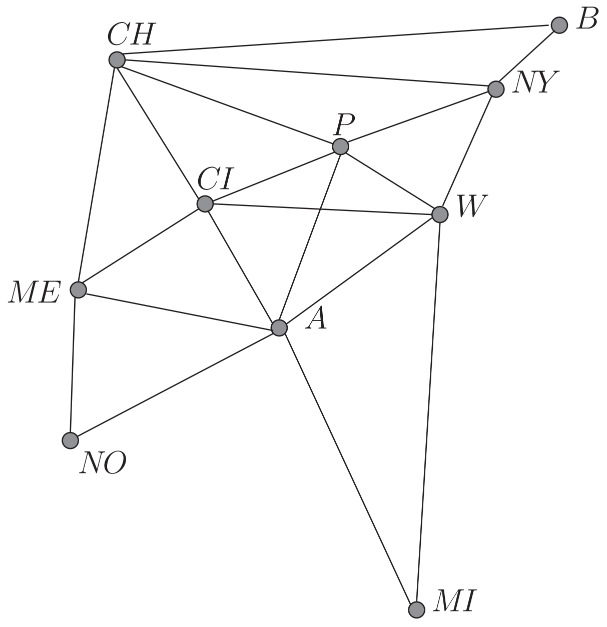
4.3 Graphs

In this chapter we introduce a fundamental structural idea of discrete mathematics, that of a graph. Many situations in the applications of discrete mathematics may be modeled by the use of a graph, and many algorithms have their most natural description in terms of graphs. It is for this reason that graphs are important to the computer scientist. Graph theory is an ideal subject for learning to understand proof by induction because induction, especially strong induction, seems to enter into the majority of proofs in graph theory.

Edges and Paths

Exercise 4.3-1 In Figure 4.1, you see a stylized map of some cities in the eastern United States (Boston, New York, Pittsburgh, Cincinnati, Chicago, Memphis, New Orleans, Atlanta, Washington DC, and Miami). A company has major offices with data pro-

Figure 4.1: A stylized map of some eastern US cities.



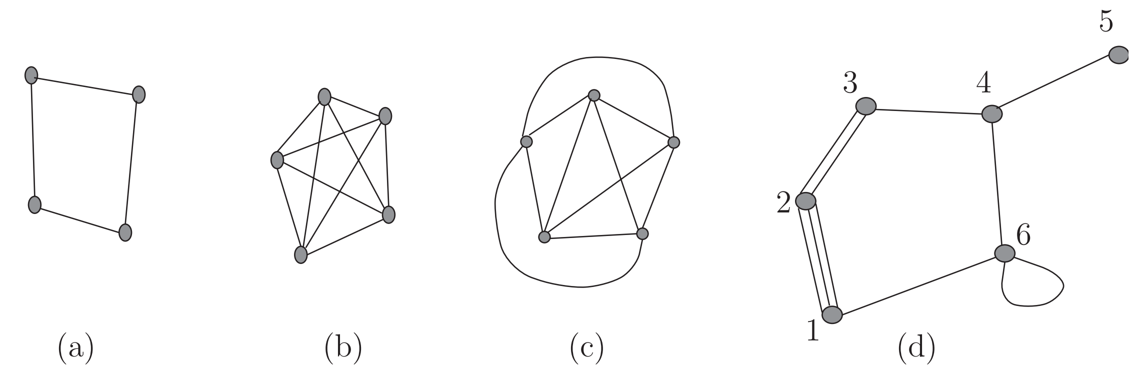
cessing centers in each of these cities, and as its operations have grown, it has leased dedicated communication lines between certain pairs of these cities to allow for efficient communication among the computer systems in the various cities. Each grey dot in the figure stands for a data center, and each line in the figure stands for a dedicated communication link. What is the minimum number of links that could be used in sending a message from B (Boston) to NO (New Orleans)? Give a route with this number of links.

Exercise 4.3-2 Which city or cities has or have the most communication links emanating from them?

Exercise 4.3-3 What is the total number of communication links in the figure?

