Chapter 12: Graph Theory

Figure 1: Networks connect cities around the globe.

In this chapter, you will learn the fundamental skills needed to work with graphs used in an area of mathematics known as graph theory. You can think of these graphs as a kind of map. Maps have served many purposes over the course of history. You probably use GPS maps to navigate to various destinations. A scientist from ancient Greece named Ptolemy wanted an accurate map of the world to make more accurate astrological predications. In recent years, neurobiologists have mapped the cerebral cortex to better understand the human brain. Social network analysts map online interactions to assist advertisers in reaching target audiences. Like other maps, the graphs you will study in this chapter can serve many purposes, but they do not have a lot of the details you might expect in a map such as size, shape, and distance between objects. All of that is stripped away so that we can focus on one element of maps, the connections between objects.

12.1 Graph Basics

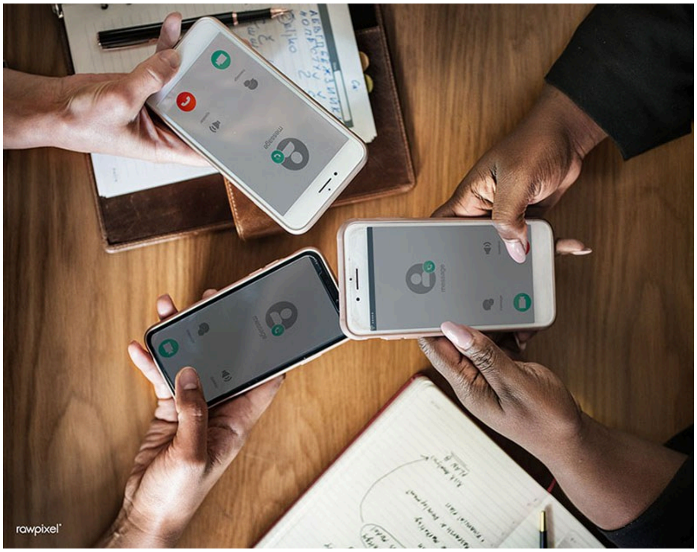


Figure 2: Cell phone networks connect individuals.

Learning Objectives

After completing this section, you should be able to:

1. Identify parts of a graph.
2. Model applications of graph basics.

When you hear the word, *graph*, what comes to mind? You might think of the xy-coordinate system you learned about earlier in this course, or you might think of the line graphs and bar charts that are used to display data in news reports. The graphs we discuss in this chapter are probably very different from what you think of as a graph. They look like a bunch of dots connected by short line segments. The dots represent a group of objects and the line segments represent the connections, or relationships, between them. The objects might be bus stops, computers, Snapchat accounts, family members, or any other objects that have direct connections to each other. The connections can be physical or virtual, formal or casual, scientific or social. Regardless of the kind of connections, the purpose of the graph is to help us visualize the structure of the entire network to better understand the interactions of the objects within it.

Parts of a Graph

In a **graph**, the objects are represented with dots and their connections are represented with lines like those in . displays a **simple graph** labeled G and a **multigraph** labeled H. The dots are called **vertices** an individual dot is a **vertex**, which is one object of a set of objects, some of which may be connected. We often label vertices with letters. For example, Graph G has vertices a, b, c, and d, and Multigraph H has vertices, e, f, g, and h. Each line segment or connection joining two vertices is referred to as an **edge**. H is considered a multigraph because it has a double edge between *f* and *h*, and a double edge between *h* and *g*. Another reason *H* is called a multigraph is that it has a **loop** connecting vertex *e* to itself; a loop is an edge that joins a vertex to itself. Loops and double edges are not allowed in a simple graph.

To sum up, a simple graph is a collection of vertices and any edges that may connect them, such that every edge connects two vertices with no loops and no two vertices are joined by more than one edge. A multigraph is a graph in which there may be loops or pairs of vertices that are joined by more than one edge. In this chapter, most of our work will be with simple graphs, which we will call graphs for convenience.

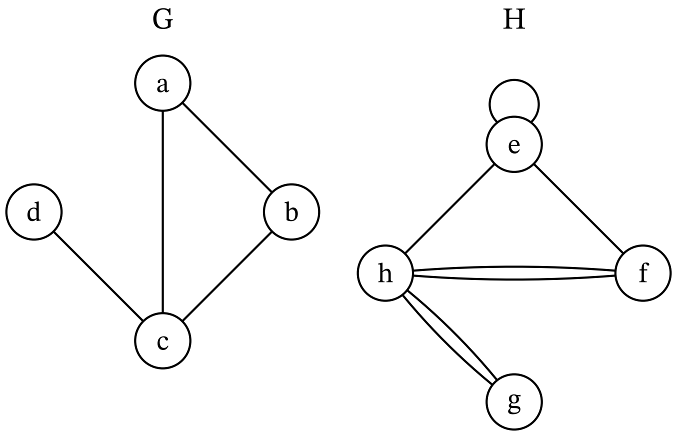


Figure 3: A Graph and a Multigraph

It is not necessary for the edges in a graph to be straight. In fact, you can draw an edge any way you want. In graph theory, the focus is on which vertices are connected, not how the connections are drawn. In a graph, each edge can be named by the two letters of the associated vertices. The four edges in Graph *X* in are *ab, ac, ad*, and *ae*. The order of the letters is not important when you name the edge of a graph. For example, *ab* refers to the same edge as *ba*.

CHECKPOINT

A graph may have vertices that are not joined to other vertices by edges, such as vertex f in Graph X in , but any edge must have a vertex at each end.

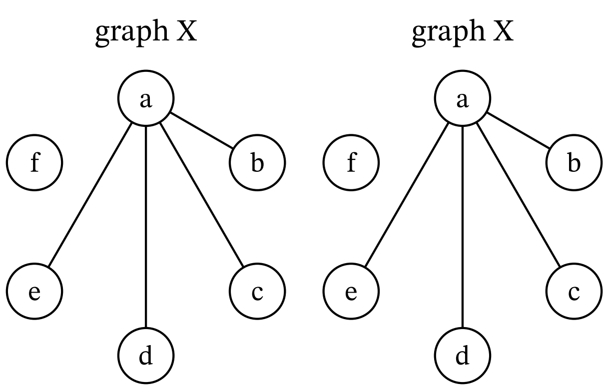


Figure 4: Different Representations of the Same Graph

Example 1 **Identifying Edges and Vertices**

Name all the vertices and edges of graph *F* in .

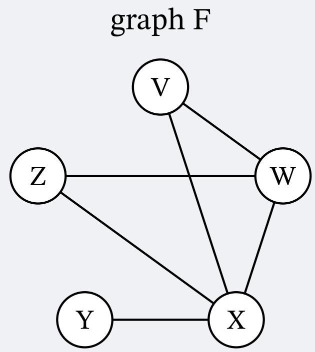


Figure 5: Graph *F*

Solution

The vertices are *v, w, x, y*, and *z*. The edges are *vw, vx, wx, wz, xy*, and *xz*.

CHECKPOINT

When listing the vertices and edges in a graph, work in alphabetical order to avoid accidentally listing the same item twice. When you are finished, count the number of vertices or edges you listed and compare that to the number of vertices or edges on the graph to ensure you didn't miss any.

Since the purpose of a graph is to represent the connections between objects, it is very important to know if two vertices share a common edge. The two vertices at either end of a given edge are referred to as **neighboring**, or **adjacent**. For example, in , vertices *x* and *w* are adjacent, but vertices *y* and *w* are not.

Example 2 **Identifying Vertices That Are Not Adjacent**

Name all the pairs of vertices of graph *F* in that are *not* adjacent.

Solution

The pairs of vertices that are not adjacent in graph *F* are *v* and *y, v* and *z, w* and *y*, and *y* and *z*.

PEOPLE IN MATHEMATICS

Sergey Brin and Laurence Page

The "Google boys," Sergey Brin and Laurence Page, transformed the World Wide Web in 1998 when they used the mathematics of graph theory to create an algorithm called Page Rank, which is known as the Google Search Engine today. The two computer scientists identified webpages as vertices and hyperlinks on those pages as edges because hyperlinks connect one website to the next. The number of edges influences the ranking of a website on the Google Search Engine because the websites with more links to "credible sources" are ranked higher. ("Page Rank: The Graph Theory-based Backbone of Google," September 20, 2011, Cornell University, Networks Blog.

Analyzing Geographical Maps with Graphs

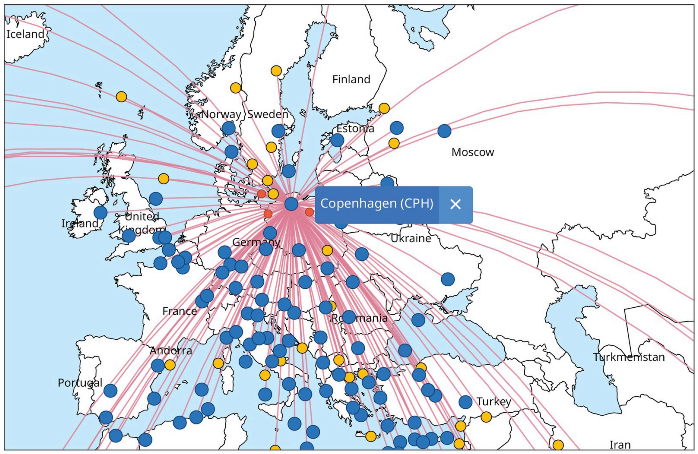


Figure 6: Commercial airlines' route systems create a global network.

When graphs are used to model and analyze real-world applications, the number of edges that meet at a particular vertex is important. For example, a graph may represent the direct flight connections for a particular airport as in . Representing the connections with a graph rather than a map shifts the focus away from the relative positions and toward which airports are connected. In , the vertices are the airports, and the edges are the direct flight paths. The number of flight connections between a particular airport and other South Florida airports is the number of edges meeting at a particular vertex. For example, Key West has direct flights to three of the five airports on the graph. In graph theory terms, we would say that vertex FYW has **degree** 3. The degree of a vertex is the number of edges that connect to that vertex.

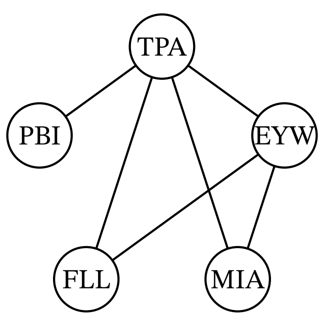


Figure 7: Direct Flights between South Florida Airports

Example 3 **Determining the Degree of a Vertex**

Determine the degree of each vertex of Graph J in . If graph J represents direct flights between a set of airports, do any of the airports have direct flights to two or more of the other cities on the graph?

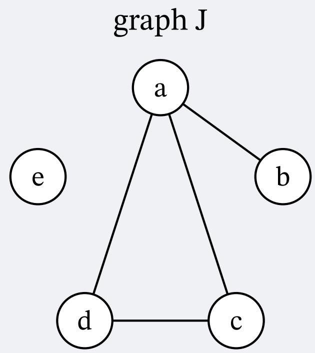


Figure 8: Graph J

Solution

For each vertex, count the number of edges that meet at that vertex. This value is the degree of the vertex. In , the dashed edges indicate the edges that meet at the marked vertex.

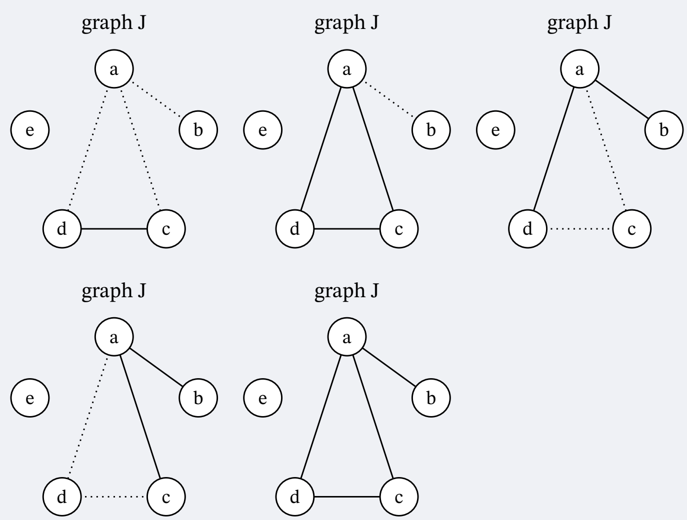


Figure 9: Degrees of Vertices of Graph J

Vertex a has degree 3, vertex b has degree 1, vertices c and d each have degree 2, and vertex e has degree 0. Airports a, c, and d have direct flights to two or more of the other airports.

Graphs are also used to analyze regional boundaries. The states of Utah, Colorado, Arizona, and New Mexico all meet at a single point known as the "Four Corners," which is shown in the map in .

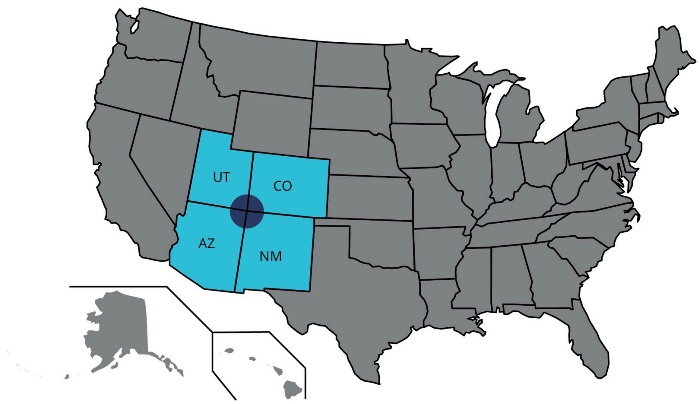


Figure 10: Map of the Four Corners

In , each vertex represents one of these states, and each edge represents a shared border. States like Utah and New Mexico that meet at only a single point are *not* considered to have a shared border. By representing this map as a graph, where the connections are shared borders, we shift our perspective from physical attributes such as shape, size and distance, toward the existence of the relationship of having a shared boundary.

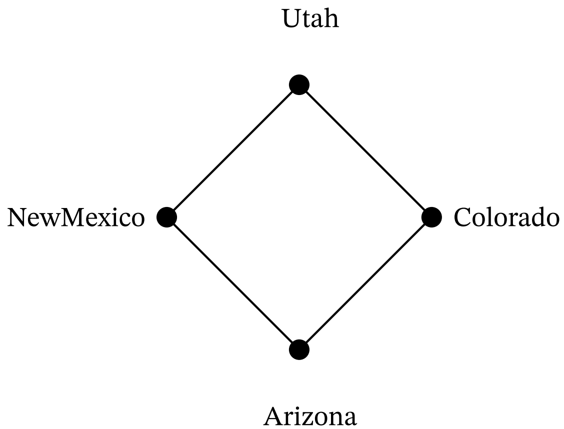


Figure 11: Graph of the Shared Boundaries of Four Corners States

Example 4 **Graphing the Midwestern States**

A map of the Midwest is given in . Create a graph of the region in which each vertex represents a state and each edge represents a shared border.

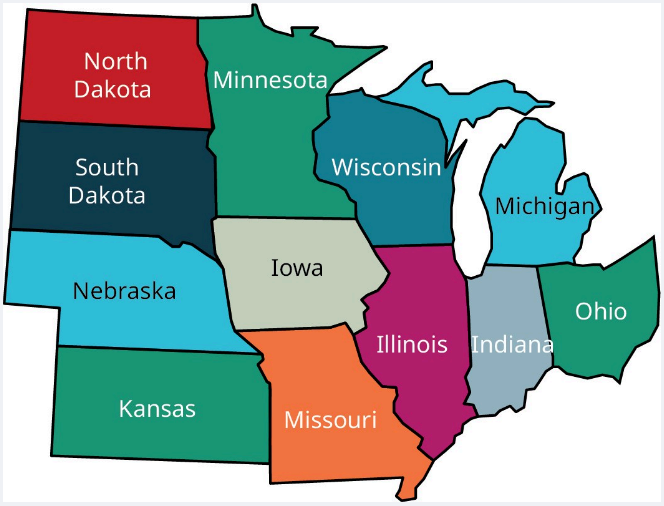


Figure 12: Map of Midwestern States

Solution

Step 1: For each state, draw and label a vertex as in .

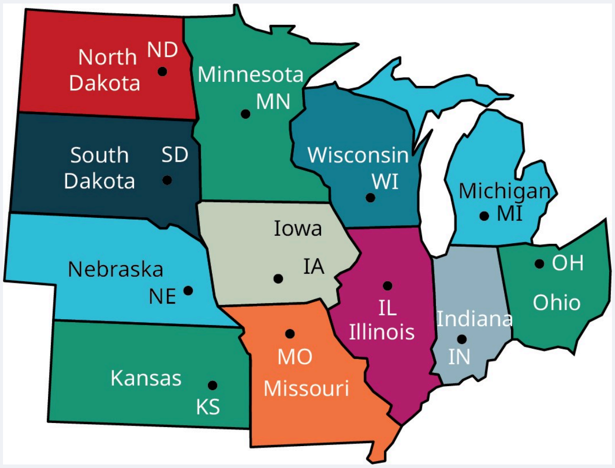


Figure 13: Vertex Assigned to Each Midwestern State

Step 2: Draw edges between any two states that share a common land border as in .

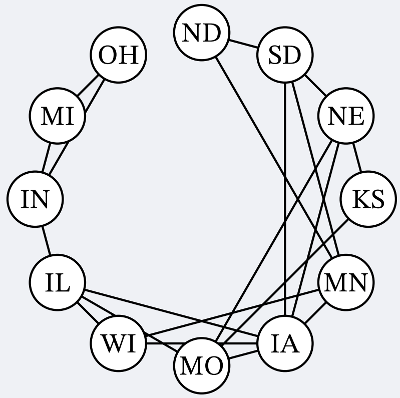


Figure 14: Edge Assigned to Each Pair of Midwestern States with Common Border

The graph is given in .

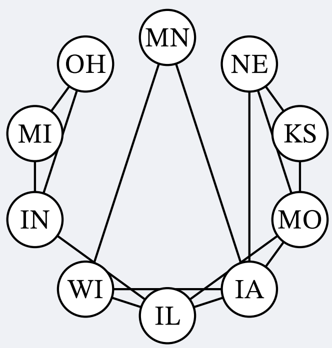


Figure 15: Final Graph Representing Common Boundaries between Midwestern States

VIDEO

Graph Theory: Create a Graph to Represent Common Boundaries on a Map

Graphs of Social Interactions

Geographical maps are just one of many real-world scenarios which graphs can depict. Any scenario in which objects are connected to each other can be represented with a graph, and the connections don't have to be physical. Just think about all the connections you have to people around the world through social media! Who is in your network of Twitter followers? Whose Snapchat network are you connected to?

Example 5 **Graphing Chloe's *Roblox* Friends**

Roblox is an online gaming platform. Chloe is interested to know how many people in her network of *Roblox* friends are also friends with each other so she polls them. Explain how a graph or multigraph might be drawn to model this scenario by identifying the objects that could be represented by vertices and the connections that could be represented by edges. Indicate whether a graph or a multigraph would be a better model.

Solution

The objects that are represented with vertices are *Roblox* friends. A *Roblox* friendship between two friends will be represented as an edge between a pair of vertices. There will be no double edges because it is not possible for two friends to be linked twice in *Roblox* they are either friends or they are not. Also, a player cannot be a friend to themself, so there is no need for a loop. Since there are no double edges or loops, this is best represented as a graph.

WHO KNEW?

Using Graph Theory to Reduce Internet Fraud

Could graphs be used to reduce Internet fraud? At least one researcher thinks so. Graph theory is used every day to analyze our behavior, particularly on social network sites. Alex Buetel, a computer scientist from Carnegie Mellon University in Pittsburgh, Pennsylvania, published a research paper in 2016 that discussed the possibilities of distinguishing the normal interactions from those that might be fraudulent using graph theory. Buetel wrote, “To more effectively model and detect abnormal behavior, we model *how* fraudsters work, catching previously undetected fraud on Facebook, Twitter, and Tencent Weibo and improving classification accuracy by up to 68%.” In the same paper, the researcher discusses how similar techniques can be used to model many other applications and even, “predict *why* you like a particular movie.” (Alex Beutel, “User Behavior Modeling with Large-Scale Graph Analysis,” http://reports-archive.adm.cs.cmu.edu/anon/2016/CMU-CS-16-105.pdf, May 2016, CMU-CS-16-105, Computer Science Department, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA)

Key Terms

- vertex
- edge
- loop
- graph (simple graph)
- multigraph
- adjacent (neighboring)
- degree

Key Concepts

- Graphs and multigraphs represent objects as vertices and the relationships between the objects as edges.
- The degree of a vertex is the number of edges that meet it and the degree can be zero.

- An edge must have a vertex at each end.
- Multigraphs may contain loops and double edges, but simple graphs may not.

Videos

- Graph Theory: Create a Graph to Represent Common Boundaries on a Map

12.2 Graph Structures

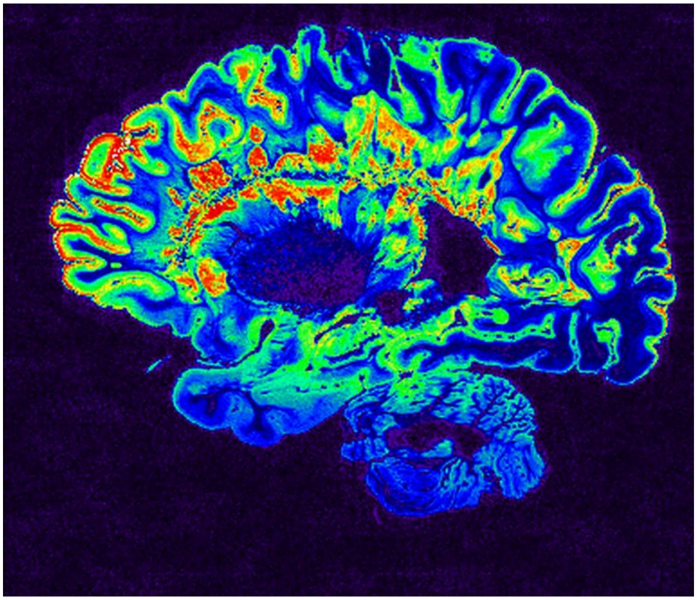


Figure 16: Neuroimaging shows brain activity.

Learning Objectives

After completing this section, you should be able to:

1. Describe and interpret relationships in graphs.
2. Model relationships with graphs.

   Graph theory is used in neuroscience to study how different parts of the brain connect. Neurobiologists use functional magnetic resonance imaging (fMRI) to measure levels of blood in different parts of the brain, called nodes. When nodes are active at the same time, it suggests there is a functional connection between them so they form a network. This network can be represented as a graph where the vertices are the nodes and the functional connections are the edges between them. (Mikey Taylor, "Graph Theory & Machine Learning in Neuroscience," Medium.com, June 24, 2020.

Importance of the Degrees of Vertices

One reason scientists study these networks is to determine how successful the communication within a network continues to be when it experiences failures in nodes and connections. Graphs can be used to study the resilience of these networks. (Mikey Taylor, “Graph Theory & Machine Learning in Neuroscience,” Medium.com, June 24, 2020)

Example 1 **Using Graphs to Understand Relationships**

The graphs in represent neural networks, where the vertices are the nodes, and the edges represent functional connections between them. Which graph do you think would represent a network with the highest resistance to failure? Which graph would probably be the most vulnerable to failure? How might this relate to the degrees of the vertices?

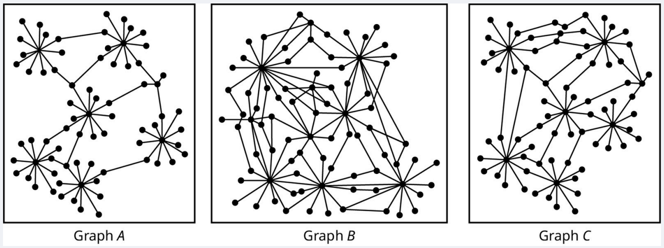


Figure 17: Models of Neural Networks

Solution

Graph B represents a network that would have the highest resistance to failure because there are more edges connecting the nodes indicating there are more connections between nodes than on either of the other graphs. This would make the network more resistant to failure because there are more options to work around a faulty edge or node. Graph A would probably be the most susceptible to failure because the network depends on a few particular edges and vertices for connections. There are more vertices of higher degree in Graph B than in Graph A because there are more edges connecting the nodes.

Relating the Number of Edges to the Degrees of Vertices

In the applications of graph theory to neuroscience and sociology in and YOUR TURN 12.6, there was a correlation between the degrees of vertices and the resilience of a network. Researchers in many fields have also observed a direct relationship between the number of edges in a graph and the degrees of the vertices. To begin to understand this relationship, consider a graph with five vertices and zero edges as in .

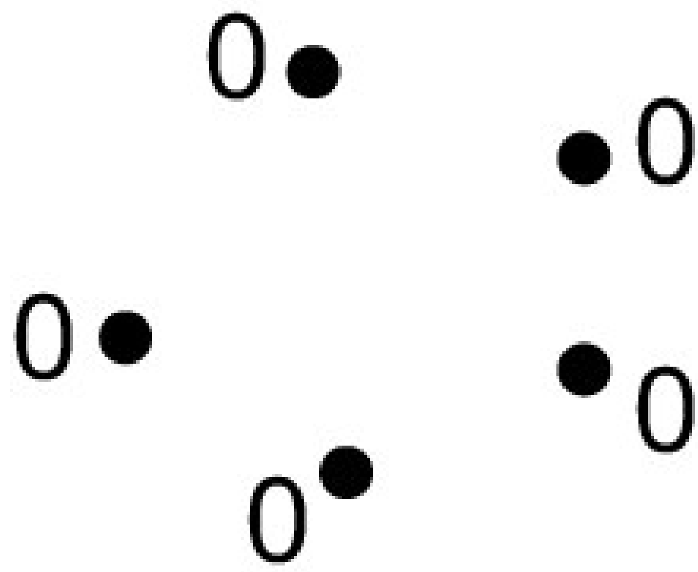


Figure 18: Graph with Five Vertices and zero Edges

Instead of being marked with a name, each vertex in is marked with its degree. In this case, all of the degrees are 0 so the sum of the degrees is also zero. Suppose that we add an edge between any two existing vertices and indicate the degrees of the vertices. This gives us a graph with five vertices and one edge like the graph in .

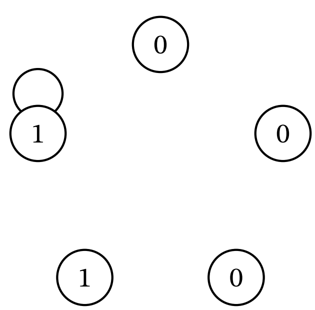


Figure 19: Graph with Five Vertices and One Edge

Note that the degrees of two vertices increased, each by 1. So, the sum of the degrees is now 2. Suppose that we continue in this way, adding one edge at a time and making note of the number of edges and the sum of the degrees of the vertices as in .

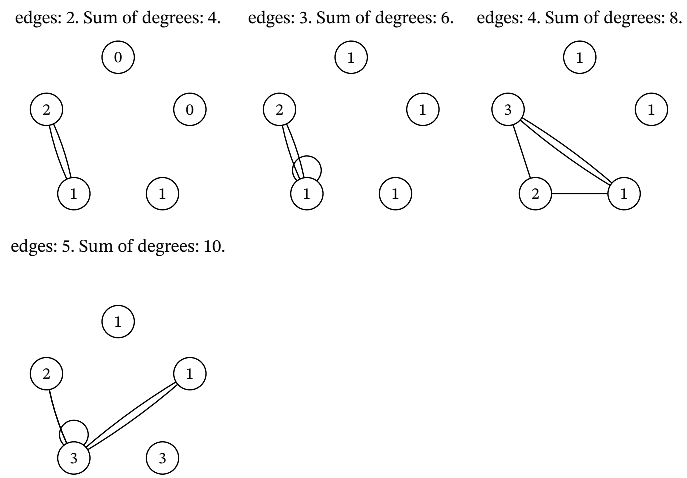


Figure 20: Comparing Number of Edges to Sum of Degrees

demonstrates a characteristic that is true of all graphs of any shape or size. When the number of edges is increased by one, the sum of the degrees increases by two. This happens because each edge has two ends and each end increases the degree of one vertex by one unit. As a result, the sum of the degrees of the vertices on any graph is always twice the number of edges. This relationship is known as the **Sum of Degrees Theorem**.

FORMULA

For the Sum of Degrees Theorem,

$$\text{sum of the degrees} = 2 \times \text{number of edges}$$

or

$$\text{number of edges} = \frac{\text{sum of degrees}}{2}$$

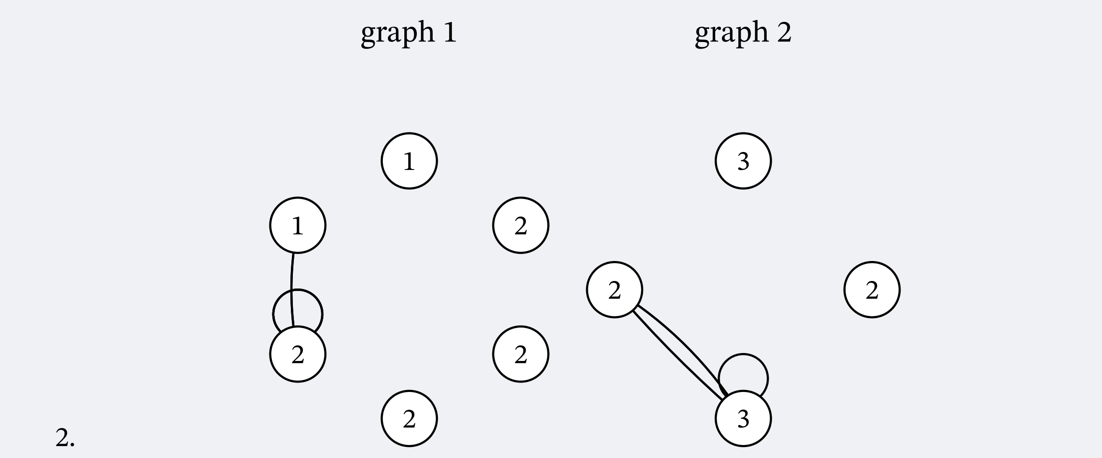
Example 2 **Finding the Sum of Degrees**

Suppose that a graph has five edges.

1. Find the sum of the degrees of the vertices.
2. Draw two different graphs that demonstrate this conclusion.

Solution

1. The sum of the degrees is twice the number of edges: $2 \times 5 = 10$. The sum of the degrees is 10.



2.

Completeness

Suppose that there were five strangers in a room, A, B, C, D, and E, and each one would be introduced to each of the others. How many introductions are necessary? One way to begin to answer this question is to draw a graph with each vertex representing an individual in the room and each edge representing an introduction as in .

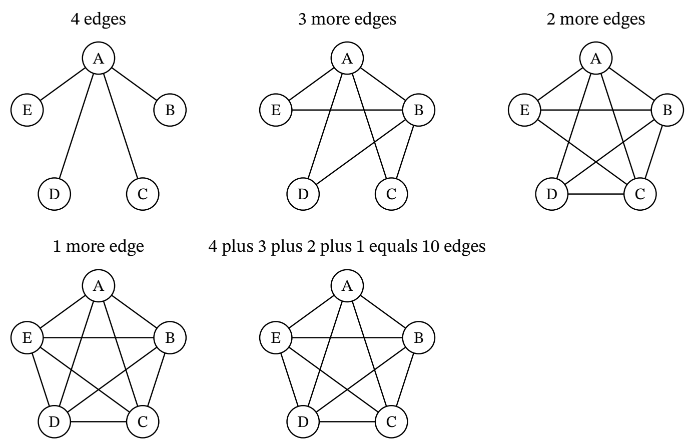


Figure 22: Model of Introductions between Five Strangers

Let's approach the problem by thinking about how many new people Person A would meet, then Person B, and so on, making sure not to repeat any introductions. The first graph in shows Person A meeting Persons B, C, D, and E, for a total of 4 introductions. The next graph shows that Person B still has to meet Persons C, D, and E, for a total of 3 more introductions. The next graph shows that Person C still has to meet Persons D and E, which is 2 more introductions. The next graph shows that Person D only remains to meet Person E, which is 1 more introduction. The final graph has $4 + 3 + 2 + 1 = 10$ edges representing 10 introductions. The last graph is an example of a **complete** graph because each pair of vertices is joined by an edge. Another way of saying this is that the graph is complete because each vertex is adjacent to every other vertex. shows complete graphs with three, four, five, and six vertices.

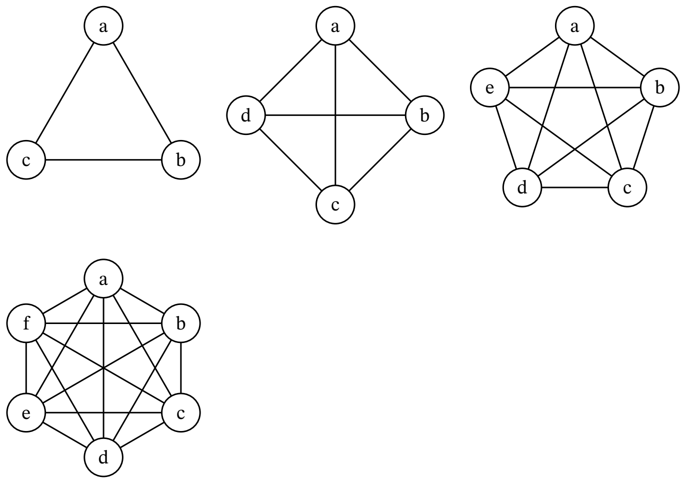


Figure 23: Complete Graphs with Up to Six Vertices

Suppose we want to know the number of introductions necessary in a room with six people. This would be represented by a complete graph with six vertices, and the total number of introductions would be $5 + 4 + 3 + 2 + 1 = 15$, the number of edges in the graph. In fact, you can always find the number of introductions in a room with n people by adding all the whole numbers from 1 to $n - 1$.

FORMULA

The number of edges in a complete graph with n vertices is the sum of the whole numbers from 1 to $n - 1$, $1 + 2 + 3 + \cdots + (n - 1)$.

Suppose that we want to determine how many introductions are necessary in a room with 500 strangers. In other words, suppose that we want to determine the number of edges in a complete graph with 500 vertices. Adding up all the numbers from 1 to 499 could take a long time! In the next example, we use the Sum of Degrees Theorem to make the problem more manageable.

Example 3 **Using the Sum of Degrees Theorem**

Use the Sum of Degrees Theorem to determine the number of introductions required in a room with

1. 6 strangers.
2. 10 strangers.
3. n strangers.

Solution

1. Since there are 6 strangers, there are 6 vertices. Since each individual must meet 5 other individuals, there are 5 edges meeting at each vertex which means each vertex

has degree 5. Since there are 6 vertices of degree 5, the sum of degrees is $6 \cdot 5 = 30$. By the Sum of Degrees Theorem, the number of edges is half the sum of the degrees, which is $\frac{30}{2} = 15$. So, the total number of introductions is 15.

2. Since there are 10 strangers, there are 10 vertices. Since each individual must meet 9 other individuals, there are 9 edges meeting at each vertex which means each vertex has degree 9. Since there are 10 vertices of degree 9, the sum of degrees is $10 \cdot 9 = 90$. By the Sum of Degrees Theorem, the number of edges is half the sum of the degrees, which is $\frac{90}{2} = 45$. So, the total number of introductions is 45.
3. Since there are n strangers, there are n vertices. Since each individual must meet $n-1$ other individuals, there are $n-1$ edges meeting at each vertex which means each vertex has degree $n-1$. Since there are n vertices of degree $n-1$, the sum of degrees is $n(n-1)$. By the Sum of Degrees Theorem, the number of edges is half the sum of the degrees, which is $\frac{n(n-1)}{2}$. So, the total number of introductions is $\frac{n(n-1)}{2}$.

Now we have a shorter way to calculate the number of introductions in a room with n strangers, and the number of edges on a complete graph with n vertices. Let's update our formula.

FORMULA

The number of edges in a complete graph with n vertices is $1 + 2 + 3 + \cdots + (n-1) = \frac{n(n-1)}{2}$.

Subgraphs

Sometimes a graph is a part of a larger graph. For example, the graph of South Florida Airports from is part of a larger graph that includes Orlando International Airport in Central Florida, which is shown in

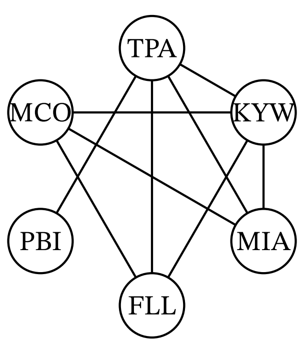


Figure 24: Orlando and South Florida Airports

The graph in includes an additional vertex, MCO, and additional edges shown with dashed lines. The graph of direct flights between South Florida airports from is called a **subgraph** of the graph that also includes direct flights between Orlando and the same South Florida airports in .

In general terms, if Graph B consists entirely of a set of edges and vertices from a larger Graph A, then B is called a subgraph of A.

Example 4 **Identifying a Subgraph**

In , Graph G is given, along with four diagrams. Determine whether each diagram is or is not a subgraph of Graph G and explain why.

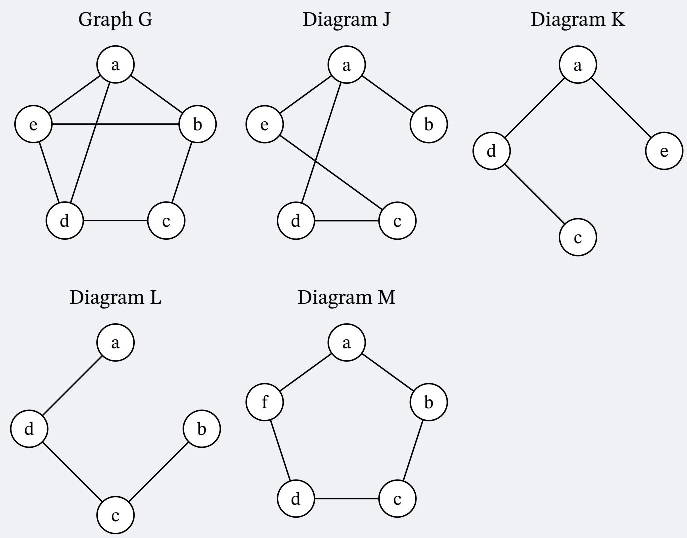


Figure 25: Graph G and Potential Subgraphs

Solution

- Diagram J is not a subgraph of Graph G because edge ec is not in Graph G.
- Diagram K is a subgraph of Graph G because all of its vertices and edges were part of Graph G.
- Diagram L is not a graph at all, because there is a line extending from vertex a that does not connect a to another vertex. So, Diagram L cannot be a subgraph.
- Diagram M is not a subgraph of Graph G because it contains a vertex f, which is not part of G.

Identifying and Naming Cycles

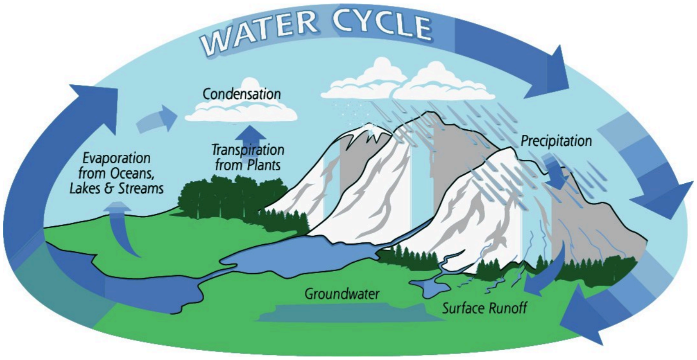


Figure 26: The water cycle begins and ends with water.

When you think of a cycle in everyday life, you probably think of something that begins and ends the same way. For example, the water cycle begins with water in a lake or ocean, which evaporates into water vapor, condenses into clouds, and then returns to earth as rain or some other form of precipitation that settles into lakes or oceans. A **cycle** in graph theory is similar in that it begins and ends in the same way: It is a series of connected edges that begin and end at the same vertex but otherwise never repeat any vertices.

In a cycle, there are always the same number of vertices as edges, and all vertices must be of degree 2. Cycles are often referred to by the number of vertices. For example, a cycle with 5 vertices can be called a 5-cycle. Cycles can also be named after polygons based on the number of edges. For example a 5-cycle is also called a pentagon. lists these names for cycles of size 3 through size 10.

Cycle Category	Number of Edges	Example
triangle	3	
quadrilateral	4	
pentagon	5	
hexagon	6	
heptagon	7	
octagon	8	
nonagon	9	
decagon	10	

There are many more polygon names, including a megagon that has a million edges and a googolgon that has 10^{100} edges, but usually we just say n-gon when the number n is past 10. For example, a cycle with 11 edges could be called an 11-gon.

Notice that the 10-cycle, or decagon, appears to cross over itself in . Remember, graphs can be drawn differently but represent the same connections. In , the same decagon is transformed into a graph that does not appear to overlap itself. We have done this without changing any of the connections so both diagrams represent the same relationships, and both diagrams are considered decagons.

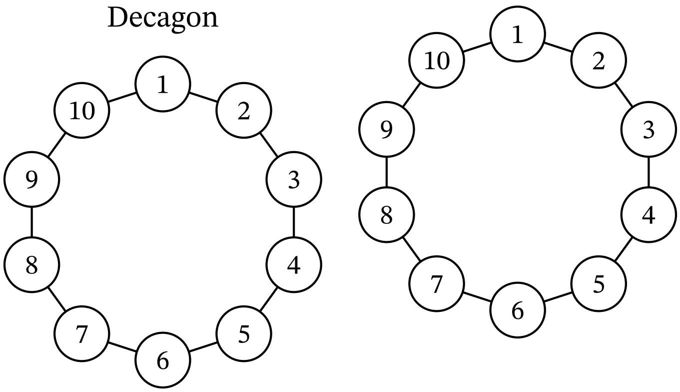


Figure 27: Transformation of a Decagon

WHO KNEW?

Googol to Google

Did the name "googolgon" ring a bell for you? If so, there is a good reason for it. The creators of Google.com renamed their search engine Google, a misspelled reference to the number "googol" alluding to the enormous number of calculations their algorithm can tackle. A googol is a 1 followed by 100 zeroes, or 10^{100}. (William L. Hosch, "Google, American Company," https://www.britannica.com/topic/Google-Inc, Encyclopedia Britannica online)

Cyclic Subgraphs and Cliques

When cycles appear as subgraphs within a larger graph, they are called **cyclic subgraphs**. Cyclic subgraphs are named by listing their vertices sequentially. The vertex where you begin is not important. Graph K in has two triangle cycles (g, h, j) and (h, i, j), and one quadrilateral cycle (g, h, i, j).

