CHAPTER 5

Basic Concepts in Graph Theory

Introduction

The concepts in this chapter are essential for understanding later discussions involving graphs, so be sure that you understand them. It is not necessary to memorize all the concepts since you can refer back to them if necessary; however, make sure that you understand them when you study them now so that referring back to them will simply be a memory refresher, not a new learning experience.

Since the basic concepts in Section 2.1 (p. 41) are used in this chapter, you may wish to review them before continuing.

5.1 What is a Graph?

There are various types of graphs, each with its own definition. Unfortunately, some people apply the term "graph" rather loosely, so you can't be sure what type of graph they're talking about unless you ask them. After you have finished this chapter, we expect you to use the terminology carefully, not loosely. To motivate the various definitions, we'll begin with some examples.

Example 5.1 A computer network Computers are often linked with one another so that they can interchange information. Given a collection of computers, we would like to describe this linkage in fairly clean terms so that we can answer questions such as "How can we send a message from computer A to computer B using the fewest possible intermediate computers?"

We could do this by making a list that consists of pairs of computers that are connected. Note that these pairs are unordered since, if computer C can communicate with computer D, then the reverse is also true. (There are sometimes exceptions to this, but they are rare and we will assume that our collection of computers does not have such an exception.) Also, note that we have implicitly assumed that the computers are distinguished from each other: It is insufficient to say that "A PC is connected to a Mac." We must specify which PC and which Mac. Thus, each computer has a unique identifying label of some sort.

For people who like pictures rather than lists, we can put dots on a piece of paper, one for each computer. We label each dot with a computer's identifying label and draw a curve connecting two

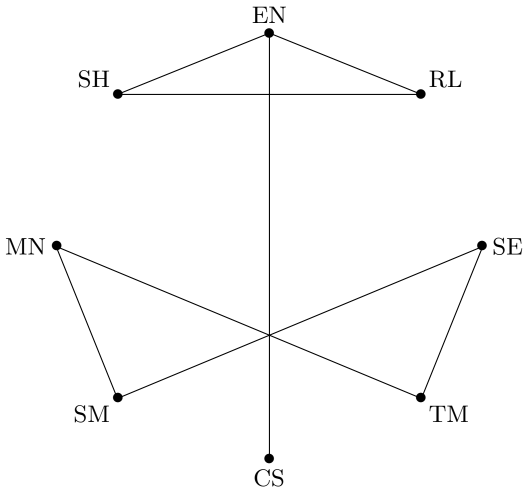


Figure 5.1 Computers connected by networks. Computers (vertices) are indicated by dots (•) with labels. The connections (edges) are indicated by lines. When lines cross, they should be thought of as cables that lie on top of each other—not as cables that are joined.

dots if and only if the corresponding computers are connected. Note that the shape of the curve does not matter (it could be a straight line or something more complicated) because we are only interested in whether two computers are connected or not. Figure 5.1 shows such a picture. Each computer has been labeled by the initials of its owner.

Recall that $\mathcal{P}_2(V)$ stands for the set of all two element subsets of the set V. Based on our computer example we have

> Definition 5.1 Simple graph *A* **simple graph** *G is a set V, called the* **vertices** *of G, and a subset E of $\mathcal{P}_2(V)$ (i.e., a set E of 2 element subsets of V), called the* **edges** *of G. We can represent this by writing $G = (V, E)$.*

In our case, the vertices are the computers and a pair of computers is in E if and only if they are connected. □

Example 5.2 Routes between cities Imagine four cities named, with characteristic mathematical charm, A, B, C and D. Between these cities there are various routes of travel, denoted by a, b, c, d, e, f and g. A picture of this situation is shown in Figure 5.2. Looking at it, we see that there are three routes between cities B and C. These routes are named d, e and f. Figure 5.2 is intended to give us a picture of only the interconnections between cities. It leaves out many aspects of the situation that might be of interest to a traveler. For example, the nature of these routes (rough road, freeway, rail, etc.) is not portrayed. Furthermore, unlike a typical map, no claim is made that the picture represents in any way the distances between the cities or their geographical placement relative to each other. The object shown in Figure 5.2 is called a *graph.* Edges a and b are called *parallel*, as are edges d, e and f.

Following our previous example, one is tempted to list the pairs of cities that are connected; in other words, to extract a simple graph from the information. Unfortunately, this does not describe the problem adequately because there can be more than one route connecting a pair of cities; e.g., d, e and f connecting cities B and C in the figure. How can we deal with this? Definition 5.2 is a precise definition of a graph of the type required to handle this type of problem.

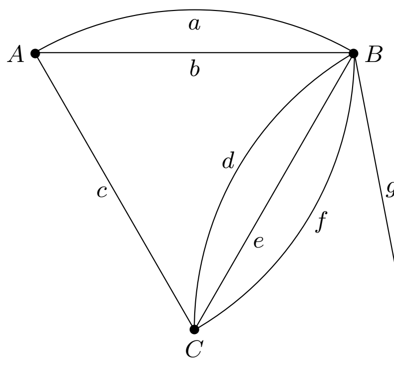


Figure 5.2 Capital letters indicate cities and lower case indicate routes.

Definition 5.2 Graph *A* **graph** *is a triple* $G = (V, E, \varphi)$ *where* V *and* E *are finite sets and* φ *is a function with range* $\mathcal{P}_2(V)$ *and with domain* E*. We call* E *the set of* **edges** *of the graph* G*. The set* V *is called the set of* **vertices** *of* G*.*

In Figure 5.2, $G = (V, E, \varphi)$ where

$$V = \{A, B, C, D\}, \quad E = \{a, b, c, d, e, f, g\}$$

and

$$\varphi = \begin{pmatrix} a & b & c & d & e & f & g \\ \{A,B\} & \{A,B\} & \{A,C\} & \{B,C\} & \{B,C\} & \{B,C\} & \{B,D\} \end{pmatrix}.$$

Definition 5.2. tells us that to specify a graph G it is necessary to specify the sets V and E and the function φ. We have just specified V and φ in *set theoretic* terms. Figure 5.2 specifies the same V and φ in pictorial terms. The set V is represented clearly in Figure 5.2 by dots (•), each of which has a city name adajcent to it. Similarly, the set E is also represented clearly. The function φ is determined from Figure 5.2 by comparing the name attached to a route with the two cities connected by that route. Thus, the route name d is attached to the route with endpoints B and C. This means that $\varphi(d) = \{B, C\}$.

Note that, since part of the definition of a function includes its range and domain, φ determines $\mathcal{P}_2(V)$ and E. Also, V can be determined from $\mathcal{P}_2(V)$. Consequently, we could have said that a graph is a function φ whose domain is a finite set and whose range is $\mathcal{P}_2(V)$ for some finite set V. Instead, we choose to specify V and E explicitly because the vertices and edges play a fundamental role in thinking about a graph G. □

The function φ is sometimes called the *incidence function* of the graph. The two elements of $\varphi(x) = \{u, v\}$, for any $x \in E$, are called the vertices of the edge x, and we say u and v are *joined* by x. We also say that u and v are *adjacent vertices* and that u is *adjacent to* v or , equivalently, v is adjacent to u. For any $v \in V$, if v is a vertex of an edge x then we say x is *incident* on v. Likewise, we say v is a member of x, v is on x, or v is in x. Of course, v is a member of x actually means v is a member of $\varphi(x)$.

Figure 5.3 shows two other pictorial ways of specifying the same graph as in Figure 5.2. The drawings look very different but exactly the same set V and function φ are specified in each case. It is *very important* that you understand exactly what information is needed to completely specify the graph. When thinking in terms of cities and routes between them, you naturally want the pictorial representation of the cities to represent their geographical positioning also. If the pictorial representation does this, that's fine, but it is not a part of the information required to define a graph. Geographical location is extra information. The geometrical positioning of the vertices A, B, C and D is very different in Figures 5.2 and 5.3(a). However, in each of these cases, the vertices on a given

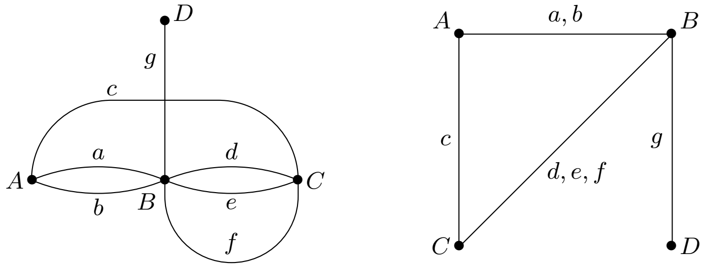


Figure 5.3 Two alternate pictorial specifications of Figure 5.2.

edge are the same and hence the graphs specified are the same. In Figure 5.3(b) a different method of specifying the graph is given. There, φ^{-1}, the inverse of φ, is given. For example, $\varphi^{-1}(\{C, B\})$ is shown to be $\{d, e, f\}$. Knowing φ^{-1} determines φ and hence determines G since the vertices A, B, C and D are also specified in Figure 5.3(b).

Warning *Some people call our "simple graph" a "graph" and some people call our "graph" a "multigraph." Still other people mean something somewhat different by the term multigraph!*

Example 5.3 Simple graphs are graphs We can easily reconcile our two definitions by realizing that a simple graph is a special case of a graph. Let $G = (V, E)$ be a simple graph. Define $\varphi : E \to E$ to be the identity map; i.e., $\varphi(e) = e$ for all $e \in E$. The graph $G' = (V, E, \varphi)$ is essentially the same as G. There is one subtle difference in the pictures: The edges of G are unlabeled but each edge of G' is labeled by a set consisting of the two vertices at its ends. □

There are still more concepts that can be called graphs. People may not be interested in which road is which in Figure 5.2, so the labels on the edges are not needed. On the other hand, some people might need more information, such as how long it takes to travel on each of the routes in Figure 5.2. There may be other situations, too; for example, some of the routes may be one way. We will meet some of these concepts later.

Exercises

5.1.1. Let (V, E, φ) be a graph and $v \in V$ a vertex. Define the *degree* of v, $d(v)$ to be the number of $e \in E$ such that $v \in \varphi(e)$; i.e., e is incident on v. Prove that $\sum_{v \in V} d(v) = 2|E|$, an even number. Conclude that the number of vertices v for which $d(v)$ is odd is even.

5.1.2. We are interested in the number of simple graphs with $V = \underline{n}$.

(a) Prove that there are $2^{\binom{n}{2}}$ such simple graphs. (That's 2 to the power $\binom{n}{2}$, not $2\binom{n}{2}$.)

(b) How many of them have exactly q edges?

(c) If we choose a simple n-vertex graph uniformly at random, what is the probability that it has exactly q edges?

5.1.3. Sometimes it is useful to allow an edge to have both its ends on the same vertex. Let $Q = (V, E, \varphi)$ be a graph where

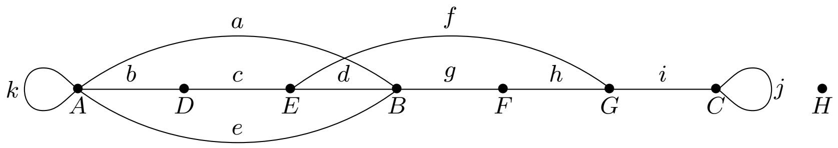
$$V = \{A, B, C, D, E, F, G, C, H\}, \quad E = \{a, b, c, d, e, f, g, h, i, j, d\}$$

and

$$\varphi = \begin{pmatrix} a & b & c & d & e & f & g & h & i & j & k \\ A & A & D & E & A & E & B & F & G & C & A \\ B & D & E & B & B & G & F & G & C & C & A \end{pmatrix}.$$

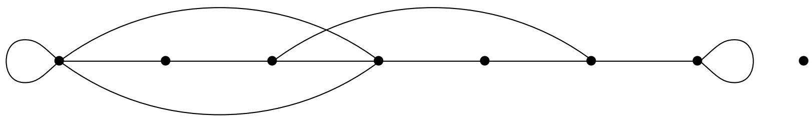
In this representation of φ, the first row specifies the edges and the the two vertices below each edge specify the vertices incident on that edge. Here is a pictorial representation $P(Q)$ of this graph.