The picture in Figure 4.1 is a drawing of what we call a "graph". A **graph** consists of a set of *vertices* and a set of *edges* with the property that each edge has two (not necessarily different) vertices associated with it and called its *endpoints*. We say the edge *joins* the endpoints, and we say two endpoints are *adjacent* if they are joined by an edge. When vertex is an endpoint of an edge, we say the edge and the vertex are *incident*. Several more examples of graphs are given in Figure 4.2. To *draw* a graph, we draw a point (in our case a grey circle) in the plane

Figure 4.2: Some examples of graphs



for each vertex, and then for each edge we draw a (possibly curved) line between the points that correspond to the endpoints of the edge. The only vertices that may be touched by the line representing an edge are the endpoints of the edge. Notice that in graph (d) of Figure 4.2 we have three edges joining the vertices marked 1 and 2 and two edges joining the vertices marked 2 and 3. We also have one edge that joins the vertex marked 6 to itself. This edge has two identical endpoints. The graph in Figure 4.1 and the first three graphs in Figure 4.2 are called simple graphs. A *simple graph* is one that has at most one edge joining each pair of distinct vertices, and no edges joining a vertex to itself.[1] You'll note in Figure 4.2 that we sometimes label the vertices of the graph and we sometimes don't. We label the vertices when we want to give them meaning, as in Figure 4.1 or when we know we will want to refer to them as in graph (d) of Figure 4.2. We say that graph (d) in Figure 4.2 has a "loop" at vertex 6 and multiple edges joining vertices 1 and 2 and vertices 2 and 3. More precisely, an edge that joins a vertex to itself is called a *loop* and we say we have *multiple edges* between vertices x and y if there is more than one edge joining x and y. If there is an edge from vertex x to vertex y in a simple graph, we denote it by $\{x, y\}$. Thus $\{P, W\}$ denotes the edge between Pittsburgh and Washington in Figure 4.1 Sometimes it will be helpful to have a symbol to stand for a graph. We use the phrase "Let $G = (V, E)$" as a shorthand for "Let G stand for a graph with vertex set V and edge set E.

The drawings in parts (b) and (c) of Figure 4.2 are different drawings of the same graph. The graph consists of five vertices and one edge between each pair of distinct vertices. It is called the complete graph on five vertices and is denoted by K_5. In general, a *complete graph* on n vertices is a graph with n vertices that has an edge between each two of the vertices. We use K_n to stand for a complete graph on n vertices. These two drawings are intended to illustrate that there are many different ways we can draw a given graph. The two drawings illustrate two

[1]The terminology of graph theory has not yet been standardized, because it is a relatively young subject. The terminology we are using here is the most popular terminology in computer science, but some graph theorists would reserve the word graph for what we have just called a simple graph and would use the word multigraph for what we called a graph.

different ideas. Drawing (b) illustrates the fact that each vertex is adjacent to each other vertex and suggests that there is a high degree of symmetry. Drawing (c) illustrates the fact that it is possible to draw the graph so that only one pair of edges crosses; other than that the only places where edges come together are at their endpoints. In fact, it is impossible to draw K_5 so that no edges cross, a fact that we shall explain later in this chapter.

In Exercise 4.3-1 the links referred to are edges of the graph and the cities are the vertices of the graph. It is possible to get from the vertex for Boston to the vertex for New Orleans by using three communication links, namely the edge from Boston to Chicago, the edge from Chicago to Memphis, and the edge from Memphis to New Orleans. We call an alternating sequence of vertices and edges in a graph a **path** if it starts and ends with a vertex, and each edge joins the vertex before it in the sequence to the vertex after it in the sequence.[2] If a is the first vertex in the path and b is the last vertex in the path, then we say the path is a path from a to b. Thus the path we found from Boston to New Orleans is $B\{B,CH\}CH\{CH,ME\},ME\{ME,NO\}NO$. Because the graph is simple, we can also use the shorter notation B,CH,ME,NO to describe the same path, because there is exactly one edge between successive vertices in this list. The *length* of a path is the number of edges it has, so our path from Boston to New Orleans has length 3. The length of a shortest path between two vertices in a tree is called the *distance* between them. Thus the distance from Boston to New Orleans in the graph of FigureStylizedMap is three. By inspecting the map we see that there is no shorter path from Boston to New Orleans. Notice that no vertex or edge is repeated on our path from Boston to New Orleans. A path is called a **simple path** if it has no repeated vertices or edges.[3]

The degree of a vertex

In Exercise 4.3-2, the city with the most communication links is Atlanta (A). We say the vertex A has "degree" 6 because 6 edges emanate from it. More generally the *degree* of a vertex in a graph is the number of times it is incident with edges of the graph; that is, the degree of a vertex x is the number of edges from x to other vertices plus twice the number of loops at vertex x. In graph (d) of Figure 4.2 vertex 2 has degree 5, and vertex 6 has degree 4. In a graph like the one in Figure 4.1, it is somewhat difficult to count the edges just because you can forget which ones you've counted and which ones you haven't.

