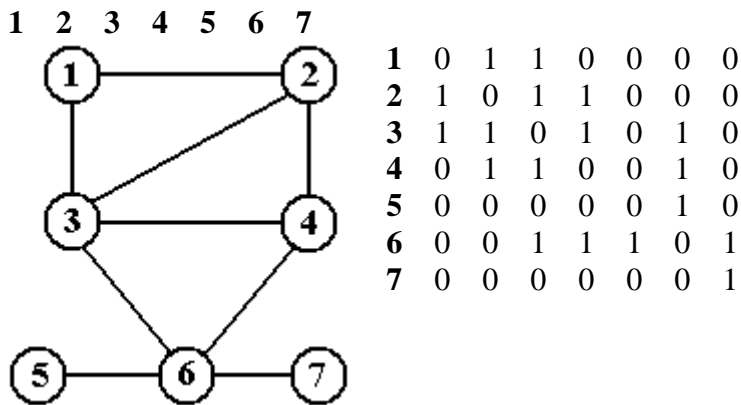


Figure 20: An Undirected Graph and Its Adjacency Matrix

We now add edge weights to this example, making it a weighted graph. Note that the only change to the adjacency matrix representation is to replace the 1 by the weight of the edge. Suppose that A is the adjacency matrix of a graph. We have two cases.

Unweighted Graph

$$A_{IJ} = 0 \text{ if no edge}$$

$$A_{IJ} = 1 \text{ if } (I, J) \text{ is an edge}$$

Weighted Graph

$$A_{IJ} = 0 \text{ if no edge}$$

$$A_{IJ} = \text{weight}(I, J) \text{ if } (I, J) \text{ is an edge}$$

We now present a weighted graph that has the same underlying undirected graph as the example in the figure above. Note that the adjacency matrix is different, it now has the weights.

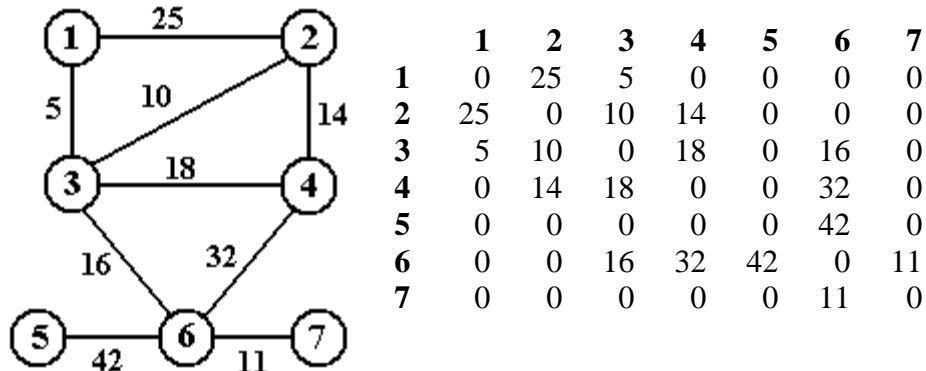


Figure 21: A Weighted Graph and Its Adjacency Matrix.

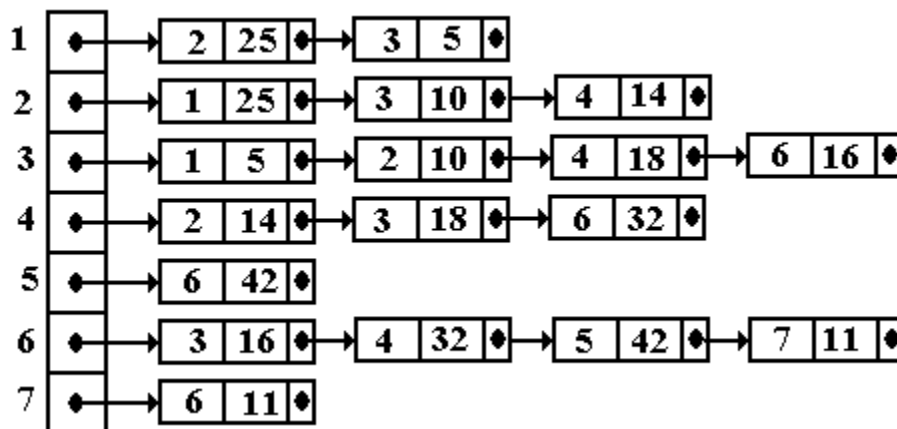


Figure 22: The Adjacency List Representation of the Same Weighted Graph

Note that the vertices in the list are kept in sorted order. This is a convention only and is not necessary. Ordered lists are easier to search, but take more time for insertion.

At this point we should note that the above adjacency matrix will cause some graph algorithms to malfunction. The problem arises when the edges represent distances or costs or some such quantity that one might want to minimize. The problem, which does not occur in the adjacency list representation, is due to the fact that a 0 is used to represent an edge that is not present. Consider a silly algorithm attempting to develop a minimum cost Hamiltonian circuit of the above graph. It might select $1 \rightarrow 4 \rightarrow 5 \rightarrow 3 \rightarrow 7 \rightarrow 2 \rightarrow 6 \rightarrow 1$ as the route with a total weight of 0. This is, of course, an impossible route as none of these edges exist.

It is easy to see that the problem does not occur when one uses the adjacency list representation of the graph. Edges that are not present simply do not have entries in the linked lists representing the open neighborhoods of each vertex. The problem is avoided.

It is also easy to see that the problem does not occur when one is using a graph to model some problem in which the edges represent flow capacities, available communication circuits, or some other measure to be maximized. In that case an edge that does not exist is identical to an edge of zero capacity; neither can be used to solve the problem.

When one is considering an adjacency matrix representation of a graph modeling a problem for which sums of edge weights are to be minimized, it is necessary to place a large value in the matrix elements that indicate non-existent edges between distinct vertices. Note that almost all algorithms will detect that a diagonal element $A[K][K]$ of a matrix is not to be used as the graph contains no loops, so only the entries for non-existent edges must be adjusted.

Many books suggest placing ∞ as an element in the adjacency matrix to represent the weight for the non-existent edges. This is great for drawings, but presents problems in the application of an algorithm, because most computers lack a consistent representation for ∞ . The approach commonly suggested is to take a very large number and use that. Here is another suggestion that will work for most algorithms.

- 1) Beginning with the adjacency matrix having 0's represent each non-existent edge, sum all the edge weights. The sum is twice the total of the edge weights, as every edge is summed twice.
- 2) Multiply that number by two and use that value to represent non-existent edges.

In the above example, the sum of the values of the adjacency matrix is 346, indicating a total edge weight of 173. We double the value of 346 to get 692 and use either that value or any larger value to represent a non-existent edge. The use of this number is based on the observation that no path through the graph will have a total distance greater than the sum of all the weights of all the edges, so here we use a number four times as big to keep the algorithms from picking any of these non-existent edges. The array below is the adjacency matrix using this approach.

0	25	5	692	692	692	692
25	0	10	14	692	692	692
5	10	0	18	692	16	692
692	14	18	0	692	32	692
692	692	692	692	0	42	692
692	692	16	32	42	0	11
692	692	692	692	692	11	0

Figure 23: The Adjusted Adjacency Matrix