$P'(Q)$:



Note that $\varphi(k) = \{A, A\} = \{A\}$. Such an edge is called a *loop*. Adding a loop to a vertex increases its degree by two. The vertex H, which does not belong to $\varphi(x)$ for any edge x (i.e., has no edge incident upon it), is called an *isolated* vertex. The degree of an isolated vertex is zero. Edges, such as a and e of Q, with the property that $\varphi(a) = \varphi(e)$ are called *parallel* edges. If all edge and vertex labels are removed from $P(Q)$ then we get the following picture $P'(Q)$:

$P'(Q)$:



The picture $P'(Q)$ represents the "form" of the graph just described and is sometimes referred to as a pictorial representation of the "unlabeled" graph associated with Q. For each of the following graphs R, where $R = (V, E, \varphi)$, $V = \{A, B, C, D, E, F, G, C, H\}$, draw a pictorial representation of R by starting with $P'(Q)$ removing and/or adding as few edges as possible and then labeling the resulting picture with the edges and vertices of R. A graph R which require no additions or removals of edges is said to be "of the same form as" or "isomorphic to" the graph Q.

(a) Let $E = \{a, b, c, d, e, f, g, h, i, j, k\}$ be the set of edges of R and

$$\varphi = \begin{pmatrix} a & b & c & d & e & f & g & h & i & j & k \\ C & C & F & A & H & E & E & A & D & A & A \\ C & G & G & H & H & H & F & H & G & D & F \end{pmatrix}.$$

(b) Let $E = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}$ be the set of edges of R and

$$\varphi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \\ A & E & E & E & F & G & H & B & C & D & E \\ G & H & E & F & G & H & B & C & D & D & H \end{pmatrix}.$$

5.1.4. Let $Q = (V, E, \varphi)$ be a graph with $|V| = n$. Let $d_1, d_2, \ldots, d_n$, where $d_1 \le d_2 \le \cdots \le d_n$ be the sequence of degrees of the vertices of Q, sorted by size. We refer to this sequence as the *degree sequence* of the graph Q. For example, if $Q = (V, E, \varphi)$ is the graph where

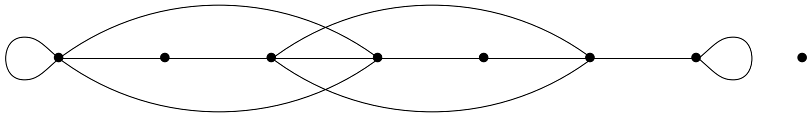
$$V = \{A, B, C, D, E, F, G, H\}, \quad E = \{a, b, c, d, e, f, g, h, i, j, k, l\}$$

and

$$\varphi = \begin{pmatrix} a & b & c & d & e & f & g & h & i & j & k & l \\ A & A & D & E & A & E & B & F & G & C & A & E \\ B & D & E & B & B & G & F & G & C & C & A & G \end{pmatrix}.$$

then $(0, 2, 2, 3, 4, 4, 4, 5)$ is the degree sequence of Q. Consider the the following unlabeled pictorial representation of Q

$P'(Q)$:



(a) Create a pictorial representation of Q by labeling $P'(Q)$ with the edges and vertices of Q.

(b) A necessary condition that a pictorial representation of a graph R can be created by labeling $P'(Q)$ with the vertices and edges of R is that the degree sequence of R be $(0, 2, 2, 3, 4, 4, 4, 5)$. True of false? Explain.

(c) A sufficient condition that a pictorial representation of a graph R can be created by labeling $P'(Q)$ with the vertices and edges of R is that the degree sequence of R be $(0, 2, 2, 3, 4, 4, 4, 5)$. True or false? Explain.

5.1.5. In each of the following problems information about the degree sequence of a graph is given. In each case, decide if a graph, *without loops*, satisfying the specified conditions exists or not. Give reasons in each case.

(a) A graph Q with degree sequence $(1, 1, 2, 3, 3, 5)$?

(b) A graph Q with degree sequence $(1, 2, 2, 3, 3, 5)$, multiple (i.e. parallel) edges allowed?

(c) A simple graph Q with degree sequence $(1, 2, 2, 3, 3, 5)$?

(d) A simple graph Q with degree sequence $(3, 3, 3, 3)$?

(e) A graph Q with degree sequence $(3, 3, 3, 3)$, no loops or parallel edges allowed?

(f) A graph Q with degree sequence $(3, 3, 3, 5)$, no loops or parallel edges allowed?

(g) A graph Q with degree sequence $(4, 4, 4, 4, 4)$, no loops or parallel edges allowed?

(h) A graph Q with degree sequence $(4, 4, 4, 4, 6)$, no loops or parallel edges allowed?

5.2 Equivalence Relations and Unlabeled Graphs

Sometimes we are interested only in the "structure" of a graph and not in the names (labels) of the vertices and edges. In this case we are interested in what is called an unlabeled graph. A picture of an unlabeled graph can be obtained from a picture of a graph by erasing all of the names on the vertices and edges. This concept is simple enough, but is difficult to use mathematically because the idea of a picture is not very precise.

The concept of an *equivalence relation* on a set is an important concept in mathematics and computer science. We used the idea in Section 4.3, but did not discuss it much there. We'll explore it more fully here and will use it to rigorously define unlabeled graphs. Later we will use it to define connected components and biconnected components. We recall the definition given in Section 4.3:

Definition 5.3 Equivalence relation *An* **equivalence relation** *on a set S is a partition of S. We say that $s, t \in S$ are* **equivalent** *if and only if they belong to the same block. If the symbol $\sim$ denotes the equivalence relation, then we write $s \sim t$ to indicate that s and t are equivalent.*

Example 5.4 To refresh your memory, we'll look at some simple equivalence relations.

Let S be any set and let all the blocks of the partition have one element. Two elements of S are equivalent if and only if they are the same. This rather trivial equivalence relation is, of course, denoted by "$=$".

Now let the set be the integers $\mathbb{Z}$. Let's try to define an equivalence relation by saying that n and k are equivalent if and only if they differ by a multiple of 24. Is this an equivalence relation? If it is we should be able to find the blocks of the partition. There are 24 of them, which we could number $0, \ldots, 23$. Block j consists of all integers which equal j plus a multiple of 24; that is, they have a remainder of j when divided by 24. Since two numbers belong to the same block if and only if they both have the same remainder when divided by 24, it follows that they belong to the same block if and only if their difference gives a remainder of 0 when divided by 24, which is the same as saying their difference is a multiple of 24. Thus this partition does indeed give the desired equivalence relation.

Now let the set be $\mathbb{Z} \times \mathbb{Z}^*$, where $\mathbb{Z}^*$ is the set of all integers except 0. Write $(a, b) \sim (c, d)$ if and only if $ad = bc$. With a moment's reflection, you should see that this is a way to check if the two fractions a/b and c/d are equal. We can label each equivalence class with the fraction a/b that it represents. In an axiomatic development of the rationals from the integers, one defines a rational number to be just such an equivalence class and proves that it is possible to add, subtract, multiply and divide equivalence classes. We won't pursue this. $\square$

In the next theorem we provide necessary and sufficient conditions for an equivalence relation. Verifying the conditions is a useful way to prove that some particular situation is an equivalence relation. Recall that a *binary relation* on a set S is a subset R of $S \times S$. Given a binary relation R, we will write $s \sim t$ if and only if $(s, t) \in R$. Thus "$\sim$" is another way to represent the binary relation.

Theorem 5.1 Equivalence relations *Let S be a set and suppose that we have a binary relation $\sim$ on S. This is an equivalence relation if and only if the following three conditions hold.*

(i) *(Reflexive) For all $s \in S$ we have $s \sim s$.*

(ii) *(Symmetric) For all $s, t \in S$ such that $s \sim t$ we have $t \sim s$.*

(iii) *(Transitive) For all $r, s, t \in S$ such that $r \sim s$ and $s \sim t$ we have $r \sim t$.*

Proof: We first prove that an equivalence relation satisfies (i)–(iii). Suppose that $\sim$ is an equivalence relation. Since s belongs to whatever block it is in, we have $s \sim s$. Since $s \sim t$ means that s and t belong to the same block, we have $s \sim t$ if and only if we have $t \sim s$. Now suppose that $r \sim s \sim t$. Then r and s are in the same block and s and t are in the same block. Thus r and t are in the same block and so $r \sim t$.

We now suppose that (i)–(iii) hold and prove that we have an equivalence relation. What would the blocks of the partition be? Everything equivalent to a given element should be in the same block. Thus, for each $s \in S$ let $B(s)$ be the set of all $t \in S$ such that $s \sim t$. We must show that the set of these sets form a partition of S.

In order to have a partition of S, we must have

(a) every $t \in S$ is in some $B(s)$ and

(b) for every $p, q \in S$, $B(p)$ and $B(q)$ are either equal or disjoint.

Since $\sim$ is reflexive, $s \in B(s)$, proving (a). Suppose $x \in B(p) \cap B(q)$ and $y \in B(p)$. We have, $p \sim x$, $q \sim x$ and $p \sim y$. Thus $q \sim x \sim p \sim y$ and so $y \in B(q)$, proving that $B(p) \subseteq B(q)$. Similarly $B(q) \subseteq B(p)$ and so $B(p) = B(q)$. This proves (b). $\square$

Suppose we have a picture of a graph $G = (V, E, \varphi)$, with the elements of V and E written next to the appropriate vertices and edges in the picture. Suppose that we have another graph $G' = (V', E', \varphi')$. We may be able to erase the elements of V and E and replace them with elements of V' and E', respectively, so that we obtain a picture of the graph G'. If this is possible, we will say that G and G' are isomorphic graphs. One can show that this relation (G is isomorphic to G') satisfies the conditions of Theorem 5.1 and so is an equivalence relation. All graphs which are isomorphic to a given graph correspond to the same unlabeled graph. The following definition and example formulate these ideas more precisely.

Definition 5.4 Graph isomorphism *Let $G = (V, E, \varphi)$ and $G' = (V', E', \varphi')$ be graphs. We say G and G' are* **isomorphic**, *written $G \sim G'$ if there are bijections*

$$\nu: V \to V' \quad \text{and} \quad \varepsilon: E \to E'$$

such that $\varphi'(\varepsilon(e)) = \nu(\varphi(e))$ for all $e \in E$, where $\nu(\{x, y\})$ is defined to be $\{\nu(x), \nu(y)\}$.