Exercise 4.3-4 Is there a relationship between the number of edges in a graph and the degrees of the vertices? If so, find it. Hint: computing degrees of vertices and number of edges in some relatively small examples of graphs should help you discover a formula. To find one proof, imagine a wild west movie in which the villain is hiding under the front porch of a cabin. A posse rides up and is talking to the owner of the cabin, and the bad guy can just barely look out from underneath the porch and count the horses hoofs. If he counts the hooves accurately, what can he do to figure out the number of horses, and thus presumably the size of the posse?

In Exercise 4.3-4, examples such as those in Figure 4.2 convince us that the sum of the degrees of the vertices is twice the number of edges. How can we prove this? One way is to count the

[2]Again, the terminology we are using here is the most popular terminology in computer science, but what we just defined as a path would be called a walk by most graph theorists.

[3]Most graph theorists reserve the word path for what we are calling a simple path, but again we are using the language most popular in computer science.

total number of incidences between vertices and edges (similar to counting the horses hooves in the hint). Each edge has exactly two incidences, so the total number of incidences is twice the number of edges. But the degree of a vertex is the number of incidences it has, so the sum of the degrees of the vertices is also the total number of of incidences. Therefore the sum of the degrees of the vertices of a graph is twice the number of edges. Thus to compute the number of edges of a graph, we can sum the degrees of the vertices and divide by two. (In the case of the hint, the horses correspond to edges and the hooves to endpoints.) There is another proof of this result that uses induction.

Theorem 4.6 *Suppose a graph has a finite number of edges. Then the sum of the degrees of the vertices is twice the number of edges.*

Proof: We induct on the number of edges of the graph. If a graph has no edges, then each vertex has degree zero and the sum of the degrees is zero, which is twice the number of edges. Now suppose $e > 0$ and the theorem is true whenever a graph has fewer than e edges. Let G be a graph with e edges and let ϵ be an edge of G.[4] Let G' be the graph (on the same vertex set as G) we get by deleting ϵ from the edge set E of G. Then G has $e-1$ edges, and so by our inductive hypothesis, the sum of the degrees of the vertices of G' is twice $e-1$. Now there are two possible cases. Either e was a loop, in which case one vertex of G' has degree two less in G' than it has in G. Otherwise e has two distinct endpoints, in which case exactly two vertices of G' have degree one less than their degree in G. Thus in both cases the sum of the degrees of the vertices in G' is two less than the sum of the degrees of the vertices in G, so the sum of the degrees of the vertices in G is $(2e-2)+2 = 2e$. Thus the truth of the theorem for graphs with $e-1$ edges implies the truth of the theorem for graphs with e edges. Therefore, by the principle of mathematical induction, the theorem is true for a graph with any finite number of edges. ■

There are a couple instructive points in the proof of the theorem. First, since it wasn't clear from the outset whether we would need to use strong or weak induction, we made the inductive hypothesis we would normally make for strong induction. However in the course of the proof, we saw that we only needed to use weak induction, so that is how we wrote our conclusion. This is not a mistake, because we used our inductive hypothesis correctly. We just didn't need to use it for every possible value it covered.

Second, instead of saying that we would take a graph with $e-1$ edges and add an edge to get a graph with e edges, we said that we would take a graph with e edges and remove an edge to get a graph with $e-1$ edges. This is because we need to prove that the result holds for *every* graph with e edges. By using the second approach we avoided the need to say that "every graph with e edges may be built up from a graph with $e-1$ edges by adding an edge," because in the second approach we started with an arbitrary graph on e edges. In the first approach, we would have proved that the theorem was true for all graphs that could be built from an $e-1$ edge graph by adding an edge, and we would have had to explicitly say that every graph with e edges could be built in this way.

In Exercise 3 the sum of the degrees of the vertices is (working from left to right)

$$2+4+5+6+5+2+5+4+2 = 40,$$

and so the graph has 20 edges.

[4]Since it is very handy to have e stand for the number of edges of a graph, we will use Greek letters such as epsilon (ϵ) to stand for edges of a graph. It is also handy to use v to stand for the number of vertices of a graph, so we use other letters near the end of the alphabet, such as w, x, y,and z to stand for vertices.