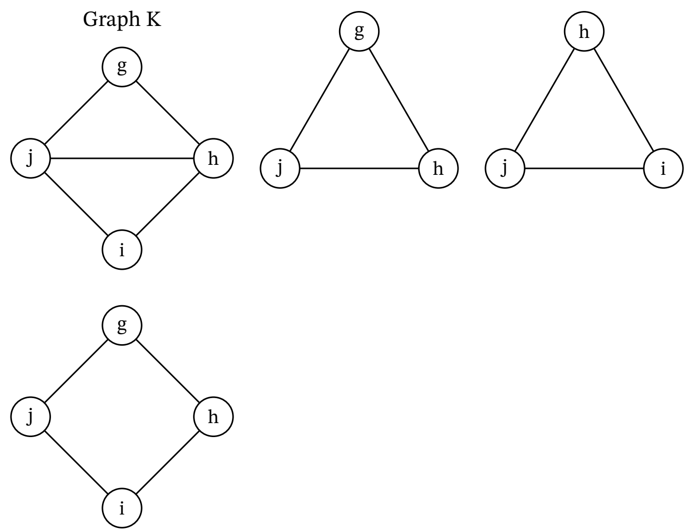


Figure 28: Cycles in Graph *K*

CHECKPOINT

The same cycle can be named in many ways, but we must keep in mind that listing two vertices consecutively indicates an edge exists between them. For example, (g, h, i, j) can also be named (h, i, j, g) or (j, i, h, g). However, you cannot name it (g, i, h, j,) which doesn't reflect the correct order of the vertices. We can see that the order (g, i, h, j) is incorrect because the cycle has no edges gi or hj.

Example 5 **Identifying Cyclic Subgraphs**

Identify the types of cyclic subgraphs in Graph H in and name them.

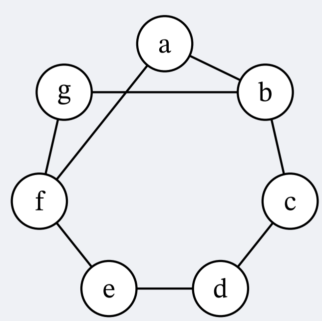


Figure 29: Graph *H*

Solution

In order to avoid missing a cycle, begin by analyzing the vertex a, then proceed in alphabetical order. Vertex a is part of two cycles, the quadrilateral cycle (a, b, f, g,) and the hexagonal cycle (a, b, c, d, e, f). Next look at vertex b. It is part of the hexagonal cycle (b, c, d, e, f, g). After checking each of the remaining vertices, we see there are no other cycles.

CHECKPOINT
In order to avoid naming the same cycle twice, consider naming cycles beginning with the letter closest to the beginning of the alphabet and then progressing clockwise.

We have seen that sociologists use graphs to study the structures of social networks. In sociology, there is a principle known as Triadic Closure. It says that if two individuals in a social network have a friend in common, then it is more likely those two individuals will become friends too. Sociologists refer to this as an open triad becoming a closed triad. This concept can be visualized as graphs in . (Chakraborty, Dutta, Mondal, and Nath, "Application of Graph Theory in Social Media, *International Journal of Computer Sciences and Engineering*, 6(10):722-729) In the open triad in , person a and person b each has a friendship with person c. In the closed triad, person a and person b have also developed a friendship. Notice that the graph representing the closed triad is a three-cycle, or a triangle, in graph theory terms.

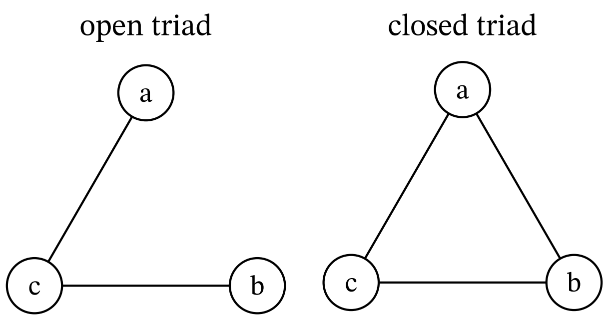


Figure 30: Graphs of Open Triad and Closed Triad

Another topic of interest to sociologists, as well as computer scientists and scientists in many fields, is the concept of a **clique**. In a social group, a clique is a subgroup who are all friends. A triad is an example of a clique with three people, but there can be cliques of any size. In graph theory, a clique is a complete subgraph.

Example 6 **Identifying Triads and Cliques**

The graphs in YOUR TURN 12.6 represent social communities. The vertices are individuals and the edges are social connections. Use the graphs to answer the questions.

1. Which graph has the most triads? Name the triangles.
2. Which graph has the fewest triads? Name the triangles.
3. How might an increase in the number of triads in a graph affect the resilience of a community? Explain your reasoning.
4. Identify a clique with more than three vertices in Graph E by naming its vertices.

Solution

1. Graph E has the most triads. The triangles are (a, b, c), (a, c, e), (a, c, f), (a, c, g), (a, e, g), (c, e, g), (e, g, h), (e, h, i), and (f, g, i).

2. Graph D has the fewest triads. There are no triangles in the graph.
3. More triads means vertices of greater degree, which indicates greater resilience. Specifically, if an edge is removed from a closed triad, there the two individuals who are no longer adjacent are still connected by way of the third member of the triad.
4. The vertices a, e, c, and g form a clique with four vertices.

Three-Cycles in Complete Graphs

Just as complete graphs have a predictable number of edges, complete graphs have a predictable number of cyclic subgraphs. Let's look at the three-cycles within complete graphs with up to six vertices, which are shown.

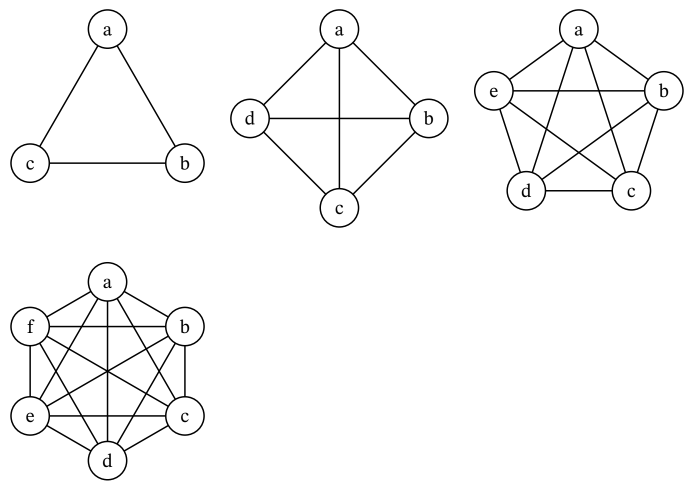


Figure 31: Complete Graphs with Up to Six Vertices

Let's list the names of all the triangles in each graph. ***Since every pair of vertices is adjacent, any three vertices on a complete graph form a triangle***. There is only one triangle in the complete graph with three vertices, (a, b, c). For the rest of the graphs, it is important that we take an organized approach. Start with the vertex that is first alphabetically, listing any triangles that include that vertex also in alphabetical order. Then, proceed to the next vertex in the alphabet, and list any triangles that include that vertex, except those that are already listed. Keep going in this way as shown.

Complete Graph With:	3 Vertices	4 Vertices	5 Vertices	6 Vertices
All a triangles	(a, b, c)	(a, b, c), (a, b, d), (a, c, d)	(a, b, c), (a, b, d), (a, b, e) (a, c, d), (a, c, e) (a, d, e)	(a, b, c), (a, b, d), (a, b, e), (a, b, f) (a, c, d), (a, c, e), (a, c, f) (a, d, e), (a, d, f) (a, e, f)
Other b triangles	None	(b, c, d)	(b, c, d), (b, c, e), (b, d, e)	(b, c, d), (b, c, e), (b, c, f) (b, d, e), (b, d, f) (b, e, f)
Other c triangles	None	None	(c, d, e)	(c, d, e), (c, d, f) (c, e, f)
Other d triangles	None	None	None	(d, e, f)
Total	1	$(2 + 1) + 1$ $= 3 + 1$ $= 4$	$(3 + 2 + 1) + (2 + 1) + 1$ $= 6 + 3 + 1$ $= 10$	$(4 + 3 + 2 + 1) + (3 + 2 + 1) + (2$ $= 10 + 6 + 3 + 1$ $= 20$

Look at the last row in . Do you see a pattern emerge for counting triangles in a complete graph? Without drawing a complete graph with 7 vertices, we can predict that it will have $(5 + 4 + 3 + 2 + 1) + (4 + 3 + 2 + 1) + (3 + 2 + 1) + (2 + 1) + 1 = 35$ triangles inside it. This pattern also appears in a famous diagram known to Western mathematicians as “Pascal’s Triangle.” displays the first 11 rows of Pascal’s Triangle. Row 0 of Pascal’s Triangle only has the number 1 in it. The first and last entries in each of the other rows are also 1s. Otherwise, all the entries are formed by adding two entries from the previous row. For example, in row 6, entry 1 is 6, which was found by adding 1 and the 5 from the previous row, and entry 2 is 15, which was found by adding the 5 and the 10 from the previous row, as shown.

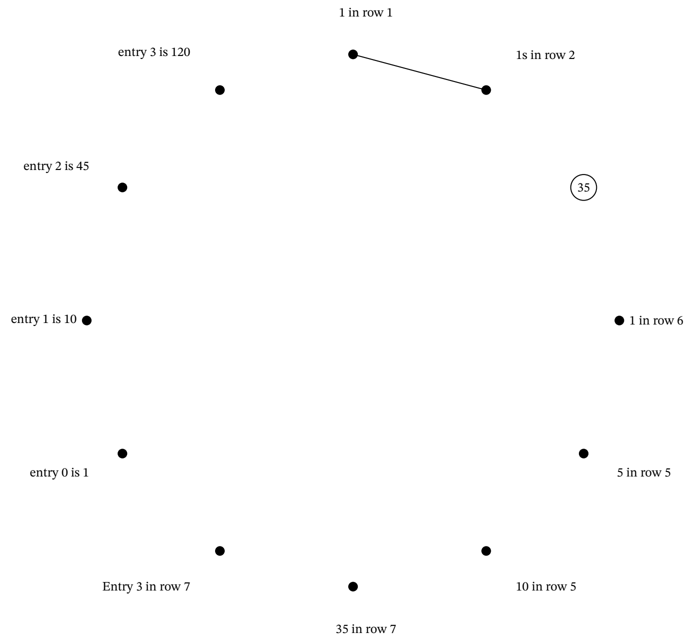


Figure 32: Pascal's Triangle

CHECKPOINT

IMPORTANT! When you count the rows and entries of Pascal's Triangle begin with row 0 and entry 0, not row 1 and entry 1.

VIDEO

The Mathematical Secrets of Pascal's Triangle by Wajdi Mohamed Ratemi

If we begin to list the third entries in each row of Pascal's triangle from the top down, we see 1, 4, 10, 20, 35, and so on. Notice that these values are exactly the number of triangles in a complete graph of sizes 3, 4, 5, 6, and 7, respectively. Specifically, entry 3 in row 7 is 35, the number of triangles in a complete graph of size 7. Let's practice using Pascal's triangle to find the number of triangles in a complete graph of a particular size.

STEPS TO FIND THE NUMBER OF TRIANGLES IN A COMPLETE GRAPH OF SIZE *n*

Step 1 Calculate the entries in row n of Pascal's Triangle using sums of pairs of entries in previous rows as needed.

Step 2 The number of triangles is entry 3 in row n. Remember to count the entries 0, 1, 2, and 3, from left to right.

Example 7 **Using Pascal's Triangle to Find the Number of Triangles in a Complete Graph**

Use Pascal's Triangle in to find the number of triangles in a complete graph with 11 vertices.

Solution

Step 1: Identify the row number and calculate its entries. Since the complete graph has 11 vertices, we will need to look in row 11 of Pascal's Triangle. Use row 10 in to find the entries in row 11 as shown.

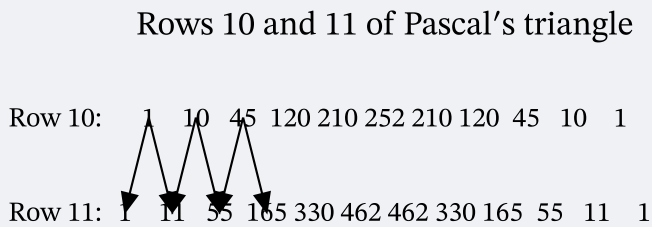


Figure 33: Rows 10 and 11 of Pascal's Triangle

Step 2: Find entry number 3. The number of triangles is in entry 3. Remember to count the entries beginning with entry 0 as shown.

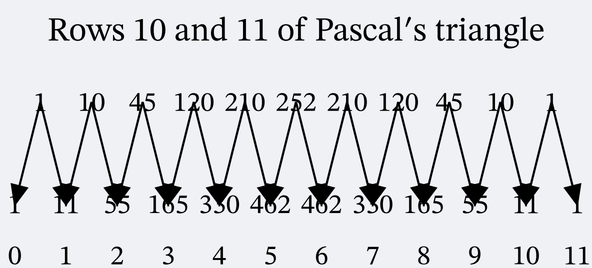


Figure 34: Identifying an Entry in Pascal's Triangle

Entry 3 in row 11 is 165. So, there are 165 triangles in a complete graph with 11 vertices.

TECH CHECK

Since the entries in Pascal's Triangle are useful in many applications, many resources such as online and traditional calculator functions have been developed to assist in calculating the values. The website dCode.xyz has a tool that automatically calculates entries in Pascal's Triangle using these steps:

Step 1: Make sure to select Index 0.

Step 2: Select your preference, display the triangle until a particular row (line), display only one row (line), or calculate the value at coordinates (which means give the value of a single entry).

Step 3: Enter row number (line number) you are interested, or enter the row number and entry number in parentheses: (row number, entry number).

Step 4: Select Calculate.

Step 5: Retrieve the answer from the results box.

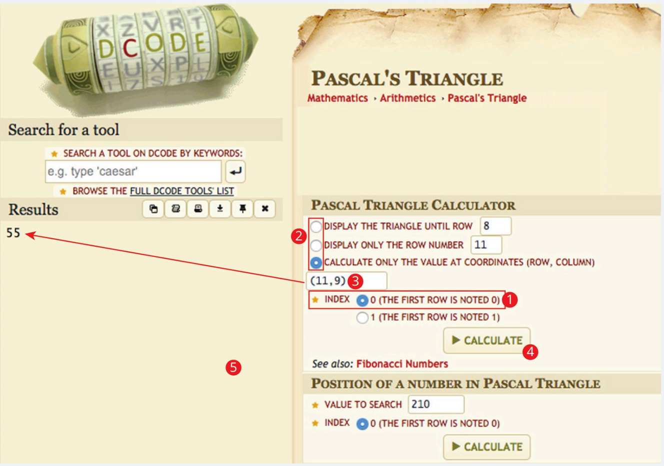


Figure 35: Pascal's Triangle Tool on dCode.xyz

Key Terms

- complete
- subgraph
- cycle
- cyclic subgraph
- clique

Key Concepts

- The sum of the degrees of the vertices in a graph is twice the number of edges.
- In a complete graph every pair of vertices is adjacent.
- A subgraph is part of a larger graph.
- Cycles are a sequence of connected vertices that begin and end at the same vertex but never visit any vertex twice.

Formulas

For the Sum of Degrees Theorem, $\text{sum of the degrees} = 2 \times \text{number of edges}$ or $\text{number of edges} = \frac{\text{sum of degrees}}{2}$

The number of edges in a complete graph with n vertices is the sum of the whole numbers from 1 to $n-1$, $1+2+3+\cdots+(n-1)$.

The number of edges in a complete graph with n vertices is $1+2+3+\cdots+(n-1)=\frac{n(n-1)}{2}$.

Videos

- The Mathematical Secrets of Pascal's Triangle by Wajdi Mohamed Ratemi

12.3 Comparing Graphs

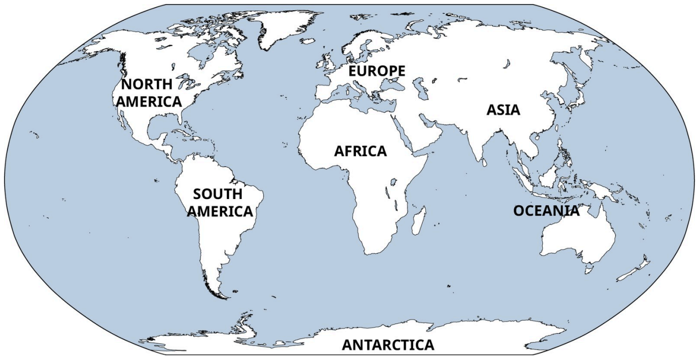


Figure 36: A flat map represents the surface of Earth in two dimensions.

Figure 37: A globe represents the surface of Earth in three dimensions.

Learning Objectives

After completing this section, you should be able to:

1. Identify the characteristics used to compare graphs.
2. Determine when two graphs represent the same relationships
3. Explore real-world examples of graph isomorphisms
4. Find the complement of a graph

Maps of the same region may not always look the same. For example, a map of Earth on a flat surface looks distorted at the poles. When the same regions are mapped on a spherical globe, the countries that are closer to the polls appear smaller without the distortion. Despite these differences, the two maps still communicate the same relationships between nations such as shared boundaries, and relative position on the earth. In essence, they are the same map. This means that for every point on one map, there is a corresponding point on the other map in the same relative location. In this section, we will determine exactly what characteristics need to be the shared by two graphs in order for us to consider them "the same."

When Are Two Graphs Really the Same Graph?

In arithmetic, when two numbers have the same value, we say they are equal, like ½ = 0.5. Although ½ and 0.5 look different, they have the same value, because they are assigned the same position on the real number line. When do we say that two graphs are equal?

shows Graphs *A* and *F*, which are identical except for the labels. Graphs are visual representations of connections. As long as two graphs indicate the same pattern of connections, like Graph *A* and Graph *F*, they are considered to be equal, or in graph theory terms, **isomorphic**.

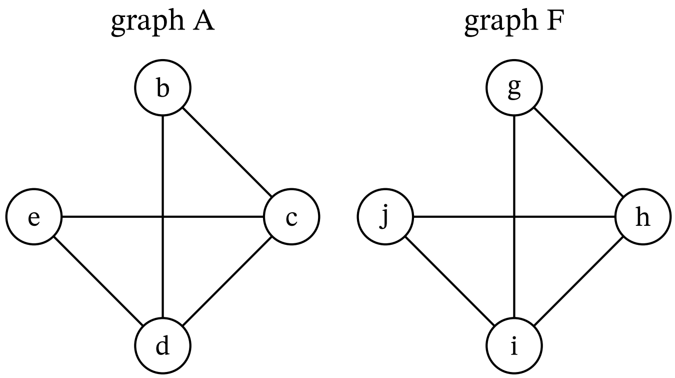


Figure 38: Identical Graphs with Different Labels

Two graphs are isomorphic if either one of these conditions holds:

1. One graph can be transformed into the other without breaking existing connections or adding new ones.
2. There is a correspondence between their vertices in such a way that any adjacent pair in one graph corresponds to an adjacent pair in the other graph.

   It is important to note that if either one of the isomorphic conditions holds, then both of them do. When we need to decide if two graphs are isomorphic, we will need to make sure that one of them holds. For example, shows how Graph *T* can be bent and flipped to look like Graph *Z*, which means that Graphs *T* and *Z* satisfy condition 1 and are isomorphic.

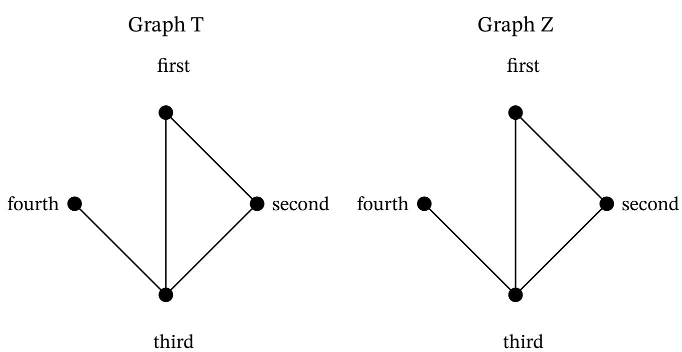


Figure 39: Graph *T* Transformed Into Graph *Z*

Also, notice that the vertices that were adjacent in the first graph are still adjacent in the transformed graph as shown. For example, vertex 3 is still adjacent to vertex 4, which means they are still neighboring vertices joined by a single edge.

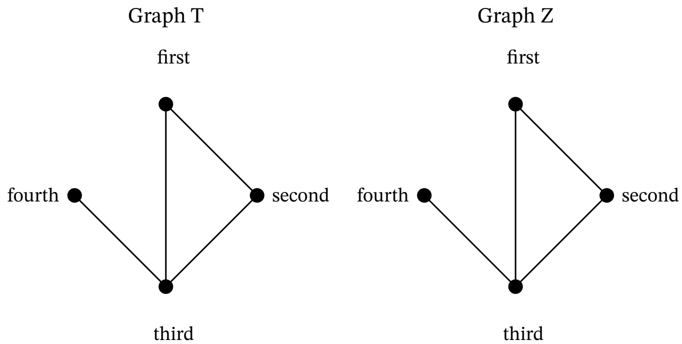


Figure 40: Adjacent Vertices Are Still Adjacent

When Are Two Graphs Really Different?

Verifying that two graphs are isomorphic can be a challenging process, especially for larger graphs. It makes sense to check for any obvious ways in which the graphs might differ so that we don't spend time trying to verify that graphs are isomorphic when they are not. If two graphs have any of the differences shown in , then they cannot be isomorphic.

Unequal number of vertices	
Unequal number of edges	
Unequal number of vertices of a particular degree	
Different cyclic subgraphs	

Recognizing Isomorphic Graphs

Isomorphic graphs that represent the same pattern of connections can look very different despite having the same underlying structure. The edges can be stretched and twisted. The graph can be rotated or flipped. For example, in , each of the diagrams represents the same pattern of connections.

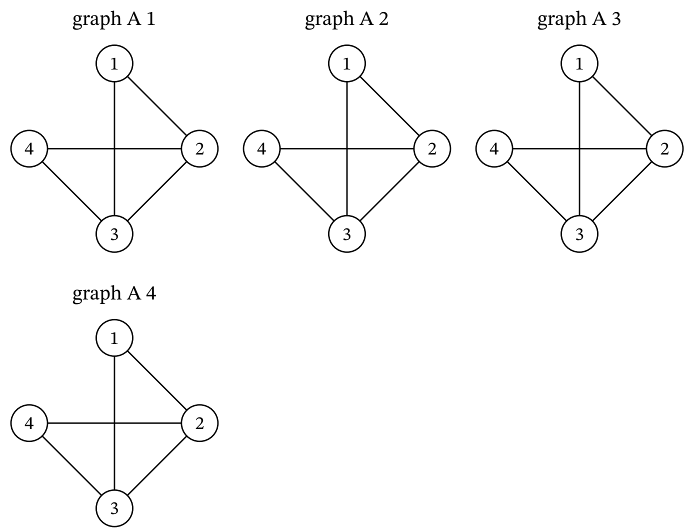


Figure 41: Four Representations of the Same Graph

Looking at , how can we know that these graphs are isomorphic? We will start by checking for any obvious differences. Each of the graphs in has four vertices and five edges; so, there are no differences there. Next, we will focus on the degrees of the vertices, which have been labeled in .

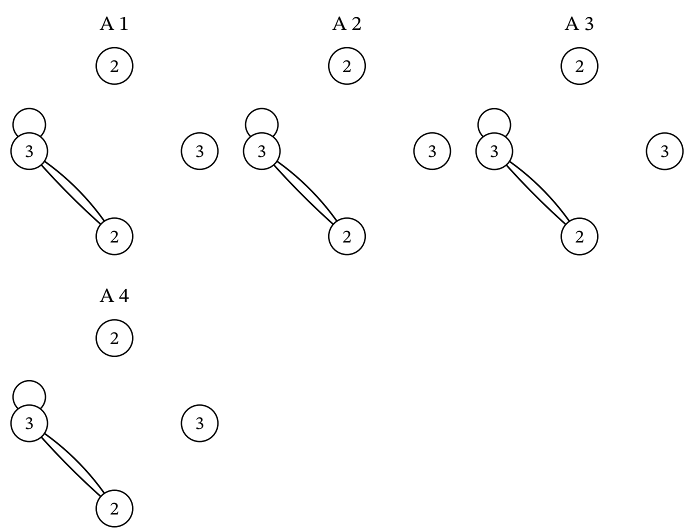


Figure 42: Graphs with Vertices of the Same Degrees

As shown in , each graph has two vertices of degree 2 and two vertices of degree 3; so, there are no differences there. Now, let’s check for cyclic subgraphs. These are highlighted in .

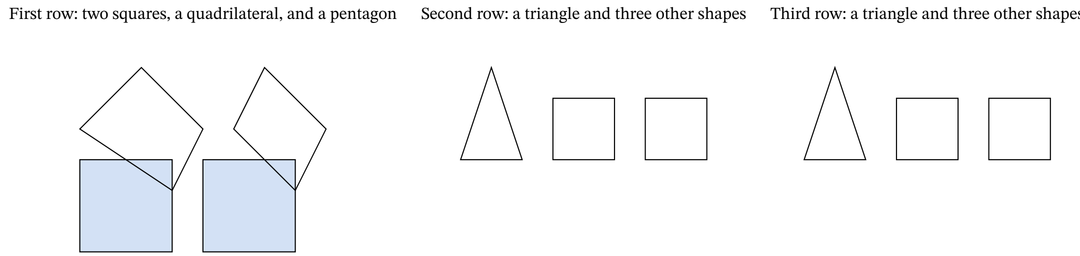


Figure 43: Graphs with the Same Cyclic Subgraphs

As shown in , each graph has two triangles and one quadrilateral; so, no differences there either. It is beginning to look likely that these graphs are isomorphic, but we will have to look further to be sure.

To know with certainty that these graphs are isomorphic, we need to confirm one of the two conditions from the definition of isomorphic graphs. With smaller graphs, you may be able to visualize how to stretch and twist one graph to get the other to see if condition 1 holds. Imagine the edges are stretchy and picture how to pull and twist one graph to form the other. If you can do this without breaking or adding any connections, then the graphs are really the same. demonstrates how to change graph A_4 to get A_3, graph A_3 to get A_2, and graph A_2 to get A_1.

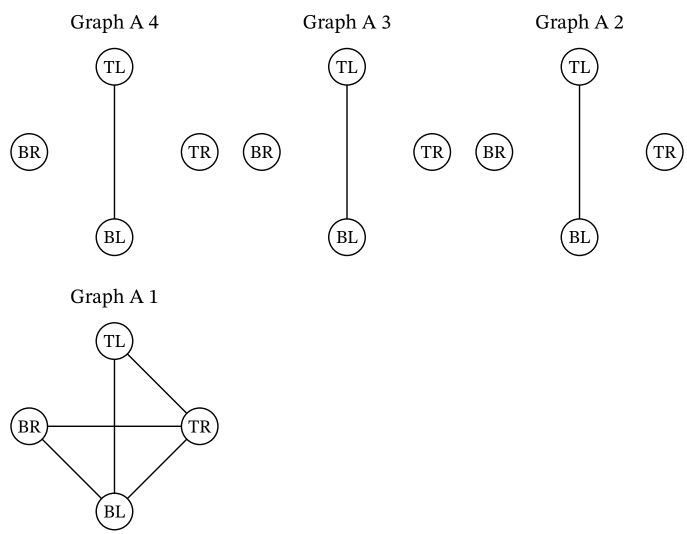


Figure 44: Transforming Graphs

Now that we have used visual analysis to see that condition 1 holds for graphs A_1, A_2, A_3, and A_4 in , we know that they are isomorphic. In , one of the edges of graph A_4 crossed another edge of the graph. By transforming it into graph A_3, we have “untangled” it. Graphs that can be untangled are called **planar** graphs. The complete graph with five vertices is an example of a **nonplanar** graph-that means that, no matter how hard you try, you can’t untangle it. But, when you try to figure out if two graphs are the same, it can be helpful to untangle them as much as possible to make the similarities and differences more obvious.

Example 1 **Identifying Isomorphic Graphs**

Which of the three graphs in are isomorphic, if any? Justify your answer.

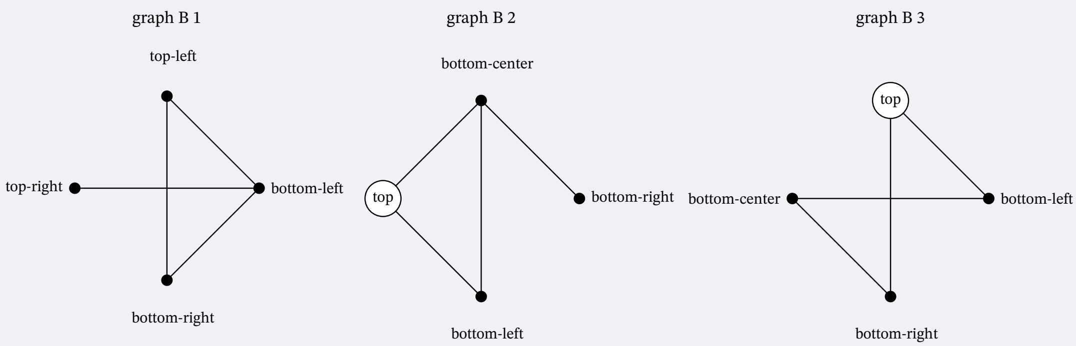


Figure 45: Three Similar Graphs

Solution

Step 1: Check for differences in number of vertices, number of edges, degrees of vertices, and types of cycles to see if an isomorphism is possible.

- Vertices: They all have the same number of vertices, 4.
- Edges: They all have the same number of edges, 4.
- Degrees: Graph B_1 and Graph B_2 each have a vertex of degree 3, while Graph B_3 does not. So, Graph B_3 is not isomorphic to either of the other two graphs, but Graph B_1 and Graph B_2 could possibly be isomorphic.
- Cycles: Focus on any cycles in Graph B_1 and Graph B_2. Each graph has a triangle as shown.

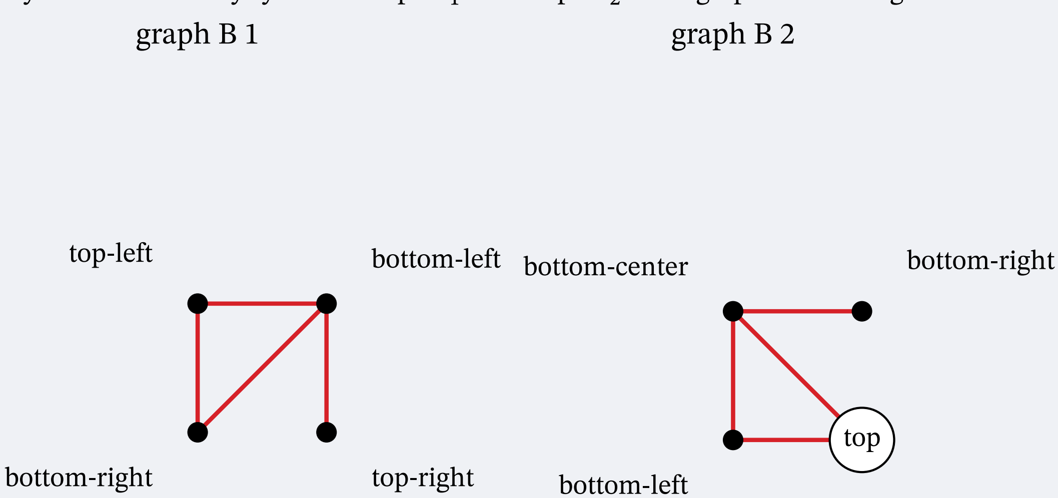


Figure 46: Cycles of Graph B_1 and Graph B_2

Step 2: If no differences were found and an isomorphism is possible, verify one of the conditions in the definition of isomorphic.

Since we were able to determine that Graph B_1 and Graph B_2 have no obvious differences, and they are relatively small graphs, we will attempt to transform one graph into the other, which would verify condition 1. Graph B_1 can be transformed into Graph B_2 without breaking or adding connections as shown. Begin by untangling graph B_1. Then rotate or flip as needed to see that the graphs match.

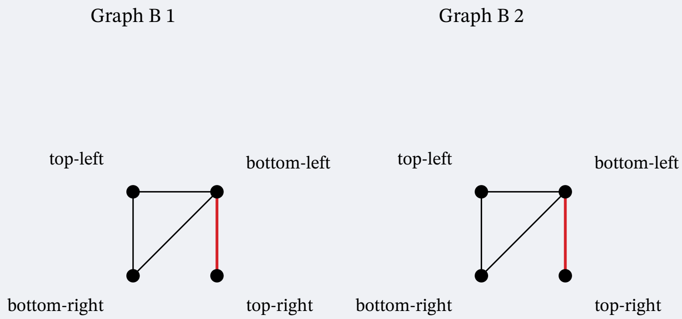


Figure 47: Transformation of Graph B_1

So, Graph B_1 and Graph B_2 have the same structure and are isomorphic.

Have you ever noticed that many popular board games may look different but are really the same game? A good example is the many variations of the board game *Monopoly*®, which was submitted to the U.S. Patent Office in 1935. Although the rules have been revised a bit, a very similar game board is still in use today. There have been many versions of Monopoly over the years. Many have been stylized to reflect a popular theme, such as a show or sports team, while retaining the same game board structure. If we were to represent these different versions of the game using a graph, we would find that the graphs are isomorphic. . Let's analyze some game boards using graph theory to determine if they have the same structure despite having different appearances.

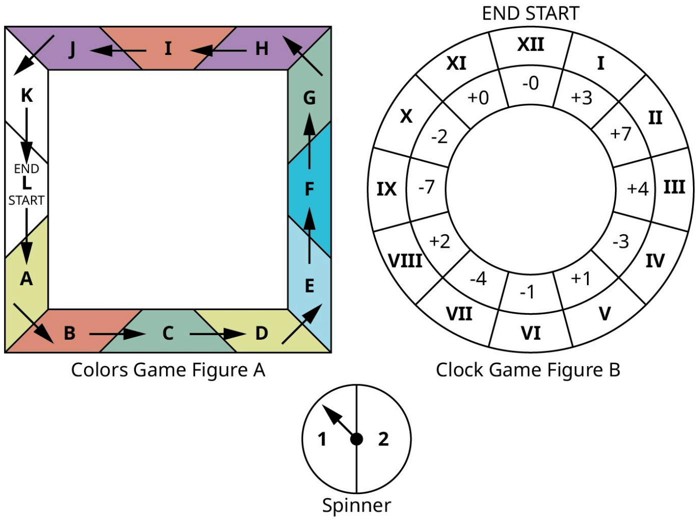


Figure 48: Two Game Boards

Example 2 **Deciding If Graphs are Isomorphic**

A teacher uses games to teach her students about colors and numbers as shown.

In the Colors Game, shown in Figure *A*, each player begins in the space marked START and proceeds in a counterclockwise direction. On each turn, the player spins a spinner marked 1 and 2 and moves forward the number of spaces shown on the spinner. If the player lands on a space marked with any color other than white, the player must move forward or back to the other space of the same color. The first player to land in or pass the space marked END wins.

In the Clock Game, shown in Figure *B*, each player begins in the space marked START and proceeds in a clockwise direction. On each turn, the player spins a spinner marked 1 and 2 and moves forward the number of spaces shown on the spinner. In the same turn, the player must read the number in the space and move forward the number of spaces indicated by a positive value or backward the number of spaces indicated by a negative value. Then the turn ends. The first player to land in or pass the space marked END wins.

Draw a graph or multigraph to represent each game in which the vertices are the spaces and the edges represent the ability of a player to move between the spaces either by a spin or as dictated by a marked color or number. (We will ignore the direction of motion for simplicity.) Transform one of the graphs to show that it is isomorphic to the other and explain what this tells you about the games.

Solution

The graph representing the Clock Game can be transformed as shown in so that we can see that the graphs are isomorphic. There is a correspondence between their vertices in such a way that any adjacent pair in one graph corresponds to an adjacent pair in the other graph.

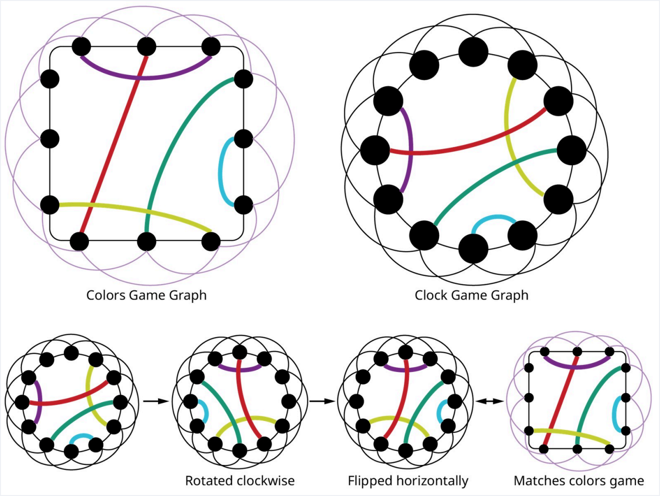


Figure 49: Graphs for Clock Game and Colors Game

This tells us that the games are essentially the same game with the same moves even though the games appear to be different.

PEOPLE IN MATHEMATICS

Elizabeth Magie

Game designer, engineer, comedian, and political activist, Elizabeth Magie designed a game called "Landlord's Game" to educate fellow citizens about the dangers of monopolies and the benefits of wealth redistribution through a land tax. Magie, whose father had campaigned with Abraham Lincoln, was a proponent of a land tax, an idea popularized by Henry George's 1879 book, *Progress and Poverty*. She designed the game to be played by two sets of rules for comparison. In one version, the goal was to dominate opponents by creating monopolies, leaving one wealthy player standing in the end. In the other version, all the players were rewarded when a monopoly was created through a simulated land tax. She patented this game for the first time in 1904. Does the game board in look familiar?

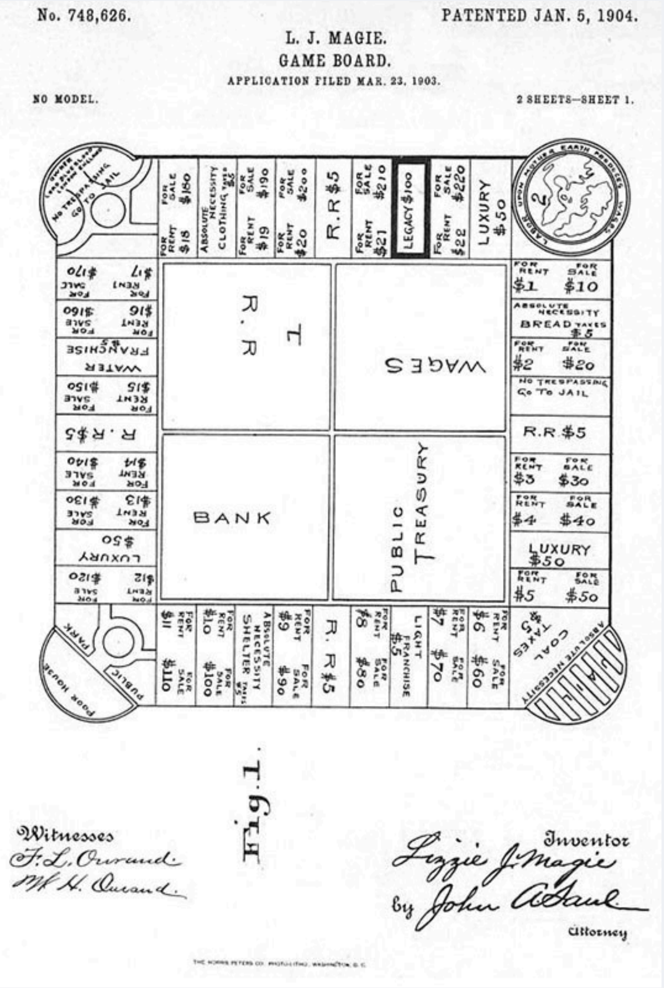


Figure 50: *Landlord's Game* Patent

That's right! A modified version of the game was obtained from friends by a man named Charles Darrow who renamed it Monopoly and sold to Parker Brothers. Parker Brothers later paid Magie $500 for the right to her patent on it. The property tax version was left behind, and the modern game of Monopoly was born. (Mary Pilon, *Monopoly's Lost Female Inventor*, September 1, 2018, National Women's History Museum, *"Monopoly's* Lost Female Inventor,"

Identifying and Naming Isomorphisms

When two graphs are isomorphic, meaning they have the same structure, there is a correspondence between their vertices, which can be named by listing corresponding pairs of vertices. This list of corresponding pairs of vertices in such a way that any adjacent pair in one graph corresponds to an adjacent pair in the other graph is called an **isomorphism**. Consider the isomorphic graphs in . In , we could replace the labels Graph F with the labels from Graph A and have an identical graph, as in .

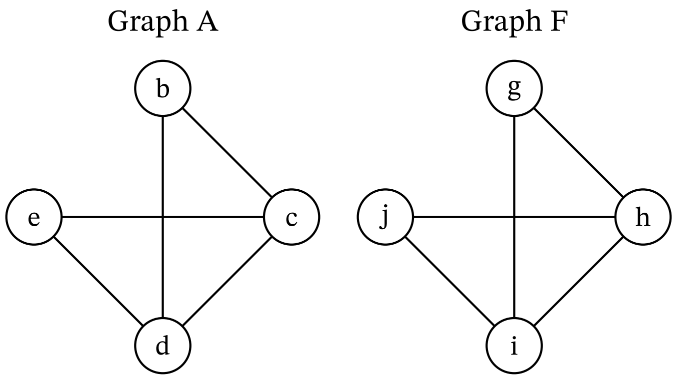


Figure 51: Identical Graphs with Different Labels

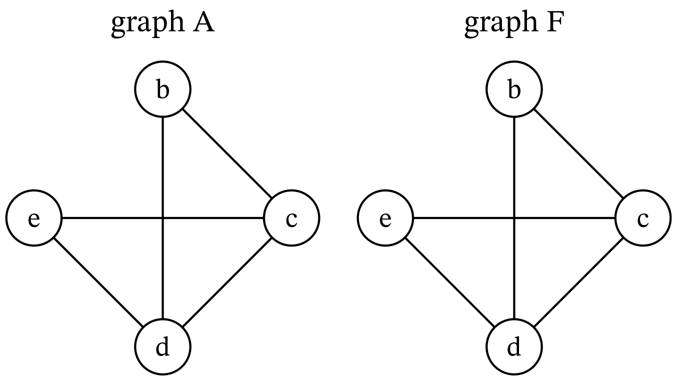


Figure 52: Corresponding Vertices

So, we can identify an isomorphism between Graph A and Graph F by listing the corresponding pairs of vertices: b-g, c-h, d-i, and e-j. Notice that b is adjacent to c and g is adjacent to h. This must be the case since b corresponds to g and c corresponds to h. The same is true for other pairs of adjacent vertices.

An isomorphism between graphs is not necessarily unique. There can be more than one isomorphism between two graphs. We can see how to form a different isomorphism between Graph A and Graph F from by rotating Graph F clockwise and comparing the rotated version of F to Graph A as in YOUR TURN 12.13. Now, we can see that a second isomorphism exists, which has the correspondence: b-j, d-h, c-i, and e-g as shown.