Let $G = (V, E)$ and $G' = (V', E')$ be simple graphs. They are isomorphic if there is a bijection $\nu: V \to V'$ such that

$$\{u, v\} \in E \quad \textit{if and only if} \quad \{\nu(u), \nu(v)\} \in E'.$$

Let's see what our definition means intuitively. Suppose we have a picture of a graph $G = (V, E, \varphi)$, with the elements of V and E written next to the appropriate vertices and edges in the picture. We can replace the vertex set by a new set V', writing the elements of V' where the elements of V were. The same thing can be done with the edges and a new set E'. This defines two functions $\nu: V \to V'$ and $\varepsilon: E \to E'$. The condition $\varphi'(\varepsilon(e)) = \nu(\varphi(e))$ says that we get φ' by simply looking at the picture to see what the ends of an edge are.

$$\begin{tikzcd} E_1 \arrow[r, shift left, "\varepsilon_1"] \arrow[d, "\varphi_1"] & E_2 \arrow[l, shift left, "\varepsilon_1^{-1}"] \arrow[d, "\varphi_2"] \\ \mathcal{P}_2(V_1) \arrow[r, shift left, "\nu_1"] & \mathcal{P}_2(V_2) \arrow[l, shift left, "\nu_1^{-1}"] \end{tikzcd} \qquad \begin{tikzcd} E_1 \arrow[r, "\varepsilon_1"] \arrow[d, "\varphi_1"] & E_2 \arrow[r, "\varepsilon_2"] \arrow[d, "\varphi_2"] & E_3 \arrow[d, "\varphi_3"] \\ \mathcal{P}_2(V_1) \arrow[r, "\nu_1"] & \mathcal{P}_2(V_2) \arrow[r, "\nu_2"] & \mathcal{P}_2(V_3) \end{tikzcd}$$

Figure 5.4 Functions involved in proving the equivalence of graphs. The left diagram is used for the reflexive law. The right diagram is used for the transitive law.

Example 5.5 Unlabeled graphs Let S be the set of all graphs and let $G = (V, E, \varphi)$ and $G' = (V', E', \varphi')$ be in S. We will prove that graph isomorphism is an equivalence relation on S.

Before we give our proof, let's look at how an equivalence class can be interpreted. Draw a picture of G and then erase the names of the vertices and edges. We could call what is left the "structure" of G or, as is more commonly done, an *unlabeled graph.* The use of an equivalence relation to define an unlabeled graph may seem a bit round-about: Why not just be satisfied with the picture? Thinking about the picture is fine for thinking about unlabeled graphs and for the proofs we'll write; however, there are reasons for presenting this definition:

- If we wanted to use the picture approach in a more formal way, we'd have to say precisely what a picture of a graph was and when two pictures represented the same graph. We faced this problem a bit in the previous section.
- Equivalence relations arise from isomorphisms in many areas of mathematics and there is often no picture that one can use to describe the equivalence relation. Thus, seeing the formalism for graphs is good preparation for seeing it in other courses.
- If we wanted a theorem-proving computer program to work with unlabeled graphs, we'd need to give it a formal definition. (It would be much simpler to work with labeled graphs.) This is an example of how an intuitively simple concept—unlabeled graphs in this case—can be very difficult to express in terms that computer software can deal with.

We now give a formal proof that we have an equivalence relation.

First: $G \sim G$ because we can take ν and ε to be the identity functions; i.e., $\nu(x) = x$ for all $x \in V$.

Second: Given that $G_1 \sim G_2$, we must prove $G_2 \sim G_1$, the reflexive law. Let ν_1 and ε_1 be the bijections guaranteed by the definition of $G_1 \sim G_2$. Then $\nu_1^{-1}\colon V_2 \to V_1$ and $\varepsilon_1^{-1}\colon E_2 \to E_1$ are bijections. Let $e_2 \in E_2$ and $e_1 = \varepsilon_1^{-1}(e_2)$. Then, by $\varphi_2(\varepsilon_1(e_1)) = \nu_1(\varphi_1(e_1))$, we have $\varphi_1(e_1) = \nu_1^{-1}(\varphi_2(\varepsilon_1(e_1)))$ and so

$$\varphi_1(\varepsilon_1^{-1}(e_2)) \;=\; \varphi_1(e_1) \;=\; \nu_1^{-1}(\varphi_2(\varepsilon_1(e_1))) \;=\; \nu_1^{-1}(\varphi_2(e_2)).$$

Thus $G_2 \sim G_1$.

Figure 5.4 may help you follow what we've just done: The definition of $G_1 \sim G_2$ says that starting at $e \in E_1$, going to E_2 via ε_1 and then to $\mathcal{P}_2(V_2)$ via φ_2 ends up at the same pair of vertices as going from e to $\mathcal{P}_2(V_1)$ and then to $\mathcal{P}_2(V_2)$ via φ_1 and ν_1, respectively. In other words, the two routes from E_1 to $\mathcal{P}_2(V_2)$ end up in the same place. We proved that the two routes from E_2 to $\mathcal{P}_2(V_1)$ must then also end up in the same place.

Third: We must prove the transitive law. Suppose that $G_1 \sim G_2 \sim G_3$. We then have the bijections $\nu_i\colon V_i \to V_{i+1}$ and $\varepsilon_i\colon E_i \to E_{i+1}$ for $i = 1, 2$. Furthermore, $\varphi_{i+1}(\varepsilon_i(e_i)) = \nu_i(\varphi_i(e_i))$ for all $e_i \in E_i$. Let $\nu(v_1) = \nu_2(\nu_1(v_1))$ and $\varepsilon(e_1) = \varepsilon_2(\varepsilon_1(e_1))$. It is easily verified that ν and ε are bijections since ν_i and ε_i are. Finally, since $\varepsilon_1(e_1) \in E_2$,

$$\varphi_3(\varepsilon(e_1)) \;=\; \varphi_3(\varepsilon_2(\varepsilon_1(e_1))) \;=\; \nu_2(\varphi_2(\varepsilon_1(e_1))) \;=\; \nu_2(\nu_1(\varphi_1(e_1))) \;=\; \nu(\varphi_1(e_1)).$$

Thus $G_1 \sim G_3$. $\square$

Exercises

5.2.1. Suppose that $G = (V, E, \varphi)$ and $G' = (V', E', \varphi')$ are equivalent; i.e., give the same unlabeled graph. Prove the following.

(a) $|V| = |V'|$ and $|E| = |E'|$.

(b) $d(k, G) = d(k, G')$ for all k, where $d(k, H)$ is the number of vertices of degree k in H.

5.2.2. From our discussion, it may seem to be an easy matter to decide if two graphs represent the same unlabeled graph. This is not true even for relatively small graphs. Divide the following graphs into equivalence classes and justify your answer; i.e., explain why you have the classes that you do. In all cases $V = \underline{4}$.

(a) $\varphi = \begin{pmatrix} a & b & c & d & e & f \\ \{1,2\} & \{1,2\} & \{2,3\} & \{3,4\} & \{1,4\} & \{2,4\} \end{pmatrix}$

(b) $\varphi = \begin{pmatrix} A & B & C & D & E & F \\ \{1,2\} & \{1,4\} & \{1,4\} & \{1,2\} & \{2,3\} & \{3,4\} \end{pmatrix}$

(c) $\varphi = \begin{pmatrix} u & v & w & x & y & z \\ \{2,3\} & \{1,3\} & \{3,4\} & \{1,4\} & \{1,2\} & \{1,2\} \end{pmatrix}$

(d) $\varphi = \begin{pmatrix} P & Q & R & S & T & U \\ \{3,4\} & \{2,4\} & \{1,3\} & \{3,4\} & \{1,2\} & \{1,2\} \end{pmatrix}$

5.2.3. Let $M(n, \mathbb{R})$ be the $n \times n$ matrices over the real numbers.

(a) For two matrices $A, B \in M(n, \mathbb{R})$, write $A \simeq B$ if and only if there is some nonsingular $P \in M(n, \mathbb{R})$ such that $B = PAP^{-1}$. Prove that this is an equivalence relation.

(b) For two matrices $A, B \in M(n, \mathbb{R})$, write $A \sim B$ if and only if there is some nonsingular $P \in M(n, \mathbb{R})$ such that $B = PAP^t$, where P^t is the transpose of P. Prove that this is an equivalence relation.

5.2.4. Which of the following define equivalence relations? If an equivalence relation is not defined, why not?

(a) For all s and t, $s \sim t$.

(b) Among the students at a university who have selected precisely one major, two are equivalent if and only if they have the same major.

(c) Among the students at a university, two students are equivalent if they have a class in common.

(d) For the real numbers, two numbers are equivalent if they differ by less that 0.001.

(e) For the real numbers, two numbers are equivalent if they agree in their decimal expansions through the third digit after the decimal place.

*5.2.5. Define the concept of a equivalence relation for simple graphs so that you can introduce the notion of an unlabeled simple graph. Prove that you have, indeed, defined an equivalence relation.
Hint. You need only introduce ν, not ε.

*5.2.6. We say that an ordinary function $f \in B^A$ has its domain A and its range B *labeled*. For convenience,we let $A = \{1, \ldots, a\}$ and $B = \{1, \ldots, b\}$.

(a) Suppose that $f, g \in B^A$. Write $f \sim g$ if there is a permutation π on A such that $f(x) = g(\pi(x))$ for all $x \in A$. Prove that this is an equivalence relation. We call the equivalence class a function with unlabeled domain. Prove that the set of nondecreasing functions from A to B is a system of representatives for these equivalence classes; that is, this set contains exactly one function from each equivalence class.

(b) Using the idea in (a), define the notion of a function with unlabeled range and prove that you have an equivalence relation. Call a function $f\colon A \to B$ a "restricted growth function" if $f(1) = 1$ and, for $a > j \geq 1$, $f(j+1)$ is at most one larger than the maximum of $f(1)$, $f(2)$, $\ldots$, $f(j)$. Prove that the restricted growth functions form a system of representatives for the equivalence classes you have defined.

(c) Using the previous ideas, define the notion of a function with unlabeled domain and range and prove that you have an equivalence relation. Call a function $f\colon A \to B$ a "partition function" if $f(i) \leq f(i+1)$ for $a > i \geq 1$ and $|f^{(-1)}(j)| \geq |f^{(-1)}(j+1)|$ for $b > j \geq 1$. Prove that the partition functions give one representative from each equivalence class

*5.2.7. This problem uses the ideas and notation from the previous problem. Construct a table with four rows marked with the four possibilities "A (un)labeled and B (un)labeled" and with the columns marked with "all," "injections" and "surjections." Each of the twelve positions is to be interpreted as the number of (equivalence classes of) functions in B^A satisfying the conditions. We use $|A| = a$ and $|B| = b$ and use U and L to indicate labeled and unlabeled. The start of a table is shown below. Verify these entries and complete the table. How can the number of (equivalence classes of) bijections be found from the table?

A	B	all	injections	surjections
L	L	?	$b(b-1)\cdots(b-a+1)$	?
L	U	$\sum_{k\leq b} S(a,k)$	?	?
U	L	?	?	$\binom{a-1}{a-b}$
U	U	?	1	?

5.3 Paths and Subgraphs

An important concept for describing the structure of a graph is the concept of a path.

Definition 5.5 Path, trail, walk and vertex sequence *Let $G = (V, E, \varphi)$ be a graph. Let $e_1, e_2, \ldots, e_{n-1}$ be a sequence of elements of E (edges of G) for which there is a sequence $a_1, a_2, \ldots, a_n$ of distinct elements of V (vertices of G) such that $\varphi(e_i) = \{a_i, a_{i+1}\}$ for $i = 1, 2, \ldots, n-1$. The sequence of edges $e_1, e_2, \ldots, e_{n-1}$ is called a* **path** *in G. The sequence of vertices $a_1, a_2, \ldots, a_n$ is called the* **vertex sequence** *of the path. (Note that since the vertices are distinct, so are the edges.) If we require that $e_1, \ldots, e_{n-1}$ be distinct, but not that $a_1, \ldots, a_n$ be distinct, the sequence of edges is called a* **trail**. *If we do not even require that the edges be distinct, it is called a* **walk**.