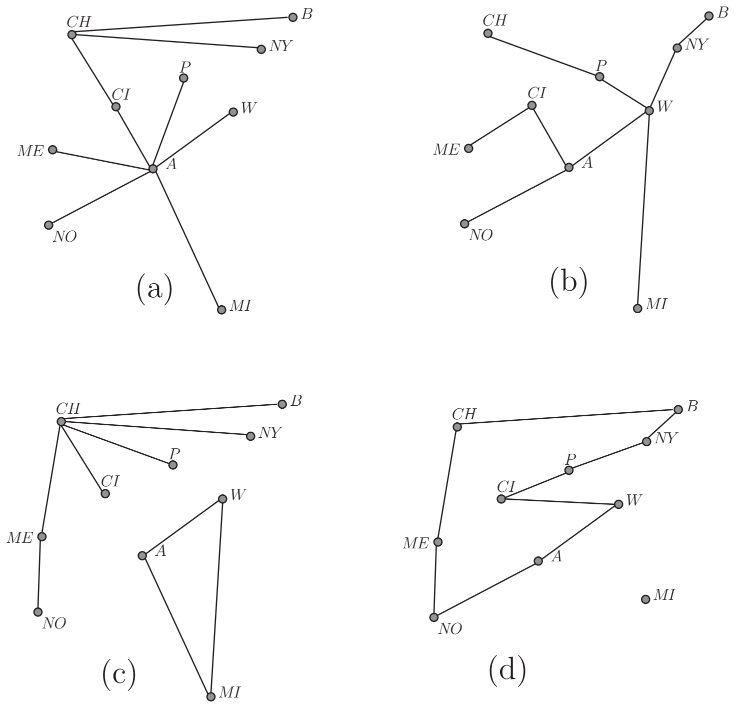
Connectivity

All of the examples we have seen so far have a property that is not common to all graphs, namely that there is a path from every vertex to every other vertex.

Exercise 4.3-5 The company with the computer network in Figure 4.1 needs to reduce its expenses. It is currently leasing each of the communication lines shown in the Figure. Since it can send information from one city to another through one or more intermediate cities, it decides to only lease the minimum number of communication lines it needs to be able to send a message from any city to any other city by using any number of intermediate cities. What is the minimum number of lines it needs to lease? Give two examples of subsets of the edge set with this number of edges that will allow communication between any two cities and two examples of a subset of the edge set with this number of edges that will not allow communication between any two cities.

Some experimentation with the graph convinces that if we keep eight or fewer edges, there is no way we can communicate among the cities (we will explain this more precisely later on), but that there are quite a few sets of nine edges that suffice for communication among all the cities. In Figure 4.3 we show two sets of nine edges each that allow us to communicate among all the cities and two sets of nine edges that do not allow us to communicate among all the cities.

Figure 4.3: Selecting nine edges from the stylized map of some eastern US cities.

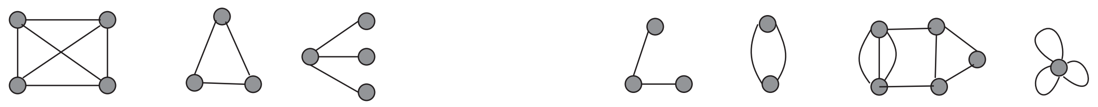


Notice that in graphs (a) and (b) it is possible to get from any vertex to any other vertex by a path. A graph is called *connected* there is a path between each two vertices of the graph.

Notice that in graph (c) it is not possible to find a path from Atlanta to Boston, for example, and in graph (d) it is not possible to find a path from Miami to any of the other vertices. Thus these graphs are not connected; we call them disconnected.