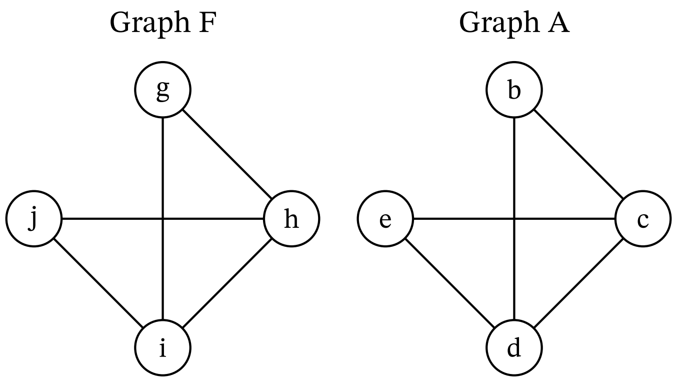


Figure 53: Transforming Graph *F* into Graph *A*

CHECKPOINT

When you name isomorphisms, one way to check that your answer is reasonable is to make sure that the degrees of corresponding vertices are equal.

VIDEO

Determine If Two Graphs Are Isomorphic and Identify the Isomorphism

Example 3 **Identifying Isomorphisms**

In Example 1, we showed that the Graphs B_1 and B_2 in are isomorphic. In , labels have been assigned to the vertices of Graphs B_1 and B_2. Identify an isomorphism between them by listing corresponding pairs of vertices.

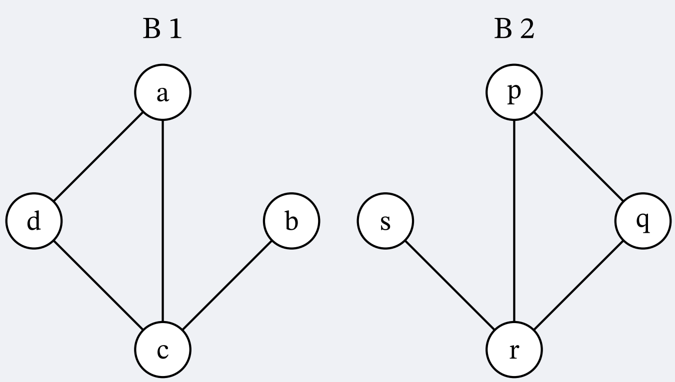


Figure 54: Two Isomorphic Graphs

Solution

showed how to transform Graph B_1 to get Graph B_2. In , we will do the same, but this time we will include the labels.

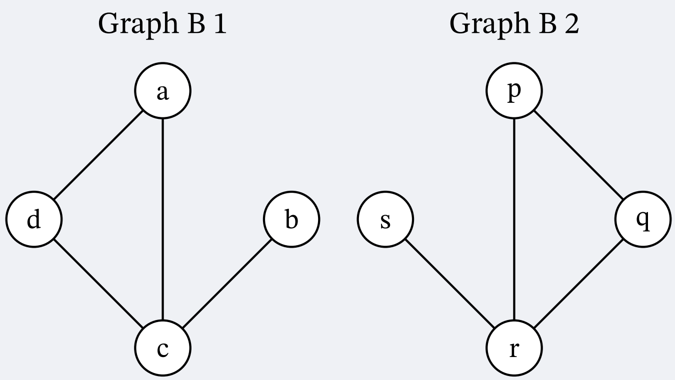


Figure 55: Transform Graph B_1 into Graph B_2

From YOUR TURN 12.13, we can see the corresponding vertices: *a*-*q*, *d*-*p*, *c*-*r*, and *b*-*s*, which is an isomorphism of the two graphs.

Example 4 **Recognizing an Isomorphism**

Determine whether Graphs *G* and *S* in are isomorphic. If not, explain how they are different. If so, name the isomorphism.

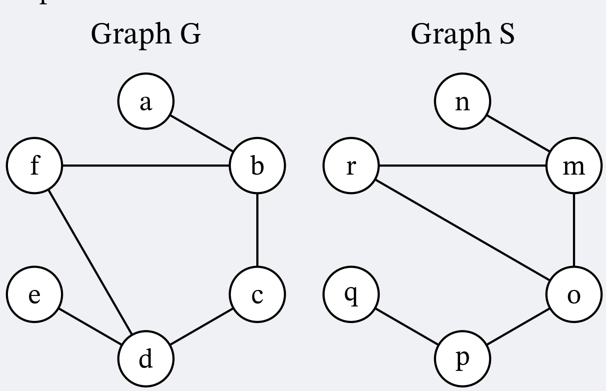


Figure 56: Graph *G* and Graph *S*

Solution

Step 1: Check for any differences.

- Vertices: Graphs *G* and *S* each have six vertices.
- Edges: Graphs *G* and *S* each have six edges.
- Degrees: Each graph also has two vertices of degree 1, two vertices of degree 2, and two vertices of degree 3.
- Cycles: From , we can see that Graph *G* contains a quadrilateral cycle (*b*, *f*, *d*, *c*) but Graph *S* has no quadrilaterals. Also, Graph *S* contains a triangle cycle (*m*, *r*, *o*) but Graph *G* has no triangles. This means that the graphs are not isomorphic.

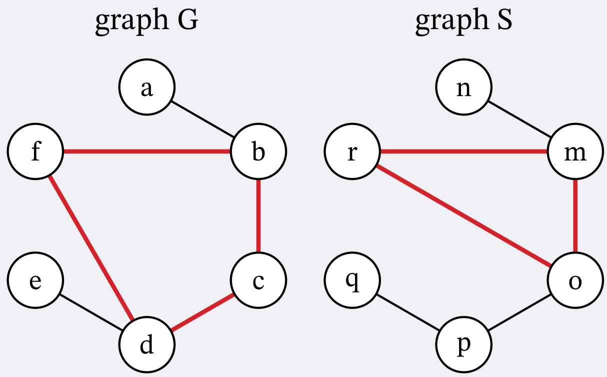


Figure 57: Comparing Subgraphs

Step 2: This step is not necessary because we now know Graphs G and S are not isomorphic.

Complementary Graphs

Suppose that you are a camp counselor at Camp Woebegone and you are holding a camp Olympics with four events. The campers have signed up for the events. You drew a graph in to help you visualize which events have campers in common.

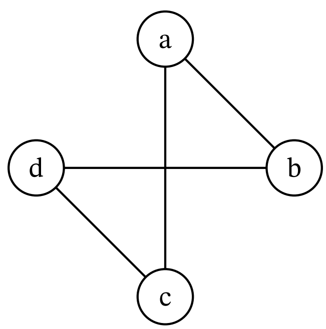


Figure 58: Graph of Events with Campers in Common

Graph E in shows that some of the same campers will be in events a and b, as well as b and d, c and d, and a and c. What do you think the graph would look like that represented the events that do not have campers in common? It would have the same vertices, but any pair of adjacent edges in Graph E, would not be adjacent in the new graph, and vice versa. This is called a **complementary** graph, as shown. Two graphs are complementary if they have the same set of vertices, but any vertices that are adjacent in one, are not adjacent in the other. In this case, we can say that one graph is the **complement**of the other.

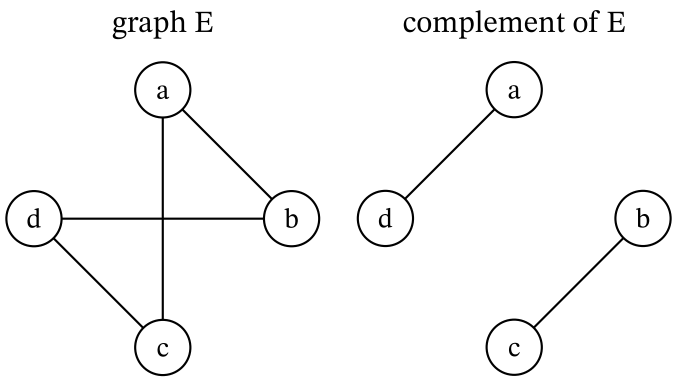


Figure 59: Graphs of Camp Olympics

One way to find the complement of a graph is to draw the complete graph with the same number of vertices and remove all the edges that were in the original graph. Let's say you wanted to find the complement of Graph *E* from , and you didn't already know it was Graph *F*. You could start with the complete graph with four vertices and remove the edges that are in Graph *E* as shown.

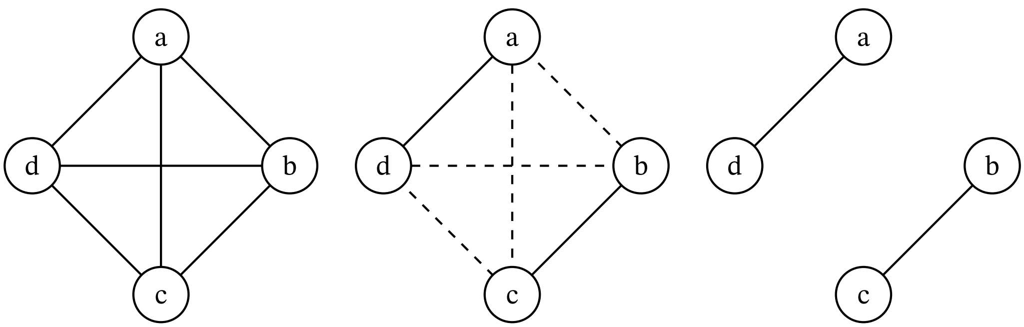


Figure 60: Use Complete Graph to Find Complement

Example 5 **Finding a Complement**

A particular high school has end-of-course exams in (E3) English 3, (E4) English 4, (M) Advanced Math, (C) Calculus, (W) World History, (U) U.S. History, (B) Biology, and (P) Physics. No English 3 students are taking English 4, World History, or Biology; no English 4 students are also in Calculus, Advanced Math, U.S. History, or Physics; no Physics students are also taking Advanced Math; No World History students are also taking U.S. History; and no Advanced Math students are also taking Calculus.

1. Create a graph in which the vertices represent the exams, and an edge between a pair of vertices indicates that there are no students taking both exams.
2. Find the complement of the graph in part 1.
3. Explain what the graph in part 2 represents.

Solution

1. In , we have drawn a vertex for each exam and edges between any vertices that have no students in common.

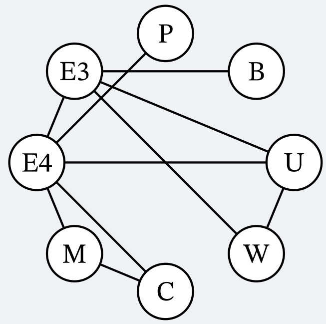


Figure 61: Graph of Exams with No Students in Common

2. One way to get the complement of the graph in is to draw a complete graph with the same number of vertices and remove the edges they have in common as shown.

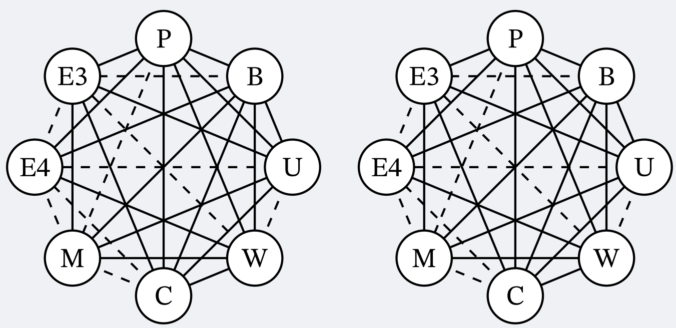


Figure 62: Remove Unwanted Edges from Complete Graph

The final graph of the complement is in .

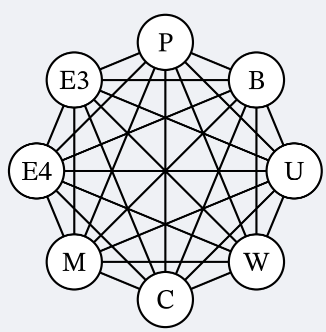


Figure 63: Graph of Exams with Students in Common

1. In the graph in , the vertices are still the exams, and a pair of adjacent vertices represents a pair of exams that have students in common.

When two graphs are really the same graph, they have the same missing edges. So, when two graphs have a lot of edges, it may actually be easier to determine if they are isomorphic by looking at which edges are missing rather than which edges are included. In other words, we can determine if two graphs are isomorphic by checking if their complements are isomorphic.

Example 6 **Using a Complement to Find an Isomorphism**

Use to answer each question.

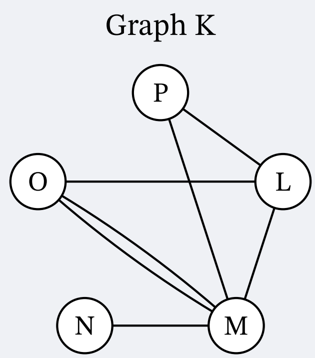


Figure 64: Graph *K*

1. Find the complement of Graph *K*.
2. Identify an isomorphism between the complement of Graph *K* from part 1, and the complement of Graph *H* in YOUR TURN 12.17.
3. Confirm that the correspondence between the vertices you found in part 2 also gives an isomorphism between Graph *H* from YOUR TURN 12.17, and Graph *K* from .

Solution

1. The complement of Graph *K* can be found by removing the edges of Graph *K* from a complete graph with the same vertices as shown.

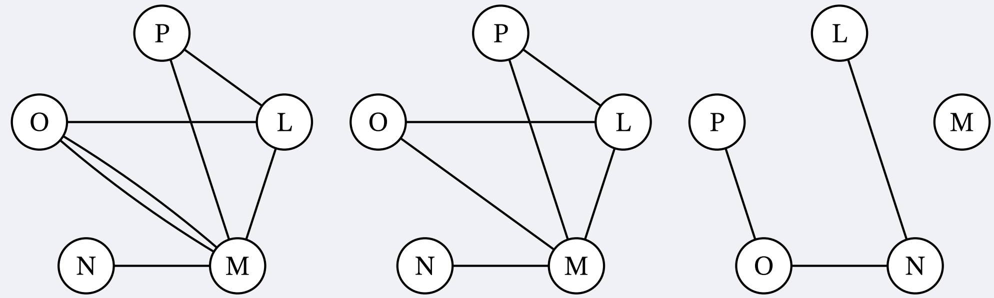


Figure 65: Find Complement of Graph *K*

2. An isomorphism between the complement of Graph *K* and the complement of Graph *H* is *A*-*M*, *C*-*L*, *E*-*N*, *B*-*O*, and *D*-*P*, which is confirmed by transforming the complement of Graph *K* in .

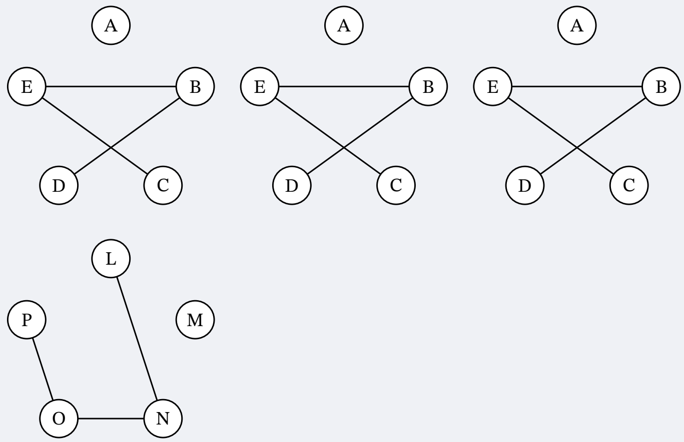


Figure 66: Isomorphism between Complements of *K* and *H*

3. shows how Graph *K* can be transformed into Graph *H* to confirm that the correspondence is *A*-*M*, *C*-*L*, *E*-*N*, *B*-*O*, and *D*-*P* also gives an isomorphism between Graph *K* and Graph *H*.

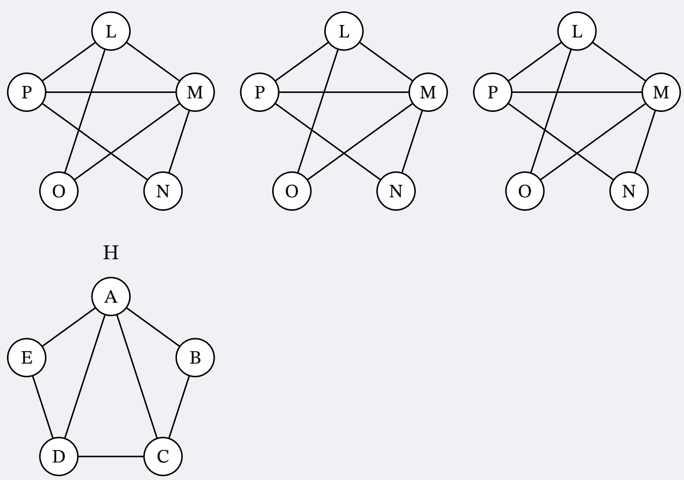


Figure 67: Isomorphism between *K* and *H*

This means we now have three conditions that guarantee two graphs are isomorphic.

First Way: One graph can be transformed into the other without breaking existing connections or adding new ones.

Second Way: There is a correspondence between their vertices in such a way that any adjacent pair in one graph corresponds to an adjacent pair in the other graph.

Third Way: Their complements are isomorphic.

If any one of these statements is true, then they are all true. If any one of these statements is false, then they are all false.

WORK IT OUT

Here is an activity you can do with a few of your classmates that will build your graph comparison skills.

Step 1: Draw a planar graph with the following characteristics: exactly five vertices, one vertex of degree four, at least two vertices of degree three, and exactly eight edges. Give names to the vertices. Make sure you do not use the same letters or numbers to label your vertices as your classmates do.

Step 2: Analyze your graph. What is the degree of each vertex? Does your graph have any cyclic subgraphs? If so, list them and indicate their sizes.

Step 3: Draw and analyze the complement of your graph. How many edges and vertices does it have? What is the degree of each vertex? Does the complementary graph have any cyclic subgraphs? If so, list them and indicate their sizes.

Step 4: Compare your graphs to each of your classmates' graphs. Does your graph have the same number of edges and vertices as the graph of your classmate? Does your graph have the same size cyclic subgraphs as the graph of your classmate? How does the complement of your graph compare to the complement of the graph of your classmate? Determine if your graph is isomorphic to your classmates' graph. If so, give a correspondence that demonstrates the isomorphism. If not, explain how you know.

Key Terms

- isomorphic
- isomorphism
- planar
- nonplanar
- complement
- complementary

Key Concepts

- Two graphs are isomorphic if they have the same structure.
- When graphs are relatively small, we can use visual inspection to identify an isomorphism by transforming one graph into another without breaking connections or adding new ones.
- An isomorphism between two graphs preserves adjacency.
- If two graphs differ in number of vertices, number of edges, degrees of vertices, or types of subgraphs, they cannot be isomorphic.
- When the complements of two graphs are isomorphic, so are the graphs themselves.

Videos

- Determine If Two Graphs Are Isomorphic and Identify the Isomorphism

12.4 Navigating Graphs

Figure 68: Visitors navigate a garden maze.

Learning Objectives

After completing this section, you should be able to:

1. Describe and identify walks, trails, paths, and circuits.
2. Solve application problems using walks, trails, paths, and circuits.
3. Identify the chromatic number of a graph.
4. Describe the Four-Color Problem.
5. Solve applications using graph colorings.

Now that we know the basic parts of graphs and we can distinguish one graph from another, it is time to really put our graphs to work for us. Many applications of graph theory involve navigating through a graph very much like you would navigate through a maze. Imagine that you are at the entrance to a maze. Your goal is to get from one point to another as efficiently as possible. Maybe there are treasures hidden along the way that make straying from the shortest path worthwhile, or maybe you just need to get to the end fast. Either way, you definitely want to avoid any wrong turns that would cause unnecessary backtracking. Luckily, graph theory is here to help!

Walks

Suppose is a maze you want to solve. You want to get from the start to the end.

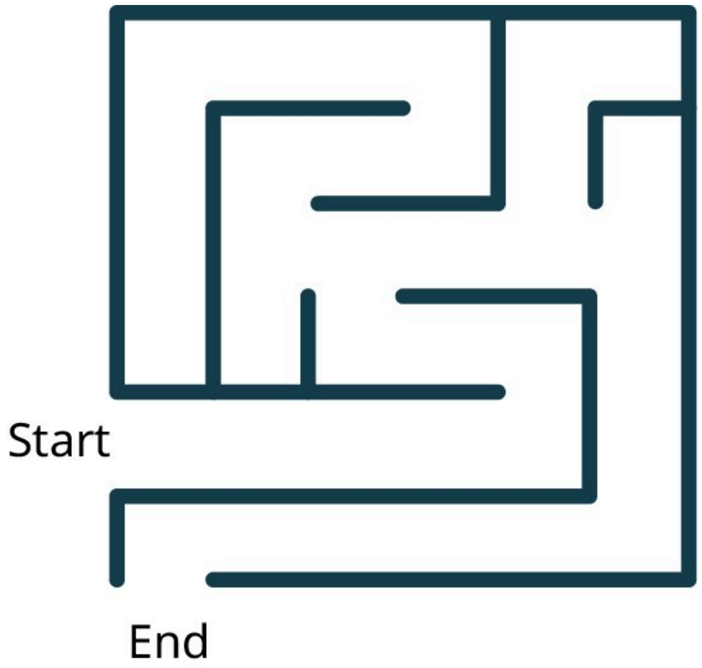


Figure 69: Your Maze

You can approach this task any way you want. The only rule is that you can't climb over the wall. To put this in the context of graph theory, let's imagine that at every intersection and every turn, there is a vertex. The edges that join the vertices must stay within the walls. The graph within the maze would look like .

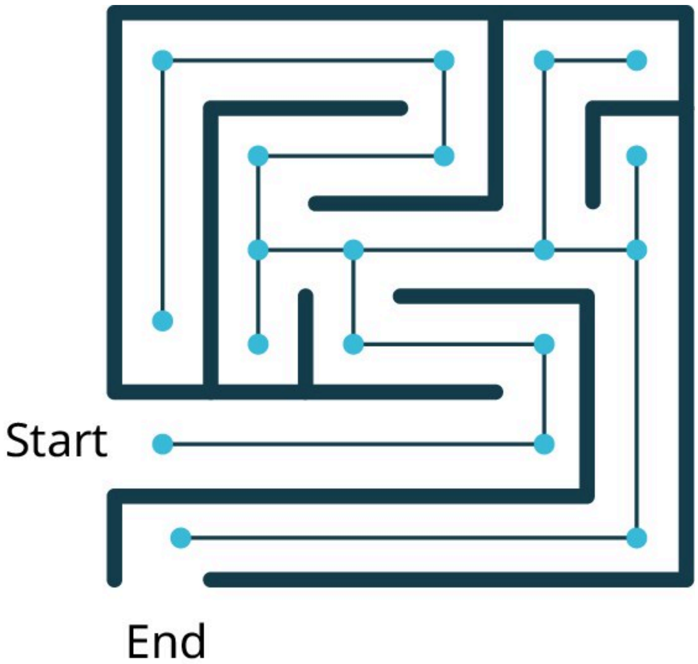


Figure 70: The Graph in Your Maze

One approach to solving a maze is to just start walking. It is not the most efficient approach. You might cross through the same intersection twice. You might backtrack a bit. It's okay. We are just out for a walk. It might look something like the black sequence of vertices and edges in .

Figure 71: The Walk Through the Graph in Your Maze

This type of sequence of adjacent vertices and edges is actually called a **walk** (or **directed walk**) in graph theory too!

A walk can be identified by naming the sequence of its vertices (or by naming the sequence of its edges if those are labeled). Let's take the graph out of the context of the maze and give each vertex a name and each edge of the walk a direction as in .

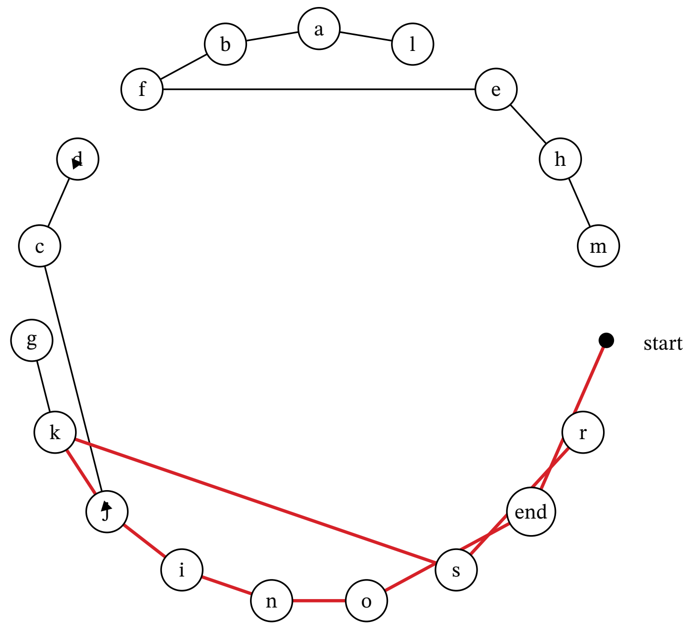


Figure 72: The Graph without Your Maze

The name of this walk from p to r is $p \rightarrow q \rightarrow o \rightarrow n \rightarrow i \rightarrow j \rightarrow c \rightarrow d \rightarrow c \rightarrow j \rightarrow k \rightarrow s \rightarrow r$. When a particular edge on our graph was traveled in both directions, it had arrows in both directions and the letters of vertices that were visited more than once had to be repeated in the name of the walk.

CHECKPOINT

Not every list of vertices or list of edges makes a path. The order must take into account the way in which the edges and vertices are connected. The list must be a sequence, which means they are in order. No edges or vertices can be skipped and you cannot go off the graph.

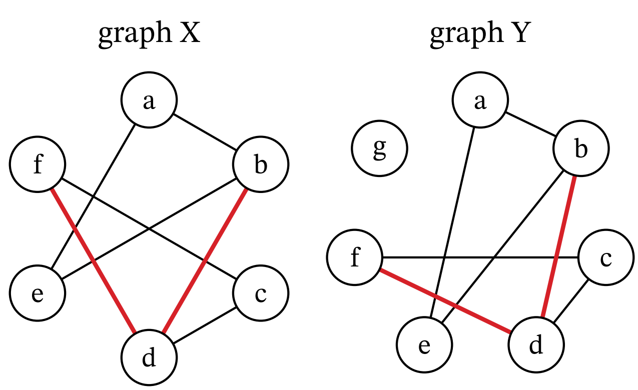


Figure 73: A Walk or Not a Walk?

The highlighted edges Graph Y in represent a walk between f and b. The highlighted edges in Graph X do *not* represent a walk between f and b, because there is a turn at a point that is not a vertex. This is like climbing over a wall when you are walking through a maze. Another way of saying this is that $b \longrightarrow d \longrightarrow f$ is not a walk, because there is no edge between b and d.

CHECKPOINT

A point on a graph where two edges cross is not a vertex.

Example 1 **Naming a Walk Through A House**

shows the floor plan of a house. Use the floor plan to answer each question.

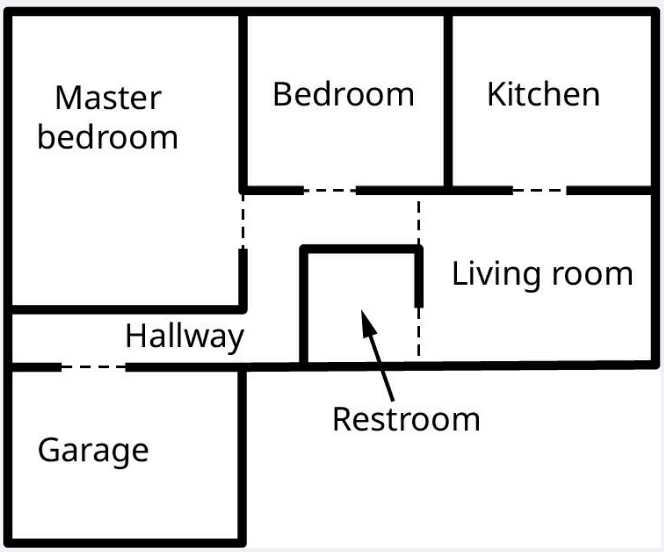


Figure 74: Floor Plan of a House

1. Draw a graph to represent the floor plan in which each vertex represents a different room (or hallway) and edges represent doorways between rooms.
2. Name a walk through the house that begins in the living room, ends in the garage and visits each room (or hallway) at least once.

Solution

1. **Step 1:** We will need a vertex for each room and it is convenient to label them according to the names of the rooms. Visualize the scenario in your head as shown. You don't have to write this step on your paper.

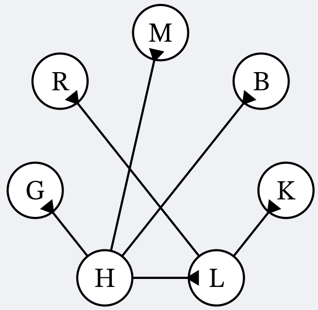


Figure 75: Assigning Vertices to Rooms

Step 2: Draw a graph to represent the scenario. Start with the vertices. Then connect those vertices that share a doorway in the floorplan as shown.

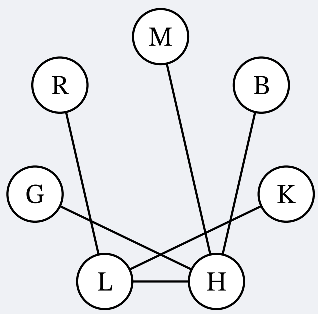


Figure 76: Graph of the Floor plan

1. **Step 1:** Draw a path that begins at vertex L, representing the living room, and ending at vertex G, representing the garage making sure to visit every room at least once. There are many ways this can be done. You may want to number the edges to keep track of their order. One example is shown.

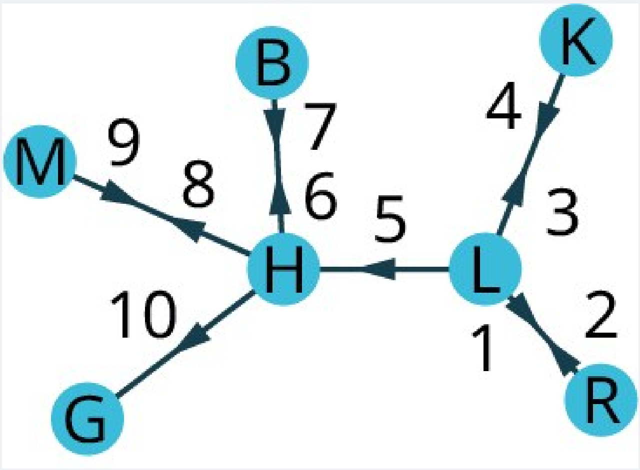


Figure 77: Draw the Path from L to G

Step 2: Name the path that you followed by listing the vertices in the order you visited them.

$L \to R \to L \to K \to L \to H \to B \to H \to M \to H \to G$

Paths and Trails

A walk is the most basic way of navigating a graph because it has no restrictions except staying on the graph. When there are restrictions on which vertices or edges we can visit, we will call the walk by a different name. For example, if we want to find a walk that avoids travelling the same edge twice, we will say we want to find a **trail** (or **directed trail**). If we want to find a walk that avoids visiting the same vertex twice, we will say, we want to find a **path** (or **directed path**).

Walks, trails, and paths are all related.

1. All paths are trails, but trails that visit the same vertex twice are not paths.
2. All trails are walks, but walks in which an edge is visited twice would not be trails.

   We can visualize the relationship as in .

(no graphs in spec)

Figure 78: Walks, Trails, and Paths

Let's practice identifying walks, trails, and paths using the graphs in .

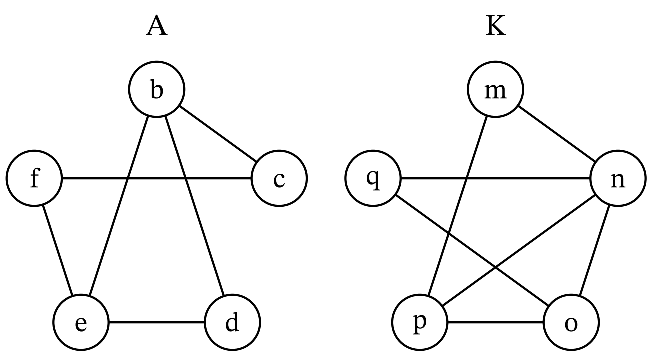


Figure 79: Graphs A and K

Example 2 **Identifying Walks, Paths, and Trails**

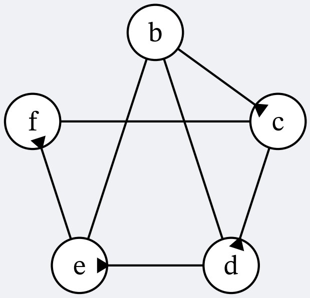
Consider each sequence of vertices from Graph A in . Determine if it is only a walk, both a walk and a path, both a walk and a trail, all three, or none of these.

1. $b \to c \to d \to e \to f$
2. $c \to b \to d \to b \to e$
3. $c \to f \to e \to d \to b \to c$
4. $b \to e \to f \to c \to b \to d$

Solution

1. First, check to see if the sequence of vertices is a walk by making sure that the vertices are consecutive. As you can see in , there is no edge between vertex *c* and vertex *d*.

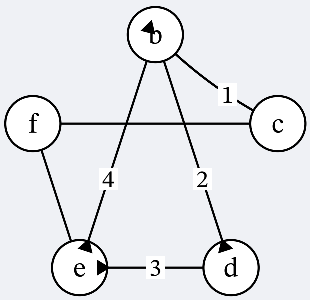
Graph A



This means that the sequence is not a walk. If it is not a walk, then it can't be a path and it cannot be a trail, so, it is none of these.

1. First, check to see if the sequence is a walk. As you can see in , the vertices are consecutive.

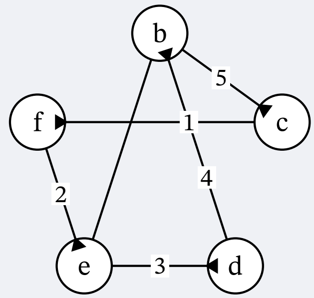
Graph A



This means that the sequence is a walk. Since the vertex *b* is visited twice, this walk is not a path. Since edge *bd* is traveled twice, this walk is not a trail. So, the sequence is only a walk.

1. First, check to see if the sequence is a walk. We can see in that the vertices are consecutive, which means it is a walk.

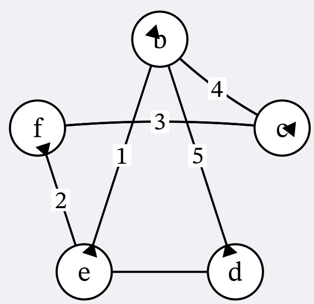
Graph A



Next, check to see if any vertex is visited twice. Remember, we do *not* consider beginning and ending at the same vertex to be visiting a vertex twice. So, no vertex was visited twice. This means we have a walk that is also a path. Next check to see if any edge was visited twice; none were. So, the sequence is a walk, a path, and a trail.

1. First, check to see if the sequence is a walk. We can see in that the vertices are consecutive, which means it is a walk.

Graph A



Next check to see if any vertex is visited twice. Since vertex *b* is visited twice, this is not a path. Finally, check to see if any edges are traveled twice. Since no edges are traveled twice, this is a trail. So, the sequence of vertices is a walk and a trail.

VIDEO

Walks, Trails, and Paths in Graph Theory

Circuits

In many applications of graph theory, such as creating efficient delivery routes, beginning and ending at the same location is a requirement. When a walk, path, or trail end at the same location or vertex they began, we call it **closed**. Otherwise, we call it **open** (does not begin and end at the same location or vertex). Some examples of closed walks, closed trails, and closed paths are given in in the following table.

DESCRIPTION	EXAMPLE	CHARACTERISTICS
A closed walk is a walk that begins and ends at the same vertex.	$d \longrightarrow f \longrightarrow b \longrightarrow c \longrightarrow f \longrightarrow d$	Alternating sequence of vertices and edges Begins and ends at the same vertex
A closed trail is a trail that begins and ends at the same vertex. It is commonly called a **circuit**.	$d \longrightarrow f \longrightarrow b \longrightarrow c \longrightarrow f \longrightarrow e \longrightarrow d$	No repeated edges Begins and ends at the same vertex
A closed path is a path that begins and ends at the same vertex. It is also referred to as a **directed cycle** because it travels through a cyclic subgraph.	$d \longrightarrow f \longrightarrow b \longrightarrow c \longrightarrow d$	No repeated edges or vertices Begins and ends at the same vertex

Since walks, trails, and paths are all related, closed walks, circuits, and directed cycles are related too.

1. All circuits are closed walks, but closed walks that visit the same edge twice are not circuits.
2. All directed cycles are circuits, but circuits in which a vertex is visited twice are not directed cycles.

   We can visualize the relationship as in .

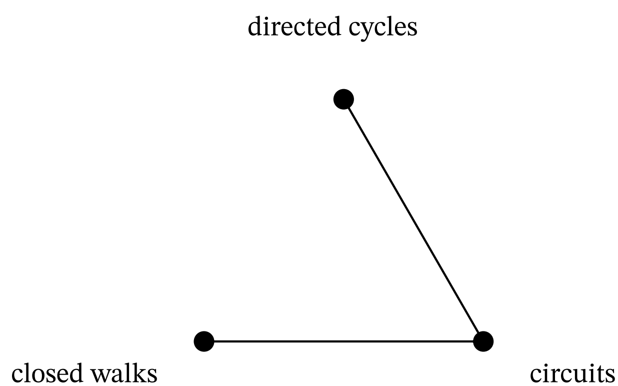


Figure 84: Closed Walks, Circuits, and Directed Cycles

The same circuit can be named using any of its vertices as a starting point. For example, the circuit $d \longrightarrow f \longrightarrow b \longrightarrow c \longrightarrow d$ can also be referred to in the following ways.

$a \longrightarrow b \longrightarrow c \longrightarrow d \longrightarrow a$ is the same as

$$\{b \to c \to d \to a \to b$$
$$c \to d \to a \to b \to c$$
$$d \to a \to b \to c \to d$$

Let's practice working with closed walks, circuits (closed trails), and directed cycles (closed paths). In the graph in , the vertices are major central and south Florida airports. The edges are direct flights between them.

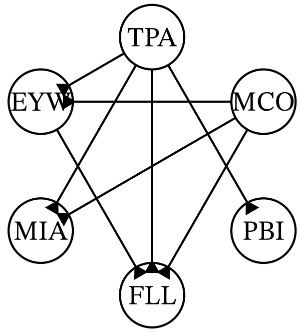


Figure 85: Major Central and South Florida Airports

Example 3 **Determining a Closed Walk, Circuit, or Directed Cycle**

Suppose that you need to travel by air from Miami (MIA) to Orlando (MCO) and you were restricted to flights represented on the graph. For the trip to Orlando, you decide to purchase tickets with a layover in Key West (EYW) as shown in , but you still have to decide on the return trip. Determine if your roundtrip itinerary is a closed walk, a circuit, and/or a directed cycle, based on the return trip described in each part.

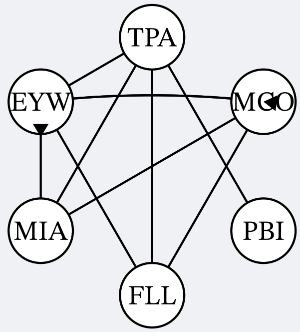


Figure 86: MIA to EYW to MCO

1. You returned to Miami (MIA) by reversing your route.
2. Your direct flight back left Orlando (MCO) but was diverted to Fort Lauderdale (FLL)! From there you flew to Tampa (TPA) before returning to Miami (MIA).

Solution

1. The whole trip was $MIA \longrightarrow EYW \longrightarrow MCO \longrightarrow EYW \longrightarrow MIA$. This is a closed walk, because it is a walk that begins and ends at the same vertex. It is not a circuit, because it repeats edges. If it is not a circuit, then it cannot be a directed cycle.
2. The whole trip was $MIA \longrightarrow EYW \longrightarrow MCO \longrightarrow FLL \longrightarrow TPA \longrightarrow MIA$. This is a closed walk, because it is a walk that begins and ends at the same vertex. It is a circuit

because no edges were repeated. It is also a directed cycle because no vertices were repeated either. So, it is all three!

VIDEO

Closed Walks, Closed Trails (Circuits), and Closed Paths (Directed Cycles) in Graph Theory

Graph Colorings

In this section so far, we have looked at how to navigate graphs by proceeding from one vertex to another in a sequence that does not skip any vertices, but in some applications we may want to skip vertices. Remember the camp Olympics at Camp Woebegone in Comparing Graphs? You were planning a camp Olympics with four events. The campers signed up for the events. You drew a graph to help you visualize which events have campers in common. The vertices of Graph *E* in represent the events and adjacent vertices indicate that there are campers who are participating in both.

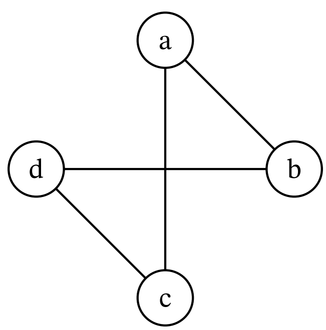


Figure 87: Graphs of Camp Olympics

In this case, we do NOT want events represented by two adjacent vertices to occur in the same timeslot, because that would prevent the campers who wanted to participate in both from doing so. We can use the graph in to count the timeslots we need so there are no conflicts. Let's assign each timeslot a different color. We could categorize events that happen at 1 pm as Red; 2 pm, Purple; 3 pm, Blue; and 4 pm, Green. Then assign different colors to any pair of adjacent vertices to ensure that the events they represent do not end up in the same timeslot. shows several of the ways to do this while obeying the rule that no pair of adjacent vertices can be the same color.

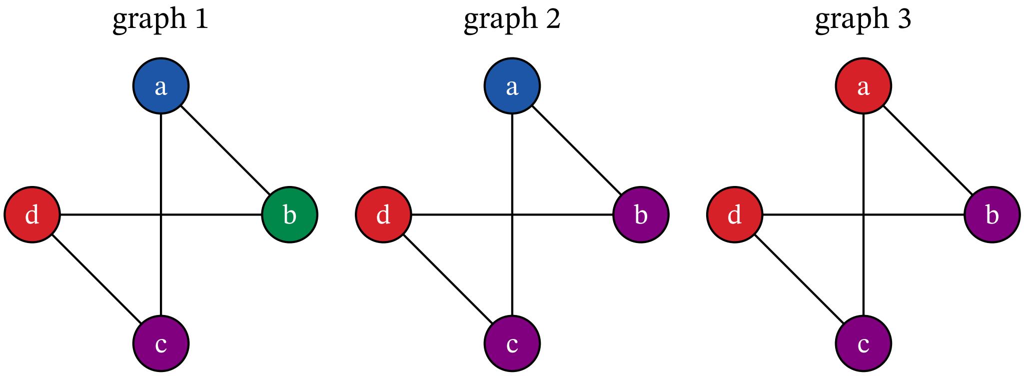


Figure 88: Graph Colorings

In , the graphs with vertices colored so that no adjacent vertices are the same color are called **graph colorings**. Notice that Graph 3 has the fewest colors, which means it shows us how to have the fewest number of timeslots. The events marked in red, *a* and *d*, can be held at the same time because they are not adjacent and do not have conflicts. Also, the events marked in purple, *b* and *c*, can be held at the same time. We would not need green or blue timeslots at all!