Note that the definition of a path requires that it not intersect itself (i.e., have repeated vertices), while a trail may intersect itself. Although a trail may intersect itself, it may not have repeated edges, but a walk may. If $P = (e_1, \ldots, e_{n-1})$ is a path in $G = (V, E, \varphi)$ with vertex sequence $a_1, \ldots, a_n$ then we say that P is a *path from a_1 to a_n*. Similarly for a trail or a walk.

In the graph of Figure 5.2 (p. 123), the sequence c, d, g is a path with vertex sequence A, C, B, D. If the graph is of the form $G = (V, E)$ with $E \subseteq \mathcal{P}_2(V)$, then the vertex sequence alone specifies the sequence of edges and hence the path. Thus, in Figure 5.1 (p. 122), the vertex sequence MN, SM, SE, TM specifies the path {MN, SM}, {SM, SE}, {SE, TM}.

Note that every path is a trail and every trail is a walk, but not conversely. However, we can show that, if there is a walk between two vertices, then there is a path. This rather obvious result can be useful in proving theorems, so we state it as a theorem.

Theorem 5.2 *Suppose $u \neq v$ are vertices in $G = (V, E, \varphi)$. The following are equivalent:*

(a) *There is a walk from u to v.*

(b) *There is a trail from u to v.*

(c) *There is a path from u to v.*

Furthermore, given a walk from u to v, there is a path from u to v all of whose edges are in the walk.

Proof: Since every path is a trail, (c) implies (b). Since every trail is a walk, (b) implies (a). Thus it suffices to prove that (a) implies (c). Let $e_1, e_2, \ldots, e_k$ be a walk from u to v. We use induction on n, the number of repeated vertices in a walk. If the walk has no repeated vertices, it is a path. This starts the induction at $n = 0$. Suppose $n > 0$. If u and or v is repeated, take the part of the walk that starts at the last occurrence of u and ends at the first occurrence of v, since this walk has less than n repeated vertices, there is a path. Let r be a repeated vertex different from u and v. Suppose it first appears in edge e_i and last appears in edge e_j. Then $e_1, \ldots, e_i, e_j, \ldots, e_k$ is a walk from u to v in which r is not a repeated vertex. Hence there are less than n repeated vertices in this walk from u to v and so there is a path by induction. Since we constructed the path by removing edges from the walk, the last statement in the theorem follows. □

Another basic notion is that of a subgraph of $G = (V, E, \varphi)$, which we will soon define. First we need some terminology about functions. By a *restriction* φ' of φ to $E' \subseteq E$, we mean the function φ' with domain E' and satisfying $\varphi'(x) = \varphi(x)$ for all $x \in E'$.

Definition 5.6 Subgraph *Let $G = (V, E, \varphi)$ be a graph. A graph $G' = (V', E', \varphi')$ is a* **subgraph** *of G if $V' \subseteq V$, $\varphi(e') \in \mathcal{P}_2(V')$ for all $e' \in E'$, and φ' is the restriction of φ to E' having range $\mathcal{P}_2(V')$.*

The fact that G' is itself a graph means that $\varphi(x) \in \mathcal{P}_2(V')$ for each $x \in E'$.

Example 5.6 For the graph $G = (V, E, \varphi)$ of Figure 5.2 (p. 123), Let $G' = (V', E', \varphi')$ be defined by $V' = \{A, B, C\}$, $E' = \{a, b, c, f\}$, and by φ' being the restriction of φ to E' with range $\mathcal{P}_2(V')$. Notice that φ' is determined completely from knowing V', E' and φ. Thus, to specify a subgraph G', the key information is V' and E'.

As another example from Figure 5.2, we let $V' = V$ and $E' = \{a, b, c, f\}$. In this case, the vertex D is not a member of any edge of the subgraph. Such a vertex is called an *isolated vertex* of G'. □

One way of specifying a subgraph is to give a set of edges $E' \subseteq E$ and take V' to be the set of all vertices on some edge of E'. In other words , V' is the union of the sets $\varphi(x)$ over all $x \in E'$. Such a subgraph is called the *subgraph induced by* E'. The first of Examples 5.6 is the subgraph induced by $E' = \{a, b, c, f\}$. Likewise, given a set $V' \subseteq V$, we can take E' to be the set of all edges $x \in E$ such that $\varphi(x) \subseteq V'$. The resulting subgraph is called the *subgraph induced by* V'. Referring to Figure 5.2 (p. 123), the edges of the subgraph induced by $V' = \{C, B\}$, are $E' = \{d, e, f\}$.

Look again at Figure 5.2. In particular, consider the path c, a with vertex sequence C, A, B. Notice that the edge d has $\varphi(d) = \{C, B\}$. The subgraph $G' = (V', E', \varphi')$, where $V' = \{C, A, B\}$ and $E' = \{c, a, d\}$ is called a *cycle* of G. In general, whenever there is a path in G, say $e_1, \ldots, e_{n-1}$ with vertex sequence $a_1, \ldots, a_n$, and an edge x with $f(x) = \{a_1, a_n\}$, then the subgraph induced by the edges $e_1, \ldots, e_{n-1}, x$ is called a *cycle* of G. The formal definition is:

Definition 5.7 Cycle *Let $G = (V, E, \varphi)$ be a graph and let $e_1, \ldots, e_{n-1}$ be a path with vertex sequence $a_1, \ldots, a_n$. If x is an edge of G such that $\varphi(x) = \{a_1, a_n\}$, then the subgraph G' of G induced by the set of edges $\{e_1, \ldots, e_{n-1}, x\}$ is called a* **cycle** *of G. The* **length** *of the cycle is n.*

In our definitions, a path is a *sequence* of edges but a cycle is a *subgraph* of G. In actual practice, we will not need to make such fine distinctions, so we may think of a cycle as a path, except that it starts and ends at the same vertex. Cycles are closely related to the existence of unique paths between vertices:

Theorem 5.3 *Two vertices $u \neq v$ are on a cycle of G if and only if there are two paths from u to v that have no vertices in common except the endpoints u and v.*

Proof: Suppose u and v are on a cycle. Follow the cycle in some direction from u to v to obtain one path. Then follow the cycle in the opposite direction from u to v to obtain another. Since a cycle has no repeated vertices, the only vertices that lie in both paths are u and v. On the other hand, a path from u to v followed by a path from v to u is a cycle if the paths have no vertices in common other than u and v. $\square$

Definition 5.8 Connected graph *Let $G = (V, E, \varphi)$ be a graph. If for any two distinct elements u and v of V there is a path P from u to v then G is a* **connected graph**. *If $|V| = 1$, then G is connected.*

We make two observations about the definition.

- Because of Theorem 5.2, we can replace "path" in the definition by "walk" or "trail" if we wish.
- The last sentence in the definition is not really needed. To see this, suppose $|V| = 1$. Now G is connected if, for any two *distinct* elements u and v of V, there is a path from u to v. This is trivially satisfied since we cannot find two distinct elements in the one element set V.

The graph of Figure 5.1 (p. 122) is not connected. (There is no path from EN to TM, for example.) The subgraph of this graph induced by the edges {{SH, EN}, {EN, RL}, {EN, CS}} is a connected graph with no cycles. Notice in Figure 5.1, that the relation defined on pairs of vertices u, v by "there exists a path from u to v" partitions the vertices into two subsets: $V_1 =$ {EN, SH, RL, CS} and $V_2 =$ {MN, SM, SE, TM}. Any two vertices in V_1 can be joined by a path and the same is true for any two vertices in V_2. There is no path connecting a vertex in V_1 to a vertex in V_2.

This is the case in general for a graph $G = (V, E, \varphi)$: The vertex set is partitioned into subsets $V_1, V_2, \ldots, V_m$ such that if u and v are in the same subset then there is a path from u to v and if they are in different subsets there is no such path. The subgraphs $G_1 = (V_1, E_1, \varphi_1), \ldots, G_m = (V_m, E_m, \varphi_m)$ induced by the sets $V_1, \ldots, V_m$ are called the *connected components* of G. Every edge of G appears in one of the connected components. To see this, suppose that $\{u, v\}$ is an edge and note that the edge is a path from u to v and so u and v are in the same induced subgraph, G_i. By the definition of induced subgraph, $\{u, v\}$ is in G_i.

***Example 5.7 Connected components as an equivalence relation** If you've read Section 2, you may have realized that the definition of connected components is a bit sloppy: We need to know that the partitioning into such subsets can actually occur. To see that this is not trivially obvious, define two integers to be "connected" if they have a common factor. Thus 2 and 6 are connected and 3 and 6 are connected, but 2 and 3 are not connected and so we cannot partition the set $V = \{2, 3, 6\}$ into "connected components". We must use some property of the definition of graphs and paths to show that the partitioning of vertices is possible. One way to do this is to construct an equivalence relation.

For $u, v \in V$, write $u \sim v$ if and only if either $u = v$ or there is a walk from u to v. We will use Theorem 5.1 (p. 127) to prove that this is an equivalence relation. It is clear that $\sim$ is reflexive and symmetric. We now prove that it is transitive. Let $u \sim v \sim w$. The walk from u to v followed by the walk from v to w is a walk from u to w. This completes the proof that $u \sim v$ is an equivalence relation. The relation partitions V into subsets $V_1, \ldots, V_m$. □

Exercises

5.3.1. Let C be the set of courses at the university and S the set of students. Let $V = C \cup S$ and let $\{s, c\} \in E$ if and only if student s is enrolled in course c.

(a) Prove that $G = (V, E)$ is a simple graph.

(b) Prove that every cycle of G has an even number of edges.

5.3.2. A graph $G = (V, E)$ is called *bipartite* if V can be partitioned into two sets A and B such that each edge has one vertex in A and one vertex in B. (A partition means that $A \cup B = V$ and $A \cap B = \emptyset$.)

(a) Prove that the example in Exercise 5.3.1 is a bipartite graph.

(b) Prove that every cycle in a bipartite graph has even length.

(c) Suppose that G is a connected bipartite graph. Develop an algorithm to partition the vertices of G into sets A and B such that each edge has one vertex in A and one vertex in B. Prove that your algorithm is correct.

(d) Extend your algorithm to all bipartite graphs.

(e) Prove that the number of ways to choose A and B in a bipartite graph with k connected components is 2^k.

*(f) Prove that a graph is bipartite if and only if every cycle in the graph has even length.

*5.3.3. A *cut edge* or *isthmus* of a connected graph $G = (V, E, \varphi)$ is an edge e such that the removal of e from G leaves a graph which is not connected. A *cut vertex* or *articulation point* of G is a vertex v such that the subgraph induced by $V - \{v\}$ is not connected. For example, in Figure 5.2, edge g is an isthmus and vertex B is a cut vertex. For this problem, assume that G is simple.

(a) If e is a cut edge of G and $v \in \varphi(e)$ is also on another edge $f \neq e$, prove that v is a cut vertex of G.

(b) Give an example of a connected graph that has a cut vertex but does not have an isthmus.

(c) Prove that an edge e of G is a cut edge if and only if it does not lie on a cycle.
Hint. Look for a path that does not contain e but connects the two vertices in $\varphi(e)$.