We say two vertices are *connected* if there is a path between them, so a graph is connected if each two of its vertices are connected. Thus in Graph (c) the vertices for Boston and New Orleans are connected. The relationship of being connected is an equivalence relation (in the sense of Section 1.4). To show this we would have to show that this relationship divides the set of vertices up into mutually exclusive classes; that is, that it partitions the vertices of the graph. The class containing Boston, for example is all vertices connected to Boston. If two vertices are in that set, they both have paths to Boston, so there is a path between them using Boston as an intermediate vertex. If a vertex x is in the set containing Boston and another vertex y is not, then they cannot be connected or else the path from y to x and then on to Boston would connect y to Boston, which would mean y was in the class containing Boston after all. Thus the relation of being connected partitions the vertex set of the graph into disjoint classes, so it is an equivalence relation. Though we made this argument with respect to the vertex Boston in the specific case of graph (c) of Figure **??**, it is a perfectly general argument that applies to arbitrary vertices in arbitrary graphs. We call the equivalence relation of being connected to the *connectivity* relation. There can be no edge of a graph between two vertices in different equivalence classes of the connectivity relation because then everything in one class would be connected to everything in the other class, so the two classes would have to be the same. Thus we also end up with a partition of the edges into disjoint sets. If a graph has edge set E, and C is an equivalence class of the connectivity relation, then we use $E(C)$ to denote the set of edges whose endpoints are both in C. Since no edge connects vertices in different equivalence classes, each edge must be in some set $E(C)$. The graph consisting of an equivalence class C of the connectivity relation together with the edges $E(C)$ is called a *connected component* of our original graph. From now on our emphasis will be on connected components rather than on equivalence classes of the connectivity relation. Notice that graphs (c) and (d) of Figure 4.3 each have two connected components. In graph (c) the vertex sets of the connected components are $\{NO, ME, CH, CI, P, NY, B\}$ and $\{A, W, MI\}$. In graph (d) the connected components are $\{NO, ME, CH, B, NY, P, CI, W, A\}$ and $\{MI\}$. Two other examples of graphs with multiple connected components are shown in Figure 4.4.

Figure 4.4: A simple graph with three connected components and a graph with four connected components.



Cycles

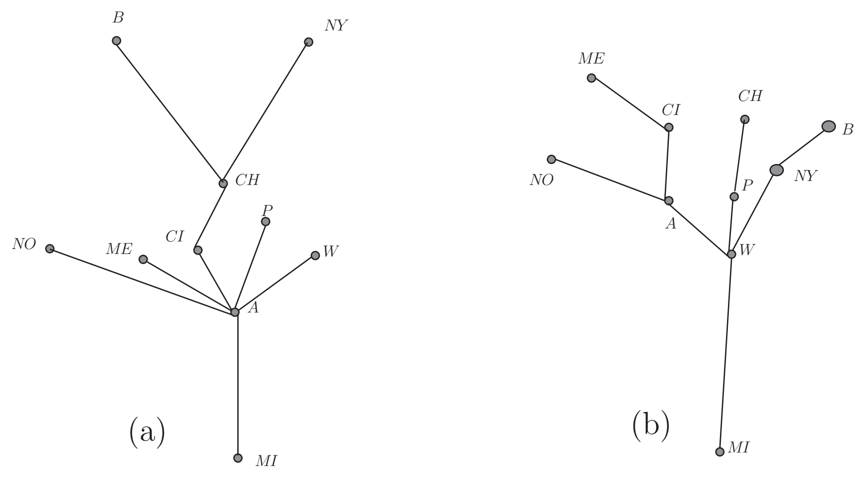
In graphs (c) and (d) of Figure 4.3 we see a feature that we don't see in graphs (a) and (b), namely a path that leads from a vertex back to itself. A path that starts and ends at the same vertex is called a *closed path*. A closed path with at least one edge is called a *cycle* if, except for the last vertex, all of its vertices are different. The closed paths we see in graphs (c) and (d) of Figure 4.3 are cycles. Not only do we say that $\{NO, ME, CH, B, NY, P, CI, W, A, NO\}$ is a cycle in in graph (d) of Figure 4.3, but we also say it is a cycle in the graph of Figure 4.1. The way we distinguish between these situations is to say the cycle $\{NO, ME, CH, B, NY, P, CI, W, A, NO\}$ is an induced cycle in Figure 4.3 but not in Figure 4.1. More generally, a graph H is called a *subgraph* of the graph G if all the vertices and edges of H are vertices and edges of G, and we call H an *induced subgraph* of G if every vertex of H is a vertex of G, and every edge of G connecting vertices of H is an edge of H. Thus the first graph of Figure 4.4 has an induced K_4 and an induced cycle on three vertices.

We don't normally distinguish which point on a cycle really is the starting point; for example we consider the cycle $\{A, W, MI, A\}$ to be the same as the cycle $\{W, MI, A, W\}$. Notice that there are cycles with one edge and cycles with two edges in the second graph of Figure 4.4. We call a graph G a *cycle* on n vertices or an n-cycle and denote it by C_n if it has a cycle that contains all the vertices and edges of G and a *path* on n vertices and denote it by P_n if it has a path that contains all the vertices and edges of G. Thus drawing (a) of Figure 4.2 is a drawing of C_4. The second graph of Figure 4.4 has an induced P_3 and an induced C_2 as subgraphs.