A graph that uses *n* colors is called an ***n*-coloring**. The smallest number of colors needed to color a particular graph is called its **chromatic number**.

Graph colorings can be used in many applications like the scheduling scenario at camp Woebegone. Let's look at how they work in more detail. shows two different colorings of a particular graph. Coloring A is called a four-coloring, because it uses four colors, red (R), green (G), blue (B), and purple (P). Coloring B is called a three-coloring because it uses three. The colors allow us to visually subdivide the graphs into groups. The only rule is that adjacent vertices are different colors so that they are in different groups.

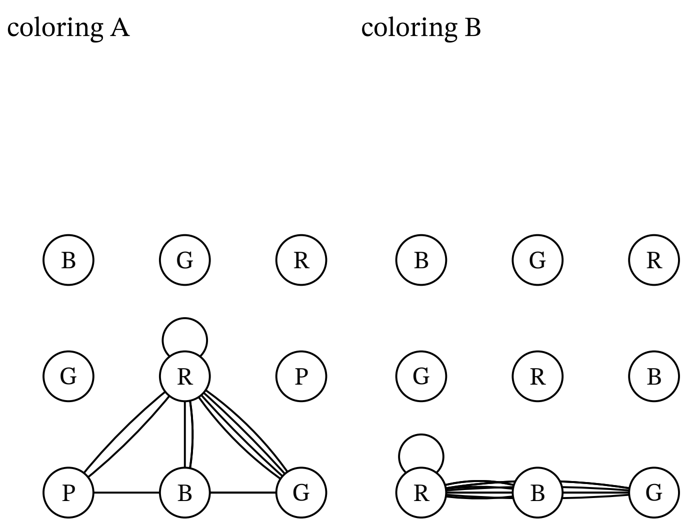


Figure 89: Two Colorings of the Same Graph

It turns out that a three-coloring is the best we can do with the graph in . No matter how many different patterns you try with only two colors, you will never find one in which the adjacent vertices are always different colors. In other words, the graph has a chromatic number of three. For large graphs, computer assistance is usually required to find the chromatic numbers. There is no formula for finding the chromatic number of a graph, but there are some facts that are helpful in .

Fact	Example
Recall that planar graphs are untangled; that is, they can be drawn on a flat surface so that no two edges are crossing.**If a graph is planar, it can be colored with four colors or possibly fewer.**	
Each vertex in a complete graph is adjacent to every other vertex forcing us to color them all different colors.**The chromatic number of a complete graph is the same as the number of vertices.**	
If a graph has a clique, a complete subgraph, then each vertex in the clique must have a different color and the vertices outside the clique may or may not have even more colors. **The chromatic number of a graph is *at least* the number of vertices in its biggest clique.**	

Creating Colorings to Solve Problems

Let's see how these facts can help us color the graph in .

- Since the graph is planar, the chromatic number is *no more than* four.
- The graph is not complete, but it has complete subgraphs of three vertices. In other words, it has triangles like the one shown with blue vertices in . This means that the chromatic number is *at least* three.

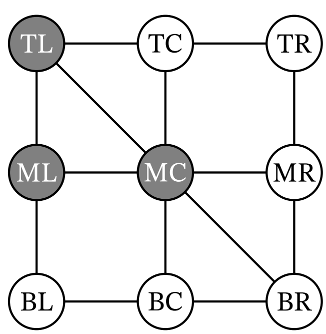


Figure 90: A Graph with Triangles

We know we can color this graph in three or four colors. It is usually best to start by coloring the vertex of highest degree as shown. In this case, we used red (R). The color is not important.

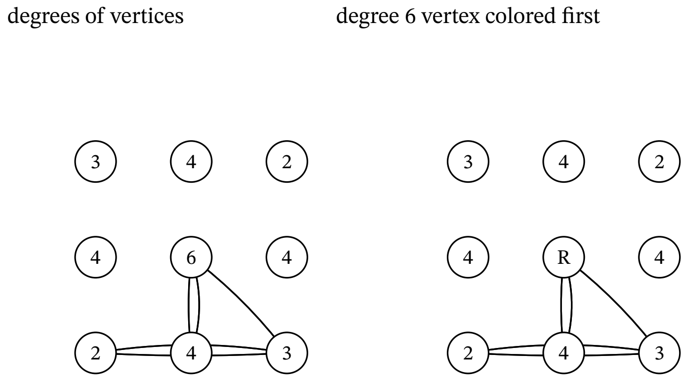


Figure 91: Color Highest Degree Vertex First

We want to color as many of the vertices the same color as possible; so, we look at all the vertices that are not adjacent to the red vertex and begin to color them, red starting with the one among them of highest degree. Since the only vertices that are not adjacent are both degree 2, choose either one and color it red as shown.

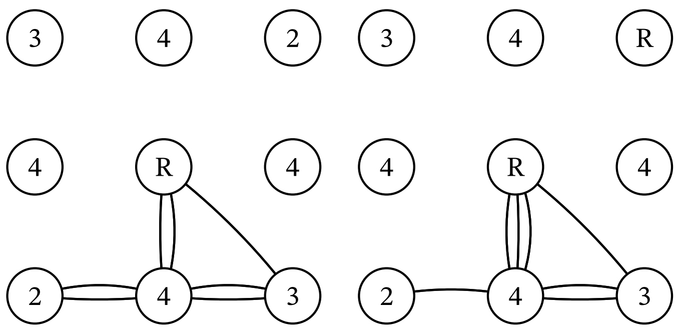


Figure 92: Color More Red from Highest to Lowest Degree

Now, there is only one vertex left that is not adjacent to a red; so, color it red. Of the remaining vertices, the highest degree is four; so, color one of the vertices of degree four in a different color. These two steps are shown.

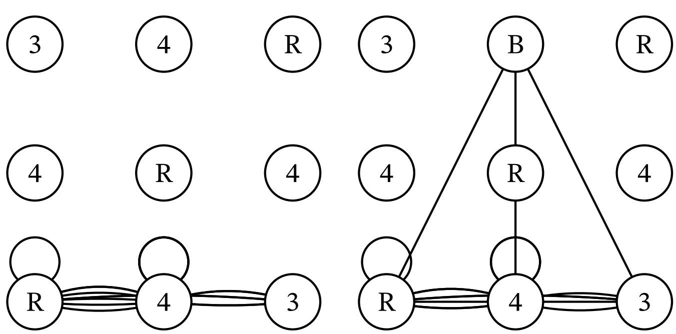


Figure 93: Complete the Reds and Start Another Color

Repeat the same procedure. There are three remaining vertices that are not adjacent to a blue. Color as many blue as possible, with priority going to vertices of higher degree as shown.

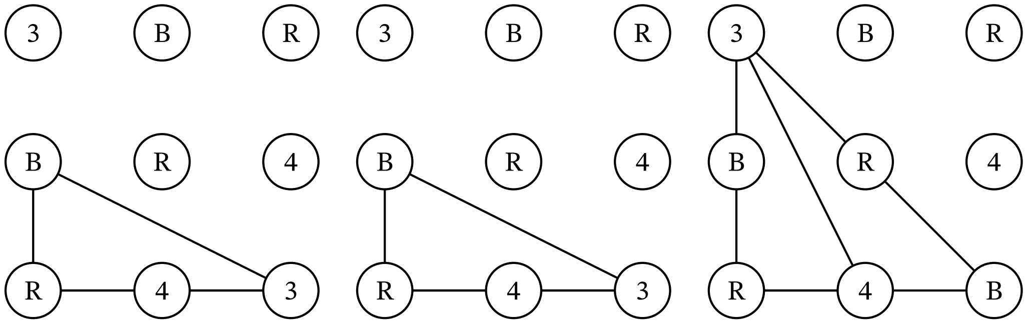


Figure 94: Repeat Procedure for Color Blue

All the remaining vertices are adjacent to blue. So, it is time to repeat the procedure with another color as shown.

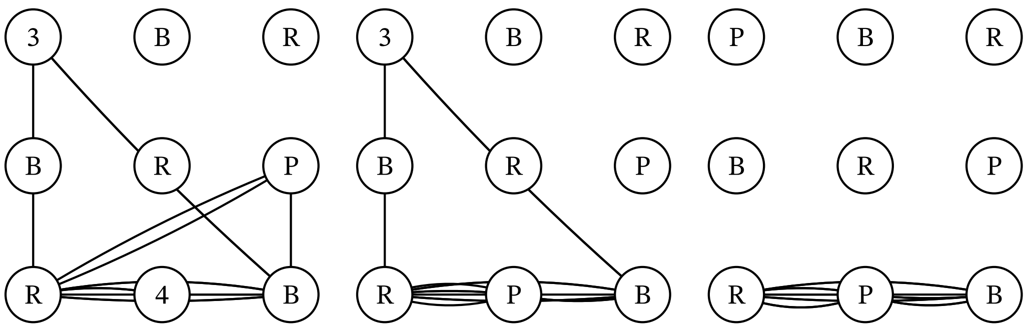


Figure 95: Repeat Procedure for Color Purple

All the vertices are now colored with a three-coloring so we know the chromatic number is *at most* three, but we knew the chromatic number was *at least* three because the graph has a triangle. So, we are now certain it is *exactly*three.

VIDEO

Coloring Graphs Part 1: Coloring and Identifying Chromatic Number

Suppose the vertices of the graph in represented the nine events in the Camp Woebegone Olympics, and edges join any events that have campers in common, but nonadjacent vertices do not.

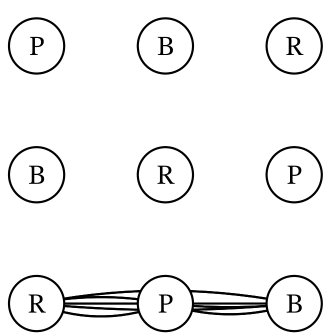


Figure 96: Coloring for Nine Event Camp Woebegone Olympics

Any vertices of the same color are nonadjacent; so, they have no conflicts. Since this graph is a three-coloring, all nine events could be scheduled in just three timeslots!

Example 4 **Understanding Chromatic Numbers**

In Example 5 in Section 12.3, we discussed a high school, which holds end-of-course exams in (E3) English 3, (E4) English 4, (M) Advanced Math, (C) Calculus, (W) World History, (U) U.S. History, (B) Biology, and (P) Physics. We were given a list of courses that had no students in common. We used that information to find the graph in , which shows edges between exams with students in common. Use the graph we found in Example 5 in Section 12.3 to answer each question.

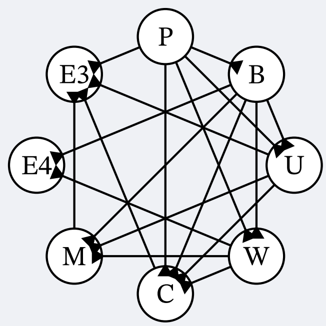


Figure 97: Graph of Exams with Students in Common

1. The graph contains a clique of size 4 formed by the vertices P, E3, C, and U. What does this tell you about the chromatic number?
2. The graph is not planar, meaning that you cannot untangle it. What does this tell you about the chromatic number?
3. Create a coloring by coloring vertex of highest degree first, coloring as many other vertices as possible each color from highest to lowest degree, then repeating this process for the remaining vertices.
4. Do you know what the minimum number of timeslots is? If so, what is it and how do you know? If not, what are the possibilities?

Solution

1. We would need four different colors just for the clique with four vertices; so, the chromatic number is *at least* four.
2. It is possible for the chromatic number to be greater than four.
3. The process is shown.

This is the original graph. Vertex B has highest degree.	Color vertex B. Vertex E3 is the only remaining vertex that is NOT adjacent to B.	Color vertex E3 the same color. Vertices P, U, W, and C are the remaining vertices with highest degree. Pick one to color.
Color vertex P a new color. Vertices E4 and M are NOT adjacent to P, and M has higher degree.	Color vertex M the same color. Vertex E4 is the only remaining vertex NOT adjacent to P or M.	Color vertex E4 the same color. Vertices U, W, and C are the remaining vertices with highest degree. Pick one to color.
Color U a new color. Vertex W is the only remaining vertex NOT adjacent to U.	Color vertex W the same color. Vertex C is the only remaining vertex that has NOT been colored.	Color vertex C a new color. The coloring is final. We used four colors.

The last graph in is the final coloring.

1. Yes, the minimum number of times slots is the chromatic number. We knew the chromatic number had to be at least four because there was a clique with four vertices. Now we have found a four-coloring of the graph which tells us that the chromatic number is at most four. So, we know four must be the chromatic number.

CHECKPOINT

The procedure used in Example 22 to color the graph is not guaranteed to result in a graph that has the minimum number of colors possible, but it is usually results in a coloring that is close to the chromatic number.

The Four Color Problem

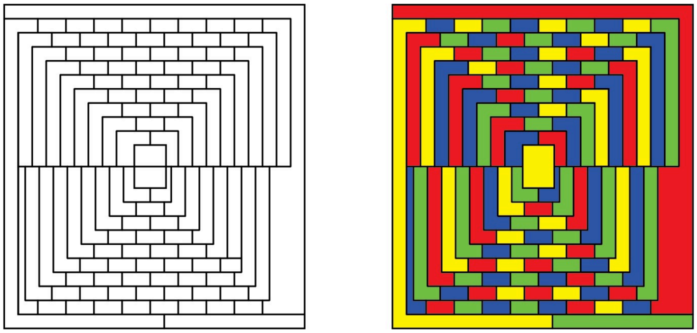


Figure 98: Using only four colors, no two adjacent regions have the same color.

The idea of coloring graphs to solve problems was inspired by one of the most famous problems in mathematics, the "four color problem." The idea was that, no matter how complicated a map might be, only four colors were needed to color the map so that no two regions that shared a boundary would be the same color. For many years, everyone suspected this to be true, because no one could create a map that needed more than four colors, but they couldn't prove it was true in general. Finally, graphs were used to solve the problem!

VIDEO

The Four Color Map Theorem – Numberphile

We saw how maps can be represented as graphs in Graph Basics. from Example 4 in Section 12.1 shows a map of the midwestern region of the United States.

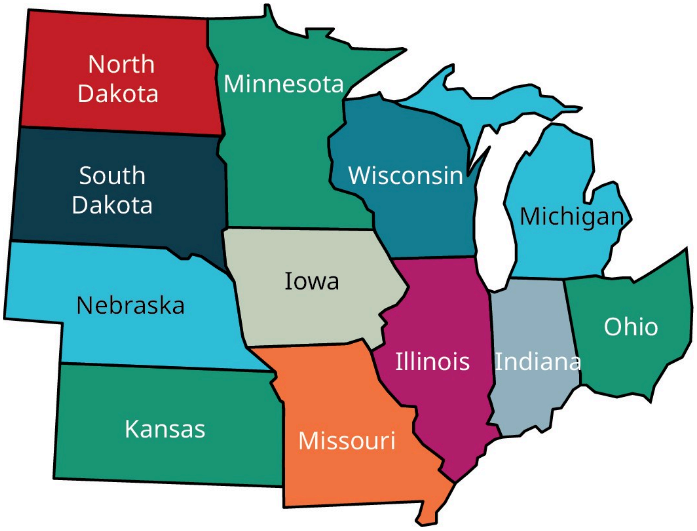


Figure 99: Map of Midwestern States

shows how this map can be associated with a graph in which each vertex represents a state and each edge indicates the states that share a common boundary. shows the final graph.

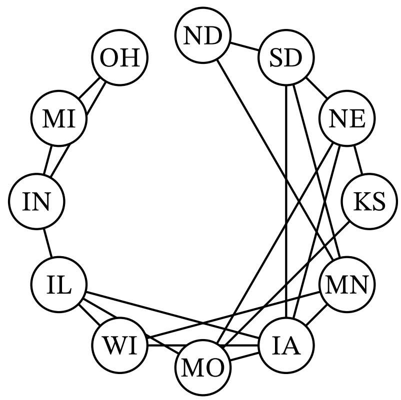


Figure 100: Edge Assigned to Each Pair of Midwestern States with Common Border

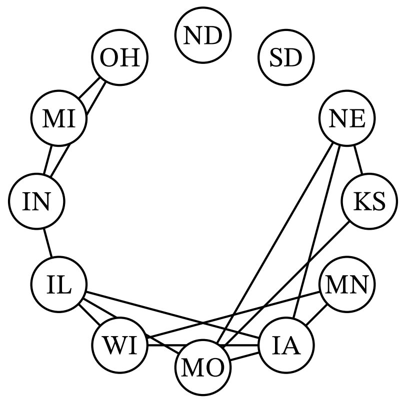


Figure 101: Final Graph Representing Common Boundaries between Midwestern States

Notice that the graph representing the common boundaries between midwestern states is planar, meaning that it can be drawn on a flat surface without edges crossing. As we have seen, any planar graph has a chromatic number of four or less. This very well-known fact is called the **Four-Color Theorem**, or **Four-Color Map Theorem**.

VIDEO

Coloring Graphs Part 2: Coloring Maps and the Four Color Problem

PEOPLE IN MATHEMATICS

Four-Color Theorem

In the 19th century, Francis Guthrie was coloring a map of British counties and noticed that he could color so that no two adjacent counties were the same color using only four colors. This seemed to be the case for other maps as well, but he could not figure out why. He shared this with his brother, Frederick, who subsequently took the problem to his professor, August De Morgan, one of the most famous mathematicians of all time. De Morgan never was able to solve the problem, but he shared it with his colleagues. The problem continued to intrigue and perplex mathematicians for over a hundred years, inspiring discoveries in several areas of mathematics. It was finally proven by Kenneth Appel and Wolfgang Haken from the University of Illinois in 1976 using a combination of computer-aided calculations and graph theory. This was the first time in history that a mathematical theorem was proven using a computer! (Jesus Najera, “The Four-Color Theorem,” *Cantor’s Paradise*, a Medium publication, October 22, 2019)

Example 5 **Using Four Colors or Fewer to Color a Graph**

Find a coloring of the graph in , which uses four colors or fewer. Use the resulting coloring as a guide to recolor the map in . How many colors did you use? Does this support the conclusion of the Four-Color Theorem? If so, how?

Solution

The steps to color the graph are shown.

Step 1: Graph with degrees of vertices labeled. Vertex IA has highest degree.	**Step 2:** Color vertex IA any color. Vertices ND, KS, MI, OH, and IN are NOT adjacent to IA. MI and IN have highest degree, 3.	**Step 3:** Color either MI or IN the same color as IA. Vertex ND and KS are the only remaining vertices not adjacent to a red vertex. They both have degree 2.
Step 4: Since KS and ND are not adjacent to each other, we can save a step and color both red. The highest degree of remaining vertices is four.	**Step 5:** Choose one of the vertices of degree 4, SD, to color with a new color, blue. The vertices WI, IL, MO, IN,and OH are NOT adjacent to blue. WI, IL, and MO have the highest degree.	**Step 6:** Choose one of WI, IL, or MO to color. We color WI blue. MO, IN and OH are NOT adjacent to blue. MO has the highest degree of these.
Step 7: Color MO blue. All remaining vertices are adjacent to blue. Choose a new color. Four vertices remain, MN, NE, IL, and OH. MN, NE, and IL have the highest degree.	**Step 8:** Since MN, IL, and NE are not adjacent, save steps and color all three the new color, purple. Vertex OH is the only remaining vertex that has NOT been colored.	**Step 9:** Since vertex OH is not adjacent to purple, color it purple. This is the final graph. We used three colors.

The final graph in shows how we would color the map. In we have colored the map to correspond to the colors on the graph.

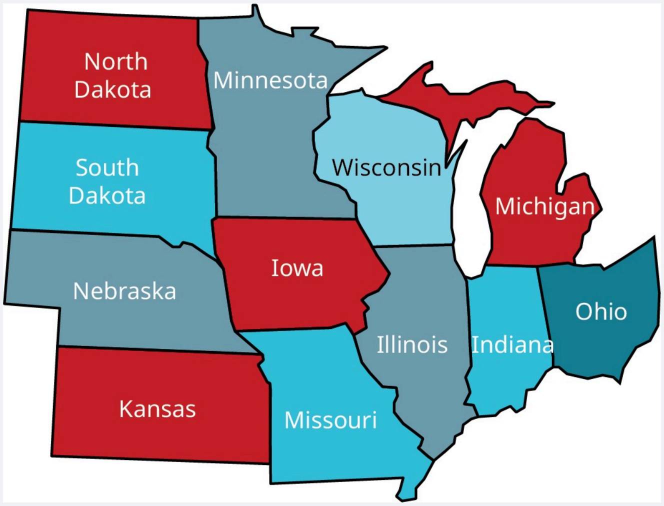


Figure 102: Midwestern States in Three Colors

We used three colors to color the graph. This supports the Four-Color Theorem, because the graph is planar and its chromatic number is less than four.

WHO KNEW?

Coloring A Möbius Strip

Have you ever heard of a möbius strip? It is a flat object with only one side. This might sound theoretical, but you can make one for yourself. Take a narrow strip of paper, make a half twist in it, and tape the ends together. To prove to yourself that it has only one side, pick a point anywhere on the paper and start drawing a line. You can draw the line continuously without lifting your pen from the paper and eventually, you will get back to the point at which you started. Since you never had to flip over the paper, you have just proved it has one side!

Now, you might be wondering what this has to do with Graph Theory. It turns out that a map drawn on a möbius strip cannot necessarily be drawn using four colors. This doesn't contradict the Four-Color Theorem, which only applies to graphs that are planar. A möbius strip does not live on a "plane," or flat surface, which means the maps are not always planar like those drawn on a flat piece of paper. In the case of a möbius strip, it is the Six-Color Map Theorem! is an example of a möbius strip with a map that requires five colors.

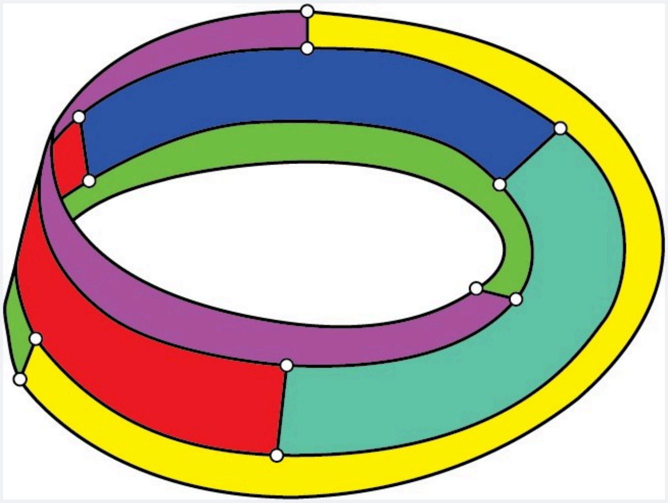


Figure 103: A Five-Coloring on a Möbius Strip

VIDEO

Neil deGrasse Tyson Explains the Möbius Strip

Key Terms

- walk (directed walk)
- trail (directed trail)
- path (directed path)
- closed
- open
- closed walk
- circuit (closed trail)
- directed cycle (closed path)
- coloring (graph coloring)
- n-coloring
- chromatic number

Key Concepts

- Walks, trails, and paths are ways to navigate through a graph using a sequence of connected vertices and edges.
- Closed walks, circuits, and directed cycles are ways to navigate from a vertex on a graph and return to the same vertex.

- Colorings are a way to organize the vertices of a graph into groups so that no two members of a group are adjacent.
- Maps can be represented with planar graphs, which can always be colored using four colors or fewer.

Videos

- Walks, Trails, and Paths in Graph Theory
- Closed Walks, Closed Trails (Circuits), and Closed Paths (Directed Cycles) in Graph Theory
- Coloring Graphs Part 1: Coloring and Identifying Chromatic Number
- The Four Color Map Theorem – Numberphile
- Coloring Graphs Part 2: Coloring Maps and the Four Color Problem
- Neil deGrasse Tyson Explains the Möbius Strip

12.5 Euler Circuits

Figure 104: Delivery trucks move goods from place to place.

Learning Objectives

After completing this section, you should be able to:

1. Determine if a graph is connected.
2. State the Chinese postman problem.
3. Describe and identify Euler Circuits.
4. Apply the Euler Circuits Theorem.
5. Eulerize graphs in real-world applications.

The delivery of goods is a huge part of our daily lives. From the factory to the distribution center, to the local vendor, or to your front door, nearly every product that you buy has been shipped multiple times to get to you. If the cost and time of that delivery is too great, you will not be able to afford the product. Delivery personnel have to leave from one location, deliver the goods to various places, and then return to their original location and do all of this in an organized way without losing money. How do delivery services find the most efficient delivery route? The answer lies in graph theory.

Connectedness

Before we can talk about finding the best delivery route, we have to make sure there is a route at all. For example, suppose that you were tasked with visiting every airport on the graph in by plane. Could you accomplish that task, only taking direct flight paths between airports listed on this graph? In other words, are all the airports connected by paths? Or are some of the airports disconnected from the others?

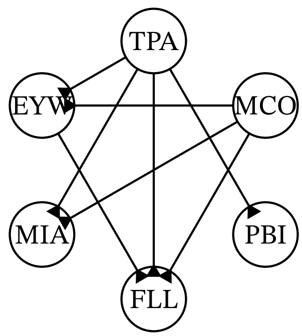


Figure 105: Major Central and South Florida Airports

In , we can see TPA is adjacent to PBI, FLL, MIA, and EYW. Also, there is a path between TPA and MCO through FLL. This indicates there is a path between each pair of vertices. So, it is possible to travel to each of these airports only taking direct flight paths and visiting no other airports. In other words, the graph is **connected** because there is a path joining every pair of vertices on the graph. Notice that if one vertex is connected to each of the others in a graph, then all the vertices are connected to each other. So, one way to determine if a graph is connected is to focus on a single vertex and determine if there is a path between that vertex and each of the others. If so, the graph is connected. If not, the graph is **disconnected,** which means at least one pair of vertices in a graph is not joined by a path. Let's take a closer look at graph X in . Focus on vertex a. There is a path between vertices a and b, but there is no path between vertex a and vertex c. So, Graph X is disconnected.

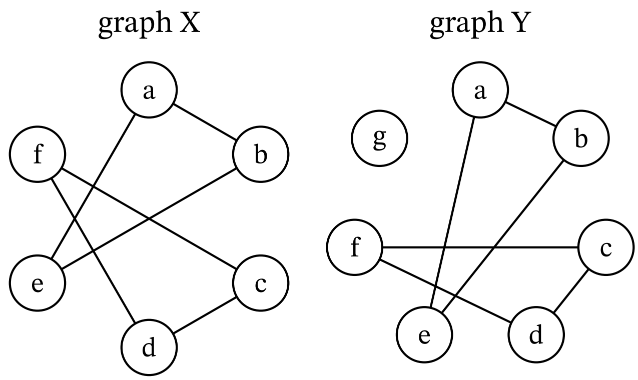


Figure 106: Connected vs. Disconnected

When you are working with a planar graph, you can also determine if a graph is connected by untangling it. If you draw it so that so that none of the edges are overlapping, like we did with Graph *X* in , it is easier to see that the graph is disconnected.

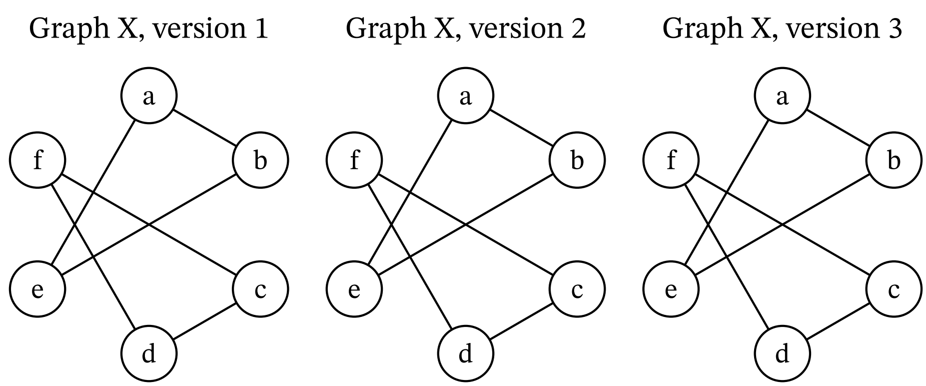


Figure 107: Untangling Graph *X*

Versions 2 and 3 of Graph *X* in each have the same number of vertices, number of edges, degrees of the vertices, and pairs of adjacent vertices as version 1. In other words, each version retains the same structure as the original graph. Since versions 2 and 3 of Graph *X*, do not have overlapping edges, it is easier to identify pairs of vertices that do not have paths between them, and it is more obvious that Graph *X* is disconnected. In fact, there are two completely separate, disconnected subgraphs, one with the vertices in $\{a, b, e\}$, and the other with the vertices $\{c, d, f\}$

These sets of vertices together with all of their edges are called **components** of Graph *X*. A component of a graph is a subgraph in which there is a path between each pair of vertices in the subgraph, but no edges between any of the vertices in the subgraph and a vertex that is not in the subgraph.

Now let's focus on Graph *Y* from . As in Graph *X*, there is a path between vertices a and b, as well as between vertices a and e, but Graph *Y* is different from Graph *X* because of vertex g. Not only is there a path between vertices a and g, but vertex g bridges the gap between a and c with the path $a \rightarrow b \rightarrow g \rightarrow c$. Similarly, there is a path between vertices a and d and vertices a and f: $a \rightarrow b \rightarrow g \rightarrow d$, $a \rightarrow b \rightarrow g \rightarrow d \rightarrow f$. Since there is a path between vertex a and every other vertex, Graph *Y* is connected. You can also see this a bit more clearly by untangling Graph *Y* as in . Even when Y is drawn so that the edges do not overlap, the graph cannot be drawn as two separate, unconnected pieces. In other words, Graph Y has only one component with the vertices $\{a, b, c, d, e, f\}$.

We can give an alternate definition of **connected** and **disconnected** using the idea of components. A graph is connected if it has only one component. A graph is disconnected if it has more than one component. These alternate definitions are equivalent to the previous definitions. This means that you can confirm a graph is connected or disconnected either by checking to see if there is a path between each vertex and each other vertex, or by identifying the number of components. A graph is connected if it has only one component.

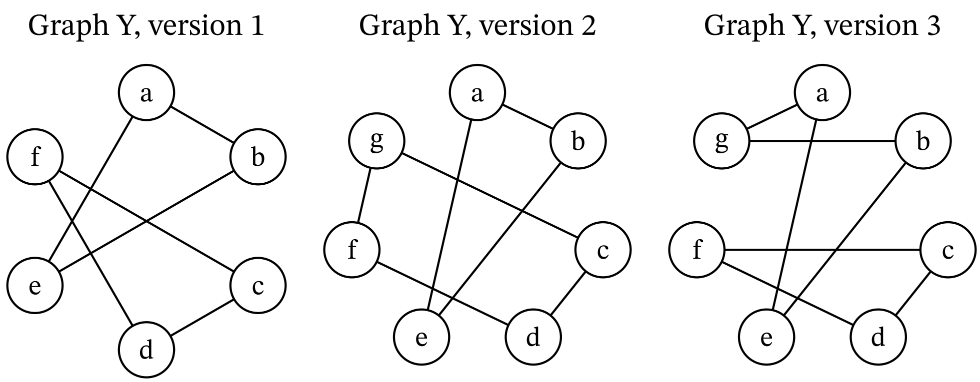


Figure 108: Untangling Graph Y

Example 1 **Determining If a Graph Is Connected or Disconnected**

Use to answer each question.

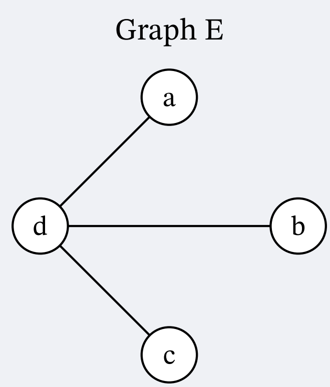


Figure 109: Graph E

1. Find a path between vertex a and every other vertex on the graph, if possible.
2. Identify all the components of Graph E.
3. Determine whether the graph is connected or disconnected and explain how you know.

Solution

1. The paths are $a \longrightarrow d \longrightarrow b$, $a \longrightarrow d \longrightarrow c$, and $a \longrightarrow d$.

2. There is only one component in Graph E. It has the vertices $\{a, b, c, d\}$.
3. The graph is connected, because there is a path between vertex a and every other vertex. We can also show that Graph E is connected because it has only one component.

VIDEO

Connected and Disconnected Graphs in Graph Theory

Example 2

The U.S. Interstate Highway System extends throughout the 48 contiguous states. It also has routes in the states of Hawaii and Alaska, and the commonwealth of Puerto Rico. Consider a graph representing the U.S. Interstate Highway System, in which there is a vertex for each of the 50 states and Puerto Rico, and an edge is drawn between any two vertices representing states that are connected by a highway in that system. Would this graph be connected or disconnected? Explain your reasoning.

Solution

Hawaii, Alaska, and Puerto Rico are geographically separate from the 48 contiguous states, and each from each other. This means that there are vertices on the graph with no path joining them to the other vertices. So the graph is disconnected.

Origin of Euler Circuits

The city of Konigsberg, modern day Kaliningrad, Russia, has waterways that divide up the city. In the 1700s, the city had seven bridges over the various waterways. The map of those bridges is shown. The question as to whether it was possible to find a route that crossed each bridge exactly once and return to the starting point was known as the Konigsberg Bridge Problem. In 1735, one of the most influential mathematicians of all time, Leonard Euler, solved the problem using an area of mathematics that he created himself, graph theory!

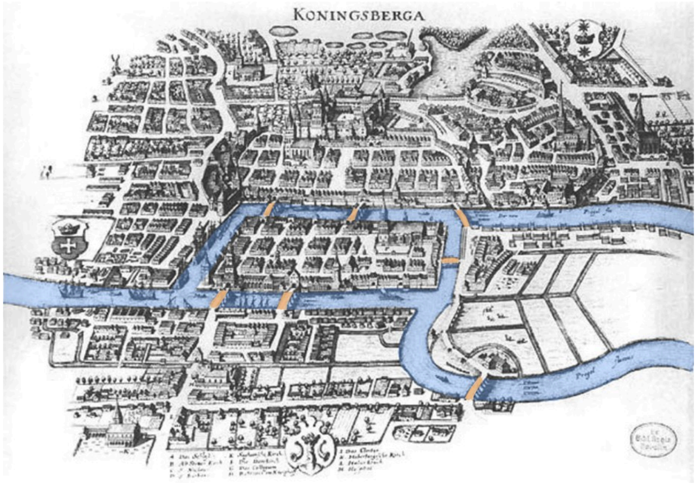


Figure 110: Map of the Bridges of Konigsberg in 1700s

Euler drew a multigraph in which each vertex represented a land mass, and each edge represented a bridge connecting them, as shown. Remember from Navigating Graphs that a circuit is a trail, so it never repeats an edge, and it is closed, so it begins and ends at the same vertex. Euler pointed out that the Konigsberg Bridge Problem was the same as asking this graph theory question: Is it possible to find a circuit that crosses every edge? Since then, circuits (or closed trails) that visit every edge in a graph exactly once have come to be known as **Euler circuits** in honor of Leonard Euler.

VIDEO

Recognizing Euler Trails and Euler Circuits

Euler was able to prove that, in order to have an Euler circuit, the degrees of all the vertices of a graph have to be even. He also proved that any graph with that characteristic must have an Euler circuit. So, saying that a connected graph is Eulerian is the same as saying it has vertices with all even degrees, known as the **Eulerian circuit theorem**.

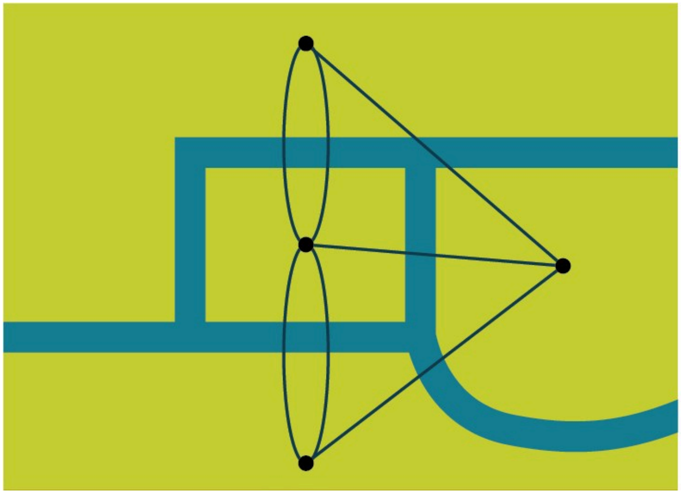


Figure 111: Graph of Konigsberg Bridges

To understand why the Euler circuit theorem is true, think about a vertex of degree 3 on any graph, as shown.

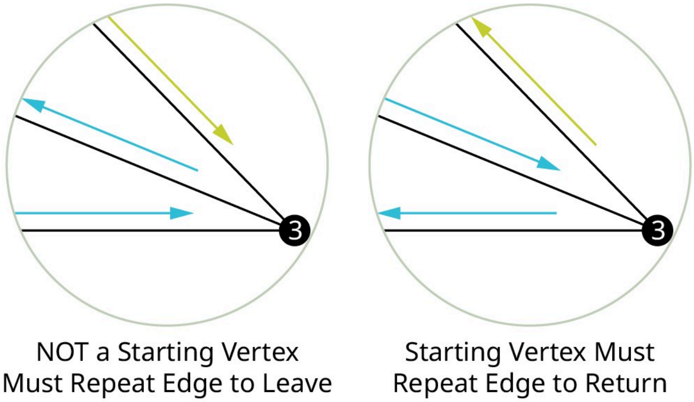


Figure 112: A Vertex of Degree 3

First imagine the vertex of degree 3 shown in is not the starting vertex. At some point, each edge must be traveled. The first time one of the three edges is traveled, the direction will be toward the vertex, and the second time it will be away from the vertex. Then, at some point, the third edge must be traveled coming in toward the vertex again. This is a problem, because the only way to get back to the starting vertex is to then visit one of the three edges a second time. So, this vertex cannot be part of an Euler circuit.

Next imagine the vertex of degree 3 shown in is the starting vertex. The first time one of the edges is traveled, the direction is away from the vertex. At some point, the second edge will be traveled coming in toward the vertex, and the third edge will be the way back out, but the starting vertex is also the ending vertex in a circuit. The only way to return to the vertex is now to travel one of the edges a second time. So, again, this vertex cannot be part of an Euler circuit.

For the same reason that a vertex of degree 3 can never be part of an Euler circuit, a vertex of any odd degree cannot either. We can use this fact and the graph in to solve the Konigsberg Bridge Problem. Since the degrees of the vertices of the graph in are not even, the graph is not Eulerian and it cannot have an Euler circuit. This means it is not possible to travel through the city of Konigsberg, crossing every bridge exactly once, and returning to your starting position.

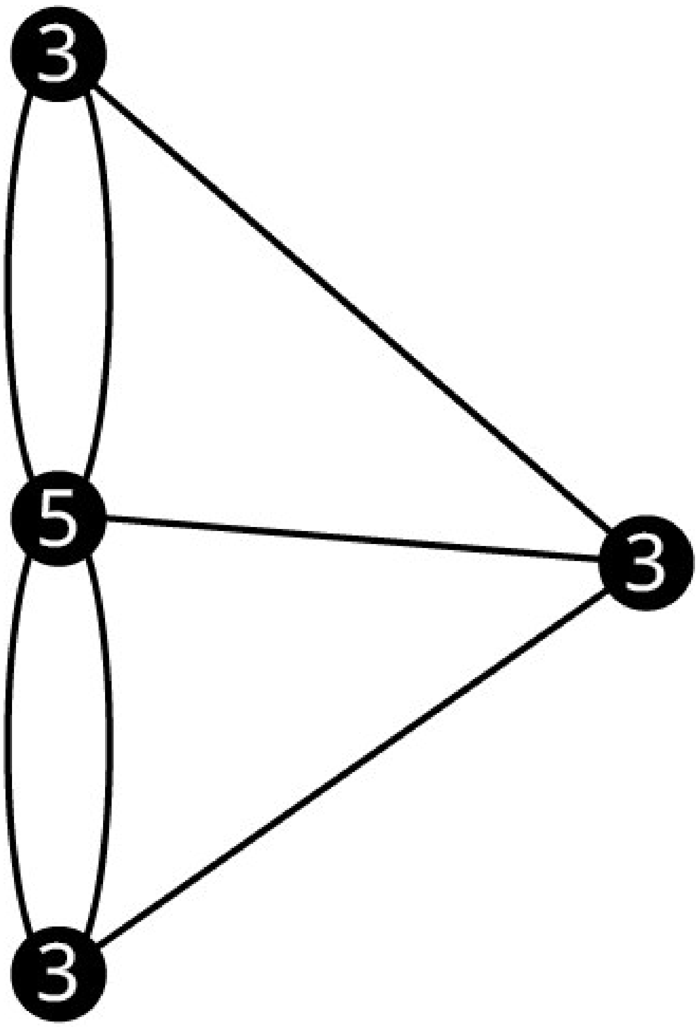


Figure 113: Degrees of Vertices in Konigsberg Bridge Multigraph

VIDEO

Existence of Euler Circuits in Graph Theory

Chinese Postman Problem

At Camp Woebegone, campers travel the waterways in canoes. As part of the Camp Woebegone Olympics, you will hold a canoeing race. You have placed a checkpoint on each of the 11 different streams. The competition requires each team to travel each stream, pass through the checkpoints in any order, and return to the starting line, as shown in the .

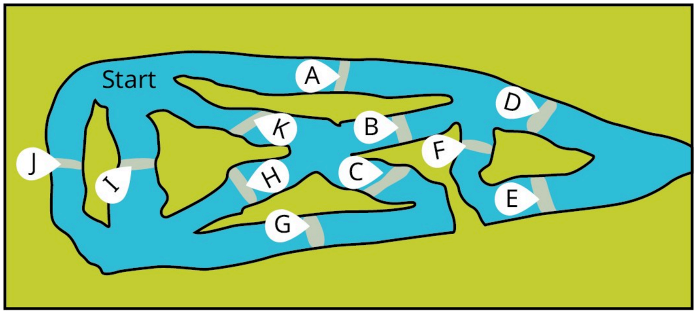


Figure 114: Map of Canoe Event Checkpoints

Since the teams want to go as fast as possible, they would like to find the shortest route through the course that visits each checkpoint and returns to the starting line. If possible, they would also like to avoid backtracking. Let's visualize the course as a multigraph in which the vertices represent turns and the edges represent checkpoints as in .