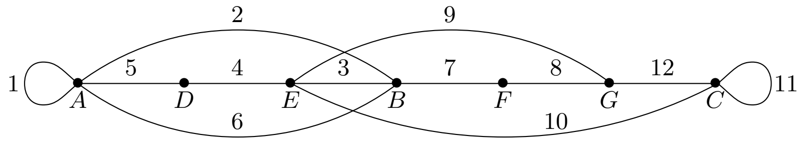
(d) (*Challenge*) Formulate and prove a result similar to the previous one for cut vertices.

5.3.4. A *circuit* (or "closed trail") in a graph $G = (V, E, \varphi)$ is defined exactly as is a cycle in Definition 5.7 except that the "path with vertex sequence $a_1, \ldots, a_n$" is replaced by a "trail with vertex sequence $a_1, \ldots, a_n$." In the next section, we'll define a tree as a connected graph without cycles. Suppose that in this definition and in Definition 5.8 "path"is replaced by "trail" and "cycle" is replaced by "circuit." Would the new definitions of *tree* and *connected graph* describe the same structures as the old definition? Explain.

5.3.5. We are going to describe a process for constructing a graph $G = (V, E, \varphi)$ (with loops allowed). Start with $V = \{v_1\}$ consisting of a single vertex and with $E = \emptyset$. Add an edge e_1, with $\varphi(e_1) = \{v_1, v_2\}$, to E. If $v_1 = v_2$, we have a graph with one vertex and one edge, else we have a graph with two vertices and one edge. Keep track of the vertices and edges in the order added. Here (v_1, v_2) is the sequence of vertices in the order added and the (e_1) is the sequence of edges in order added. Suppose we continue this process to construct a sequence of vertices (not necessarily distinct) added and sequence of *distinct* edges added. At the point where k distinct edges have been added, if v is the last vertex added, then we add a new edge e_{k+1}, different from all previous edges, with $\varphi(e_{k+1}) = \{v, v'\}$ where either v' is a vertex already added or a new vertex. Here is a picture of this process carried out with the edges numbered in the order added, where the vertex sequence is

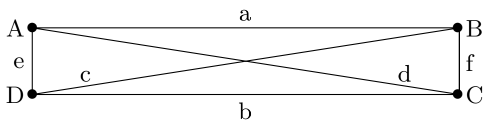
$$(A, A, B, E, D, A, B, F, G, E, C, C, G)$$

$P'(Q)$:

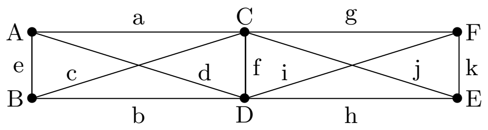


Such a graph is called *Eulerian* or a "graph with an Eulerian trail." By construction, if G is a graph with an Eulerian trail, then there is a trail in G that includes every edge in G. If there is a circuit in G that includes every edge of G then G is called an Eulerian circuit graph or graph with an Eulerian circuit. Thinking about the above example, if a graph has an Eulerian trail but no Eulerian circuit, then all vertices of the graph have even degree except the start vertex and end vertex of the Eulerian trail (they have odd degree). If a graph has an Eulerian circuit then all vertices have even degree. The converses in each case are also true (but take a little work to show). In each of the following graphs, find the longest trail (most edges) and longest circuit. If the graph has an Eulerian circuit or trail, say so.

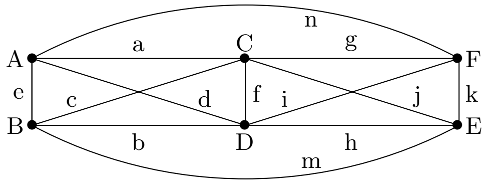
(a)



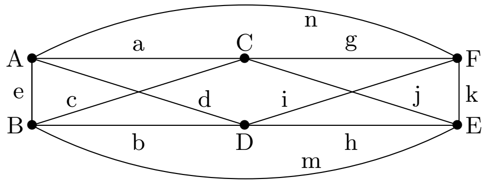
(b)



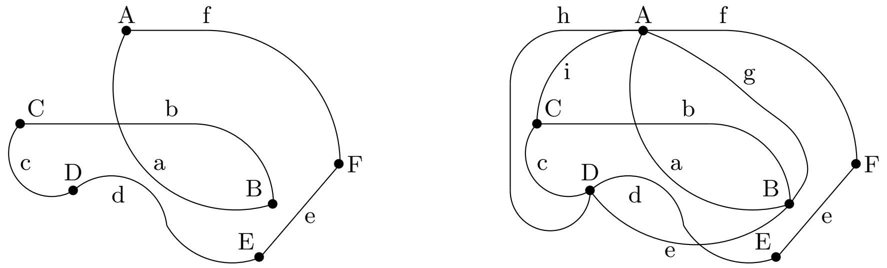
(c)



(d)

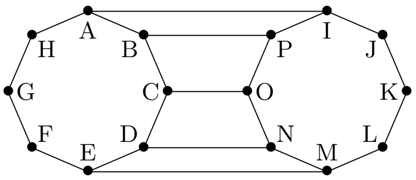


5.3.6. Suppose we start with a graph $G' = (V, E', \varphi')$ that is a cycle and then add additional edges, without adding any new vertices, to obtain a graph $G = (V, E, \varphi)$. As an example, consider

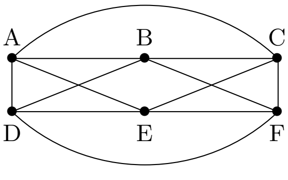


where the first graph $G' = (V, E', \varphi')$ is the cycle induced by the edges $\{a, b, c, d, e, f\}$. A graph that can be constructed from such a two-step process is called a *Hamiltonian* graph. The cycle G' is called a *Hamiltonian* cycle. Alternatively, a cycle in a graph $G = (V, E, \varphi)$ is a Hamiltonian cycle for G if every element of V is a vertex of the cycle. A graph $G = (V, E, \varphi)$ is Hamiltonian if it has a subgraph that is a Hamiltonian cycle for G. For each of the following graphs $G = (V, E, \varphi)$, find a cycle in G of maximum length. State whether or not the graph is Hamiltonian.

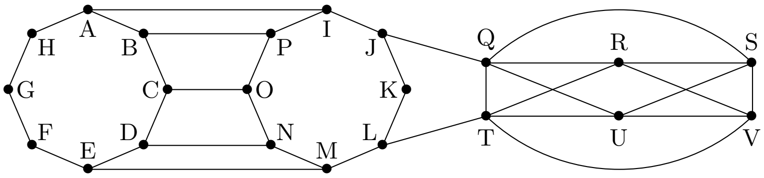
(a)



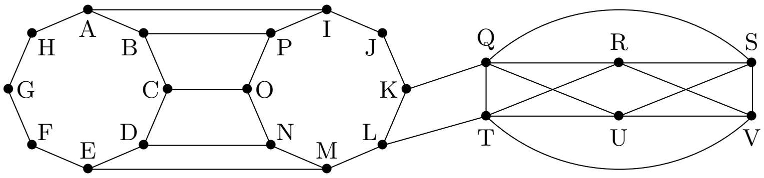
(b)



(c)



(d)



5.4 Trees

Trees play an important role in a variety of algorithms. We have already met decision trees in Chapter 3. In this section, we define trees precisely and look at some of their properties. We study trees further in Section 6.1 and Chapter 9.

Definition 5.9 (Free) Tree *If G is a connected graph without any cycles then G is called a* **tree**. *(If $|V| = 1$, then G is connected and hence is a tree.) A tree is also called a* **free tree**.

The graph of Figure 5.2 (p. 123) is connected but is not a tree. The subgraph of this graph induced by the edges $\{a, e, g\}$ is a tree. If G is a tree, then φ is an injection since if $e_1 \neq e_2$ and $\varphi(e_1) = \varphi(e_2)$, then $\{e_1, e_2\}$ induces a cycle. Because of this, we can think of a tree as a simple graph when we are not interested in names of the edges.

It's natural to ask how many trees can be formed using an n-set V for the vertices. In Example 5.10 (p. 143), we'll prove that the answer is n^{n-2}. Another proof is given in Exercise 5.4.12.

Since the notion of a tree is so important, it will be useful to have some equivalent definitions of a tree. We state them as a theorem

Theorem 5.4 Definitions of tree *If G is a connected graph, the following are equivalent.*

(a) *G is a tree.*

(b) *G has no cycles.*

(c) *For every pair of vertices $u \neq v$ in G, there is exactly one path from u to v.*

(d) *Removing any edge from G gives a graph which is not connected.*

(e) *The number of vertices of G is one more than the number of edges of G.*

Proof: By the definition of a tree, (a) and (b) are equivalent.

Theorem 5.3 can be used to prove that (b) and (c) are equivalent. We leave that as an exercise.

If $\{u, v\}$ is an edge, it follows from (c) that the edge is the only path from u to v and so removing it disconnects the graph. Hence (c) implies (d). We leave it as an exercise to prove that (d) implies (b). This shows that (a), (b), (c), and (d) are all equivalent.

All that remains is (e).

We first show that (b) implies (e). We will use induction on the number of vertices of G. If G has one vertex, it has no edges and so we are done. Otherwise, we claim that G has a vertex u of degree 1; that is, it lies on only one edge $\{u, w\}$. We prove this claim shortly. Remove u and $\{u, v\}$ to obtain a graph H with one less edge and one less vertex. Since G is connected and has no cycles, the same is true of H. Since H has fewer vertices than G, the induction hypothesis tells us that (e) is true for H: there is one more vertex than edge in H. Since H was obtained from G by removing one edge and one vertex, (e) is true for G. It remains to prove the existence of u. Suppose no such u exists; that is, suppose that each vertex lies on at least two edges. We will derive a contradiction. Start at any vertex v_1 of G leave v_1 by some edge e_1 to reach another vertex v_2. Leave v_2 by some edge e_2 different from the edge used to reach v_2. Continue with this process. Since each vertex lies on at least two edges, the process never stops. Hence we eventually repeat a vertex, say

$$v_1, e_1, v_2, \ldots, v_k, e_k, \ldots, v_n, e_n, v_{n+1} = v_k.$$

The edges $e_k, \ldots, e_n$ form a cycle, which is a contradiction.

Now suppose G is a connected graph which is not a tree. It suffices to prove that G has at least as many edges as it has vertices. Why? If we do so, we will have shown

$$((\text{a}) \text{ is false}) \quad \text{implies} \quad ((\text{e}) \text{ is false})$$

and hence the contrapositive ((e) is true) implies ((a) is true). On with the proof! By (d) we can remove an edge from G to get a new graph which is still connected. If this is not a tree, repeat the process and keep doing so until we reach a tree. Since (a) implies (b) and (b) implies (e), the number of vertices is now one more than the number of edges. Since we removed edges from G but did not remove vertices, G must have at least as many edges as vertices. $\square$

Example 5.8 Symmetry in graphs and trees Let $G = (V, E)$ be a simple graph. Suppose $\nu : V \to V$ is a isomorphism of G to G. Graph isomorphism is defined in Definition 5.4 (p. 128). We call an isomorphism from something to itself an "endomorphism" or a "symmetry." How much can a symmetry move the vertices of a graph?