Trees

The graphs in parts (a) and (b) of Figure 4.3 are called trees. We have redrawn them slightly in Figure 4.5 so that you can see why they are called trees. We've said these two graphs are called

Figure 4.5: A visual explanation of the name tree.



trees, but we haven't given a definition of trees. In the examples in Figure 4.3, the graphs we have called trees are connected and have no cycles.

Definition 4.1 *A connected graph with no cycles is called a tree.*

Other Properties of Trees

In coming to our definition of a tree, we left out a lot of other properties of trees we could have discovered by a further analysis of Figure 4.3

Exercise 4.3-6 Given two vertices in a tree, how many distinct simple paths can we find between the two vertices?

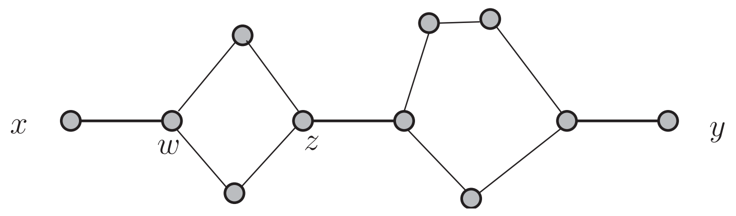
Exercise 4.3-7 Is it possible to delete an edge from a tree and have it remain connected?

Exercise 4.3-8 How many edges does a tree with v vertices have?

Exercise 4.3-9 Does every tree have a vertex of degree 1? If the answer is yes, explain why. If the answer is no, try to find additional conditions that will guarantee that a tree satisfying these conditions has a vertex of degree 1.

For Exercise 4.3-6, suppose we had two distinct paths from a vertex x to a vertex y. They begin with the same vertex x and might have some more edges in common as in Figure 4.6. Let w be the last vertex after (or including) x the paths share before they become different. The paths must come together again at y, but they might come together earlier. Let z be the first vertex the paths have in common after w. Then there are two paths from w to z that have only w and z in common. Taking one of these paths from w to z and the other from z to w gives us a cycle, and so the graph is not a tree. We have shown that if a graph has two distinct paths from x to y, then it is not a tree. By contrapositive inference, then, if a graph is a tree, it does not have two distinct paths between two vertices x and y. We state this result as a theorem.

Figure 4.6: A graph with multiple paths from x to y.



Theorem 4.7 *There is exactly one path between each two vertices in a tree.*

Proof: By the definition of a tree, there is at least one path between each two vertices. By our argument above, there is at most one path between each two vertices. Thus there is exactly one path. ■

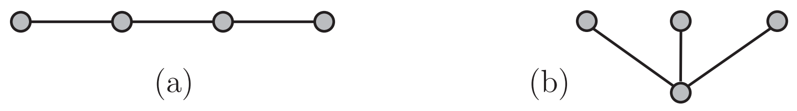
For Exercise 4.3-7, note that if ϵ is an edge from x to y, then x, ϵ, y is the unique path from x to y in the tree. Suppose we delete ϵ from the edge set of the tree. If there were still a path from x to y in the resulting graph, it would also be a path from x to y in the tree, which would contradict Theorem 4.7. Thus the only possibility is that there is no path between x and y in the resulting graph, so it is not connected and is therefore not a tree. In fact, you may have guessed an even stronger statement that we give in the lemma that follows.

Lemma 4.8 *Removing one edge from the edge set of a tree gives a graph with two connected components, each of which is a tree.*

Proof: Suppose as before the lemma that ϵ is an edge from x to y. We have seen that the graph G we get by deleting ϵ from the edge set of the tree is not connected, so it has at least two connected components. If w is in the tree, there is a unique path from w to y by Theorem 4.7. Either ϵ is an edge of that path or it is not. If ϵ is on the path, then it must be the last edge of the path since y is its endpoint and is the last vertex of the path. Thus if ϵ is on the path, there is a unique path from w to x that does not use ϵ. Therefore in the graph G we get by deleting ϵ from the edge set of the tree, there is a unique path from w to x. Otherwise ϵ is not on the path from w to y and so in the graph G we get by deleting ϵ, there is a path from w to y. Thus each vertex in G is connected by a path to either x or y. Therefore G has exactly two connected components. Since neither has any cycles, both are trees. ■