Canoe Event Checkpoints

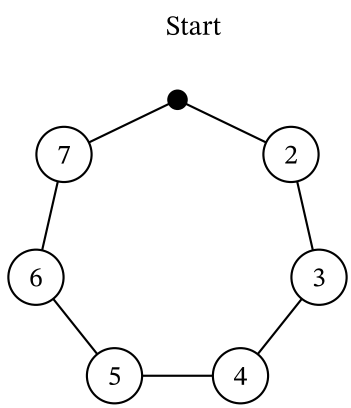


Figure 115: Multigraph of Canoe Event

The teams would like to find a closed walk that repeats as few edges as possible while still visiting every edge. If they never repeat an edge, then they have found a closed trail, which is a circuit. That circuit must cover all edges; so, it would be an Euler circuit. The task of finding a shortest circuit that visits every edge of a connected graph is often referred to as the **Chinese postman problem**. The name Chinese postman problem was coined in honor of the Chinese mathematician named Kwan Mei-Ko in 1960 who first studied the problem.

If a graph has an Euler circuit, that will always be the best solution to a Chinese postman problem. Let's determine if the multigraph of the course has an Euler circuit by looking at the degrees of the vertices in . Since the degrees of the vertices are all even, and the graph is connected, the graph is Eulerian. It is possible for a team to complete the canoe course in such a way that they pass through each checkpoint exactly once and return to the starting line.

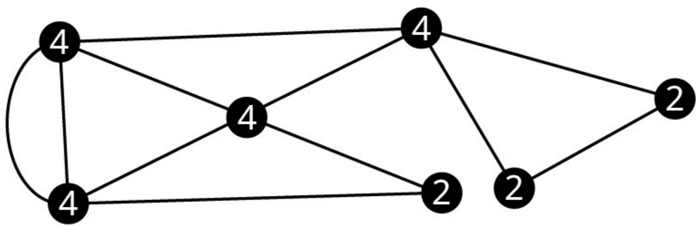


Figure 116: Degrees of Vertices in Graph of Canoe Event

Example 3 **Understanding Eulerian Graphs**

A postal delivery person must deliver mail to every block on every street in a local subdivision. is a map of the subdivision. Use the map to answer each question.

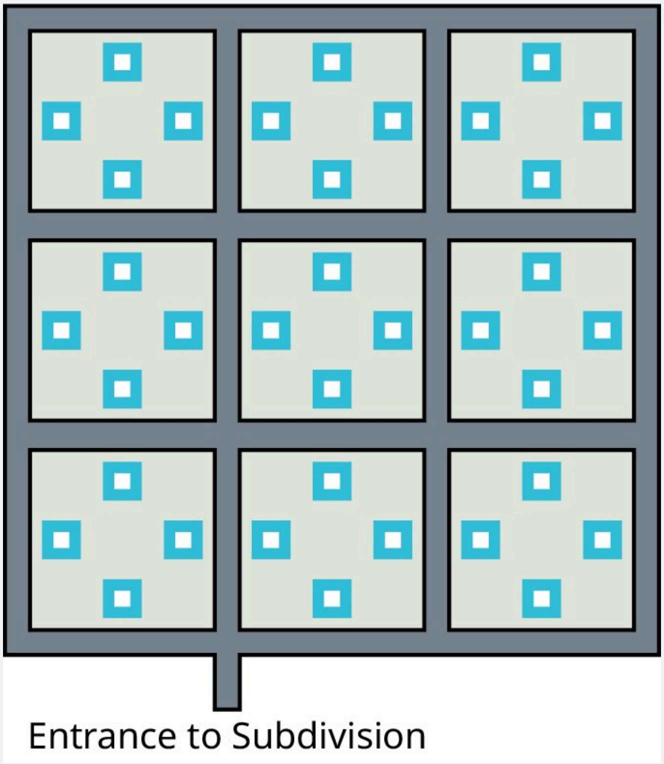


Figure 117: Map of Subdivision

1. Draw a graph or multigraph to represent the subdivision in which the vertices represent the intersections, and the edges represent streets.
2. Is your graph connected? Explain how you know.
3. Determine the degrees of the vertices in the graph.
4. Is your graph an Eulerian graph?

5. Is it possible for the postal delivery person to visit each block on each street exactly once? Justify your answer.

Solution

1. The graph is in .

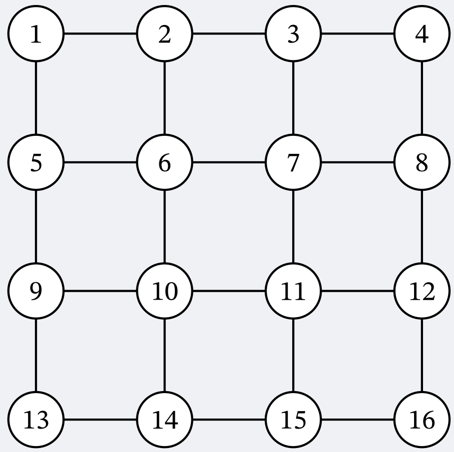


Figure 118: Graph of Subdivision

2. The graph is connected. It has only one component and there is a path between each pair of vertices.
3. There are four corner vertices of degree 2, there are eight exterior vertices of degree 3, and there are four interior vertices of degree 4.
4. The graph is not Eulerian because it has vertices of odd degree.
5. No. Since the graph is not Eulerian, there is no Euler circuit, which means that there is no route that would pass through every edge exactly once.

Identifying Euler Circuits

Solving the Chinese postman problem requires finding a shortest circuit through any graph or multigraph that visits every edge. In the case of Eulerian graphs, this means finding an Euler circuit. The method we will use is to find any circuit in the graph, then find a second circuit starting at a vertex from the first circuit that uses only edges that were not in the first circuit, then find a third circuit starting at a vertex from either of the first two circuits that uses only edges that were not in the first two circuits, and then continue this process until all edges have been used. In the end, you will be able to link all the circuits together into one large Euler circuit.

Let's find an Euler circuit in the map of the Camp Woebegone canoe race. In , we have labeled the edges of the multigraph so that the circuits can be named. In a multigraph it is necessary to name circuits using edges and vertices because there can be more than one edge between adjacent vertices.

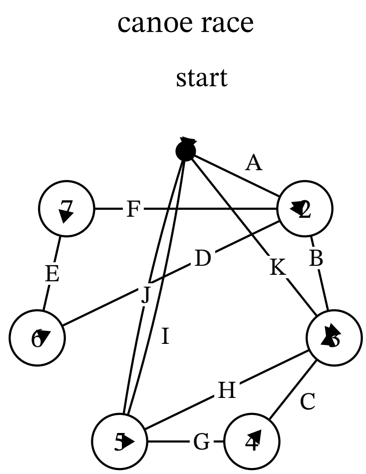


Figure 119: Multigraph of Canoe Race with Vertices and Edges Labeled

We will begin with vertex 1 because it represents the starting line in this application. In general, you can start at any vertex you want. Find any circuit beginning and ending with vertex 1. Remember, a circuit is a trail, so it doesn't pass through any edge more than once. shows one possible circuit that we could use as the first circuit, $1 \longrightarrow A \longrightarrow 2 \longrightarrow B \longrightarrow 3 \longrightarrow C \longrightarrow 4 \longrightarrow G \longrightarrow 5 \longrightarrow J \longrightarrow 1$.

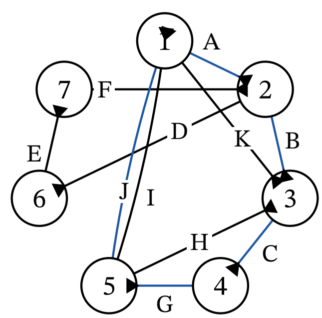


Figure 120: First Circuit

From the edges that remain, we can form two more circuits that each start at one of the vertices along the first circuit. Starting at vertex 3 we can use $3 \longrightarrow H \longrightarrow 5 \longrightarrow I \longrightarrow 1 \longrightarrow K \longrightarrow 3$ and starting at vertex 2 we can use $2 \longrightarrow D \longrightarrow 6 \longrightarrow E \longrightarrow 7 \longrightarrow F \longrightarrow 2$, as shown.

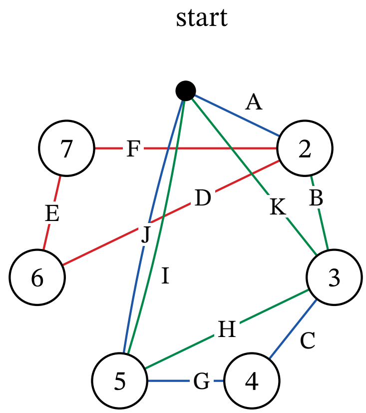


Figure 121: Second and Third Circuits

Now that each of the edges is included in one and only once circuit, we can create one large circuit by inserting the second and third circuits into the first. We will insert them at their starting vertices 2 and 3

$$1 \rightarrow A \rightarrow \boxed{2} \rightarrow B \rightarrow \boxed{3} \rightarrow C \rightarrow 4 \rightarrow G \rightarrow 5 \rightarrow J \rightarrow 1$$

becomes

$$1 \rightarrow A \rightarrow \boxed{2 \rightarrow D \rightarrow 6 \rightarrow E \rightarrow 7 \rightarrow F \rightarrow 2} \rightarrow B \rightarrow$$
$$\boxed{3 \rightarrow H \rightarrow 5 \rightarrow I \rightarrow 1 \rightarrow K \rightarrow 3} \rightarrow C \rightarrow 4 \rightarrow G \rightarrow 5 \rightarrow J \rightarrow 1$$

Finally, we can name the circuit using vertices, $1 \longrightarrow 2 \longrightarrow 6 \longrightarrow 7 \longrightarrow 2 \longrightarrow 3 \longrightarrow 5 \longrightarrow 1 \longrightarrow 3 \longrightarrow 4 \longrightarrow 5 \longrightarrow 1$, or edges, $A \longrightarrow D \longrightarrow E \longrightarrow F \longrightarrow B \longrightarrow H \longrightarrow I \longrightarrow K \longrightarrow C \longrightarrow \mathrm{G} \longrightarrow J$.

Let's review the steps we used to find this Eulerian Circuit.

Steps to Find an Euler Circuit in an Eulerian Graph

Step 1 - Find a circuit beginning and ending at any point on the graph. If the circuit crosses every edges of the graph, the circuit you found is an Euler circuit. If not, move on to step 2.

Step 2 - Beginning at a vertex on a circuit you already found, find a circuit that only includes edges that have not previously been crossed. If every edge has been crossed by one of the circuits you have found, move on to Step 3. Otherwise, repeat Step 2.

Step 3 - Now that you have found circuits that cover all of the edges in the graph, combine them into an Euler circuit. You can do this by inserting any of the circuits into another circuit with a common vertex repeatedly until there is one long circuit.

Example 4 **Finding an Euler Circuit**

Use to answer each question.

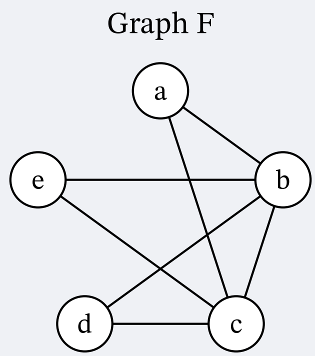


Figure 122: Graph F

1. Verify the Graph F is Eulerian.
2. Find an Euler circuit that begins and ends at vertex c.
   Solution
   1. The graph is connected because it has only one component. Also, each of the vertices in graph F has even degree as shown. So, the graph is Eulerian.

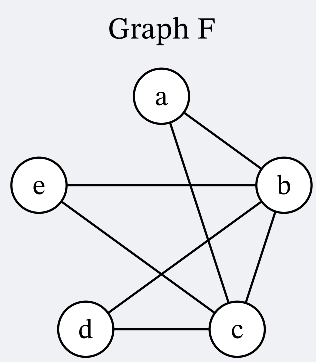


Figure 123: Degrees of Vertices in Graph F

   2. **Step 1:** Beginning at vertex c, identify circuit $c \longrightarrow e \longrightarrow b \longrightarrow c$ as shown. Since this circuit does not cover every edge in the graph, move on to Step 2.

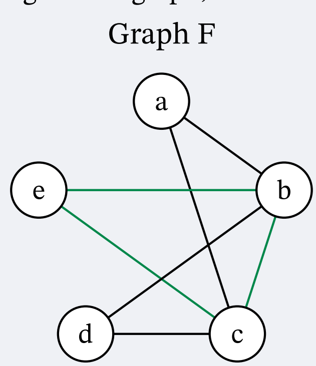


Figure 124: First Circuit in Graph F

Step 2: Find another circuit beginning at one of the vertices in the first circuit, using only edges that have not been used in the first circuit. It is possible to do this using either vertex c or vertex b. In , we have used vertex b as the starting and ending point for a second circuit, $b \longrightarrow d \longrightarrow c \longrightarrow a \longrightarrow b$. Since all edges have been crossed by one of the two circuits, move on to Step 3.

Graph F

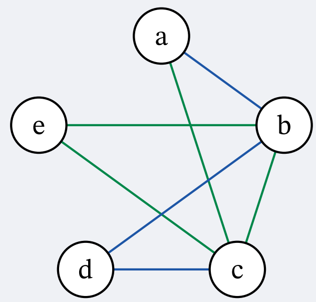


Figure 125: Second Circuit in Graph F

Step 3: Combine the two circuits into one. Replace vertex b in the first circuit with the whole second circuit that begins at vertex b.

$$c \rightarrow e \rightarrow \boxed{b} \rightarrow c$$

becomes

$$c \rightarrow e \rightarrow \boxed{b \rightarrow d \rightarrow c \rightarrow a \rightarrow b} \rightarrow c$$

An Euler circuit that begins at vertex c is $c \longrightarrow e \longrightarrow b \longrightarrow d \longrightarrow c \longrightarrow a \longrightarrow b \longrightarrow c$.

Eulerization

The Chinese postman problem doesn't only apply to Eulerian graphs. Recall the postal delivery person who needed to deliver mail to every block of every street in a subdivision. We used the map in to create the graph in .

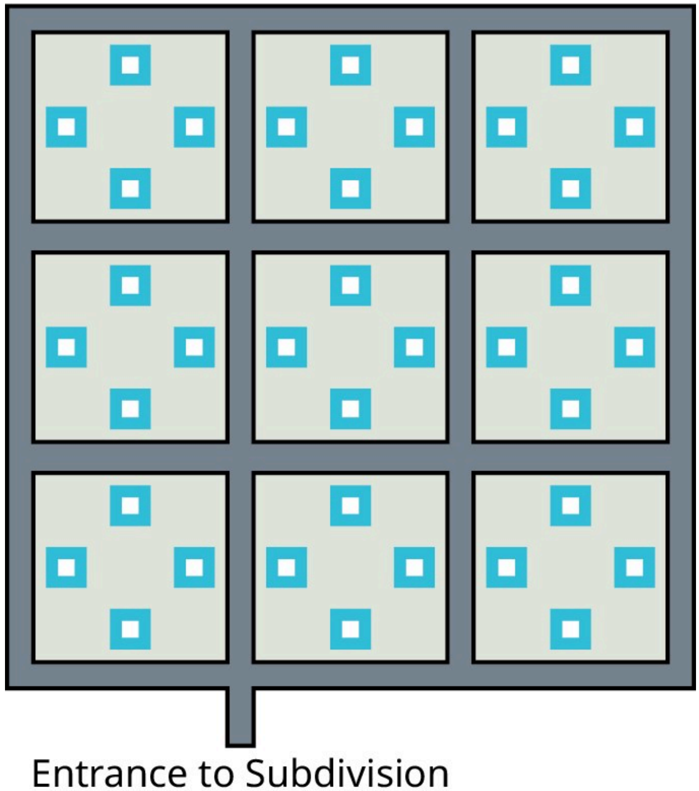


Figure 126: Map of Subdivision

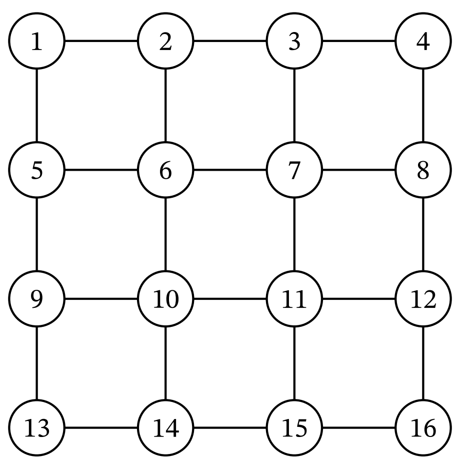


Figure 127: Graph of Subdivision

Since the graph of the subdivision has vertices of odd degree, there is no Euler circuit. This means that there is no route through the subdivision that visits every block of every street without repeating a block. What should our delivery person do? They need to repeat as few blocks as possible. The technique we will use to find a closed walk that repeats as few edges as possible is called **eulerization**. This method adds duplicate edges to a graph to create vertices of even degree so that the graph will have an Euler circuit.

In , the eight vertices of odd degree in the graph of the subdivision are circled in green. We have added duplicate edges between the pairs of vertices, which changes the degrees of the vertices to even degrees so the resulting multigraph has an Euler circuit. In other words, we have eulerized the graph.

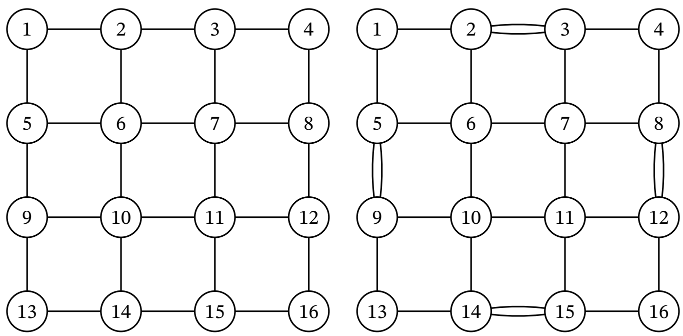


Figure 128: An Eulerized Graph

The duplicate edges in the eulerized graph correspond to blocks that our delivery person would have to travel twice. By keeping these duplicate edges to a minimum, we ensure the shortest possible route. It can be challenging to determine the fewest duplicate edges needed to eulerize a graph, but you can never do better than half the number of odd vertices. In the graph in , we have found a way to fix the eight vertices of odd degree with only four duplicate edges. Since four is half of eight, we will never do better.

CHECKPOINT

IMPORTANT! The duplicate edges that we add indicate places where the route will pass twice. An entirely new edge between two vertices that were not previously adjacent would indicate that our postal delivery person created a new road through someone's property! So, ***we can duplicate existing edges, but we cannot create new ones****.*

Example 5 **Eulerizing Graphs**

Use Graph A and multigraphs B, C, D, and E given in to answer the questions.

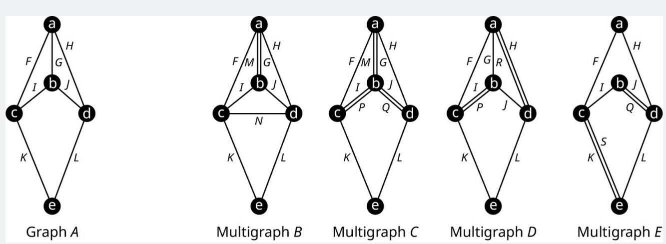


Figure 129: Graph A and Multigraphs B through E

1. Which of the multigraphs are not eulerizations of Graph A? Explain your answer.

2. Which eulerization of Graph A uses the fewest duplicate edges? How many does it use?
3. Is it possible to eulerize Graph A using fewer duplicate edges than your answer to part 2? If so, give an example. If not, explain why not.

Solution

1. Multigraph B is not an eulerization of A because the edge N between vertices c and d is not a duplicate of an existing edge. Multigraph E is not an eulerization of A because vertices b and e have odd degree.
2. Multigraph C uses 3 duplicate edges while multigraph D only uses 2. So, D uses the fewest.
3. Since there were four vertices of odd degree in Graph A, the fewest number of edges that could possibly eulerize the graph is half of four, which is two. So, it is not possible to eulerize Graph A using fewer edges.

CHECKPOINT

IMPORTANT! Since only duplicate edges can be added to eulerize a graph, it is not possible to eulerize a disconnected graph.

WORK IT OUT

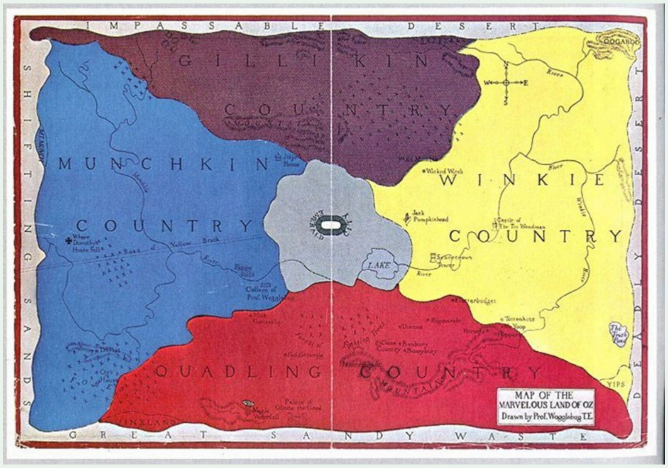


Figure 130: Map of the Magical Land of Oz

In *The Wonderful Wizard of Oz*, written by L. Frank Baum and illustrated by W. W. Denslow, the region of the Emerald City lies at the center of the magical land of Oz, with Gillikin Country to the north, Winkie Country to the east, Munchkin Country to the west, and Quadling Country to the south. Munchkin Country and Winkie Country each shares a border with Gillikin Country and Quadling Country. Let's apply graph theory to Dorothy's famous journey through Oz. Draw a graph in which each vertex is one of the regions of Oz. Then answer each question:

1. Is there an Euler trail circuit that Dorothy could follow, instead of the yellow brick road, to lead her from the land of the Munchkins, through all the regions of Oz and back, passing over each border exactly once? If not, how could the graph be Eulerized most efficiently?
2. What is the chromatic number of the graph? Find a graph coloring that demonstrates your answer and use it to draw and color a graph of Oz.

Key Terms

- connected
- component
- disconnected
- Euler circuit
- Eulerian graph
- Chinese postman problem
- Eulerization

Key Concepts

- A connected graph has only one component.
- The Euler circuit theorem states that an Euler circuit exists in every connected graph in which all vertices have even degree, but not in disconnected graphs or any graph with one or more vertices of odd degree.
- The Chinese postman problem asks how to find the shortest closed trail that visits all edges at least once.
- If an Euler circuit exists, it is always the best solution to the Chinese postman problem.
- Eulerization is the process of adding duplicate edges to a graph so that the new multigraph has an Euler circuit.
- The minimum number of duplicated edges needed to eulerize a graph is half the number of odd vertices or more.

Videos

- Connected and Disconnected Graphs in Graph Theory
- Recognizing Euler Trails and Euler Circuits
- Existence of Euler Circuits in Graph Theory

12.6 Euler Trails

Figure 131: The Pony Express mail route spanned from California to Missouri.

Learning Objectives

After completing this section, you should be able to:

1. Describe and identify Euler trails.
2. Solve applications using Euler trails theorem.
3. Identify bridges in a graph.
4. Apply Fleury's algorithm.
5. Evaluate Euler trails in real-world applications.

We used Euler circuits to help us solve problems in which we needed a route that started and ended at the same place. In many applications, it is not necessary for the route to end where it began. In other words, we may not be looking at circuits, but trails, like the old Pony Express trail that led from Sacramento, California in the west to St. Joseph, Missouri in the east, never backtracking.

Euler Trails

If we need a trail that visits every edge in a graph, this would be called an **Euler trail**. Since trails are walks that do not repeat edges, an Euler trail visits every edge exactly once.

Example 1 **Recognizing Euler Trails**

Use to determine if each series of vertices represents a trail, an Euler trail, both, or neither. Explain your reasoning.

Graph H

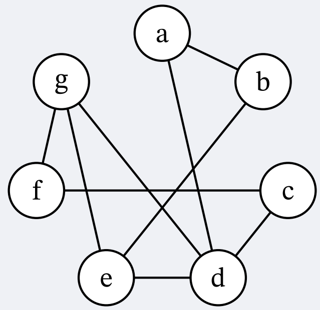


Figure 132: Graph H

1. $a \to b \to e \to g \to f \to c \to d \to e$
2. $a \to b \to e \to g \to f \to c \to d \to e \to b \to a \to d \to g$
3. $g \to d \to a \to b \to e \to d \to c \to f \to g \to e$

Solution

1. It is a trail only. It is a trail because it is a walk that doesn't cover any edges twice, but it is not an Euler trail because it didn't cover edges *ad* or *dg*.
2. It is neither. It is not a trail because it visits *ab* and *be* twice. Since it is not a trail, it cannot be an Euler trail.
3. It is both. It is a trail because it is a walk that doesn't cover any edges twice, and it is an Euler trail because it visits all the edges.

The Five Rooms Puzzle

Just as Euler determined that only graphs with vertices of even degree have Euler circuits, he also realized that the only vertices of odd degree in a graph with an Euler trail are the starting and ending vertices. For example, in , Graph H has exactly two vertices of odd degree, vertex g and vertex e. Notice the Euler trail we saw in Excercise 3 of Example 1 began at vertex g and ended at vertex e.

This is consistent with what we learned about vertices off odd degree when we were studying Euler circuits. We saw that a vertex of odd degree couldn't exist in an Euler circuit as depicted in . If it was a starting vertex, at some point we would leave the vertex and not be able to return without repeating an edge. If it was not a starting vertex, at some point we would return and not be able to leave without repeating an edge. Since the starting and ending vertices in an Euler trail are not the same, the start is a vertex we want to leave without returning, and the end is a vertex we want to return to and never leave. Those two vertices must have odd degree, but the others cannot.

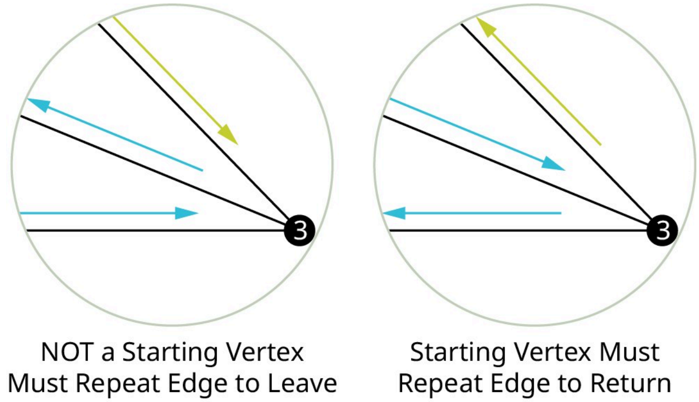


Figure 133: A Vertex of Degree 3

Let's use the Euler trail theorem to solve a puzzle so you can amaze your friends! This puzzle is called the "Five Rooms Puzzle." Suppose that you were in a house with five rooms and the exterior. There is a doorway in every shared wall between any two rooms and between any room and the exterior as shown. Could you find a route through the house that passes through each doorway exactly once?

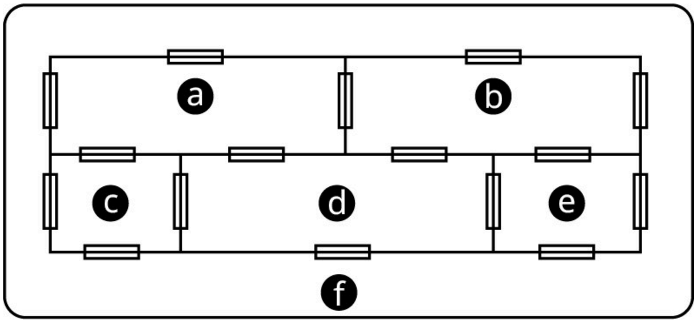


Figure 134: Five Rooms Puzzle

Let's represent the puzzle with a graph in which vertices are rooms (or the exterior) and an edge indicates a door between two rooms as shown.

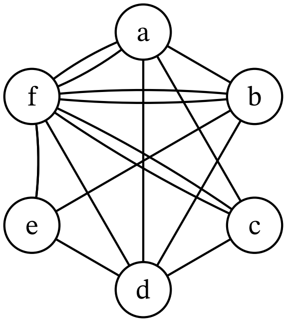


Figure 135: Graph of Five Rooms Puzzle

To pass through each doorway exactly once means that we cross every edge in the graph exactly once. Since we have not been asked to start and end at the same position, but to visit each edge exactly once, we are looking for an Euler trail. Let's check the degrees of the vertices.

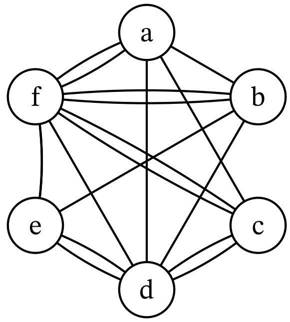


Figure 136: Degrees of Vertices in Five Rooms Puzzle

Since there are more than two vertices of odd degree as shown in , the graph of the five rooms puzzle contains no Euler path. Now you can amaze and astonish your friends!

Bridges and Local Bridges

Now that we know which graphs have Euler trails, let's work on a method to find them. The method we will use involves identifying **bridges** in our graphs. A bridge is an edge which, if removed, increases the number of components in a graph. Bridges are often referred to as cut-edges. In , there are several examples of bridges. Notice that an edge that is not part of a cycle is always a bridge, and an edge that is part of a cycle is never a bridge.

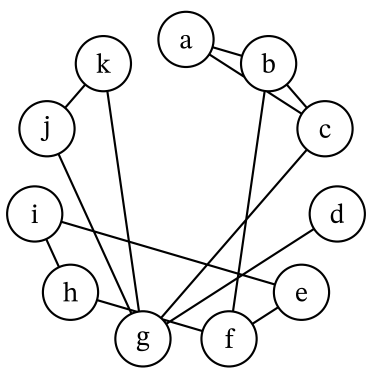


Figure 137: Graph with Bridges

Edges *bf*, *cg*, and *dg* are "bridges"

The graph in is connected, which means it has exactly one component. Each time we remove one of the bridges from the graph the number of components increases by one as shown. If we remove all three, the resulting graph in has four components.

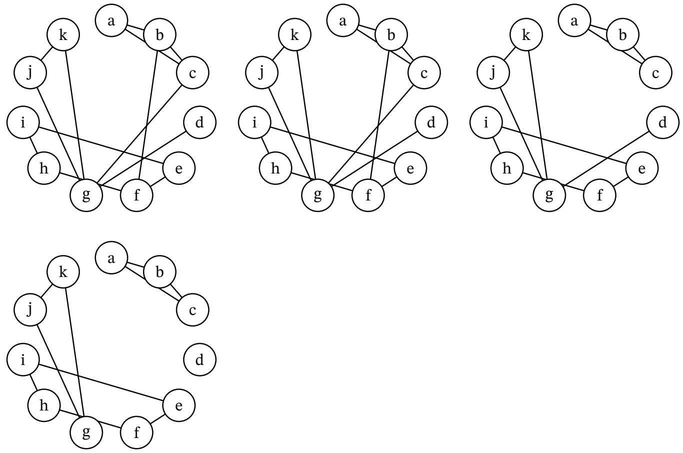


Figure 138: Removing a Bridge Increases Number of Components

In sociology, bridges are a key part of social network analysis. Sociologists study two kinds of bridges: local bridges and regular bridges. Regular bridges are defined the same in sociology as in graph theory, but they are unusual when studying a large social network because it is very unlikely a group of individuals in a large social network has only one link to the rest of the network. On the other hand, a **local bridge** occurs much more frequently. A local bridge is a friendship between two individuals who have no other friends in common. If they lose touch, there is no single individual who can pass information between them. In graph theory, a **local bridge** is an edge between two vertices, which, when removed, increases the length of the shortest path between its vertices to more than two edges. In , a local bridge between vertices b and e has been removed. As a result, the shortest path between b and e is $b \to i \to j \to k \to e$, which is four edges. On the other hand, if edge ab were removed, then there are still paths between a and b that cover only two edges, like $a \to i \to b$.

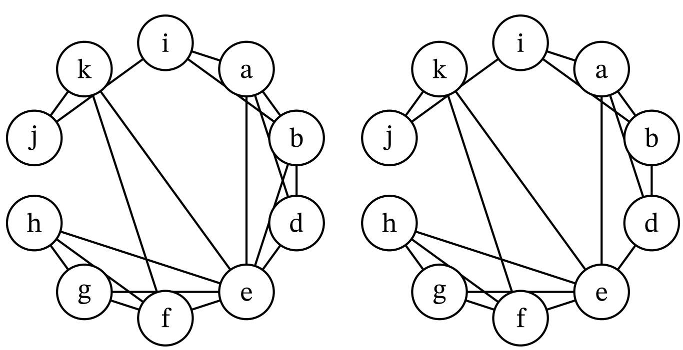


Figure 139: Removing a Local Bridge

The significance of a local bridge in sociology is that it is the shortest communication route between two groups of people. If the local bridge is removed, the flow of information from one group to another becomes more difficult. Let's say that vertex b is Brielle and vertex e is Ella. Now, Brielle is less likely to hear about things like job opportunities that Ella may know about. This is likely to impact Brielle as well as the friends of Brielle.

Example 2 **Identifying Bridges and Local Bridges**

Use the graph of a social network in to answer each question.

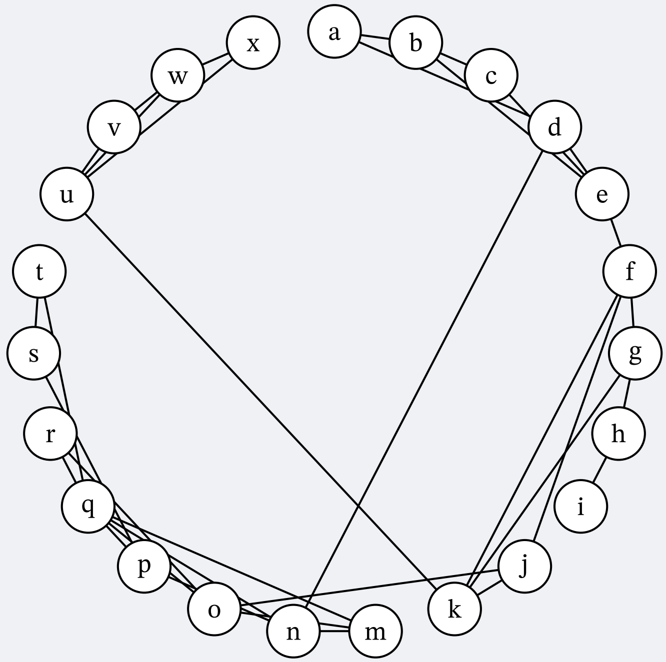


Figure 140: Graph of a Social Network

1. Identify any bridges.
2. If all bridges were removed, how many components would there be in the resulting graph?
3. Identify one local bridge.
4. For the local bridge you identified in part 3, identify the shortest path between the vertices of the local bridge if the local bridge were removed.

Solution

1. The edges *ku, gh*, and *hi* are bridges.
2. If the bridges were all removed, there would be four components in the resulting graph, {*i*}, {*h*}, {*u*, *v*, *w*, *x*}, and {*a*, *b*, *c*, *d*, *e*, *f*, g, *j*, *k*, *m*, *n*, *o*, *p*, *q*, *r*, *s*, *t*} as shown.

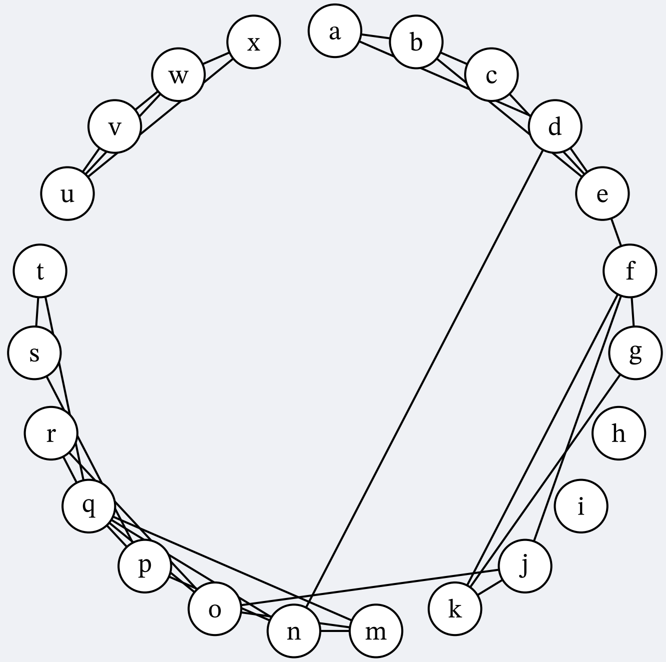


Figure 141: Graph of Social Network without Bridges

3. Three local bridges are *dn, ef*, and *qt*, among others.
4. If *dn* were removed, the shortest path between d and n would be $d \longrightarrow e \longrightarrow f \longrightarrow j \longrightarrow o \longrightarrow m \longrightarrow n$.

VIDEO

Bridges and Local Bridges in Graph Theory

Finding an Euler Trail with Fleury's Algorithm

Now that we are familiar with bridges, we can use a technique called **Fleury's algorithm**, which is a series of steps, or algorithm, used to find an Euler trail in any graph that has exactly two vertices of odd degree.

Here are the steps involved in applying Fleury's algorithm.

Step 1: Begin at either of the two vertices of odd degree.

Step 2: Remove an edge between the vertex and any adjacent vertex that is NOT a bridge, unless there is no other choice, making a note of the edge you removed. Repeat this step until all edges are removed.

Step 3: Write out the Euler trail using the sequence of vertices and edges that you found. For example, if you removed *ab, bc, cd, de*, and *ef*, in that order, then the Euler trail is $a \longrightarrow b \longrightarrow c \longrightarrow d \longrightarrow e \longrightarrow f$.

shows the steps to find an Euler trail in a graph using Fleury's algorithm.

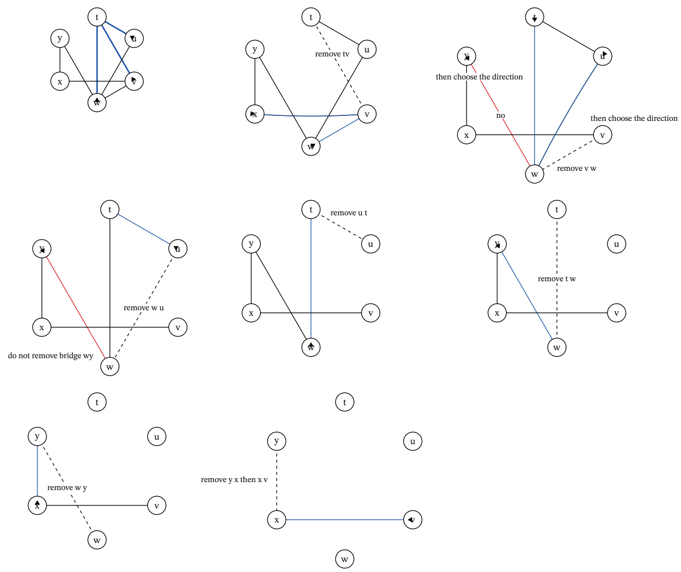


Figure 142: Using Fleury's Algorithm To Find Euler Trail

The Euler trail that was found in is $t \longrightarrow v \longrightarrow w \longrightarrow u \longrightarrow t \longrightarrow w \longrightarrow y \longrightarrow x \longrightarrow v$.

Example 3 **Finding an Euler Trail with Fleury's Algorithm**

Use Fleury's Algorithm to find an Euler trail for Graph J in .

Graph J

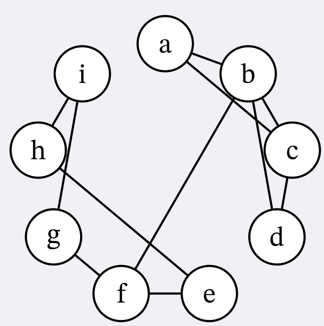


Figure 143: Graph J

Solution

Step 1: Choose one of the two vertices of odd degree, c or f, as your starting vertex. We will choose c.

Step 2: Remove edge ca, cb, or cd. None of these are cut edges so we can select any of the three. We will choose cb as shown in to be the first edge removed.

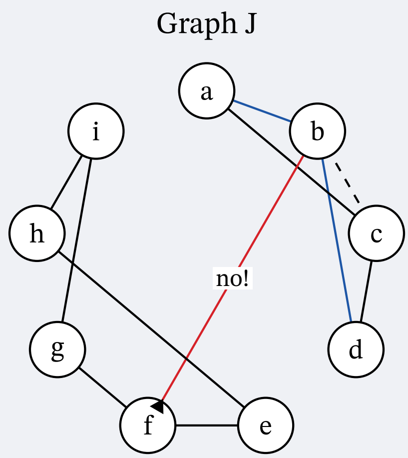


Figure 144: Graph *J* with *cb* Removed

Repeat Step 2 The next choice is to remove edge *ba, bd*, or *bf* as shown in , but *bf* is not an option since it is a bridge. We will choose *ba* as shown in to be the second edge removed.

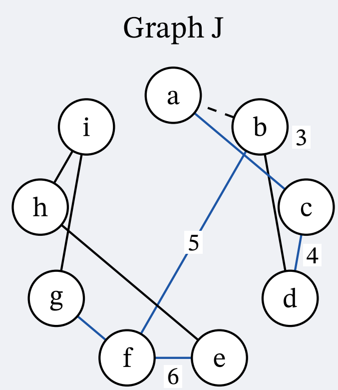


Figure 145: Graph *J* with *cb* and *ba* Removed

Repeat Step 2 for the third, fourth, fifth, sixth, and seventh edges. As shown in , until we get to the seventh edge there is only one option each time, *ac, cd, db*, and *bf* in that order. For the seventh edge, we must choose between *fe* and *fg*. Neither of these are bridges. We choose *fe*. shows that *ac, cd, db, bf,* and *fe* have been removed.

Graph J

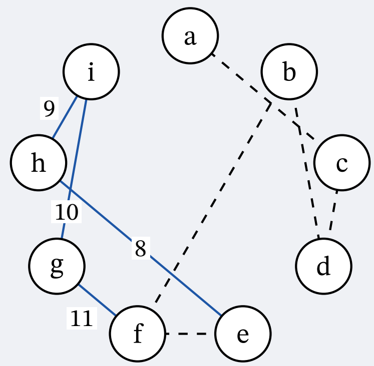


Figure 146: Graph *J* with Seven Edges Removed

Repeat Step 2 for the eight, ninth, tenth, and eleventh edges. As shown in , there is only one option for each of these edges, *eh, hi, ig*, and *gf*, in that order.

Step 3: Write out the Euler trail using the vertices in the sequence that the edges were removed. We removed *cb, ba, ac, cd, db, bf, fe, eh, hi, ig*, and *gf*, in that order. The Euler trail is $c \longrightarrow b \longrightarrow a \longrightarrow c \longrightarrow d \longrightarrow b \longrightarrow f \longrightarrow e \longrightarrow h \longrightarrow i \longrightarrow g \longrightarrow f$.