It turns out that most graphs have only the trivial endomorphism $\nu(v) = v$ for all $v \in V$, so the vertices can't move at all. On the other hand, if the graph has no edges ($E = \emptyset$) or all possible edges ($E = \mathcal{P}_2(V)$), then every permutation of the vertices in V is a symmetry since the condition $\{u, v\} \in E$ if and only if $\{\nu(u), \nu(v)\} \in E$ in Definition 5.4 is easily seen to hold. What about graphs that are encountered in practice?

The most common graphs in computer science are trees. Most trees have symmetries. For example, suppose $\{u, v\}$, $\{u, w\} \in E$ and v and w are leaves in a tree $T = (V, E)$. (Note that V may have many more vertices besides u, v and w.) We leave it for you to verify that

$$\nu(x) = \begin{cases} w, & \text{if } x = v, \\ v, & \text{if } x = w, \\ x, & \text{otherwise,} \end{cases}$$

is an isomorphism of T. Only the vertices v and w moved. While all vertices might move in an isomorphism of a tree, there are always some that either don't move or don't move "far." We'll prove

Theorem 5.5 *If ν is an isomorphism of the tree $T = (V, E)$ then either*

(a) *there is a vertex v with $\nu(v) = v$ or*

(b) *there is an edge $\{u, v\}$ with $\nu(u) = v$ and $\nu(v) = u$.*

In other words, either a vertex or an edge does not move.

Suppose (a) is not true. We'll define a map $f : V \to E$ and use the Pigeonhole Principle, Theorem 2.5 (p. 55).

If $x \in V$, $\nu(x) \neq x$. Since T is a tree, there is a unique path from x to $\nu(x)$. Let $f(x)$ be the first edge on the path. Since $|E| = |V| - 1$, the Pigeonhole Principle tells us that there must be two vertices v_1 and v_2 with $f(v_1) = f(v_2)$. Thus v_1 and v_2 are the ends of an edge $e = \{v_1, v_2\}$. Since ν is an isomorphism, $\nu(e) = \{\nu(v_1), \nu(v_2)\}$ is also an edge.

We claim $\nu(e) = e$. Draw a picture and try to understand why this is so. (Remember that paths between tree vertices are unique.)

* * * Stop and think about this! * * *

If $\nu(e) \neq e$, here is a way to get from v_1 to v_2 without using e. There is a path P_i from v_i to $\nu(v_i)$ that starts with the edge e. Start at v_1 and follow the part of P_2 after e to $\nu(v_1)$. Traverse the edge $\nu(e) = \{\nu(v_1), \nu(v_2)\}$ from $\nu(v_1)$ to $\nu(v_2)$. Since $e = \{v_1, v_2\}$ is the first edge on P_2, you can follow P_1 backwards until you reach v_2. You have just walked from v_1 to v_2 without using e. Thus there is a path from v_1 to v_2 that does not use e. Since there can be only one path from v_1 to v_2, namely e, our assumption that $\nu(e) \neq e$ must be wrong. Defining $u = v_1$ and $v = v_2$ completes the proof. □

The decision trees in Chapter 3 have some special properties. First, they have a starting point. Second, the edges (decisions) out of each vertex are ordered. We now formalize these concepts.

Definition 5.10 Rooted graph *A pair (G, v), consisting of a graph G and a specified vertex v, is called a* **rooted graph** *with* **root** *v.*

A rooted tree is sometimes simply referred to as a tree, but we will never do so.

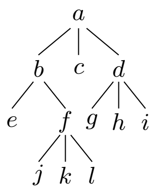


Figure 5.5 A rooted plane tree with root a at the top, as usual. Linear ordering of siblings is left to right.

Definition 5.11 Parent, child, sibling and leaf *Let (T, r) be a rooted tree. If w is any vertex other than r, let $r = v_0, v_1, \ldots, v_k, v_{k+1} = w$, be the unique path from r to w. We call v_k the* **parent** *of w and call w a* **child** *of v_k. Vertices with the same parent are* **siblings**. *A vertex with no children is a* **leaf**.

Definition 5.12 Rooted plane tree *Let (T, r) be a rooted tree. For each vertex, order the children of the vertex. The result is a* **rooted plane tree**, *which we abbreviate to* **RP-tree**. *An RP-tree is also called a* **tree**. *Parents and children are also called* **fathers** *and* **sons**.

Figure 5.5 shows an RP-tree. The sons of b are e and f, the parent of g is d, vertex k has no children, and the siblings of h are g and i. The decision trees of Chapter 3 are RP-trees. When we draw a tree as in Figure 5.5, the root is normally the topmost vertex and all edges are directed downward. In addition, siblings are drawn from left to right following their ordering. For example, the ordering of the siblings $\{j, l, k\}$ is j, then k and then l.

It's clear where "rooted" comes from in RP-tree, but where does "plane" come from? When such a tree is drawn on a piece of paper (a plane), we can start with the root, list its children below it in order, and so on—just like the picture of a decision tree. On the other hand, any rooted tree drawn in the plane has a natural ordering for the children of a vertex v provided v is not the root: Starting at the edge from the parent of v, walk counterclockwise around v, listing the children of v in the order in which their edges are met.

We now have two different concepts that are referred to as trees: free trees (Definition 5.9) and RP-trees. How will we keep them straight? When we believe there is no chance of confusion, we may call them trees; otherwise will we call them free trees and RP-trees. Sometimes the distinction is not needed. For example, by Theorem 5.4 the statement that a tree has no cycles applies equally well to RP-trees and free trees since every RP-tree is simply a free tree with a root and orderings.

In Chapter 9 we'll discuss some recursive aspects of RP-trees including tree *traversal* and *grammars*.

Exercises

5.4.1. Complete the proof of Theorem 5.4.

5.4.2. Let $G = (V, E, \varphi)$ be a connected graph. Using Theorem 5.4 and its proof, do the following.

(a) Prove that G has a cycle if and only if there is some edge e such that the subgraph of G with vertices V and edges $E - \{e\}$ is connected.

(b) Prove that there is a subgraph of G which is a tree and has vertex set V. Such a tree is called a *spanning tree*.

(c) Prove that $|V| \leq |E| + 1$ with equality if and only if G is a tree.

5.4.3. Let $T = (V, E)$ be a tree and let $d(v)$ be the degree of a vertex as defined in Exercise 5.1.1.

(a) Prove that $\sum_{v \in V}(2 - d(v)) = 2$.
Hint. See Exercise 5.1.1.

(b) Prove that, if T has a vertex of degree $m \geq 2$, then it has at least m vertices of degree 1. Vertices of degree 1 are called *leaves* or *terminal vertices*.

(c) Give an example for all $m \geq 2$ of a tree with a vertex of degree m and only m leaves.

(d) Suppose that T has at most one vertex of degree 2. Prove that over half the vertices of T are leaves.

(e) Give an example for all $m > 0$ of a tree with m leaves, $m - 1$ other vertices and at most one vertex of degree 2.

5.4.4. In this exercise, we study how counting edges and vertices in a graph can establish that cycles exist.

(a) Using induction on n, prove:
If $n \geq 0$, a connected graph with v vertices and $v + n$ edges has at least $n + 1$ cycles.

(b) Prove that a graph with v vertices, e edges and c components has at least $c + e - v$ cycles.
Hint. Use (a) for each component.

(c) Show that (a) is best possible, even for simple graphs. In other words, for each n construct a simple graph that has n more edges than vertices but has only $n + 1$ cycles.

5.4.5. Prove that every tree with at least 3 vertices has a cut vertex and a cut edge. (The terms cut edge and cut vertex are defined in Exercise 5.3.3.)

5.4.6. Give an example of a graph that satisfies the specified condition or show that no such graph exists.

(a) A tree with six vertices and six edges

(b) A tree with three or more vertices, two vertices of degree one and all the other vertices with degree three or more.

(c) A disconnected simple graph with 10 vertices, 8 edges and a cycle.

(d) A disconnected simple graph with 12 vertices, 11 edges and no cycles.

(e) A tree with 6 vertices and the sum of the degrees of all vertices 12.

(f) A connected simple graph with 6 edges, 4 vertices, and exactly 2 cycles.

(g) A simple graph with 6 vertices, 6 edges and no cycles.

5.4.7. The *height* of a rooted tree is the the length of the longest path from a leaf of the tree to the root of the tree. A rooted tree in which each non-leaf vertex has at most two children is called a *binary tree*. If each non-leaf vertex has exactly two children, the tree is called a *full binary tree*.

(a) Show that if a binary tree has l leaves and height h then $l \leq 2^h$, or, equivalently, $\log_2(l) \leq h$.

(b) Given that a binary tree has l leaves, what can you say about the maximum value of h?

(c) Given a full binary tree with l leaves, what is the maximum height h?

(d) Given a full binary tree with l leaves, what is the minimum height h?

(e) Given a binary tree of l leaves, what is the minimal height h?

5.4.8. Prove that a full binary tree with n leaves has a total of $2n - 1$ vertices. (The concept of a full binary tree was defined in Exercise 5.4.7.)
Challenge. How many different proofs can you find?

5.4.9. In each of the following cases, state whether or not such a tree is possible.

(a) A binary tree with 35 leaves and height 100.

(b) A full binary tree with 21 leaves and height 21.

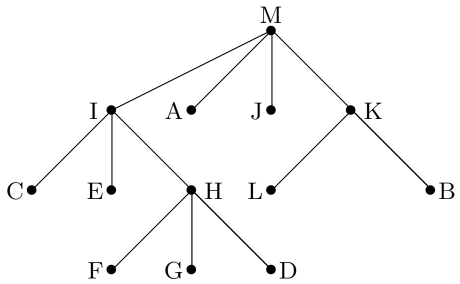
(c) A binary tree with 33 leaves and height 5.

(d) A full binary tree with 65 leaves and height 6.

5.4.10. What is the maximal number of *vertices* in a rooted tree of height h if every vertex has at most k children. What is the maximal number of *leaves* in a rooted tree of height h if every vertex has at most k children?

5.4.11. We are going to define certain important lists of vertices associated with the rooted plane tree, Figure 5.5. These lists can, in a similar fashion, be associated with any rooted plane tree. The first list is $abcdefghijkl$. This list, called the *breadth-first vertex list*, is obtained by starting at the root and reading the vertices of the tree, left to right, as one would read a book. For this definition to work on any tree, you must imagine the tree drawn with the root at the top and all vertices distance one from the root drawn at the next level down, all of distance two at the next level, etc. The second important list is called the *depth-first vertex list.* The depth-first vertex list for Figure 5.5 is $abebfjfkflfbacadgdhdida$. Note that each vertex appears in the depth-first vertex list a number of times equal to one plus the number of children of that vertex. If one extracts the sublist of first occurrences of each vertex in the depth first list, one gets the list $abefjklcdghi$. This list is called the *pre-order vertex list.* If one extracts the sublist of last occurrences of each vertex in the depth first list, one gets the list $ejklfbcghida$ This list is called the *post-order vertex list.*

(a) For the following rooted plane tree, list the breadth-first, depth-first, pre-order, and post-order vertex lists.



(b) Given that the following is the depth-first vertex list of a rooted plane tree, reconstruct the tree: $MKBKLKMIHDHGHFHIEICIM$.

(c) Is it possible to reconstruct a rooted plane tree given just its pre-order vertex list?

(d) Is it possible to reconstruct a rooted plane tree given its pre-order vertex list and its post-order vertex list?