In Exercise 4.3-8, our trees with ten vertices had nine edges. If we draw a tree on two vertices it will have one edge; if we draw a tree on three vertices it will have two edges. There are two different looking trees on four vertices as shown in Figure 4.7, and each has three edges. On the

Figure 4.7: Two trees on four vertices.



basis of these examples we conjecture that a tree on n vertices has $n-1$ edges. One approach to proving this is to try to use induction. To do so, we have to see how to build up every tree from smaller trees or how to take a tree and break it into smaller ones. Then in either case we have to figure out how use the truth of our conjecture for the smaller trees to imply its truth for the larger trees. A mistake that people often make at this stage is to assume that every tree can be built from smaller ones by adding a vertex of degree 1. While that is true for finite trees with more than one vertex (which is the point of Exercise 4.3-9), we haven't proved it yet, so we can't yet use it in proofs of other theorems. Another approach to using induction is to ask whether there is a natural way to break a tree into two smaller trees. There is: we just showed in Lemma 4.8 that if you remove an edge ϵ from the edge set of a tree, you get two connected components that are trees. We may assume inductively that the number of edges of each of these trees is one less than its number of vertices. Thus if the graph with these two connected components has v vertices, then it has $v-2$ edges. Adding ϵ back in gives us a graph with $v-1$ edges, so except for the fact that we have not done a base case, we have proved the following theorem.

Theorem 4.9 *For all integers $v \geq 1$, a tree with v vertices has $v-1$ edges.*

Proof: If a tree has one vertex, it can have no edges, for any edge would have to connect that vertex to itself and would thus give a cycle. A tree with two or more vertices must have an edge in order to be connected. We have shown before the statement of the theorem how to use the deletion of an edge to complete an inductive proof that a tree with v vertices has $v-1$ edges, and so for all $v \geq 1$, a tree with v vertices has $v-1$ edges. ■

Finally, for Exercise 4.3-9 we can now give a contrapositive argument to show that a finite tree with more than one vertex has a vertex of degree one. Suppose instead that G is a graph that is connected and all vertices of G have degree two or more. Then the sum of the degrees of the vertices is at least 2v, and so by Theorem 4.6 the number of edges is at least v. Therefore by Theorem 4.9 G is not a tree. Then by contrapositive inference, if T is a tree, then T must have at least one vertex of degree one. This corollary to Theorem 4.9 is so useful that we state it as a corollary.

Corollary 4.10 *A finite tree with more than one vertex has at least one vertex of degree one.*

Important Concepts, Formulas, and Theorems

1. *Graph.* A *graph* consists of a set of *vertices* and a set of *edges* with the property that each edge has two (not necessarily different) vertices associated with it and called its *endpoints*.

2. *Edge; Adjacent.* We say an edge in a graph *joins* its endpoints, and we say two endpoints are *adjacent* if they are joined by an edge.

3. *Incident.* When a vertex is an endpoint of an edge, we say the edge and the vertex are *incident*.

4. *Drawing of a Graph.* To *draw* a graph, we draw a point in the plane for each vertex, and then for each edge we draw a (possibly curved) line between the points that correspond to the endpoints of the edge. Lines that correspond to edges may only touch the vertices that are their endpoints.

5. *Simple Graph.* A *simple graph* is one that has at most one edge joining each pair of distinct vertices, and no edges joining a vertex to itself.

6. *Length, Distance.* The *length* of a path is the number of edges. The *distance* between two vertices in a graph is the length of a shortest path between them.

7. *Loop; Multiple Edges.* An edge that joins a vertex to itself is called a loop and we say we have multiple edges between vertices x and y if there is more than one edge joining x and y.

8. *Notation for a Graph.* We use the phrase "Let $G = (V, E)$" as a shorthand for "Let G stand for a graph with vertex set V and edge set E.

9. *Notation for Edges.* In a simple graph we use the notation $\{x, y\}$ for an edge from x to y. In any graph, when we want to use a letter to denote an edge we use a Greek letter like ϵ so that we can save e to stand for the number of edges of the graph.

10. *Complete Graph on n vertices.* A *complete graph* on n vertices is a graph with n vertices that has an edge between each two of the vertices. We use K_n to stand for a complete graph on n vertices.

11. *Path.* We call an alternating sequence of vertices and edges in a graph a *path* if it starts and ends with a vertex, and each edge joins the vertex before it in the sequence to the vertex after it in the sequence.