CHECKPOINT

TIP! To avoid errors, count the number of edges in your graph and make sure that your Euler trail represents that number of edges.

In the previous section, we found Euler circuits using an algorithm that involved joining circuits together into one large circuit. You can also use Fleury's algorithm to find Euler circuits in any graph with vertices of all even degree. In that case, you can start at any vertex that you would like to use.

Step 1: Begin at any vertex.

Step 2: Remove an edge between the vertex and any adjacent vertex that is NOT a bridge, unless there is no other choice, making a note of the edge you removed. Repeat this step until all edges are removed.

Step 3: Write out the Euler circuit using the sequence of vertices and edges that you found. For example, if you removed *ab, bc, cd, de*, and *ea*, in that order, then the Euler circuit is $a \longrightarrow b \longrightarrow c \longrightarrow d \longrightarrow e \longrightarrow a$.

VIDEO

Fluery's Algorithm to Find an Euler Circuit

CHECKPOINT

IMPORTANT! Since a circuit is a closed trail, every Euler circuit is also an Euler trail, but when *we say Euler trail in this chapter, we are referring to an open Euler trail that begins and ends at different vertices.*

Example 4 **Finding an Euler Circuit or Euler Trail Using Fleury's Algorithm**

Use Fleury's algorithm to find either an Euler circuit or Euler trail in Graph *G* in .

Graph G

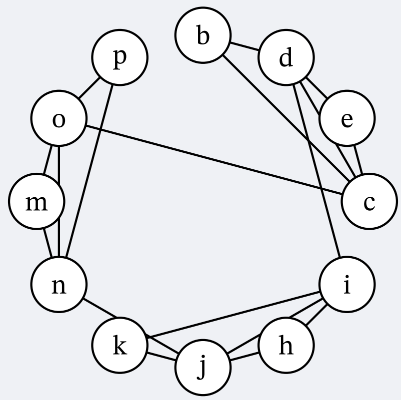


Figure 147: Graph *G*

Solution

Graph *G* has all vertices of even degree so it has an Euler circuit.

Step 1: Choose any vertex. We will choose vertex *j*.

Step 2: Remove one of the four edges that meet at vertex *j*. Since *jn* is a bridge, we must remove either *jh, ji*, or *jk*. We remove *ji* as shown.

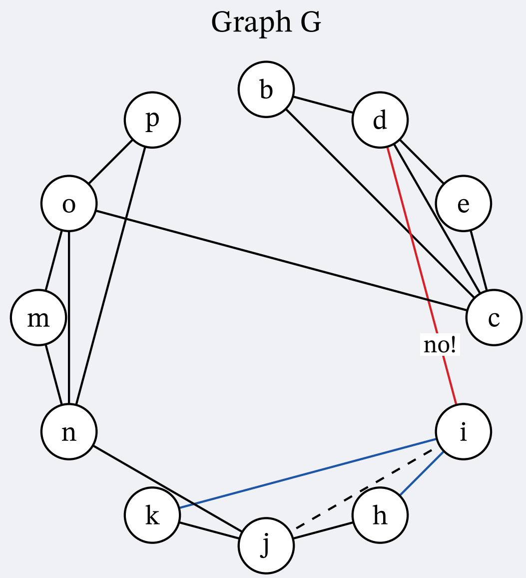


Figure 148: Graph G with 1 Edge Removed

Repeat Step 2: Since *id* is a bridge, we can remove either *ih* or *ik* next. We remove *ih*, and then the only option is to remove *hj* as shown.

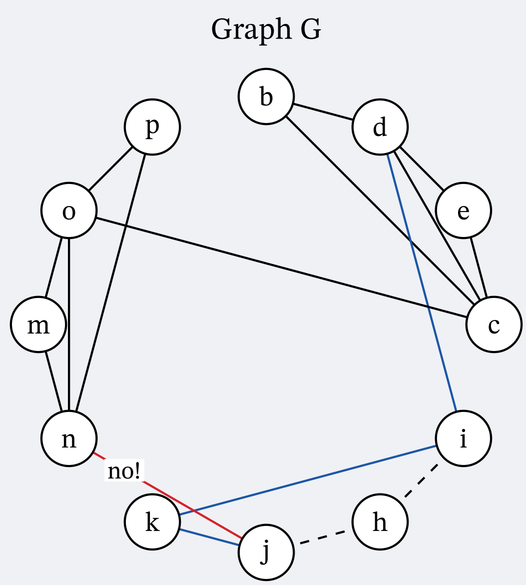


Figure 149: Graph G with 3 Edges Removed

Repeat Step 2: Since *jn* is a bridge, the next edge removed must be *jk*, and then the only option is to remove *ki* followed by *id* as shown. Even though *id* is a bridge, it can be removed because it is the only option at this point. shows Graph G with these additional edges removed.

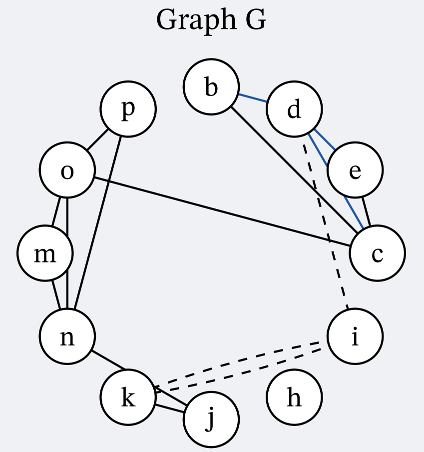


Figure 150: Graph *G* with 6 Edges Removed

Repeat Step 2: Choose any one of the edges *db, dc*, or *de*. We remove *dc* as shown.

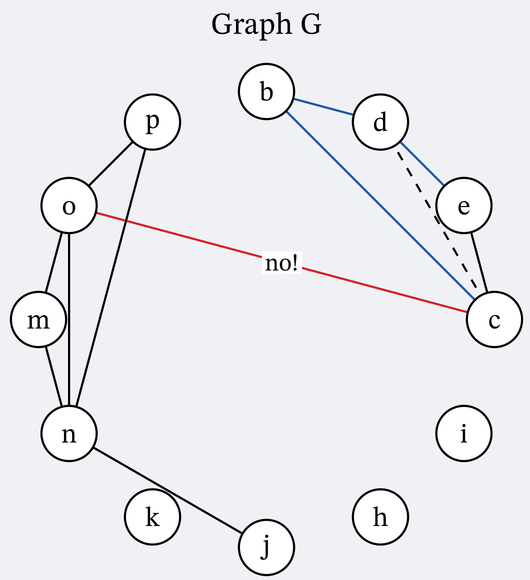


Figure 151: Graph *G* with 7 Edges Removed

Repeat Step 2: Since *co* is a bridge, choose *cb* next. We remove *cb*, then *bd*, and then *de* as shown.

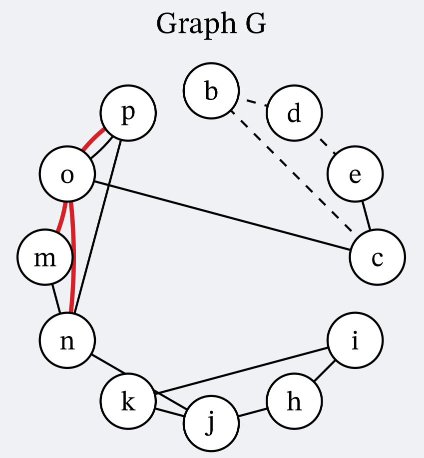


Figure 152: Graph *G* with 10 Edges Removed

Repeat Step 2: Next, remove *ec* and *co*. Then choose any of *op, pn*, or *om*. We remove on as shown.

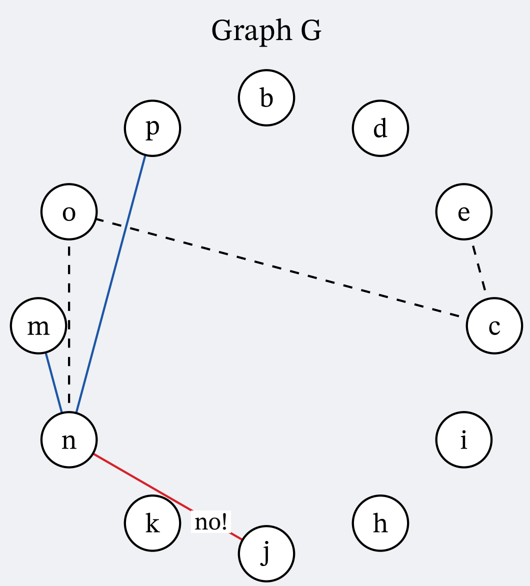


Figure 153: Graph *G* with 13 Edges Removed

Repeat Step 2: Next, remove either *nm*, *np*, or *nj*, but *nj* is a So, we remove *nm* as shown.

Graph G

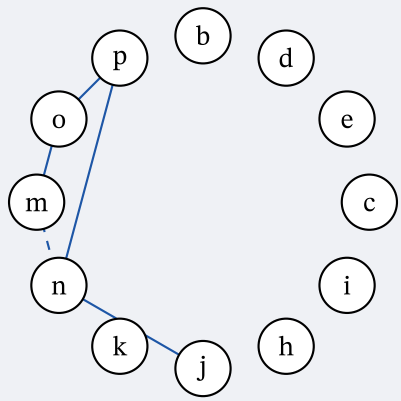


Figure 154: Graph G with 14 Edges Removed

Repeat Step 2: Next, remove *mo, op, pn*, and *nj*. And we are done!

Step 3: Notice that the algorithm brought us back to the vertex where we started, forming an Euler circuit. Write out the Euler circuit:

$j \to i \to h \to j \to k \to i \to d \to c \to b \to d \to e \to c \to o \to n \to m \to o \to p \to n \to j$

WORK IT OUT

We have discussed a lot of subtle concepts in this section. Let's make sure we are all on the same page. Work with a partner to explain why each of the following facts about bridges are true. Support your explanations with definitions and graphs.

1. When a bridge is removed from a graph, the number of components increases.
2. A bridge is never part of a circuit.
3. An edge that is part of a triangle is never a local bridge.

Key Terms

- algorithm
- Fleury's algorithm
- Euler trail
- bridge
- local bridge

Key Concepts

- An Euler trail exists whenever a graph has exactly two vertices of odd degree.
- When a bridge is removed from a graph, the number of components increases.
- A bridge is never part of a circuit.

- When a local bridge is removed from a graph, the distance between vertices increases.
- An edge that is part of a triangle is never a local bridge.

Videos

- Bridges and Local Bridges in Graph Theory
- Fluery's Algorithm to Find an Euler Circuit

12.7 Hamilton Cycles

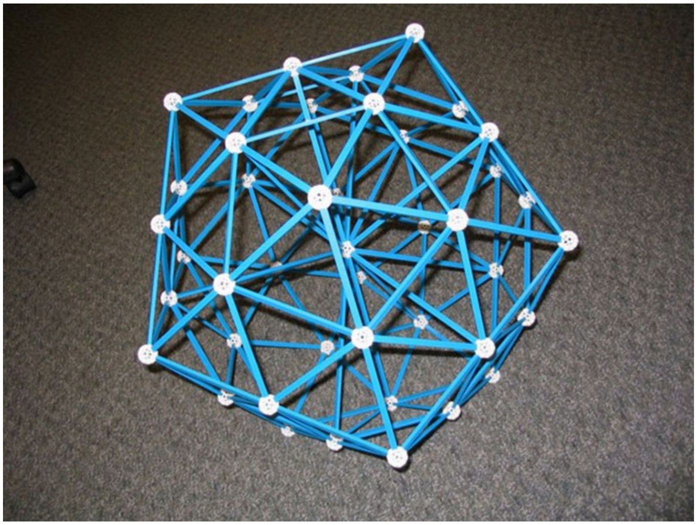


Figure 155: The symmetries of an icosahedron, with 30 edges, 20 faces, and 12 vertices, can be analyzed using graph theory.

Learning Objectives

After completing this section, you should be able to:

1. Describe and identify Hamilton cycles.
2. Compute the number of Hamilton cycles in a complete graph.
3. Apply and evaluate weighted graphs.

In Euler Circuits and Euler Trails, we looked for circuits and paths that visited each *edge* of a graph exactly once. In this section, we will look for circuits that visit each *vertex* exactly once. Like many concepts in graph theory, the idea of a circuit that visits each vertex once was inspired by games and puzzles. As early as the 9th century, Indian and Islamic intellectuals wondered whether it was possible for a knight to visit every space on a chess board of a given size, which is equivalent to visiting every vertex of a graph that represents the chess board.

In 1857, a mathematician named William Rowan Hamilton invented a puzzle in which players were asked to find a route along the edges of a dodecahedron, which visited every vertex exactly once. Let's explore how graph theory provides insight into these games as well as practical applications such as the Traveling Salesperson Problem.

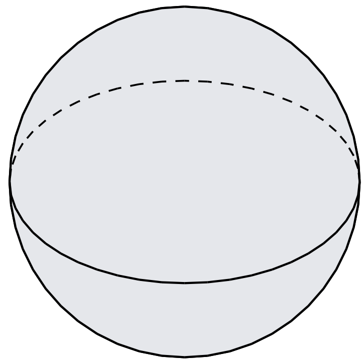


Figure 156: A Dodecahedron

Hamilton's Puzzle

Before we look at the solution to Hamilton's puzzle, let's review some vocabulary we used in . It will be helpful to remember that directed cycle is a type of circuit that doesn't repeat any edges or vertices.

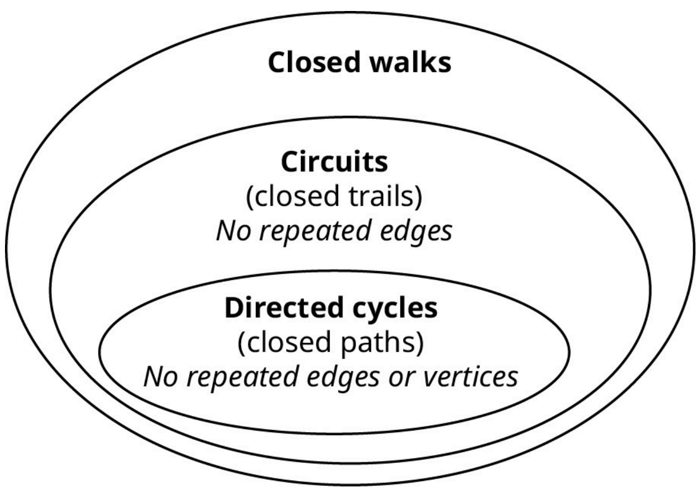


Figure 157: Closed Walks, Circuits, and Directed Cycles

The goal of Hamilton's puzzle was to find a route along the edges of the dodecahedron, which visits each vertex exactly once. A dodecahedron is a three-dimensional space figure with faces that are all pentagons as we saw in .

Since it is easier to visualize two dimensions rather than three, we will flatten out the dodecahedron and look at the edges and vertices on a flat surface. Graph A in shows a two-dimensional graph of the edges and vertices, and Graph B shows an untangled version of Graph A in which no edges are crossing. Graph B in is very similar to the design of the game board that Hamilton invented for his puzzle.

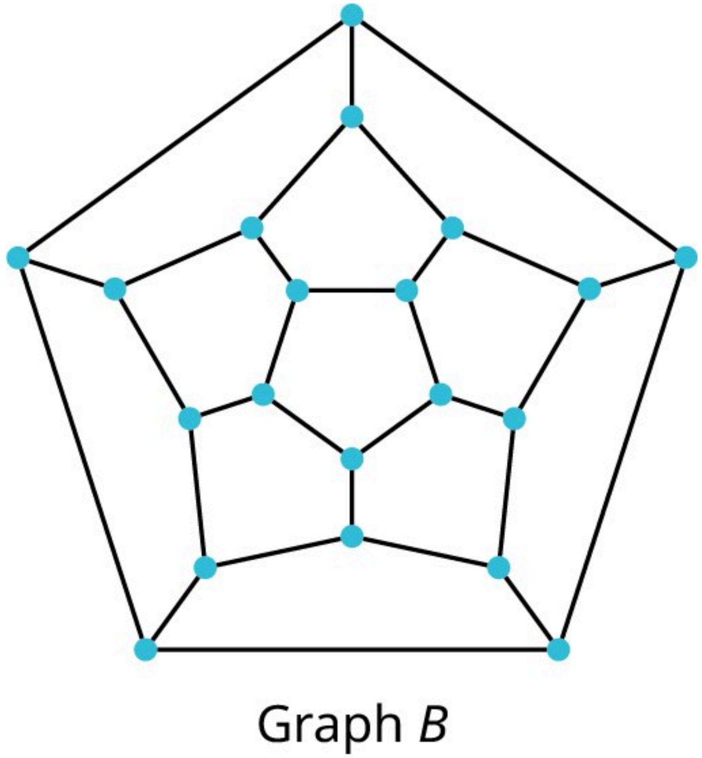


Figure 158: Graph of Edges and Vertices of Dodecahedron

We can see that this is a planar graph because it can be "untangled." In order to solve Hamilton's puzzle, we need to find a circuit that visits every vertex once. A solution is shown.

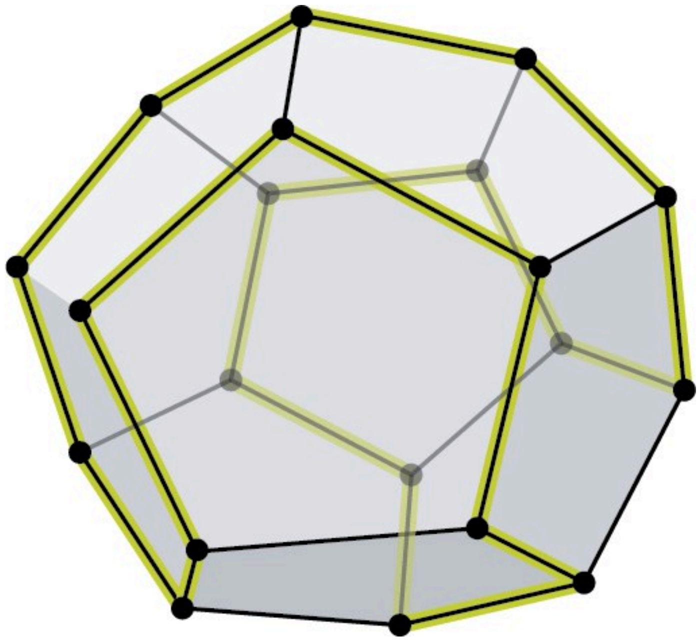


Figure 159: A Solution to Hamilton's Puzzle

A circuit that doesn't repeat any vertices, like the one in , is called a directed cycle. So, we can most accurately say that Hamilton's puzzle asks us to find a directed cycle that visits every vertex in a graph exactly once. Because Hamilton created and solved this puzzle, these special circuits were named **Hamilton cycles**, or **Hamilton circuits**.

WHO KNEW?

Icosian Game

When Hamilton invented his puzzle in the 19th century, it was originally called the Icosian game. This was a reference to an icosahedron, not a dodecahedron. Why? Hamilton was interested in the symmetries of icosahedrons, specifically regular icosahedrons, which have faces that are all equilateral triangles. It turns out that this space figure has a surprising correspondence with the regular dodecahedron, a space figure with faces that are all regular pentagons. This pair of space figures have the same number of edges, while the number of faces and vertices are swapped. The icosahedron has 30 edges, 20 faces, and 12 vertices, while the dodecahedron has 30 edges, 12 faces, and 20 vertices. By relating the vertices of the dodecahedron to the faces of the icosahedron, Hamilton was able to make the mathematical connections necessary to use graph theory and dodecahedrons to make discoveries about the symmetries of icosahedrons.

Hamilton Cycles vs. Euler Circuits

Let's practice naming and identifying Hamilton cycles, as well as distinguishing them from Euler circuits. It is important to remember that Euler circuits visit all edges without repetition, while

Hamilton cycles visit all vertices without repetition. Hamilton cycles are named by their vertices just like all circuits. An example is given in .

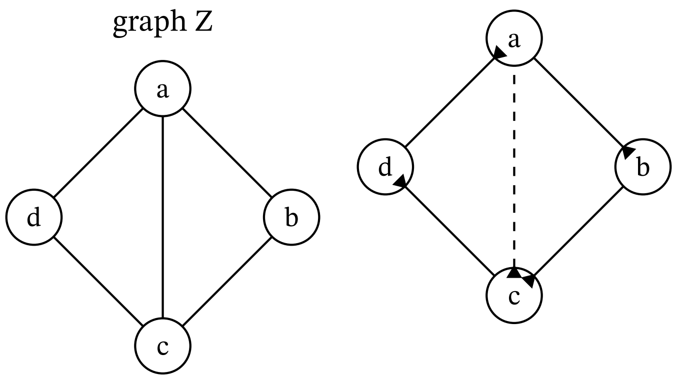


Figure 160: Hamilton Cycle in Graph Z

Notice that the Hamilton cycle $a \longrightarrow b \longrightarrow c \longrightarrow d$ for Graph Z in is NOT an Euler circuit, because it does not visit edge ac. Some Hamilton cycles are also Euler circuits while some are not, and some Euler circuits are Hamilton cycles while some are not.

CHECKPOINT

TIP! Sometimes students confuse Euler circuits with Hamilton cycles. To help you remember, think of the E in Euler as standing for Edge.

Example 1 **Differentiating between Hamilton Cycles or Euler Circuits**

Use to determine whether the given circuit is a Hamilton cycle, an Euler circuit, both, or neither.

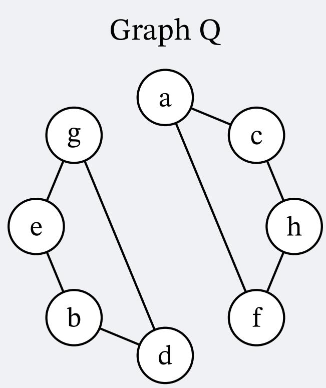


Figure 161: Graph Q

1. $a \rightarrow b \rightarrow c \rightarrow e \rightarrow h \rightarrow g \rightarrow f \rightarrow d \rightarrow a$
2. $g \rightarrow e \rightarrow h \rightarrow g \rightarrow f \rightarrow d \rightarrow a \rightarrow b \rightarrow d \rightarrow g$
3. $a \rightarrow b \rightarrow c \rightarrow e \rightarrow h \rightarrow g \rightarrow f \rightarrow d \rightarrow b \rightarrow e \rightarrow g \rightarrow d \rightarrow a$

Solution

1. This circuit is a Hamilton cycle only. It visits each vertex exactly once, so, it is a Hamilton cycle. It is not an Euler circuit because it doesn't visit all of the edges.
2. This circuit is neither Hamilton cycle nor an Euler circuit. It doesn't visit vertex *c*, so, it is not a Hamilton cycle. It also doesn't visit edges *be* and *ce*, so, it is not an Euler circuit.
3. This circuit is an Euler circuit only. It visits several vertices more than once; so, it is not a Hamilton cycle. It visits every edge exactly once, so, it is an Euler circuit.

CHECKPOINT

TIP! Euler circuits never skip a vertex, so, they only fail to be Hamilton cycles when they visit a vertex more than once. Hamilton cycles never visit any edges twice, so, they only fail to be Euler circuits when they skip edges.

Notice that the graph is a cycle. A cycle will always be Eulerian because all vertices are degree 2. Moreover, any circuit in the graph will always be both an Euler circuit and a Hamilton cycle. It is not always as easy to determine if a graph has a Hamilton cycle as it is to see that it has an Euler circuit, but there is a large group of graphs that we know will always have Hamilton cycles, the complete graphs. Since all vertices in a complete graph are adjacent, we can always find a directed cycle that visits all the vertices. For example, look at the directed six-cycle, $n \rightarrow o \rightarrow p \rightarrow q \rightarrow r \rightarrow s$, in the complete graph with six vertices in .

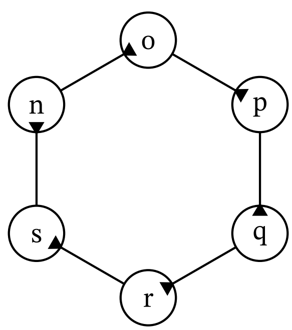


Figure 162: Directed Cycle in Complete Graph

That is not the only directed six-cycle in the graph though. We could find another just be reversing the direction, and we could find even more by using different edges. So, how many Hamilton cycles are in a complete graph with *n* vertices? Before we tackle this problem, let's look at a shorthand notation that we use in mathematics which will be helpful to us.

Factorials

In many areas of mathematics, we must calculate products like $7 \cdot 6 \cdot 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1$ or $11 \cdot 10 \cdot 9 \cdot 8 \cdot 7 \cdot 6 \cdot 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1$, products that involve multiplying all the counting numbers from a particular number down to 1. Imagine that the product happened to be all the numbers from 100 down to 1. That's a lot of writing! Instead of writing all of that out, mathematicians came up with a shorthand notation. For example, instead of $7 \cdot 6 \cdot 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1$, we write $7!$, which is read "7 **factorial**." In other words, the product of all the counting numbers from n down to 1 is called n factorial and it is written $n!$

Example 2 **Evaluating Factorials**

Evaluate $n!$ and $(n-1)!$ for $n = 4$.

Solution

$n! = 4! = 4 \cdot 3 \cdot 2 \cdot 1 = 24$ and $(n-1)! = (4-1)! = 3! = 3 \cdot 2 \cdot 1 = 6$

A common use for factorials is counting the number of ways to arrange objects. Suppose that there were three students, Aryana, Byron, and Carlos, who wanted to line up in a row. How many arrangements are possible? There are six possibilities: ABC, ACB, BAC, BCA, CAB, or CBA. Notice that there were three students being arranged, and the number of possible arrangements is three.

FORMULA

The number of ways to arrange n distinct objects is $n!$.

Example 3 **Counting Arrangements of Letters**

Find the number of ways to arrange the letters a, b, c, and d.

Solution

$4! = 4 \cdot 3 \cdot 2 \cdot 1 = 24$

Counting Hamilton Cycles in Complete Graphs

Now, let's get back to answering the question of how many Hamilton cycles are in a complete graph. In , we have drawn all the four cycles in a complete graph with four vertices. Remember, cycles can be named starting with any vertex in the cycle, but we will name them starting with vertex a.

Complete Graph	Cycle	Cycle	Cycle
Cycle Name Clockwise	*(a, b, c, d)*	*(a, b, d, c)*	*(a, c, b, d)*
Cycle Name Counterclockwise	*(a, d, c, b)*	*(a, c, d, b)*	*(a, d, b, c)*

shows that there are three unique four-cycles in a complete graph with four vertices. Notice that there were two ways to name each cycle, one reading the vertices in a clockwise direction and one reading the vertices in a counterclockwise direction. This is important to us because we are interested in Hamilton cycles, which are *directed* cycles. Although the cycles *(a, b, c, d)* and *(a, d, c, b)* are the same cycle, the directed cycles, $a \rightarrow b \rightarrow c \rightarrow d \rightarrow a$ and $a \rightarrow d \rightarrow c \rightarrow b \rightarrow a$, which travel the same route in reverse order are considered different directed cycles, as shown.

Complete Graph	Cycle	Cycle	Cycle
Clockwise Hamilton Cycle	$a \to b \to c \to d \to a$	$a \to b \to d \to c \to a$	$a \to c \to b \to d \to a$
Counter-clockwise Hamilton Cycle	$a \to d \to c \to b \to a$	$a \to c \to d \to b \to a$	$a \to d \to b \to c \to a$

The six directed four-cycles in are the only distinct Hamilton cycles in a complete graph with four vertices. Six is also the number of ways to arrange the three letters b, c, and d. (Do you see why?) The number of ways to arrange three letters is $3! = 3 \cdot 2 \cdot 1 = 6$. Similarly, the number of Hamilton cycles in a graph with five vertices is the number of ways to arrange four letters, which is $4! = 4 \cdot 3 \cdot 2 \cdot 1 = 24$. In general, to find the number of Hamilton cycles in a graph, we take one less than the number of vertices and find its factorial.

FORMULA

The number of distinct Hamilton cycles in a complete graph with n vertices is $(n-1)!$

Example 4 **Counting Hamilton Cycles in a Complete Graph**

How many Hamilton cycles are in the complete graph in ?

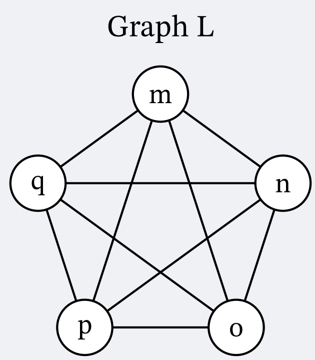


Figure 163: Complete Graph L

Solution

There are five vertices in the graph. Using $n = 5$, we have$(n-1)! = (5-1)! = 4! = 4 \cdot 3 \cdot 2 \cdot 1 = 24$ Hamilton cycles.

Weighted Graphs

Suppose that an officer in the U.S. Air Force who is stationed at Vandenberg Air Force base must drive to visit three other California Air Force bases before returning to Vandenberg. The officer needs to visit each base once. The vertices in the graph in represent the four U.S. Air Force bases, Vandenberg, Edwards, Los Angeles, and Beale. The edges are labeled to with the driving distance between each pair of cities.

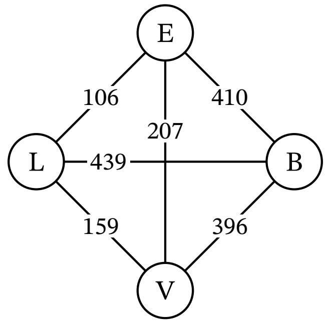


Figure 164: Graph of Four California Air Force Bases

The graph in is called a **weighted graph**, because each edge has been assigned a value or weight. The weights can represent quantities such as time, distance, money, or any quantity associated with the adjacent vertices joined by the edges. The **total weight** of any walk, trail, or path is the sum of the weights of the edges it visits.

Notice that the officer's trip can be represented as a Hamilton cycle, because each of the four vertices in the graph is visited exactly once.

Example 5 **Finding Hamilton Cycles of Lowest Weight**

Use and the given Hamilton cycles to answer the following questions.

$V \to L \to E \to B \to V$

$V \to L \to B \to E \to V$

$V \to E \to L \to B \to V$

$V \to B \to E \to L \to V$

1. Which of the Hamilton cycles (directed cycles) lie on the same cycle (undirected cycle) in the graph?
2. Find the total weight of each cycle.
3. Of the four, which of the Hamilton cycles describes the shortest trip for the officer? Describe the route.

Solution

1. $V \to L \to E \to B \to V$ and $V \to B \to E \to L \to V$ follow the same edges in reverse order.
2. Any Hamilton cycles that lie on the same cycle will have the same edges and the same total weight. $V \to L \to E \to B \to V$ and $V \to B \to E \to L \to V$ each have total weight $159 + 106 + 410 + 396 = 1071$.

$V \to L \to B \to E \to V$ has a total weight $159 + 439 + 410 + 207 = 1215$.

$V \to E \to L \to B \to V$ has a total weight $396 + 439 + 106 + 207 = 1148$.

1. Hamilton cycles $V \longrightarrow L \longrightarrow E \longrightarrow B \longrightarrow V$ and $V \longrightarrow B \longrightarrow E \longrightarrow L \longrightarrow V$ each have the lowest total weight. The officer would take the route from Vandenberg, to Los Angeles, to Edwards, to Beale, and back to Vandenberg, or reverse that route.

Key Terms

- Hamilton cycle, or Hamilton circuit
- n factorial
- weighted graph
- total weight

Key Concepts

- A Hamilton cycle is a directed cycle, or circuit, that visits each vertex exactly once.
- Some Hamilton cycles are also Euler circuits, but some are not.
- Hamilton cycles that follow the same undirected cycle in the same direction are considered the same cycle even if they begin at a different vertex.
- The number of unique Hamilton cycles in a complete graph with n vertices is the same as the number of ways to arrange $n-1$ distinct objects.
- Weighted graphs have a value assigned to each edge, which can represent distance, time, money and other quantities.

Formulas

The number of ways to arrange n distinct objects is $n!$.

The number of distinct Hamilton cycles in a complete graph with n vertices is $(n-1)!$.

12.8 Hamilton Paths

Figure 165: A school bus picks up children along a planned route.

Learning Objectives

After completing this section, you should be able to:

1. Describe and identify Hamilton paths.
2. Evaluate Hamilton paths in real-world applications.
3. Distinguish between Hamilton paths and Euler trails.

In the United States, school buses carry 25 million children between school and home every day. The total distance they travel is around 6 billion kilometers per year. In the city of Boston, Massachusetts, the 2016 budget for running those buses was $120 million dollars. In 2017, the city held a competition to find ways to cut costs and the Quantum Team from the MIT Operations Research Center came to the rescue, using a computer algorithm to identify the most efficient and least costly routes, which saved the city of Boston $5 million each year and even reduced daily CO_2 emissions by 9,000 kilograms! (*This* U.S. *city put an algorithm in charge of its school bus routes and saved $5 million*, Sean Fleming, World Economic Forum)

The problem the Quantum Team tackled involves graph theory. Imagine a graph in which vertices are the bus depot, the school, and the bus stops along a particular route. The bus must start at the depot, visit every stop exactly once, and end at the school. The route is a special kind of path that visits every vertex exactly once. Can you guess what those paths are called?

Hamilton Paths

Just as circuits that visit each vertex in a graph exactly once are called Hamilton cycles (or Hamilton circuits), paths that visit each vertex on a graph exactly once are called **Hamilton paths**. As we explore Hamilton paths, you might find it helpful to refresh your memory about the relationships between walks, trails, and paths by looking at . We know that paths are walks that don't repeat any vertices or edges. So, a Hamilton path visits every vertex without repeating any vertices or edges. shows a path from vertex A to vertex E and a Hamilton path from vertex A to vertex E.

(no graphs in spec)

Figure 166: Walks, Trails, and Paths

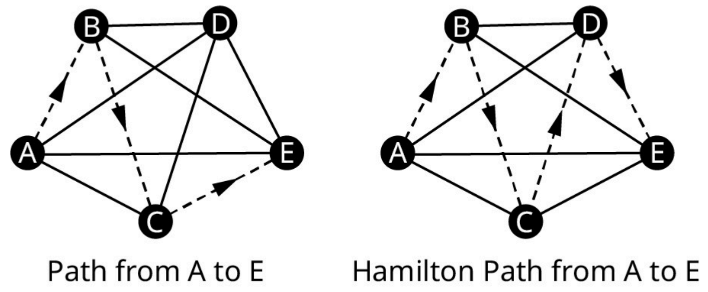


Figure 167: Path or Hamilton Path?

Example 1 **Identifying Hamilton Paths**

Which of the following sequences of vertices is a Hamilton path for Graph Q in ?

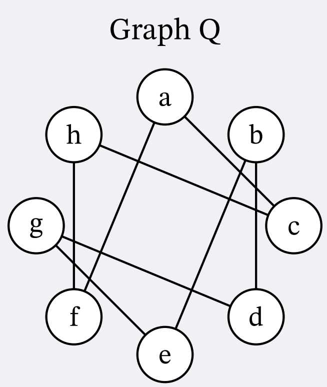


Figure 168: Graph Q

1. $a \to d \to b \to c \to e \to g \to f$
2. $c \to b \to e \to h \to g \to f \to d \to a$
3. $h \to e \to g \to d \to b \to e \to g \to f \to d \to a \to b \to c$

Solution

Sequence 1 is a path, because it is a walk that doesn't repeat any vertices or edges, but not a Hamilton path because it skips vertex h. Sequence 2 is a path that visits each vertex exactly once; so, it is a Hamilton path. Sequence 3 is a walk, but it is not a path because it visits vertices g, e, and b each more than once; so, it cannot be a Hamilton path. So, we can see that only sequence 2 is a Hamilton path.

CHECKPOINT

TIP! Since a Hamilton path visits each vertex exactly once, it must have the same number of vertices listed as appear in the graph.

Finding Hamilton Paths

Suppose you were visiting an aquarium with some friends. The map of the aquarium is given in . The letters represent the exhibits.

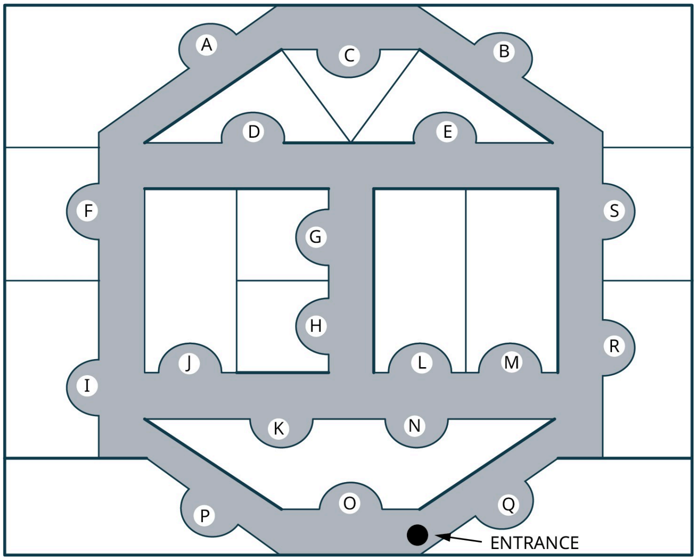


Figure 169: Map of Aquarium Exhibits

shows a graph of the aquarium in which each vertex represents an exhibit and each edge is a route between the pair of exhibits that doesn't bypass another exhibit.

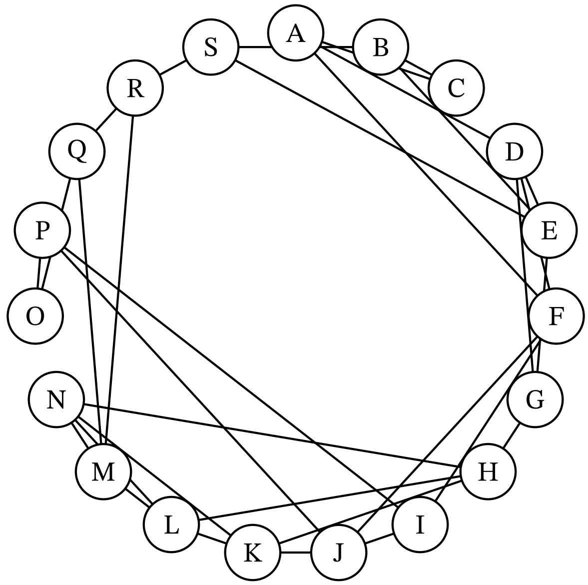


Figure 170: Graph of Aquarium

Let's see if we can plan a tour of the exhibits that visits each exhibit exactly once, beginning at exhibit *O* and ending at exhibit *C*. Suppose that, after exhibit *O*, we plan to visit exhibit *Q* and then exhibit *M*. After *M*, should we plan to visit *N*, *L*, or *R*? Take a look at . If *R* is not chosen next, that will cause a problem later on. Do you see what it is?

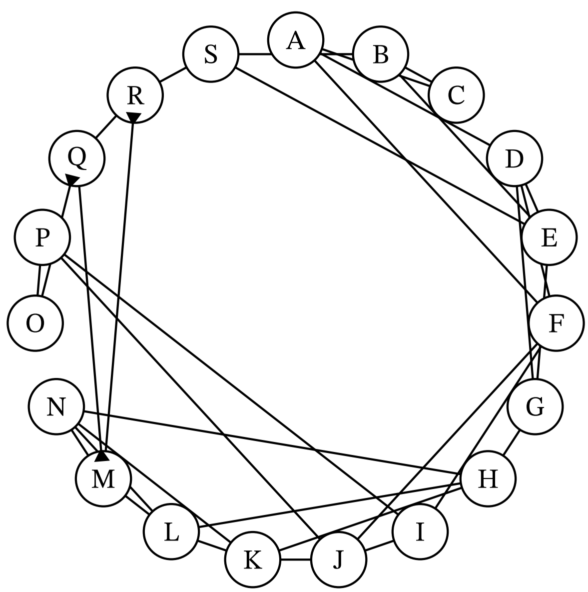


Figure 171: Choosing Vertex *L*, *N*, or *R*

If *L* or *N* is chosen next, the only way to get to *R* later will be to go from *S* to *R*, and then we will not be able to continue without repeating a vertex. So, we will pick *R* next, and then the only option is *S*. After *S* we have another choice to make. As shown in , the next choice is between *B* and *E*. Keeping in mind that the goal is to end at *C*, which would be the better choice?

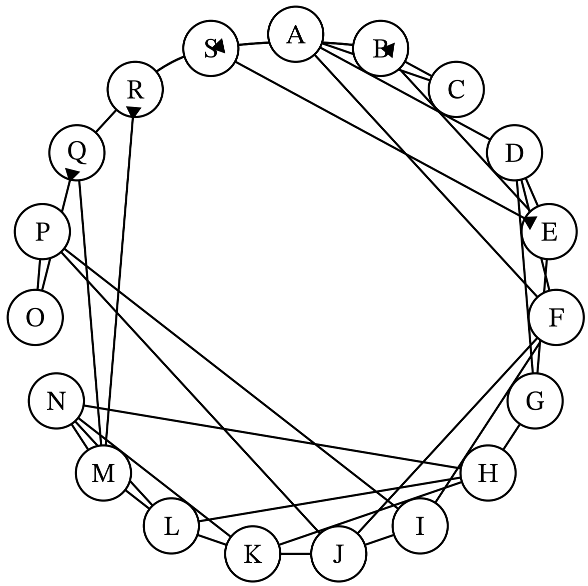


Figure 172: Choosing Vertex *B* or *E*

If you said vertex *B*, you are right! Otherwise, we will not be able to visit *B* later. After *B*, the only option is *E*. Then we can choose either *D* or *G*. Either will work fine. Let's choose *G* as shown. After *G*, you must visit *H*, but should you visit *K* or *L* after that?

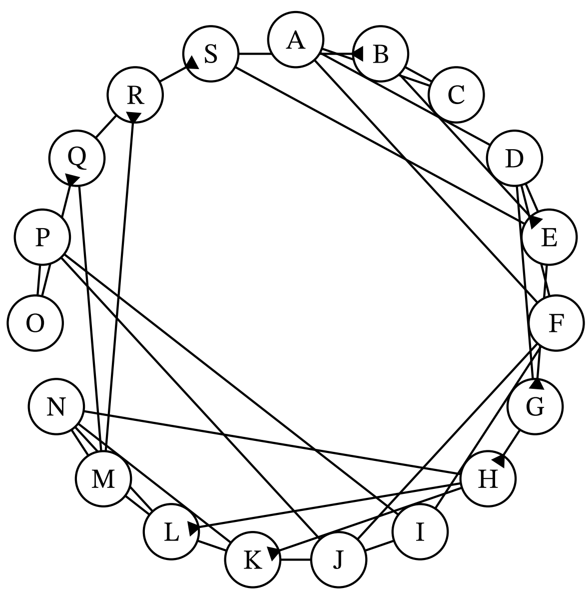


Figure 173: Choose Vertex *L* or *K*

If you said to go to vertex *L* next, you are right! Otherwise, it will be impossible to visit *N* without repeating a vertex. So, next is *L*, then *N*, then *K*, and then at *J* you have another decision to make we can see in . Should you choose *F*, *I*, or *P* next?