*5.4.12. Construct a bijection between functions from $\underline{n-2}$ to $\underline{n}$ and trees with $V = \underline{n}$ as follows. Repeatedly remove the leaf with the largest label from the tree until only a two vertex tree remains. When a leaf is removed, list the vertex that it was attached to. This is called the *Prüfer sequence* for the tree. To establish the bijection, you must prove that any list of length $n-2$ chosen from $\underline{n}$ with repetition allowed corresponds is the Prüfer sequence of a tree.
Hint. Show that the largest number *not* in the Prüfer sequence is the vertex that was first removed.

5.5 Directed Graphs (Digraphs)

In the next two sections, we take a look at two important modifications of the concept of a graph.

Look again at Figure 5.2 (p. 123). Imagine now that the symbols a, b, c, d, e, f and g, instead of standing for route names, stand for commodities (applesauce, bread, computers, etc.) that are produced in one town and shipped to another town. In order to get a picture of the flow of commodities, we need to know the directions in which they are shipped. This information is provided by Figure 5.6.

In set theoretic terms, the information needed to construct Figure 5.6 can be specified by giving a pair $D = (V, \varphi)$ where φ is a function with domain $E = \{a, b, c, d, e, f, g\}$ and range $V \times V$. Specifically,

$$\varphi = \begin{pmatrix} a & b & c & d & e & f & g \\ (B,A) & (A,B) & (C,A) & (C,B) & (B,C) & (C,B) & (D,B) \end{pmatrix}.$$

The structure given in Figure 5.6 is an example of a directed graph:

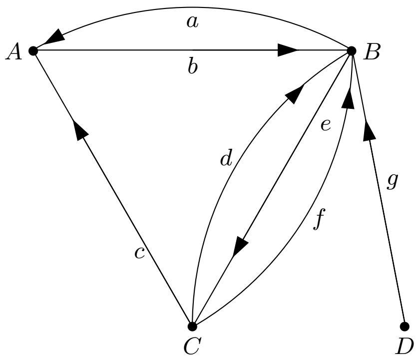


Figure 5.6 Cities with flow of commodities shown.

Definition 5.13 Directed graph *A* **directed graph** *(or* **digraph***) is a triple $D = (V, E, \varphi)$ where V and E are finite sets and φ is a function with domain E and range $V \times V$. We call E the set of* **edges** *of the digraph D and call V the set of* **vertices** *of D.*

Note that it is possible that $f(x) = (v, v)$ for $v \in V$. Such an edge x is called a *loop.* A *simple digraph*, like a simple graph, is a pair (V, E) where $E \subseteq V \times V$. Subgraphs, paths, walks, trails and cycles in directed graphs are analogous to the corresponding structures in a graph. A directed graph $D' = (V', E', \varphi')$ is a *directed subgraph* of $D = (V, E, \varphi)$ if $V' \subseteq V$, $E' \subseteq E$ and φ' is the restriction of φ to E' with range $V' \times V'$. A *directed path* in the digraph $D = (V, E, \varphi)$ is a sequence of edges $e_1, \ldots, e_{n-1}$ for which there is a sequence of distinct vertices $a_1, \ldots, a_n$ such that $\varphi(e_i) = (a_i, a_{i+1})$ for $i = 1, 2, \ldots, n-1$. The subdigraph induced by a set of edges $E' \subseteq E$ or the set of vertices $V' \subseteq V$ is defined in a way analogous to the corresponding concept for graphs. Let $D = (V, E, \varphi)$ be a directed graph and let $e_1, \ldots, e_{n-1}$ with vertex sequence $a_1, \ldots, a_n$ be a directed path. If x is an edge of D such that $\varphi(x) = (a_n, a_1)$, then the subgraph induced by the edges $\{e_1, \ldots, e_{n-1}, x\}$ is called a *directed cycle* in D. For example, the subgraph induced by the edges $\{c, b, e\}$ is a directed cycle in the digraph of Figure 5.6. The notions of *edge labeling* and *vertex labeling* extend directly to digraphs. The ideas of connected components and trees are more complicated in digraphs.

Example 5.9 Digraphs and binary relations Simple digraphs appear in mathematics under another important guise: *binary relations.* A binary relation on a set V is simply a subset of $V \times V$. Often the name of the relation and the subset are the same. Thus we speak of the binary relation $E \subseteq V \times V$. If you have absorbed all the terminology, you should be able to see immediately that (V, E) is a simple digraph and that any simple digraph (V', E') corresponds to a binary relation $E' \subseteq V' \times V'$.

What about simple graphs? We can identify $\{u, v\} \in \mathcal{P}_2(V)$ with $(u, v) \in V \times V$ and with $(v, u) \in V \times V$. A binary relation R is called "symmetric" if $(u, v) \in R$ implies $(v, u) \in R$. Thus it seems that simple graphs correspond to symmetric binary relations. This is not quite true since (u, u) would correspond to a loop. We must either allow loops or only look at symmetric binary relations that do not contain (u, u).

An equivalence relation on a set S is also a binary relation $R \subseteq S \times S$: We have $(x, y) \in R$ if and only if x and y are equivalent. Note that this is a symmetric relationship. Which simple graphs (with loops allowed) correspond to equivalence relations? There is a simple description of them. We'll let you look for it.

Functions from a set to itself are another special case of binary relations. Figure 5.7 shows a function and its associated digraph, called a *functional digraph.* Notice that some of the vertices form cycles and the remaining vertices form trees that are attached to the cycles, each tree being

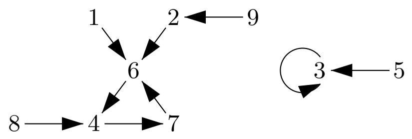


Figure 5.7 The functional digraph associated with the function $\varphi = \begin{pmatrix} 1&2&3&4&5&6&7&8&9 \\ 6&6&3&7&3&4&6&4&2 \end{pmatrix}$.

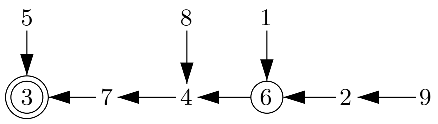


Figure 5.8 The doubly marked digraph built from the functional digraph of Figure 5.7. Removing the arrows gives a doubly marked tree.

attached by one of the vertices on a cycle. For example, the vertices 1, 2, 6, and 9 form a tree that is attached to a cycle by the vertex 6.

One can easily read powers (using composition, not multiplication) of a function from the functional digraph: The function itself comes from directed paths of length 1 and φ^k comes from directed paths of length k.

Permutations are a special case of functions. You should be able to see that, in this case, the functional digraph consists of cycles with no trees attached. $\square$

Example 5.10 The number of labeled trees Let t_n be the number of trees with vertex set $\underline{n}$. It's not hard to draw the possible trees for small values of n. If you do this, you should discover that $t_1 = 1$, $t_2 = 1$, $t_3 = 3$ and $t_4 = 16$. What is the pattern?

It turns out that $t_n = n^{n-2}$. You might try to check this for $t_5 = 125$. How can we prove this formula?

When we know the answer to a problem, we can often use some backwards reasoning or "answer analysis" to figure out how we might solve the problem. Since n^{n-2} is the number of functions from $\underline{n-2}$ to $\underline{n}$, we might try to find a bijection between such functions and trees. (This is done in Exercise 5.4.12.) Unfortunately, it is not at all clear how to proceed with this idea.

It would be much nicer if we looked at functions from $\underline{n}$ to $\underline{n}$ because these lead to functional digraphs: with the function $f \in \underline{n}^{\underline{n}}$, we associate the functional digraph (V, E) where $V = \underline{n}$ and $E = \{(x, f(x)) \,|\, x \in \underline{n}\}$. Let's pursue this and see if we can generate some ideas.

Look at the functional digraph in Figure 5.7. If we could somehow get rid of the cycles, we would have trees!

Some inspiration is needed. Here it is. The vertices on the cycles are a permutation drawn in cyclic form. For example, in Figure 5.7, the permutation is $(3)(4, 7, 6)$. Maybe we can draw the permutation in some other fashion. We can also write permutations in two line or one line form:

$$\begin{pmatrix} 3&4&6&7 \\ 3&7&4&6 \end{pmatrix} \qquad 3, 7, 4, 6.$$

We could simply take the one line form and construct a directed graph from it:

$$3 \longleftarrow 7 \longleftarrow 4 \longleftarrow 6$$

We could also drag along the other vertices that form trees attached to these cyclic vertices. This is shown in Figure 5.8, where you should ignore the circles around the vertices for the time being. We have constructed a tree!

This has one problem, we can no longer tell which vertices were on the cycles. The circles in Figure 5.8 take care of this, the double circle indicating the start of one line form and the single circle indicating the end. The unique path from the end to the start gives the one line notation, written in reverse. (The directed path is unique because we have constructed a tree.)

Actually our picture has more information than we need: The direction of the edges is determined by the fact that they are all directed toward the root, which is 3 in our example. Thus we can erase the arrowheads on the edges with no loss of information.

This entire process is reversible—Given any tree on $\underline{n}$ in which one vertex is marked with a double circle and one with a single circle, we can recover a unique function which gives this marked tree. Note that the same vertex may have the double circle and the single circle. This happens when there is only one point of the functional digraph on cycles. How many such doubly marked trees are there? By the bijection, there are n^n. Since we form a tree AND mark a vertex with a double circle AND mark a vertex with a single circle, the number is also $t_n \times n \times n$. This completes the proof. □

Exercises

5.5.1. If $D = (V, \varphi)$ is a loopless directed graph, the associated graph $G(D)$ is obtained by removing the directions of the edges. Instead of this rough geometric description, give a definition in terms of sets and functions.
Hint. Define a function with domain $\{(x, y) \mid x, y \in V \text{ and } x \neq y\}$.

5.5.2. We are interested in the number of simple digraphs with $V = \underline{n}$.

(a) Find the number of them.

(b) Find the number of them with no loops.

(c) In both cases, find the number of them with exactly q edges.

5.5.3. An *oriented simple graph* is a simple graph which has been converted to a digraph by assigning an orientation to each edge. The orientation of $\{u, v\}$ can be thought of as a mapping of it to either (u, v) or (v, u). Give an example of a simple digraph that has no loops but is not an oriented simple graph

5.5.4. We are interested in the oriented simple graphs with $V = \underline{n}$. (They are defined in Exercise 5.5.3.)

(a) Find the number of them.

(b) Find the number of them with exactly q edges.

5.5.5. Define an equivalence relation on digraphs that allows you to introduce the notion of unlabeled digraphs. Prove that you have, in fact, defined an equivalence relation.

5.5.6. A digraph is *strongly connected* if, for every two vertices v and w there is a directed path from v to w. From any digraph D, we can construct a simple graph $S(D)$ on the same set of vertices by letting $\{v, w\}$ be an edge of $S(D)$ if and only if at least one of (u, v) and (v, u) is an edge of D. You should find the first three parts of this exercise easy, if you *understand* the meaning of the various concepts—strongly connected, path, $S(D)$, etc.

(a) Prove that, if D is strongly connected, then $S(D)$ is connected.

(b) Construct an example of a digraph D such that D is not strongly connected but $S(D)$ is connected.

(c) Suppose that V_1, V_2 is a partition of the vertices V of a strongly connected digraph D; that is, $V_1 \neq \emptyset$, $V_2 \neq \emptyset$, $V_1 \cup V_2 = V$, and $V_1 \cap V_2 = \emptyset$. Prove that in D there is an edge from V_1 to V_2 and an edge from V_2 to V_1. (This means that for some $v_1, w_1 \in V_1$ and $v_2, w_2 \in V_2$, both (v_1, v_2) and (w_2, w_1) are edges of D.) We will call such a partition of V "2-way joined."