12. *Simple Path.* A path is called a *simple path* if it has no repeated vertices or edges.

13. *Degree of a Vertex.* The *degree* of a vertex in a graph is the number of times it is incident with edges of the graph; that is, the degree of a vertex x is the number of edges from x to other vertices plus twice the number of loops at vertex x.

14. *Sum of Degrees of Vertices* The sum of the degrees of the vertices in a graph with a finite number of edges is twice the number of edges.

15. *Connected.* A graph is called *connected* there is a path between each two vertices of the graph. We say two vertices are *connected* if there is a path between them, so a graph is connected if each two of its vertices are connected. The relationship of being connected is an equivalence relation on the vertices of a graph.

16. *Connected Component* If C is a subset of the vertex set of a graph, we use $E(C)$ to stand for the set of all edges *both* of whose endpoints are in C. The graph consisting of an equivalence class C of the connectivity relation together with the edges $E(C)$ is called a *connected component* of our original graph.

17. *Closed Path.* A path that starts and ends at the same vertex is called a *closed path.*

18. *Cycle.* A closed path with at least one edge is called a *cycle* if, except for the last vertex, all of its vertices are different.

19. *Tree.* A connected graph with no cycles is called a tree.

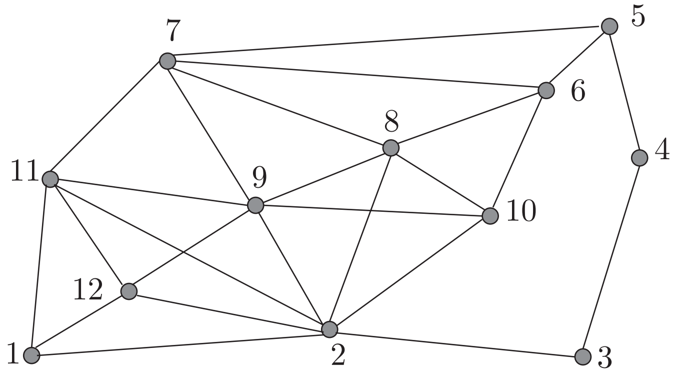
20. *Important Properties of Trees*

    (a) There is a unique path between each two vertices in a tree.
    (b) A tree on v vertices has $v - 1$ edges.
    (c) Every finite tree with at least two vertices has a vertex of degree one.

Problems

1. Find the shortest path you can from vertex 1 to vertex 5 in Figure 4.8.

Figure 4.8: A graph.



2. Find the longest simple path you can from vertex 1 to vertex 5 in Figure 4.8.

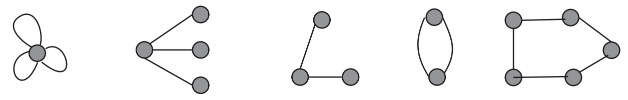
3. Find the vertex of largest degree in Figure 4.8. What is it's degree?

4. How many connected components does the graph in Figure 4.9 have?

5. Find all induced cycles in the graph of Figure 4.9.

6. What is the size of the largest induced K_n in Figure 4.9?

Figure 4.9: A graph with a number of connected components.



7. Find the largest induced K_n (in words, the largest complete subgraph) you can in Figure 4.8.

8. Find the size of the largest induced P_n in the graph in Figure 4.9.

9. A graph with no cycles is called a *forest*. Show that if a forest has v vertices, e edges, and c connected components, then $v = e + c$.

10. What can you say about a five vertex simple graph in which every vertex has degree four?

11. Find a drawing of K_6 in which only three edges cross.

12. Either prove true or find a counter-example. A graph is a tree if there is one and only one simple path between each pair of vertices.

13. Is there some number m such that if a graph with v vertices is connected and has m edges, then it is a tree? If so, what is m in terms of v?

14. Is there some number m such that a graph on n vertices is a tree if and only if it has m edges and has no cycles.

15. Suppose that a graph G is connected, but for each edge, deleting that edge leaves a disconnected graph. What can you say about G? Prove it.

16. Show that each tree on four vertices can be drawn with one of the two drawings in Figure 4.7.

17. Draw the minimum number of drawings of trees you can so that each tree on five vertices has one of those drawings. Explain why you have drawn all possible trees.

18. Draw the minimum number of drawings of trees you can so that each tree on six vertices has one of those drawings. Explaining why you have drawn all possible drawings is optional.

19. Find the longest induced cycle you can in Figure 4.8.