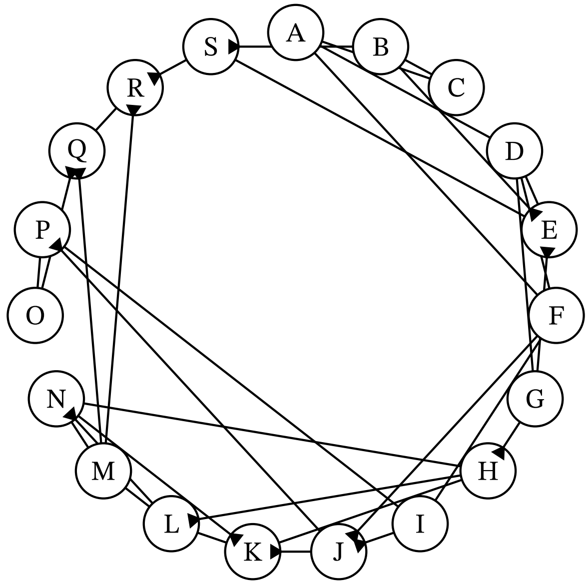


Figure 174: Choose Vertex *F*, *I*, or *P*

If you said *P*, you are right! If you choose either of the other two vertices, you will not be able to visit *P* later without passing through another vertex twice. Once *P* is chosen, vertex *I* must be next followed by *F*. Then you have to choose between *A* and *D* as shown.

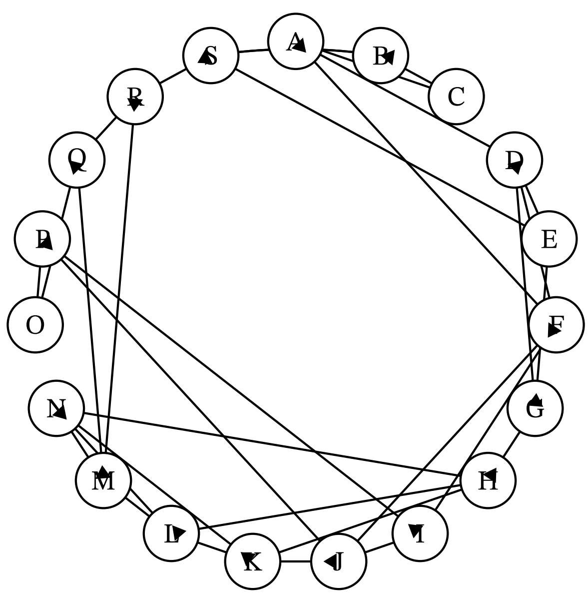


Figure 175: Choose Vertex *A* or *D*

In this case, we must go to D then to A so that we can visit C without backtracking. The complete Hamilton path is shown.

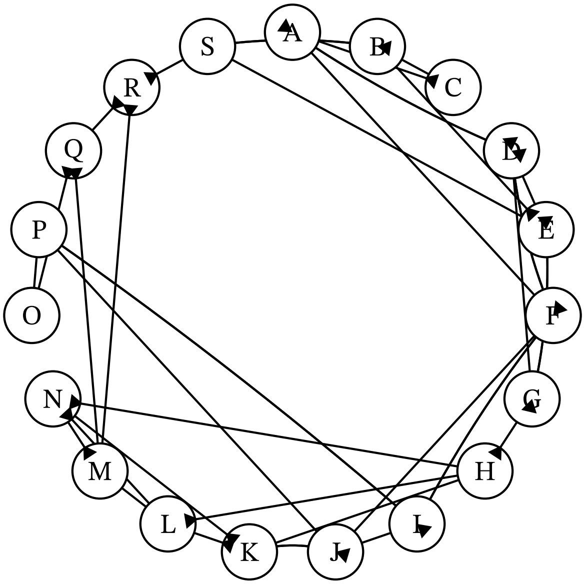


Figure 176: Complete Hamilton Path from O to C

So, one Hamilton path that begins at O and ends at C is $O \to Q \to M \to R \to S \to B \to E \to G \to H \to L \to N \to K \to J \to P \to I \to F \to D \to A \to C$.

There is no set sequence of steps that can be used to find a Hamilton path if it exists, but it does help to keep in mind where we are headed and avoid choices that will make returning to a particular vertex impossible without repeating vertices. Let's practice finding Hamilton paths.

Example 2 **Finding a Hamilton Path**

Use to find a Hamilton path between vertices C and D.

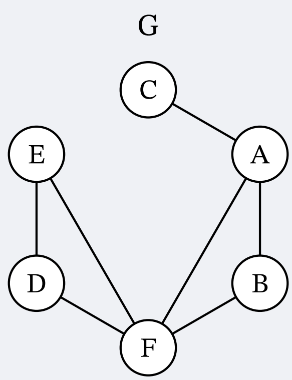


Figure 177: Graph G

Solution

If we start at vertex C, A must be next. Then we must choose between B and F. If we choose

F, we will have to backtrack to get to include B so, we must choose B. Once we choose B, we must choose F next. After F, we choose E, because we want to end at D. So, a Hamilton path between C and D is $C \rightarrow A \rightarrow B \rightarrow F \rightarrow E \rightarrow D$.

Existence of a Hamilton Path

It turns out that there is no Hamilton path between vertices A and E in Graph G in . To understand why, let's imagine there is a red apple tree on one side of a bridge and a green apple tree on the other side of the bridge. Now suppose someone asked you to pick up all the fallen apples under each tree without crossing the bridge more than once, and making sure that the first apple you pick up and the last apple you pick up are both red. You would say, that is impossible! To have the first and last apple be red would either require leaving the green apples on the ground or crossing the bridge twice.

Let's see how this relates to finding a Hamilton path between A and E in Graph G. The edge AC is a bridge because, if it were removed, the graph would become disconnected with two components, the component $\{C\}$ and the component $\{A, B, D, E, F\}$. So, we can think of the vertices A, B, D, E, and F as the red apples, vertex C as the green apple, and the edge AC is the bridge between them as in .

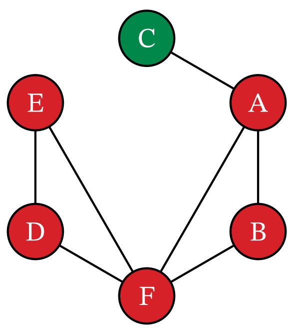


Figure 178: Bridge between Red and Green Apples

The creation of a Hamilton path requires a visit to each vertex, just as picking up all the apples requires a visit to each apple. A and E are both red apples; so, a path from A to E would both start and end at a red apple, just as you were asked to do. And you wouldn't be able to cross the bridge twice because that would mean visiting A twice, which is not allowed in a Hamilton path. So, it is impossible to find a Hamilton path from A to E just as it was impossible to pick up all the apples without crossing the bridge more than once. By the same reasoning, if a graph has a bridge, there will never be a Hamilton path that begins and ends on the same side of that bridge, meaning beginning and ending at vertices that would be in the same component if the bridge were removed from the graph.

Example 3 **Finding a Hamilton Path If One Exists**

Find a Hamilton path from vertex s to vertex v for each graph in or indicate that there is none.

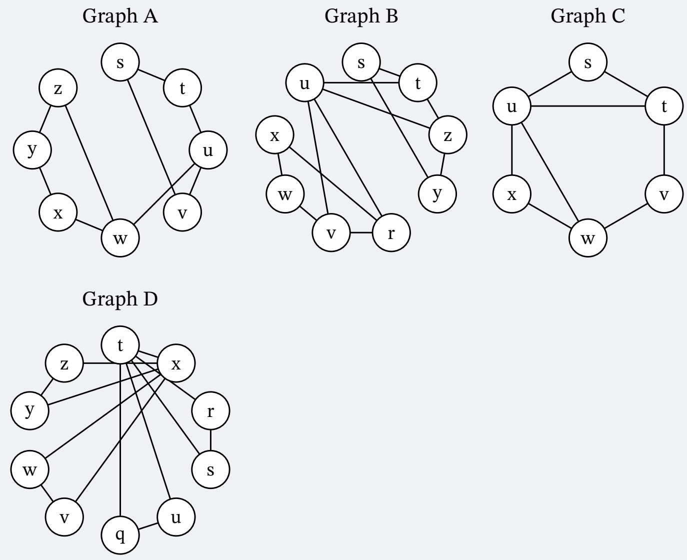


Figure 179: Graphs *A*, *B*, *C*, and *D*

Solution

Graph *A*: Edge *uw* is a bridge connecting component {*s*, *t*, *u*, *v*} to the component {*w*, *x*, *y*, *z*}. There is no Hamilton path from vertex *s* to vertex *v* because they would be part of the same component if the bridge *uw* were removed.

Graph *B*: There are no bridges in Graph *B*. The only method we have to determine if a Hamilton path from vertex *s* to vertex *v* exists is to try every possibility. From vertex *s*, we can visit either vertex *y* or vertex *t*. We will try vertex *y* first and then come back to see what happens with vertex *t*. After visiting *y*, we must visit *z* and then *u*, but then we have to decide between vertices *r*, *t*, and *v* next as shown.

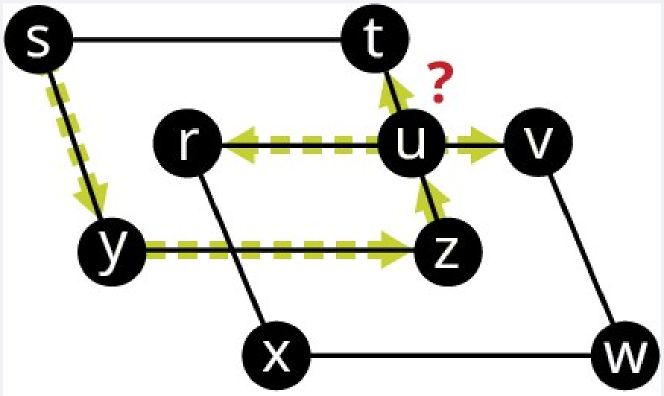


Figure 180: Choose Vertex *r*, *t*, or *v*

Vertex *v* is not an option since we want to end at *v*. Vertex *t* is not an option since that would force us to go to visit *s* a second time. So, we must go to vertex *r* next. After vertex *r*, we must visit *x*, then *w*, then *v*, but we missed vertex *t* as shown.

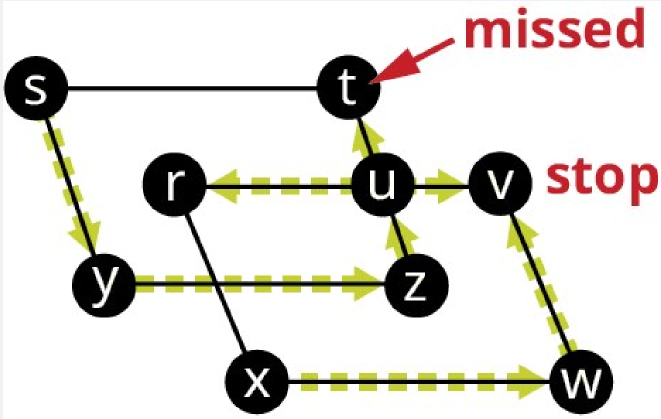


Figure 181: Missed Vertex t

Let's go back to the beginning and choose t instead of y. After t, we must go to u and then we have a choice to make between r, v, and z as shown.

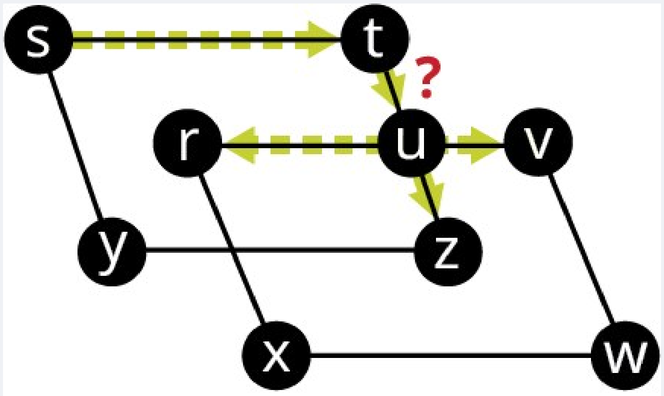


Figure 182: Choose Vertex v, r, or z

Vertex v is not an option since we want to end at v. Vertex z is not an option since that would force us to go y and then to visit s a second time. So, we must go to vertex r next. After r, we must go to x then w then v, where we have to stop even though we have missed vertices y and z, as shown.

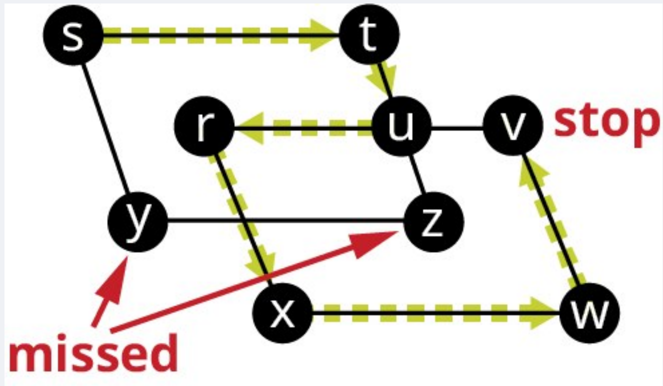


Figure 183: Missed Vertices y and z

So, we have tried every possible route and there are no Hamilton paths between s and v in Graph B.

Graph C: In Graph C, there is a Hamilton path, $s \longrightarrow t \longrightarrow u \longrightarrow x \longrightarrow w \longrightarrow v$.

Graph D: In Graph D, there is a bridge, tx, which would form components $\{r, s, t, u, q\}$ and $\{v, w, x, y, z\}$ if it were removed. Since s and v would be in different components, it is possible there is a Hamilton path between them. The only way to know is to try all possibilities. If we begin at s, we can go to r then t, or we can go directly to t, either way, we have a problem as you can see in .

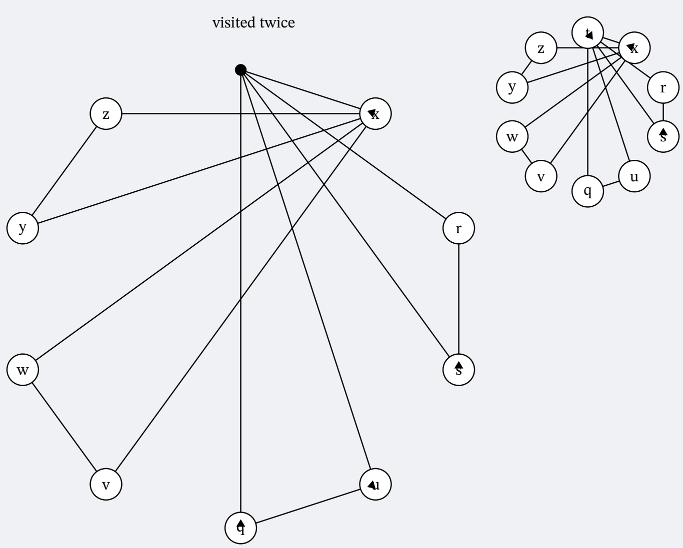


Figure 184: Vertices Visited Twice or Skipped

If we visit all the vertices in the component $\{r, s, t, u, q\}$, we will have to visit t a second time in order to cross the bridge. If we visit t only once, we have to skip some of the vertices. So, there is no Hamilton path between s and v.

There is not a short way to determine if there is a Hamilton path between two vertices on a graph that works in every situation. However, there are a few common situations that can help us to quickly determine that there is no Hamilton path. Some of these are listed in .

Scenario	Diagram
Scenario 1 If an edge ab is a bridge, then there is no Hamilton path between a pair of vertices that are on the same side of edge ab. We saw this in Graph A of Example 3.	No Hamilton path between any two vertices in the component $\{a, c, d, f\}$. No Hamilton path between any two vertices in $\{b, e, h, g, i\}$.
Scenario 2 If an edge ab is a bridge with at least three components on each side, then there is no Hamilton path beginning or ending at a or b. We saw this in Graph D of Example 3.	No Hamilton path beginning or ending at a or b.
Scenario 3 If a graph is composed of two cycles joined only at a single vertex p, and v is any vertex that is NOT adjacent to p, then there are no Hamilton paths beginning or ending at either p or v. We saw this in Graph B of Example 3.	No Hamilton path can be formed starting or ending at vertices, r, v, or u because they are not adjacent to p.

CHECKPOINT

There are also many other situations in which a Hamilton path is not possible. These are just a few that you will encounter.

Hamilton Path or Euler Trail?

We learned in Euler Trails that an Euler trail visits each edge exactly once, whereas a Hamilton path visits each vertex exactly once. Let's practice distinguishing between the two.

Example 4 **Distinguishing between Hamilton Path and Euler Trail**

Use to determine if the given sequence of vertices is a Hamilton path, an Euler trail, both, or neither.

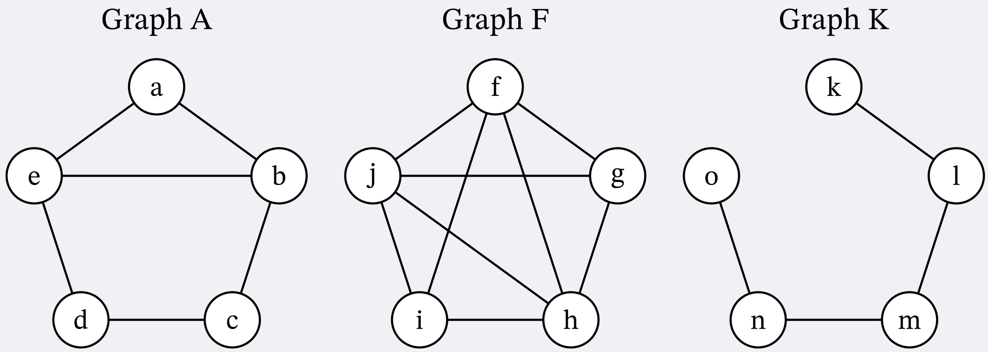


Figure 185: Graphs A, F, and K

1. Graph A, $e \rightarrow b \rightarrow a \rightarrow e \rightarrow d \rightarrow c \rightarrow b$
2. Graph F, $f \rightarrow g \rightarrow j \rightarrow h \rightarrow i$
3. Graph K, $k \rightarrow l \rightarrow m \rightarrow n \rightarrow o$

Solution

1. Since the sequence covers every edge once but visits vertices more than once, it is only an Euler trail.
2. Since the sequence visits every vertex exactly once but skips some edges, it is only a Hamilton path.
3. Since the sequence visits each edge and each vertex exactly once, it is both an Euler trail and a Hamilton path.

WORK IT OUT

As we saw in , the Emerald City lies at the center of the Magical Land of Oz, with Gillikin Country to the north, Winkie Country to the east, Munchkin Country to the west, and Quadling Country to the south. Munchkin Country and Winkie Country each shares a border with Gillikin Country and Quadling Country. Let's apply graph theory to Dorothy's famous journey through Oz one more time!

Draw a graph in which each vertex is one of the regions of Oz. Is there a Hamilton path that Dorothy could follow, instead of the yellow brick road, to lead her from the land of the Munchkins, through all the regions of Oz exactly once, and end in the Emerald City? If so, what might it be? Compare your results with those of a classmate.

Key Terms

- Hamilton path

Key Concepts

- A Hamilton path visits every vertex exactly once.
- Some Hamilton paths are also Euler trails, but some are not.

12.9 Traveling Salesperson Problem

Figure 186: Each door on the route of a traveling salesperson can be represented as a vertex on a graph.

Learning Objectives

After completing this section, you should be able to:

1. Distinguish between brute force algorithms and greedy algorithms.
2. List all distinct Hamilton cycles of a complete graph.
3. Apply brute force method to solve traveling salesperson applications.
4. Apply nearest neighbor method to solve traveling salesperson applications.

We looked at Hamilton cycles and paths in the previous sections Hamilton Cycles and Hamilton Paths. In this section, we will analyze Hamilton cycles in complete weighted graphs to find the shortest route to visit a number of locations and return to the starting point. Besides the many routing applications in which we need the shortest distance, there are also applications in which we search for the route that is least expensive or takes the least time. Here are a few less common applications that you can read about on a website set up by the mathematics department at the University of Waterloo in Ontario, Canada:

- Design of fiber optic networks
- Minimizing fuel expenses for repositioning satellites
- Development of semi-conductors for microchips
- A technique for mapping mammalian chromosomes in genome sequencing

Before we look at approaches to solving applications like these, let's discuss the two types of algorithms we will use.

Brute Force and Greedy Algorithms

An algorithm is a sequence of steps that can be used to solve a particular problem. We have solved many problems in this chapter, and the procedures that we used were different types of algorithms. In this section, we will use two common types of algorithms, a **brute force algorithm** and a **greedy algorithm**. A brute force algorithm begins by listing every possible solution and applying each one until the best solution is found. A greedy algorithm approaches a problem in stages, making the apparent best choice at each stage, then linking the choices together into an overall solution which may or may not be the best solution.

To understand the difference between these two algorithms, consider the tree diagram in . Suppose we want to find the path from left to right with the largest total sum. For example, branch A in the tree diagram has a sum of $10 + 2 + 11 + 13 = 36$.

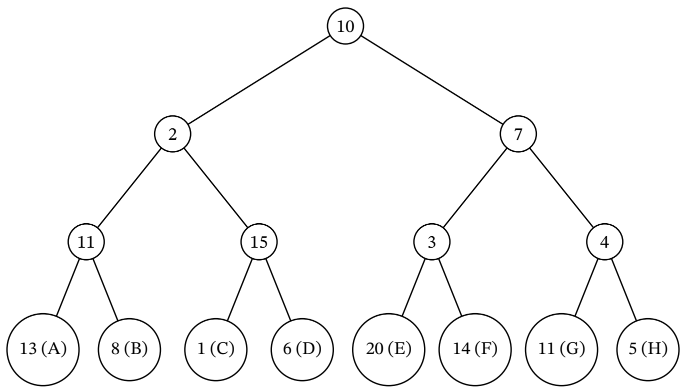


Figure 187: Points Along Different Paths

To be certain that you pick the branch with greatest sum, you could list each sum from each of the different branches:

A: $10 + 2 + 11 + 13 = 36$

B: $10 + 2 + 11 + 8 = 31$

C: $10 + 2 + 15 + 1 = 28$

D: $10 + 2 + 15 + 6 = 33$

E: $10 + 7 + 3 + 20 = 40$

F: $10 + 7 + 3 + 14 = 34$

G: $10 + 7 + 4 + 11 = 32$

H: $10 + 7 + 4 + 5 = 26$

Then we know with certainty that branch E has the greatest sum.

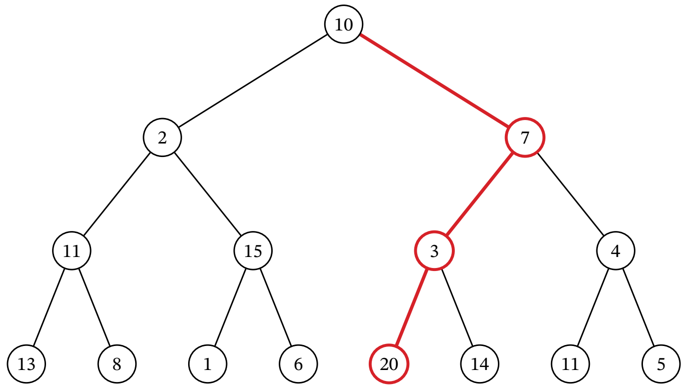


Figure 188: Branch *E*

Now suppose that you wanted to find the branch with the highest value, but you only were shown the tree diagram in phases, one step at a time.

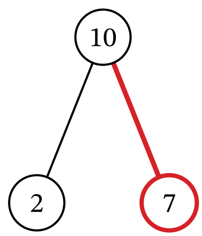


Figure 189: Tree Diagram Phase 1

After phase 1, you would have chosen the branch with 10 and 7. So far, you are following the same branch. Let's look at the next phase.

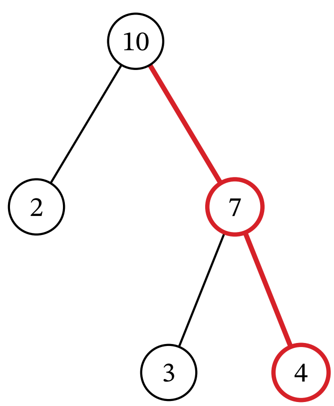


Figure 190: Tree Diagram Phase 2

After phase 2, based on the information you have, you will choose the branch with 10, 7 and 4. Now, you are following a different branch than before, but it is the best choice based on the information you have. Let's look at the last phase.

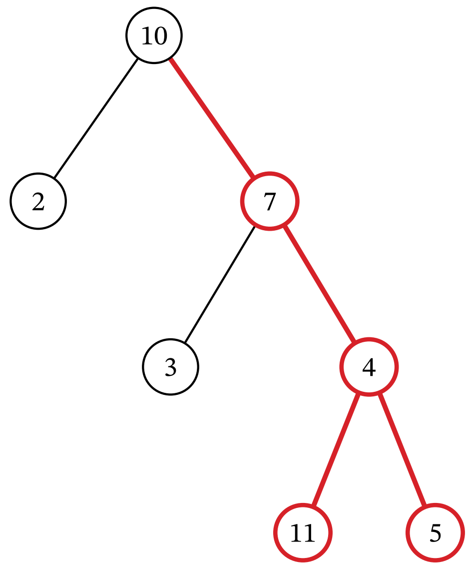


Figure 191: Tree Diagram Phase 3

After phase 3, you will choose branch *G* which has a sum of 32.

The process of adding the values on each branch and selecting the highest sum is an example of a brute force algorithm because all options were explored in detail. The process of choosing the branch in phases, based on the best choice at each phase is a greedy algorithm. Although a brute force algorithm gives us the ideal solution, it can take a very long time to implement. Imagine a tree diagram with thousands or even millions of branches. It might not be possible to check all the sums. A greedy algorithm, on the other hand, can be completed in a relatively short time, and generally leads to good solutions, but not necessarily the ideal solution.

Example 1 **Distinguishing between Brute Force and Greedy Algorithms**

A cashier rings up a sale for $4.63 cents in U.S. currency. The customer pays with a $5 bill. The cashier would like to give the customer $0.37 in change using the fewest coins possible. The coins that can be used are quarters ($0.25), dimes ($0.10), nickels ($0.05), and pennies ($0.01). The cashier starts by selecting the coin of highest value less than or equal to $0.37, which is a quarter. This leaves $\$\,0.37 - \$\,0.25 = \$\,0.12$. The cashier selects the coin of highest value less than or equal to $0.12, which is a dime. This leaves $\$\,0.12 - \$\,0.10 = \$\,0.02$. The cashier selects the coin of highest value less than or equal to $0.02, which is a penny. This leaves $\$\,0.02 - \$\,0.01 = \$\,0.01$. The cashier selects the coin of highest value less than or equal to $0.01, which is a penny. This leaves no remainder. The cashier used one quarter, one dime, and two pennies, which is four coins. Use this information to answer the following questions.

1. Is the cashier's approach an example of a greedy algorithm or a brute force algorithm? Explain how you know.

2. The cashier's solution is the best solution. In other words, four is the fewest number of coins possible. Is this consistent with the results of an algorithm of this kind? Explain your reasoning.

Solution

1. The approach the cashier used is an example of a greedy algorithm, because the problem was approached in phases and the best choice was made at each phase. Also, it is not a brute force algorithm, because the cashier did not attempt to list out all possible combinations of coins to reach this conclusion.
2. Yes, it is consistent. A greedy algorithm does not always yield the best result, but sometimes it does.

The Traveling Salesperson Problem

Now let's focus our attention on the graph theory application known as the **traveling salesperson problem (TSP)** in which we must find the shortest route to visit a number of locations and return to the starting point.

Recall from Hamilton Cycles, the officer in the U.S. Air Force who is stationed at Vandenberg Air Force base and must drive to visit three other California Air Force bases before returning to Vandenberg. The officer needed to visit each base once. We looked at the weighted graph in representing the four U.S. Air Force bases: Vandenberg, Edwards, Los Angeles, and Beal and the distances between them.

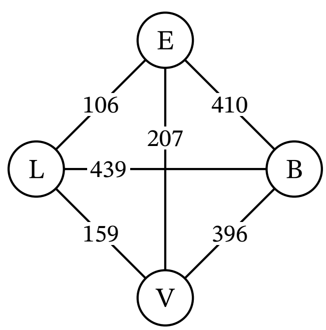


Figure 192: Graph of Four California Air Force Bases

Any route that visits each base and returns to the start would be a Hamilton cycle on the graph. If the officer wants to travel the shortest distance, this will correspond to a Hamilton cycle of lowest weight. We saw in that there are six distinct Hamilton cycles (directed cycles) in a complete graph with four vertices, but some lie on the same cycle (undirected cycle) in the graph.

Complete Graph	Cycle	Cycle	Cycle
Clockwise Hamilton Cycle	$a \to b \to c \to d \to a$	$a \to b \to d \to c \to a$	$a \to c \to b \to d \to a$
Counterclockwise Hamilton Cycle	$a \to d \to c \to b \to a$	$a \to c \to d \to b \to a$	$a \to d \to b \to c \to a$

Since the distance between bases is the same in either direction, it does not matter if the officer travels clockwise or counterclockwise. So, there are really only three possible distances as shown.

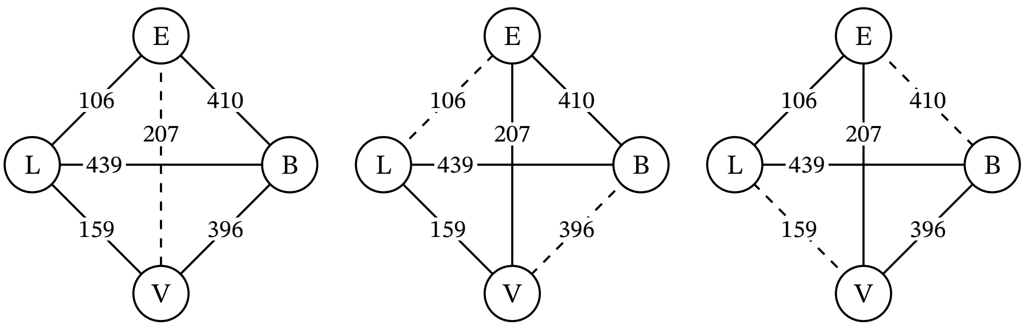


Figure 193: Three Possible Distances

The possible distances are:

$$396 + 410 + 106 + 159 = 1071$$
$$207 + 410 + 439 + 159 = 1215$$
$$396 + 439 + 106 + 207 = 1148$$

So, a Hamilton cycle of least weight is $V \to B \to E \to L \to V$ (or the reverse direction). The officer should travel from Vandenberg to Beal to Edwards, to Los Angeles, and back to Vandenberg.

Finding Weights of All Hamilton Cycles in Complete Graphs

Notice that we listed all of the Hamilton cycles and found their weights when we solved the TSP about the officer from Vandenberg. This is a skill you will need to practice. To make sure you don't miss any, you can calculate the number of possible Hamilton cycles in a complete graph. It is also helpful to know that half of the directed cycles in a complete graph are the same cycle in reverse direction, so, you only have to calculate half the number of possible weights, and the rest are duplicates.

FORMULA

In a complete graph with n vertices,

- The number of distinct Hamilton cycles is $(n-1)!$.
- There are at most $\frac{(n-1)!}{2}$ different weights of Hamilton cycles.

CHECKPOINT

TIP! When listing all the distinct Hamilton cycles in a complete graph, you can start them all at any vertex you choose. Remember, the cycle $\mathrm{a} \rightarrow \mathrm{b} \rightarrow \mathrm{c} \rightarrow \mathrm{a}$ *is the same cycle as* $\mathrm{b} \rightarrow \mathrm{c} \rightarrow \mathrm{a} \rightarrow \mathrm{b}$ *so there is no need to list both.*

Example 2 **Calculating Possible Weights of Hamilton Cycles**

Suppose you have a complete weighted graph with vertices N, M, O, and P.

1. Use the formula $(n-1)!$ to calculate the number of distinct Hamilton cycles in the graph.
2. Use the formula $\frac{(n-1)!}{2}$ to calculate the greatest number of different weights possible for the Hamilton cycles.
3. Are all of the distinct Hamilton cycles listed here? How do you know?
   Cycle 1: $N \rightarrow M \rightarrow O \rightarrow P \rightarrow N$
   Cycle 2: $N \rightarrow M \rightarrow P \rightarrow O \rightarrow N$
   Cycle 3: $N \rightarrow O \rightarrow M \rightarrow P \rightarrow N$
   Cycle 4: $N \rightarrow O \rightarrow P \rightarrow M \rightarrow N$
   Cycle 5: $N \rightarrow P \rightarrow M \rightarrow O \rightarrow N$
   Cycle 6: $N \rightarrow P \rightarrow O \rightarrow M \rightarrow N$
4. Which pairs of cycles must have the same weights? How do you know?

Solution

1. There are 4 vertices; so, $n = 4$. This means there are $(n-1)! = (4-1)! = 3 \cdot 2 \cdot 1 = 6$ distinct Hamilton cycles beginning at any given vertex.
2. Since $n = 4$, there are $\frac{(n-1)!}{2} = \frac{(4-1)!}{2} = \frac{6}{2} = 3$ possible weights.
3. Yes, they are all distinct cycles and there are 6 of them.
4. Cycles 1 and 6 have the same weight, Cycles 2 and 4 have the same weight, and Cycles 3 and 5 have the same weight, because these pairs follow the same route through the graph but in reverse.

CHECKPOINT

TIP! When listing the possible cycles, ignore the vertex where the cycle begins and ends and focus on the ways to arrange the letters that represent the vertices in the middle. Using a systematic approach is best; for example, if you must arrange the letters M, O, and P, first list all those arrangements beginning with M, then beginning with O, and then beginning with P, as we did in Example 12.42.

The Brute Force Method

The method we have been using to find a Hamilton cycle of least weight in a complete graph is a brute force algorithm, so it is called the **brute force method**. The steps in the brute force method are:

Step 1: Calculate the number of distinct Hamilton cycles and the number of possible weights.

Step 2: List all possible Hamilton cycles.

Step 3: Find the weight of each cycle.

Step 4: Identify the Hamilton cycle of lowest weight.

Example 3 **Applying the Brute Force Method**

On the next assignment, the air force officer must leave from Travis Air Force base, visit Beal, Edwards, and Vandenberg Air Force bases each exactly once and return to Travis Air Force base. There is no need to visit Los Angeles Air Force base. Use to find the shortest route.

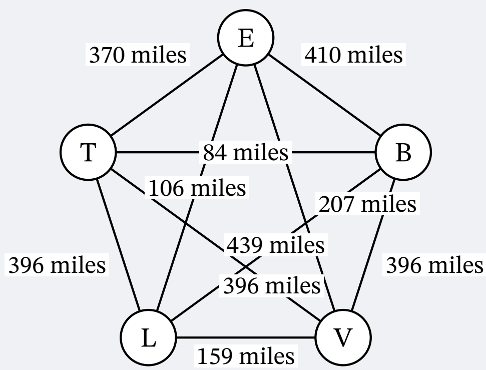


Figure 194: Distances between Five California Air Force Bases

Solution

Step 1: Since there are 4 vertices, there will be $(4-1)! = 3! = 6$ cycles, but half of them will be the reverse of the others; so, there will be $\frac{(4-1)!}{2} = \frac{6}{2} = 3$ possible distances.

Step 2: List all the Hamilton cycles in the subgraph of the graph in .

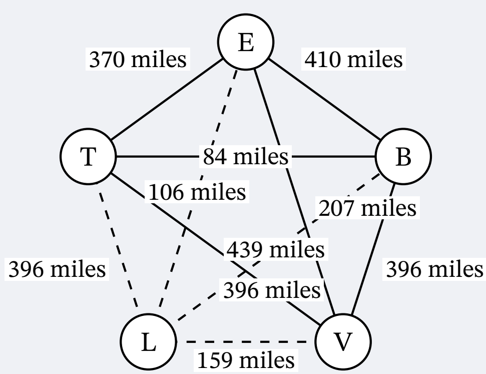


Figure 195: Subgraph with Cities *B*, *E*, *T*, and *V*

To find the 6 cycles, focus on the three vertices in the middle, *B, E,* and *V*. The arrangements of these vertices are *BEV, BVE, EBV, EVB, VBE*, and *VEB*. These would correspond to the 6 cycles:

1: $T \rightarrow B \rightarrow E \rightarrow V \rightarrow T$

2: $T \rightarrow B \rightarrow V \rightarrow E \rightarrow T$

3: $T \rightarrow E \rightarrow B \rightarrow V \rightarrow T$

4: $T \rightarrow E \rightarrow V \rightarrow B \rightarrow T$

5: $T \rightarrow V \rightarrow B \rightarrow E \rightarrow T$

6: $T \rightarrow V \rightarrow E \rightarrow B \rightarrow T$

Step 3: Find the weight of each path. You can reduce your work by observing the cycles that are reverses of each other.

1: $84 + 410 + 207 + 396 = 1097$

2: $84 + 396 + 207 + 370 = 1071$

3: $370 + 410 + 396 + 396 = 1572$

4: Reverse of cycle 2, 1071

5: Reverse of cycle 3, 1572

6: Reverse of cycle 1, 1097

Step 4: Identify a Hamilton cycle of least weight.

The second path, $T \rightarrow B \rightarrow V \rightarrow E \rightarrow T$, and its reverse, $T \rightarrow E \rightarrow V \rightarrow B \rightarrow T$, have the least weight. The solution is that the officer should travel from Travis Air Force base to Beal Air Force Base, to Vandenberg Air Force base, to Edwards Air Force base, and return to Travis Air Force base, or the same route in reverse.

Now suppose that the officer needed a cycle that visited all 5 of the Air Force bases in . There would be $(5-1)! = 4! = 4 \times 3 \times 2 \times 1 = 24$ different arrangements of vertices and $\frac{(5-1)!}{2} = \frac{4!}{2} = \frac{24}{2} = 12$ distances to compare using the brute force method. If you consider 10 Air Force bases, there would be $(10-1)! = 9! = 9 \cdot 8 \cdot 7 \cdot 6 \cdot 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 = 362,880$ different arrangements and $\frac{(10-1)!}{2} = \frac{9!}{2} = \frac{9 \cdot 8 \cdot 7 \cdot 6 \cdot 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1}{2} = 181,440$ distances to consider. There must be another way!

The Nearest Neighbor Method

When the brute force method is impractical for solving a traveling salesperson problem, an alternative is a greedy algorithm known as the **nearest neighbor method**, which always visit the closest or least costly place first. This method finds a Hamilton cycle of relatively low weight in a complete graph in which, at each phase, the next vertex is chosen by comparing the edges between the current vertex and the remaining vertices to find the lowest weight. Since the nearest neighbor method is a greedy algorithm, it usually doesn't give the best solution, but it usually gives a solution that is "good enough." Most importantly, the number of steps will be the number of vertices. That's right! A problem with 10 vertices requires 10 steps, not 362,880. Let's look at an example to see how it works.

Suppose that a candidate for governor wants to hold rallies around the state. They plan to leave their home in city *A*, visit cities *B, C, D, E*, and *F* each once, and return home. The airfare between cities is indicated in the graph in .

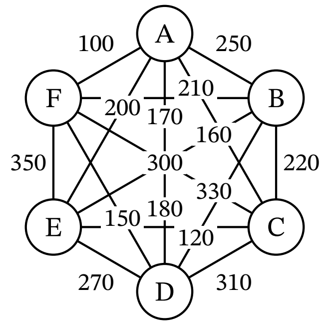


Figure 196: Airfares between Cities *A, B, C, D, E*, and *F*

Let's help the candidate keep costs of travel down by applying the nearest neighbor method to find a Hamilton cycle that has a reasonably low weight. Begin by marking starting vertex as V_1 for "visited 1st." Then to compare the weights of the edges between *A* and vertices adjacent to *A*: \$250, \$210, \$300, \$200, and \$100 as shown. The lowest of these is \$100, which is the edge between *A* and *F*.

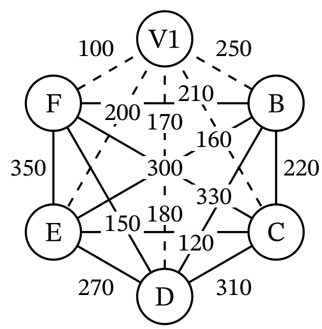


Figure 197: Finding the Second Vertex

Mark *F* as V_2 for "visited 2nd" then compare the weights of the edges between *F* and the remaining vertices adjacent to *F*: \$170, \$330, \$150 and \$350 as shown. The lowest of these is \$150, which is the edge between *F* and *D*.

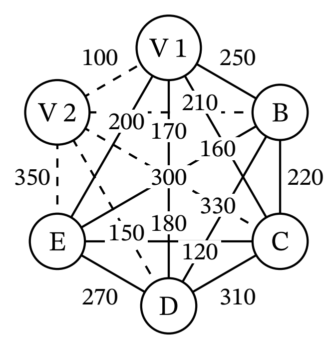


Figure 198: Finding the Third Vertex

Mark D as V_3 for "visited 3rd." Next, compare the weights of the edges between D and the remaining vertices adjacent to D: \$120, \$310, and \$270 as shown. The lowest of these is \$120, which is the edge between D and B.

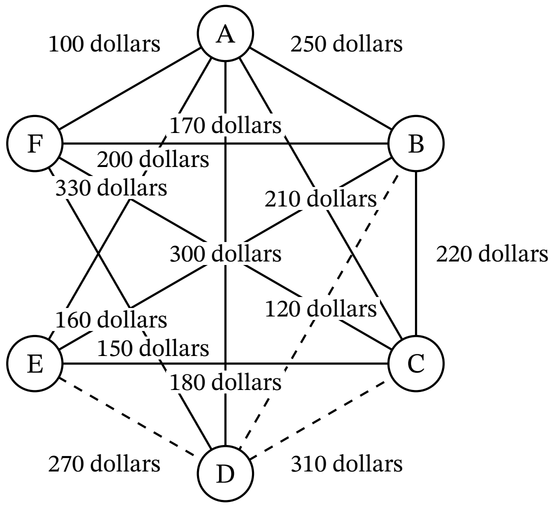


Figure 199: Finding the Fourth Vertex

So, mark B as V_4 for "visited 4th." Finally, compare the weights of the edges between B and the remaining vertices adjacent to B: \$160 and \$220 as shown. The lower amount is \$160, which is the edge between B and E.

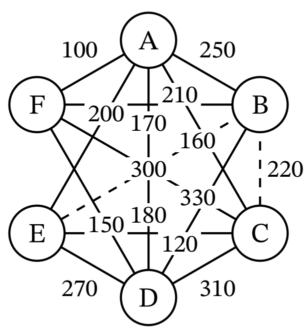


Figure 200: Finding the Fifth Vertex

Now you can mark E as V_5 and mark the only remaining vertex, which is C, as V_6. This is shown. Make a note of the weight of the edge from E to C, which is \$180, and from C back to A, which is \$210.