(d) Suppose that every partition V_1, V_2 of the vertices of D is 2-way joined. Prove that D is strongly connected.

5.5.7. Suppose that we are given a set of statements, for example (a) through (e) in Theorem 5.4, and that we have proved that some statements imply others. Construct a directed graph D as follows. The statements are the vertices V of D. For statements v and w, (v, w) is an edge of D if and only if we have a proof that statement v implies statement w. Prove the claim: "If D is strongly connected, we have proved enough to show that the statements in V are all equivalent." (See Exercise 5.5.6 for a definition of strongly connected.)

5.5.8. For any subset U of the vertices V of a directed graph D, define $d_{\text{in}}(U)$ to be the number of edges of e of D with $\varphi(e)$ of the form (w, u) where $u \in U$ and $w \notin U$. Define $d_{\text{out}}(U)$ similarly.

(a) For v a single vertex, what is $d_{\text{in}}(\{v\})$ in terms of the picture of D?

(b) Prove that $\sum d_{\text{in}}(\{v\}) = \sum d_{\text{out}}(\{v\})$, where the sums both range over all $v \in V$.

(c) Prove that $\sum_{u \in U} d_{\text{in}}(\{u\})$ equals $d_{\text{in}}(U)$ plus the number of non-loop edges of D that have both of their endpoints in U.

(d) Suppose that $d_{\text{in}}(\{v\}) = d_{\text{out}}(\{v\})$ for all $v \in V$. Prove that $d_{\text{in}}(U) = d_{\text{out}}(U)$ for all $U \subseteq V$.

5.5.9. Use the notation of Exercise 5.5.8. Suppose $d_{\text{in}}(\{v\}) = d_{\text{out}}(\{v\})$ for all $v \in V$. Prove that for every edge e of D there is a directed cycle in D that contains e.

5.5.10. Suppose that $S(D)$ is connected, where $S(D)$ is obtained from D by removing the directions of the edges. Use the notation of Exercise 5.5.8. Suppose $d_{\text{in}}(\{v\}) = d_{\text{out}}(\{v\})$ for all $v \in V$. Prove that there is a directed trail that contains every edge of D. Such a trail is called an *Eulerian trail*.

5.5.11. Let G be a connected simple graph.

(a) Suppose that the edges of G can be directed so that the resulting digraph is strongly connected. Prove that G has no isthmuses.

*(b) Suppose that G has no isthmuses. Prove that the edges of G can be directed so that the resulting graph is strongly connected. (This seems to be quite difficult given the material you have had so far. We will return to it later.)

5.5.12. Let $R \subseteq S \times S$ be a binary relation on S. Suppose that $|S| = n$.

(a) How many reflexive binary relations R are there on S?

(b) How many reflexive and symmetric relations R are there on S?

(c) The relation S is *unreflexive* if for all $x \in S$, $(x, x) \notin R$. How many unreflexive, symmetric binary relations R are there on S?

(d) How many symmetric relations R on S are not reflexive?

5.5.13. A binary relation R on S is an *order relation* if it is reflexive, antisymmetric, and transitive. R is *antisymmetric* if for all $(x, y) \in R$ with $x \neq y$, $(y, x) \notin R$. Given an order relation R, the *covering relation* H of R consists of all $(x, z) \in R$ such that there is no y, $x < y < z$, where $(x, y) \in R$. A pictorial representation of the covering relation as a directed graph is called a "Hasse diagram" of H.

(a) Show that the divides relation on $S = \{2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16\}$ is an order relation. By definition, (x, y) is in the divides relation on S is x is a factor of y. Thus, $(4, 12)$ is in the divides relation. $x|y$ is the usual notation for x is a factor of y.

(b) Draw the directed graph of the covering relation of R.

5.5.14. Let R be a binary relation on S. Let T_R be the smallest transitive relation, $R \subseteq T_R$. By "smallest" we mean that if $R \subseteq N \subset T_R$ then N is not transitive. In other words, there is no proper subset of T_R that contains R and is transitive. Note that if R is already transitive then $T_R = R$. T_R is called the *transitive closure* of R. Let $S = \{2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16\}$ and let

$$H = \{(2, 4), (2, 6), (2, 10), (2, 14), (3, 6), (3, 9), (3, 15), (4, 8), (4, 12), (5, 10), (5, 15), (6, 12), (7, 14)\}.$$

Find the transitive closure of H.

5.5.15. A graph is selected uniformly at random from all q-edge simple graphs with vertex set $\underline{n}$.

(a) What is the probability that the graph we have chosen is a tree? (Your answer will depend on q.)

(b) Show that this probability is bigger than $(2/e)^{n-1}/n$ when $q = n-1$.
Hint. You may use the fact that, by Stirling's formula, $(n-1)! > \left(\frac{n-1}{e}\right)^{n-1}$. Also note that $\binom{N}{q} < N^q/q!$.

*5.6 Computer Representations of Graphs

What is the best way to represent a graph in a computer? That question is based on the mistaken assumption that there is *one best way.* In fact, there are a variety of ways to represent a graph. We'll briefly discuss two common ones: adjacency lists and matrices. For simplicity, *we will limit our attention to simple graphs and digraphs.*

Let $G = (V,E)$ be a graph. For each $v \in V$, keep a list of those $x \in V$ such that $\{v,x\} \in E$. This is a relatively compact method for storing the structure of G. The actual implementation may be with an array or with a linked list. If $D = (V,E)$ is a simple digraph, then we keep two lists for each $v \in V$; one of those $x \in V$ such that $(x,v) \in E$ and one of those $y \in V$ such that $(v,y) \in E$.

The method of linked lists is usually used with RP-trees. If the number of possible sons a vertex can have is not limited, a variation of this method is frequently used. Here's one such variation. Each vertex v has a list of four vertices. If a vertex needed in the list does not exist, that fact is recorded in the list (e.g., by using zero). The four vertices are:

- the parent f of the vertex v;
- the first child of v in the ordering of the children of v;
- the sibling of v that immediately precedes it in the ordering of the siblings of v;
- the sibling of v that immediately follows it in the ordering of the siblings of v.

Matrices are simply doubly indexed arrays. In the most common representation of a simple graph $G = (\underline{n},E)$, the matrix $A(G)$ is $n \times n$ and

$$a_{i,j} = \begin{cases} 1 & \text{if } \{i,j\} \in E; \\ 0 & \text{otherwise.} \end{cases}$$

For a simple digraph $D = (\underline{n},E)$, we make a minor adjustment to define $A(D)$:

$$a_{i,j} = \begin{cases} 1 & \text{if } (i,j) \in E; \\ 0 & \text{otherwise.} \end{cases}$$

This representation can waste a considerable amount of space if the graph has relatively few edges. In any event, only about half of $A(G)$ is needed since $a_{i,j} = a_{j,i}$. The matrix representation is useful for some calculations of numbers related to the structure of the graph. See, for example, Exercise 5.6.3.

Exercises

5.6.1. Suppose that G is a bipartite simple graph. (See Exercise 5.3.2 for a definition.) Prove that the vertices can be numbered 1 through $n = |V|$ such that for some k the matrix $A(G)$ has a $k \times k$ block of zeroes in its upper left corner and an $(n-k) \times (n-k)$ block of zeroes in its lower right corner.

5.6.2. Suppose that G is a simple graph with connected components $G_1, \ldots, G_m$. Number the vertices of G by first numbering those in G_1, then those in G_2 and so on. Suppose that G_i has n_i vertices. Provide a description of $A(G)$ like that in Exercise 5.6.1.

5.6.3. This exercise requires familiarity with matrix multiplication. Let $G = (\underline{n}, E)$ be a simple graph. Let $a_{i,j}^{(k)}$ be the (i,j) element of the matrix $(A(G))^k$. Define a *walk* from i to j like a path, but allow repetitions; i.e., a walk is a sequence $e_1, e_2, \ldots, e_{n-1}$ and a sequence of $v_1, \ldots, v_n$ of vertices such that $v_1 = i$, $v_n = j$ and $\varphi(e_i) = \{v_i, v_{i+1}\}$. The *length* of the walk is $n-1$, the number of edges it contains, with repetitions counted.

(a) Prove that there is a walk of length k from i to j in G if and only if $a_{i,j}^{(k)} \neq 0$.

(b) Suppose that $i \neq j$. If $a_{i,j}^{(k)} \neq 0$ for some $k > 0$, let m be the smallest such k. Prove that there is a path of length m between i and j.

(c) Can you find an analog of the previous result for cycles? Be careful!

(d) Prove that G is connected if and only if $(A(G) + I)^k$ contains no zeroes for all sufficiently large k. Prove that all $k \geq n-1$ are sufficiently large.
Hint. With $B = A(G) + I$ and $A(G)^0 = I$, prove that $b_{i,j}^{(k)} = \sum_{t=0}^{k} \binom{k}{t} a_{i,j}^{(t)}$.

5.6.4. State and prove results like those in the previous exercise for directed simple graphs.

5.6.5. A matrix is called *nilpotent* if all sufficiently high powers of it consist entirely of zeroes. Find necessary and sufficient conditions on a simple digraph D for $A(D)$ to be nilpotent.

Notes and References

Because of the relatively modest mathematical prerequisites and the range of applications of graph theory, introductory texts range from high school level to graduate school level and from very applied to very theoretical. Even within one of these compartments, the breadth of graph theory results in a diversity of viewpoints. This is illustrated by the variety in the references.

The book by Ore and Wilson [10] is a delightful elementary introduction. The books by Berge [1], Bondy and Murty [3], Chartrand and Lesniak [4] and Tutte [12] are more advanced than our text. The remaining books are at a level similar to our text, although they often contain some more advanced topics. If you are particularly interested in applications, see the book by Roberts [11]. If your are interested in algorithms, see the book by McHugh [8].

Other applications of the Pigeonhole Principle to prove properties of trees can be found in [7]. Other proofs of the formula $t_n = n^{n-2}$ for trees can be found in Moon's article [9].

1. Claude Berge, *The Theory of Graphs and Its Applications*, Dover (2001).
2. Béla Bollabás, *Graph Theory: An Introductory Course*, Springer-Verlag (1971).
3. J. Adrian Bondy and U.S.R. Murty, *Graph Theory with Applications*, American Elsevier (1976).
4. Gary Chartrand and Linda Lesniak, *Graphs and Digraphs*, 4th ed., CRC Press (2004).
5. Alan Gibbons, *Algorithmic Graph Theory*, Cambridge Univ. Press (1985).
6. Ronald J. Gould and Ron Gould, *Graph Theory*, Bejamin Cummings (1988).
7. N. Graham, R.C. Entringer and L.A. Székely, New tricks for old trees: Maps and the Pigeonhole Principle, *Amer. Math. Monthly* **101** (1994) 664–667.
8. James A. McHugh, *Algorithmic Graph Theory*, Prentice-Hall (1990).
9. John W. Moon, Various proofs of Cayley's formula for counting trees, in Frank Harary (ed.), *A Seminar on Graph Theory*, Holt, Rinehart and Winston (1967).
10. Oystein Ore and Robin J. Wilson, *Graphs and Their Uses*, Mathematical Assn. of America (1990).
11. Fred S. Roberts, *Applied Combinatorics*, Prentice-Hall (1984).
12. William T. Tutte, *Graph Theory*, Encyclopedia of Mathematics and Its Applications, Vol. 21, Cambridge Univ. Press (1984).