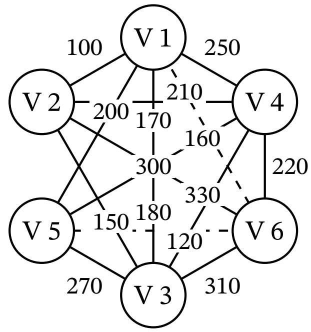


Figure 201: Finding the Sixth Vertex

The Hamilton cycle we found is $A \longrightarrow F \longrightarrow D \longrightarrow B \longrightarrow E \longrightarrow C \longrightarrow A$. The weight of the circuit is $\$\,100 + \$\,150 + \$\,120 + \$\,160 + \$\,180 + \$\,210 = \$\,920$. This may or may not be the route with the lowest cost, but there is a good chance it is very close since the weights are most of the lowest weights on the graph and we found it in six steps instead of finding 120 different Hamilton cycles and calculating 60 weights. Let's summarize the procedure that we used.

Step 1: Select the starting vertex and label V_1 for "visited 1st." Identify the edge of lowest weight between V_1 and the remaining vertices.

Step 2: Label the vertex at the end of the edge of lowest weight that you found in previous step as V_n where the subscript n indicates the order the vertex is visited. Identify the edge of lowest weight between V_n and the vertices that remain to be visited.

Step 3: If vertices remain that have not been visited, repeat Step 2. Otherwise, a Hamilton cycle of low weight is $V_1 \rightarrow V_2 \rightarrow \cdots \rightarrow V_n \rightarrow V_1$.

Example 4 **Using the Nearest Neighbor Method**

Suppose that the candidate for governor wants to hold rallies around the state but time before the election is very limited. They would like to leave their home in city A, visit cities B, C, D, E, and F each once, and return home. The airfare between cities is not as important as the time of travel, which is indicated in . Use the nearest neighbor method to find a route with relatively low travel time. What is the total travel time of the route that you found?

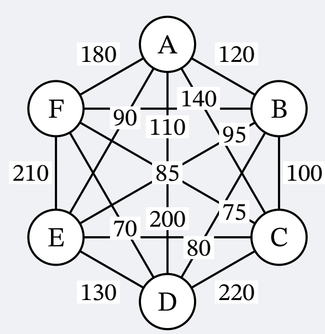


Figure 202: Travel Times between Cities A, B, C, D, E and F

Solution

Step 1: Label vertex A as V_1. The edge of lowest weight between A and the remaining vertices is 85 min between A and D.

Step 2: Label vertex D as V_2. The edge of lowest weight between D and the vertices that remain to be visited, B, C, E, and F, is 70 min between D and F.

Repeat Step 2: Label vertex F as V_3. The edge of lowest weight between F and the vertices that remain to be visited, B, C, and E, is 75 min between F and C.

Repeat Step 2: Label vertex C as V_4. The edge of lowest weight between C and the vertices that remain to be visited, B and E, is 100 min between C and B.

Repeat Step 2: Label vertex B as V_5. The only vertex that remains to be visited is E. The weight of the edge between B and E is 95 min.

Step 3: A Hamilton cycle of low weight is $A \rightarrow D \rightarrow F \rightarrow C \rightarrow B \rightarrow E \rightarrow A$. So, a route of relatively low travel time is A to D to F to C to B to E and back to A. The total travel time of this route is: 85 min $+70$ min $+75$ min $+100$ min $+95$ min $+90$ min $=$ 515 min or 8 hrs 35 min

Key Terms

- brute force algorithm
- greedy algorithm
- traveling salesperson problem (TSP)
- brute force method
- nearest neighbor method

Key Concepts

- A brute force algorithm always finds the ideal solution but can be impractical whereas a greedy algorithm is efficient but usually does not lead to the ideal solution.
- A Hamilton cycle of lowest weight is a solution to the traveling salesperson problem.
- The brute force method finds a Hamilton cycle of lowest weight in a complete graph.
- The nearest neighbor method is a greedy algorithm that finds a Hamilton cycle of relatively low weight in a complete graph.

Formulas

- In a complete graph with n vertices, the number of distinct Hamilton cycles is $(n-1)!$.
- In a complete graph with n vertices, there are at most $\frac{(n-1)!}{2}$ different weights of Hamilton cycles.

12.10 Trees

Figure 203: In graph theory, graphs known as trees have structures in common with live trees.

Learning Objectives

After completing this section, you should be able to:

1. Describe and identify trees.
2. Determine a spanning tree for a connected graph.
3. Find the minimum spanning tree for a weighted graph.
4. Solve application problems involving trees.

We saved the best for last! In this last section, we will discuss arguably the most fun kinds of graphs, trees. Have you every researched your family tree? Family trees are a perfect example of the kind of trees we study in graph theory. One of the characteristics of a family tree graph is that it never loops back around, because no one is their own grandparent!

What Is A Tree?

Whether we are talking about a family tree or a tree in a forest, none of the branches ever loops back around and rejoins the trunk. This means that a **tree** has no cyclic subgraphs, or is **acyclic**. A tree also has only one component. So, a tree is a connected acyclic graph. Here are some graphs that have the same characteristic. Each of the graphs in is a tree.

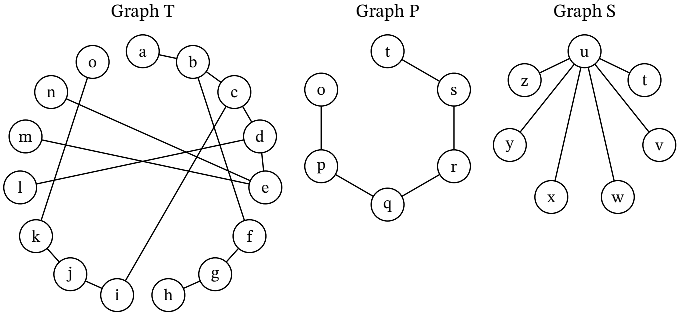


Figure 204: Graphs T, P, and S

Let's practice determining whether a graph is a tree. To do this, check if a graph is connected and has no cycles.

Example 1 **Identifying Trees**

Identify any trees in . If a graph is not a tree, explain how you know.

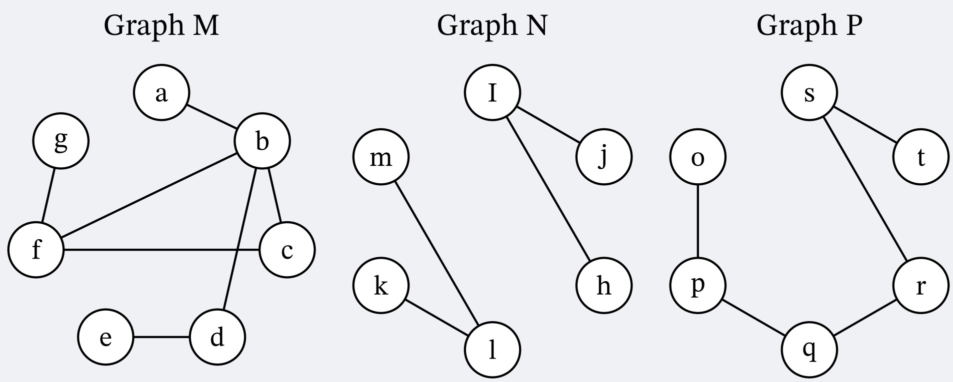


Figure 205: Graphs M, N, and P

Solution

- Graph M is not a tree because it contains the cycle (b, c, f).
- Graph N is not a tree because it is not connected. It has two components, one with vertices h, i, j, and another with vertices k, l, m.
- Graph P is a tree. It has no cycles and it is connected.

Types of Trees

Mathematicians have had a lot of fun naming graphs that are trees or that contain trees. For example, the graph in is not a tree, but it contains two components, one containing vertices a through d, and the other containing vertices e through g, each of which would be a tree on its own. This type of structure is called a **forest**. There are also interesting names for trees with certain characteristics.

- A **path graph or linear graph** is a tree graph that has exactly two vertices of degree 1 such that the only other vertices form a single path between them, which means that it can be drawn as a straight line.
- A **star tree** is a tree that has exactly one vertex of degree greater than 1 called a **root,** and all other vertices are adjacent to it.
- A **starlike tree** is a tree that has a single **root** and several paths attached to it.
- A **caterpillar tree** is a tree that has a central path that can have vertices of any degree, with each vertex not on the central path being adjacent to a vertex on the central path and having a degree of one.
- A **lobster tree** is a tree that has a central path that can have vertices of any degree, with paths consisting of either one or two edges attached to the central path.

Examples of each of these types of structures are given in .

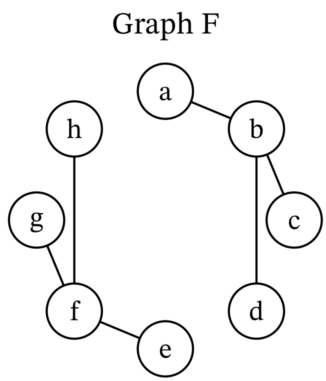


Figure 206: Forest Graph F

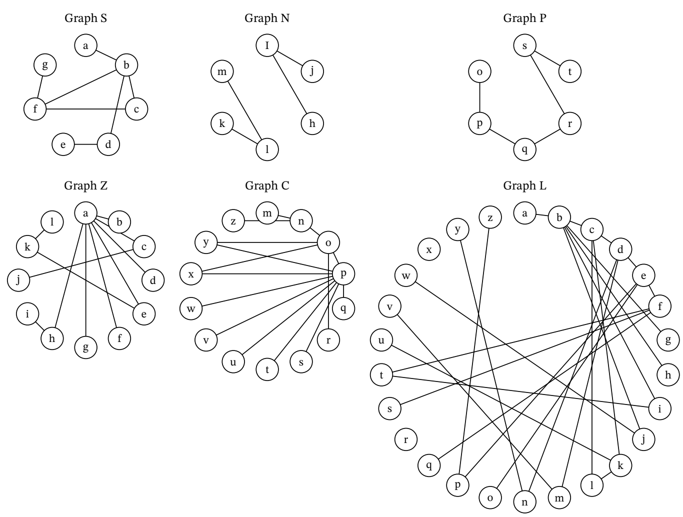


Figure 207: Six Types of Trees

Example 2 **Identifying Types of Trees**

Each graph in is one of the special types of trees we have been discussing. Identify the type of tree.

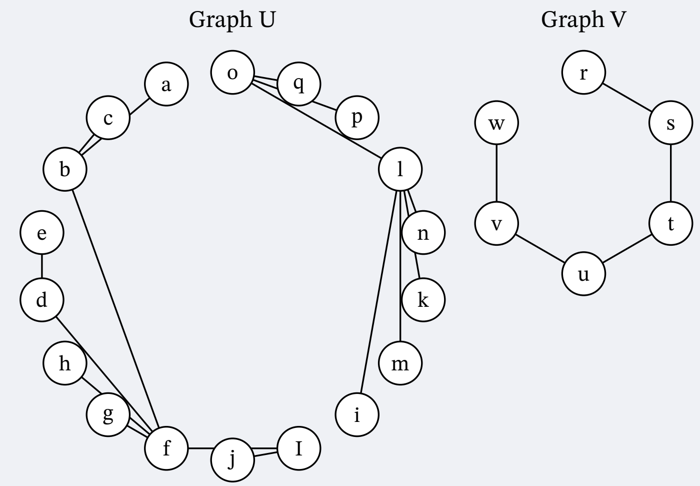


Figure 208: Graphs U and V

Solution

Graph U has a central path $a \longrightarrow b \longrightarrow d \longrightarrow f \longrightarrow i \longrightarrow l \longrightarrow o \longrightarrow q$. Each vertex that is not on the path has degree 1 and is adjacent to a vertex that is on the path. So, U is a caterpillar tree.

Graph V is a path graph because it is a single path connecting exactly two vertices of degree one, $r \longrightarrow s \longrightarrow u \longrightarrow v \longrightarrow w$.

Characteristics of Trees

As we study trees, it is helpful to be familiar with some of their characteristics. For example, if you add an edge to a tree graph between any two existing vertices, you will create a cycle, and the resulting graph is no longer a tree. Some examples are shown. Adding edge bj to Graph T creates cycle (b, c, i, j). Adding edge rt to Graph P creates cycle (r, s, t). Adding edge tv to Graph S creates cycle (t, u, v).

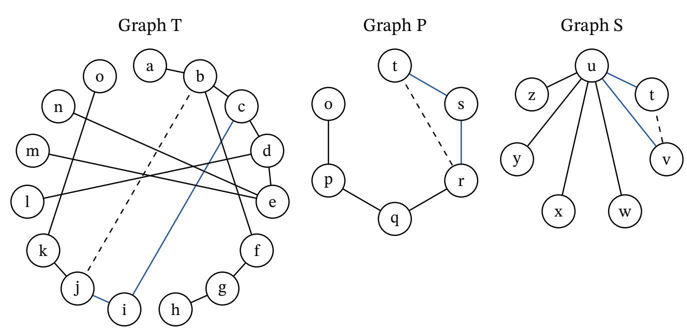


Figure 209: Adding Edges to Trees

It is also true that removing an edge from a tree graph will increase the number of components and the graph will no longer be connected. In fact, you can see in that removing one or more edges can create a forest. Removing edge qr from Graph P creates a graph with two components, one with vertices o, p and q, and the other with vertices r, s, and t. Removing edge uw from Graph

S creates two components, one with just vertex *w* and the other with the rest of the vertices. When two edges were removed from Graph *T*, edge *bf* and edge *cd*, creates a graph with three components as shown.

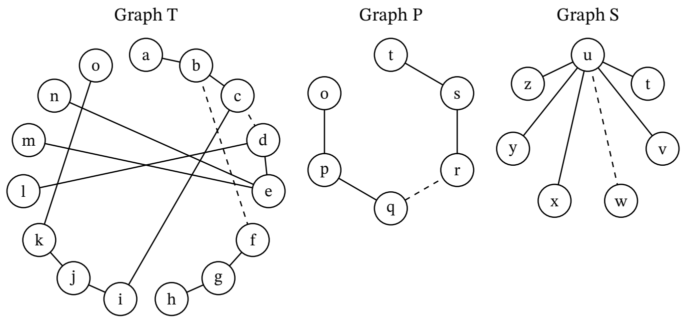


Figure 210: Removing Edges from Trees

A very useful characteristic of tree graphs is that the number of edges is always one less than the number of vertices. In fact, any connected graph in which the number of edges is one less than the number of vertices is guaranteed to be a tree. Some examples are given in .

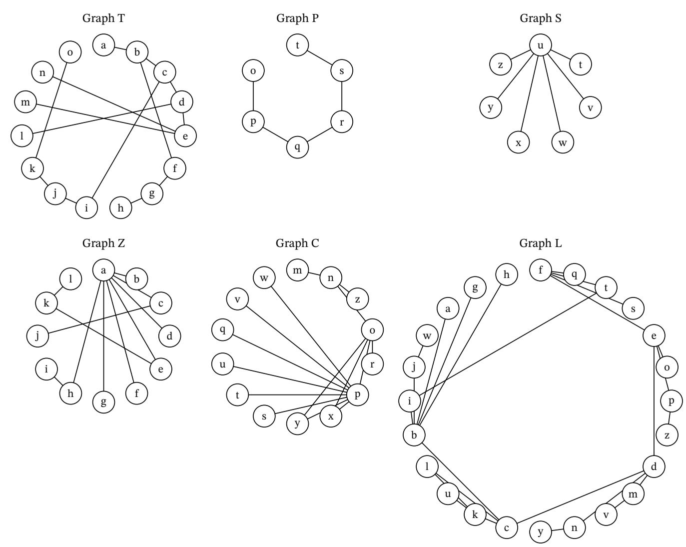


Figure 211: Number of Vertices and Edges in Trees vs. Other Graphs

FORMULA

The number of edges in a tree graph with n vertices is $n - 1$.

A connected graph with n vertices and $n - 1$ edges is a tree graph.

Example 3 **Exploring Characteristics of Trees**

Use Graphs *I* and *J* in to answer each question.

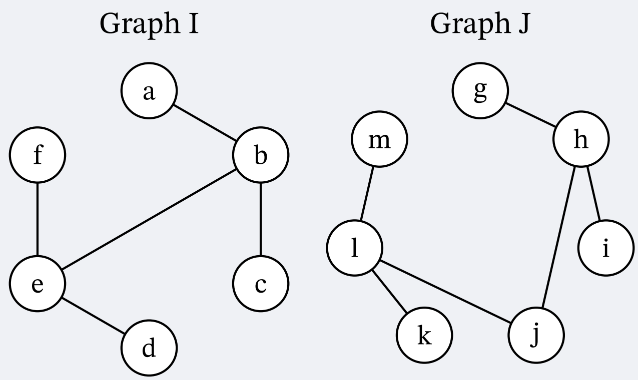


Figure 212: Graphs *I* and *J*

1. Which vertices are in each of the components that remain when edge *be* is removed from Graph *I*?
2. Determine the number of edges and the number of vertices in Graph *J*. Explain how this confirms that Graph *J* is a tree.
3. What kind of cycle is created if edge *im* is added to Graph *J*?

Solution

1. When edge *be* is removed, there are two components that remain. One component includes vertices *a*, *b*, and *c*. The other component includes vertices *d*, *e*, and *f*.
2. There are seven vertices and six edges in Graph *J*. This confirms that Graph *J* is a tree because the number of edges is one less than the number of vertices.
3. The pentagon (*i*, *h*, *j*, *l*, *m*) is created when edge *im* is added to Graph *J*.

WHO KNEW?

Graph Theory in the Movies

In the 1997 film *Good Will Hunting*, the main character, Will, played by Matt Damon, solves what is supposed to be an exceptionally difficult graph theory problem, "Draw all the homeomorphically irreducible trees of size $n = 10$." That sounds terrifying! But don't panic. Watch this great Numberphile video to see why this is actually a problem you can do at home!

VIDEO

The Problem in *Good Will Hunting* by Numberphile

Spanning Trees

Suppose that you planned to set up your own computer network with four devices. One option is to use a "mesh topology" like the one in , in which each device is connected directly to every other device in the network.

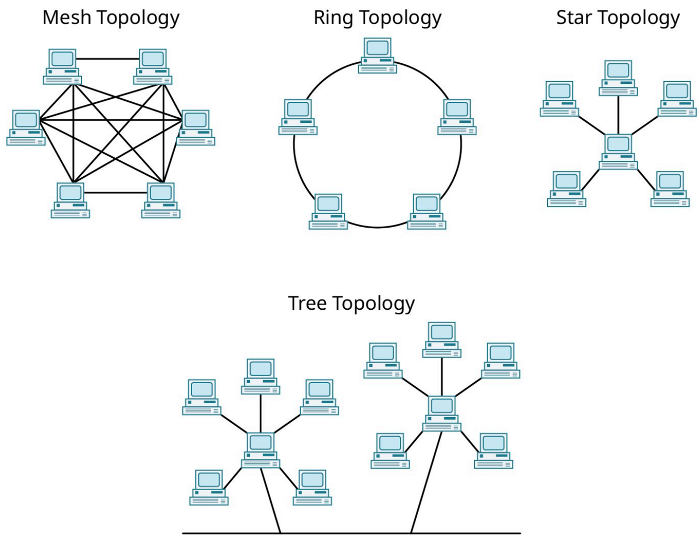


Figure 213: Common Network Configurations

The mesh topology for four devices could be represented by the complete Graph A_1 in where the vertices represent the devices, and the edges represent network connections. However, the devices could be networked using fewer connections. Graphs A_2, $A_{3,}$ and A_4 of show configurations in which three of the six edges have been removed. Each of the Graphs A_2, A_3 and A_4 in is a tree because it is connected and contains no cycles. Since Graphs A_2, A_3 and A_4 are also subgraphs of Graph A_1 that include every vertex of the original graph, they are also known as **spanning trees**.

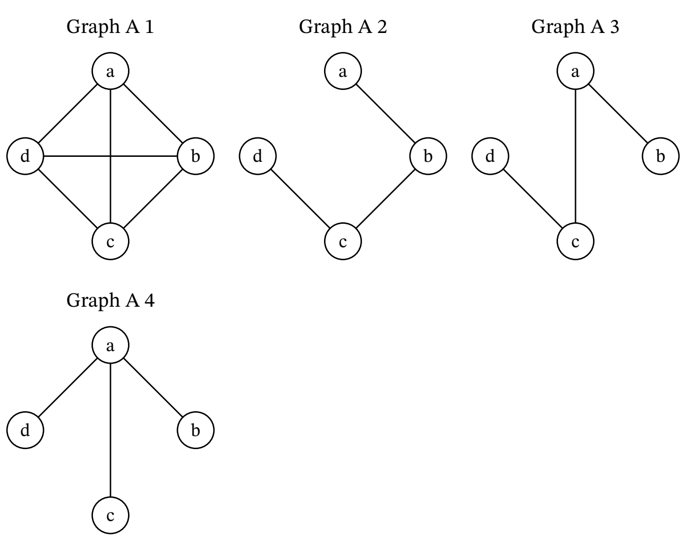


Figure 214: Network Configurations for Four Devices

By definition, spanning trees must span the whole graph by visiting all the vertices. Since spanning trees are subgraphs, they may only have edges between vertices that were adjacent in the original graph. Since spanning trees are trees, they are connected and they are acyclic. So, when deciding whether a graph is a spanning tree, check the following characteristics:

- All vertices are included.
- No vertices are adjacent that were not adjacent in the original graph.
- The graph is connected.
- There are no cycles.

Example 4 **Identifying Spanning Trees**

Use to determine which of graphs M_1, M_2, M_3, and M_4, are spanning trees of Q.

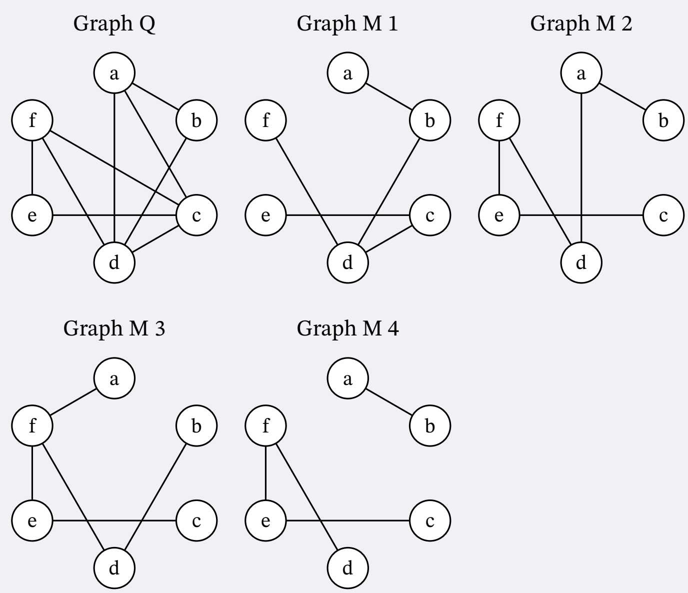


Figure 215: Graphs Q, M_1, M_2, M_3, and M_4

Solution

1. Graph M_1 is not a spanning tree of Graph Q because it has a cycle (c, d, f, e).
2. Graph M_2 is a spanning tree of Graph Q because it has all the original vertices, no vertices are adjacent in M_2 that weren't adjacent in Graph Q, Graph M_2 is connected and it contains no cycles.
3. Graph M_3 is not a spanning tree of Graph Q because vertices a and f are adjacent in Graph M_3 but not in Graph Q.
4. Graph M_4 is not a spanning tree of Graph Q because it is not connected.

   So, only graph M_2 is a spanning tree of Graph Q.

Constructing a Spanning Tree Using Paths

Suppose that you wanted to find a spanning tree within a graph. One approach is to find paths within the graph. You can start at any vertex, go any direction, and create a path through the graph stopping only when you can't continue without backtracking as shown.

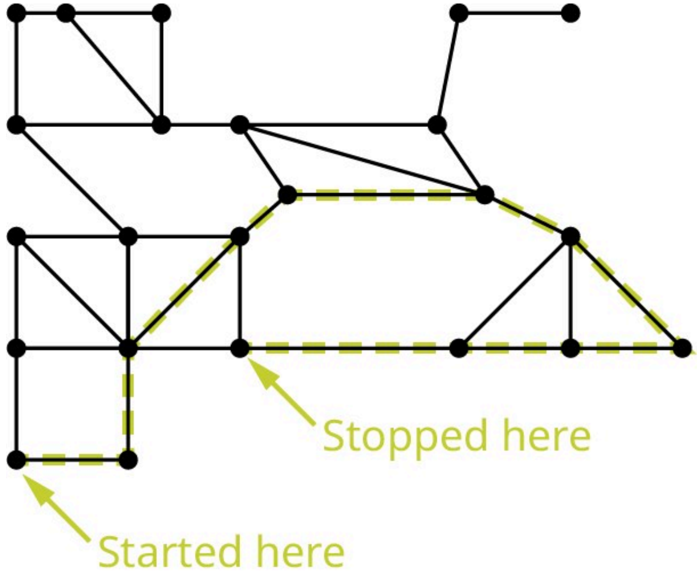


Figure 216: First Phase to Construct a Spanning Tree

Once you have stopped, pick a vertex along the path you drew as a starting point for another path. Make sure to visit only vertices you have not visited before as shown.

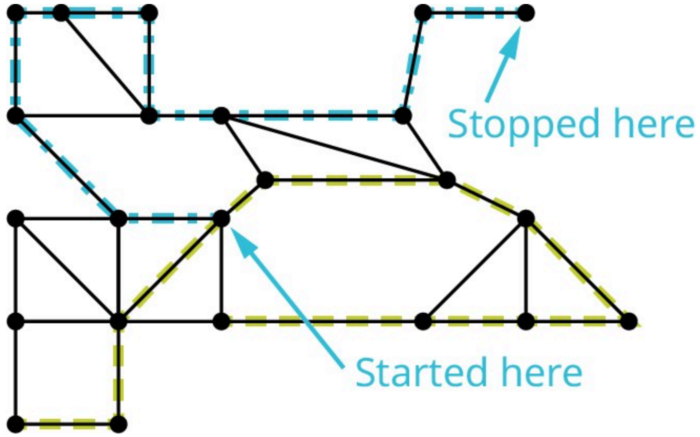


Figure 217: Intermediate Phase to Construct a Spanning Tree

Repeat this process until all vertices have been visited as shown.

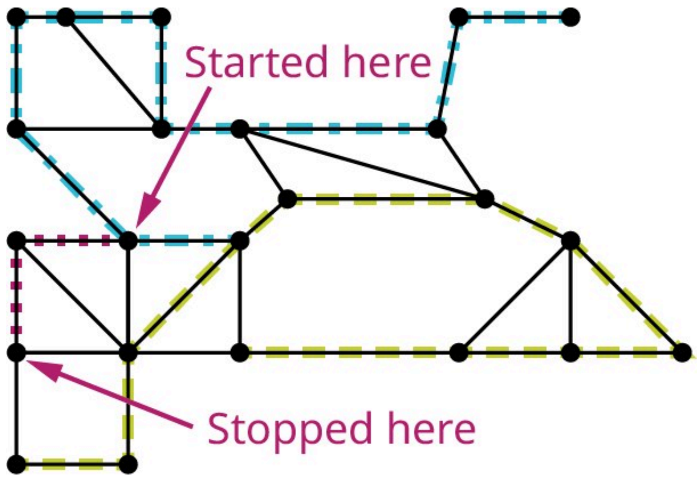


Figure 218: Final Phase to Construct a Spanning Tree

The end result is a tree that spans the entire graph as shown.

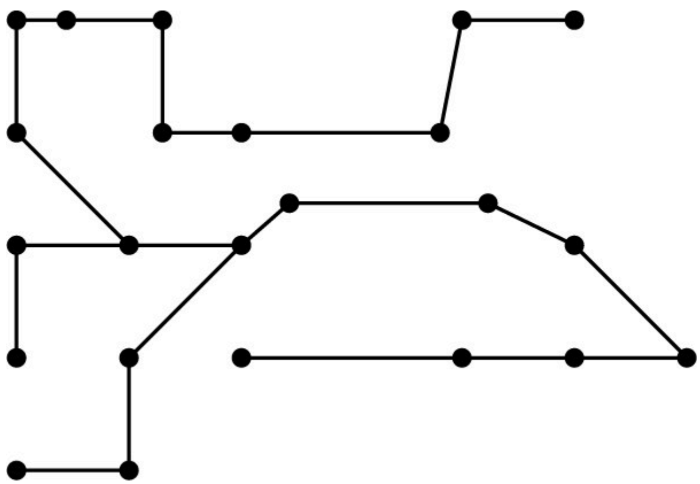


Figure 219: The Resulting Spanning Tree

Notice that this subgraph is a tree because it is connected and acyclic. It also visits every vertex of the original graph, so it is a spanning tree. However, it is not the only spanning tree for this graph. By making different turns, we could create any number of distinct spanning trees.

Example 5 **Constructing Spanning Trees**

Construct two distinct spanning trees for the graph in .

Graph L

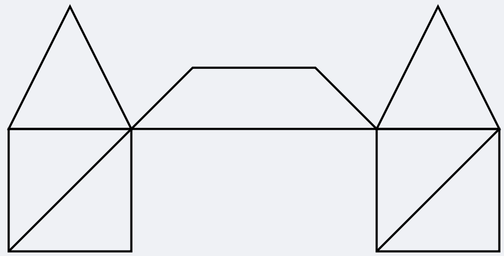


Figure 220: Graph *L*

Solution

Two possible solutions are given in and .

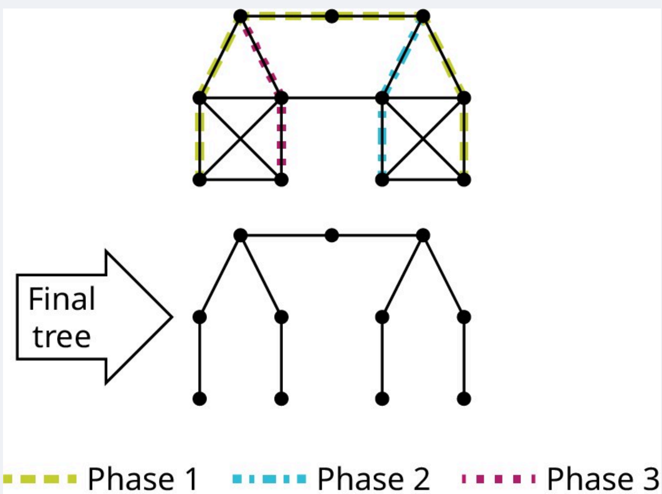


Figure 221: First Spanning Tree for Graph *L*

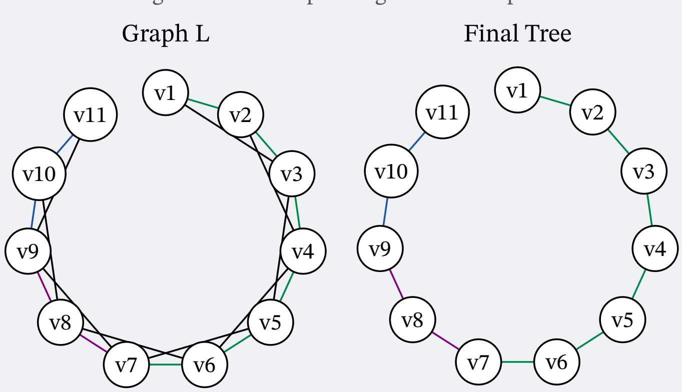


Figure 222: Second Spanning Tree for Graph *L*

Revealing Spanning Trees

Another approach to finding a spanning tree in a connected graph involves removing unwanted edges to reveal a spanning tree. Consider Graph *D* in .

Graph D

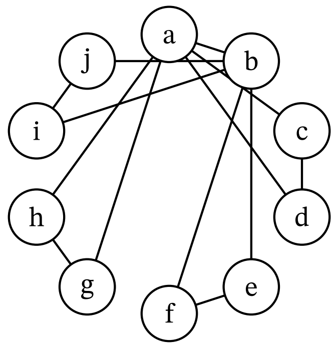


Figure 223: Graph *D*

Graph *D* has 10 vertices. A spanning tree of Graph *D* must have 9 edges, because the number of edges is one less than the number of vertices in any tree. Graph *D* has 13 edges so 4 need to be removed. To determine which 4 edges to remove, remember that trees do not have cycles. There are four triangles in Graph *D* that we need to break up. We can accomplish this by removing 1 edge from each of the triangles. There are many ways this can be done. Two of these ways are shown.

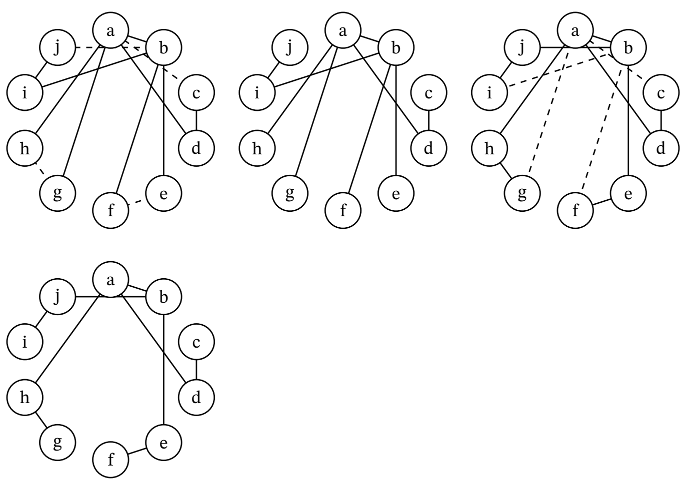


Figure 224: Removing Four Edges from Graph *D*

VIDEO

Spanning Trees in Graph Theory

Example 6 **Removing Edges to Find Spanning Trees**

Use the graph in to answer each question.

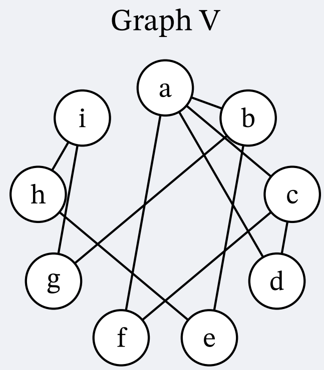


Figure 225: Graph *V*

1. Determine the number of edges that must be removed to reveal a spanning tree.
2. Name all the undirected cycles in Graph *V*.
3. Find two distinct spanning trees of Graph *V*.

Solution

1. Graph *V* has nine vertices so a spanning tree for the graph must have 8 edges. Since Graph *V* has 11 edges, 3 edges must be removed to reveal a spanning tree.
2. (a, c, d), (a, c, f), (a, d, c, f), and (b, e, h, i, g)
3. To find the first spanning tree, remove edge *ac*, which will break up both of the triangles, remove edge *cf*, which will break up the quadrilateral, and remove *be*, which will break up the pentagon, to give us the spanning tree shown.

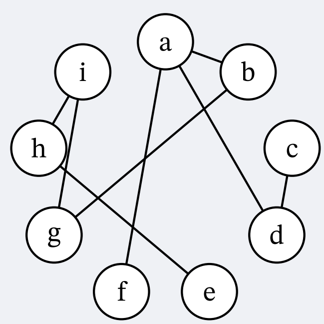


Figure 226: Spanning Tree Formed Removing *ac, cf*, and *be*

To find another spanning tree, remove *ad*, which will break up (a, c, d) and (a, d, c, f), remove *af* to break up (a, c, f), and remove *hi* to break up (b, e, h, i, g). This will give us the spanning tree in .

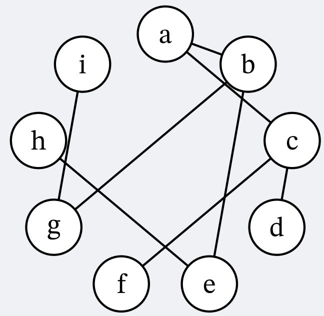


Figure 227: Spanning Tree Formed Removing *ad*, *af*, and *hi*

WHO KNEW?

Chains of Affection

Here is a strange question to ask in a math class: Have you ever dated your ex's new partner's ex? Research suggests that your answer is probably no. When researchers Peter S. Bearman, James Moody, and Katherine Stovel attempted to compare the structure of heterosexual romantic networks at a typical midwestern high school to simulated networks, they found something surprising. The actual social networks were more like spanning trees than other possible models because there were very few short cycles. In particular, there were almost no four-cycles.

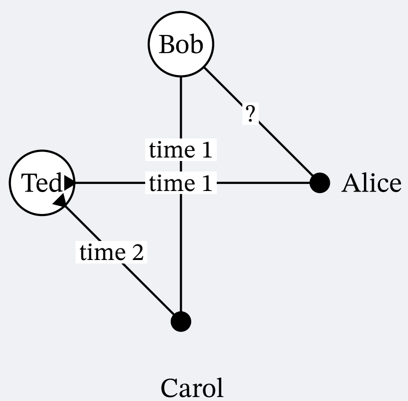


Figure 228: Chains of Affection

"...the prohibition against dating (from a female perspective) one's old boyfriend's current girlfriend's old boyfriend – accounts for the structure of the romantic network at [the highschool]."

In their article "Chains of Affection: The Structure of Adolescent Romantic and Sexual Networks," the researchers went on to explain the implications for the transmission of sexually transmitted diseases. In particular, social structures based on tree graphs are less dense and more likely to fragment. This information can impact social policies on disease prevention. (Peter S. Bearman, James Moody, and Katherine Stovel, "Chains of Affection:

The Structure of Adolescent Romantic and Sexual Networks," *American Journal of Sociology* Volume 110, Number 1, pp. 44-91, 2004)

Kruskal's Algorithm

In many applications of spanning trees, the graphs are weighted and we want to find the spanning tree of least possible weight. For example, the graph might represent a computer network, and the weights might represent the cost involved in connecting two devices. So, finding a spanning tree with the lowest possible total weight, or **minimum spanning tree**, means saving money! The method that we will use to find a minimum spanning tree of a weighted graph is called **Kruskal's algorithm**. The steps for Kruskal's algorithm are:

Step 1: Choose any edge with the minimum weight of all edges.

Step 2: Choose another edge of minimum weight from the remaining edges. The second edge does not have to be connected to the first edge.

Step 3: Choose another edge of minimum weight from the remaining edges, but do not select any edge that creates a cycle in the subgraph you are creating.

Step 4: Repeat step 3 until all the vertices of the original graph are included and you have a spanning tree.

VIDEO

Use Kruskal's Algorithm to Find Minimum Spanning Trees in Graph Theory

Example 7 **Using Kruskal's Algorithm**

A computer network will be set up with six devices. The vertices in the graph in represent the devices, and the edges represent the cost of a connection. Find the network configuration that will cost the least. What is the total cost?

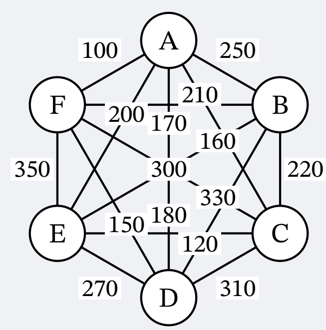


Figure 229: Graph of Network Connection Costs

Solution

A minimum spanning tree will correspond to the network configuration of least cost. We will use Kruskal's algorithm to find one. Since the graph has six vertices, the spanning tree will have six vertices and five edges.

Step 1: Choose an edge of least weight. We have sorted the weights into numerical order. The least is \$100. The only edge of this weight is edge *AF* as shown.

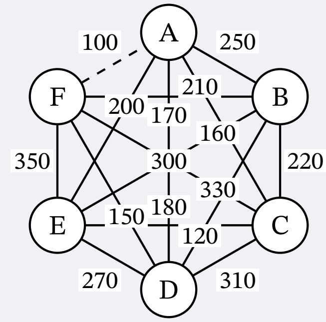


Figure 230: Step 1 Select Edge *AF*

Step 2: Choose the edge of least weight of the remaining edges, which is *BD* with \$120. Notice that the two selected edges do not need to be adjacent to each other as shown.

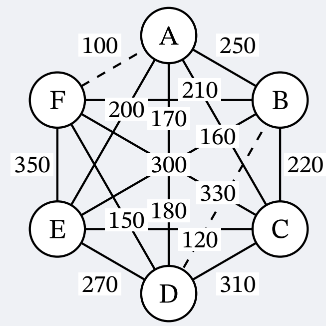


Figure 231: Step 2 Select Edge *BD*

Step 3: Select the lowest weight edge of the remaining edges, as long as it does not result in a cycle. We select *DF* with \$150 since it does not form a cycle as shown.

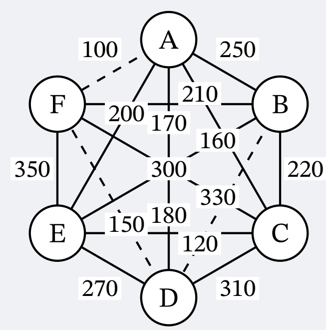


Figure 232: Step 3 Select Edge *DF*

Repeat Step 3: Select the lowest weight edge of the remaining edges, which is *BE* with \$160 and it does not form a cycle as shown. This gives us four edges so we only need to repeat step 3 once more to get the fifth edge.

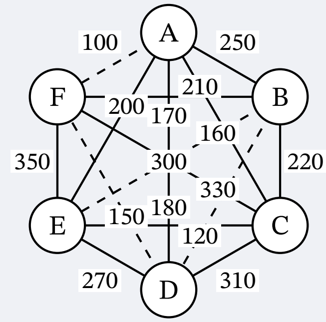


Figure 233: Repeat Step 3 Select Edge *DF*

Repeat Step 3: The lowest weight of the remaining edges is \$170. Both *BF* and *CE* have a weight of \$170, but *BF* would create cycle (b, d, f) and there cannot be a cycle in a spanning tree as shown.

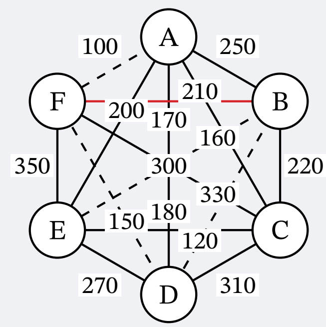


Figure 234: Repeat Step 3 Do Not Select Edge *BF*

So, we will select *CE*, which will complete the spanning tree as shown.

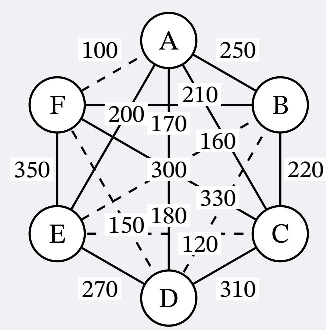


Figure 235: Repeat Step 3 Select Edge *CE*

The minimum spanning tree is shown. This is the configuration of the network of least cost. The spanning tree has a total weight of \$100 + \$120 + \$150 + \$160 + \$170 = \$700, which is the total cost of this network configuration.

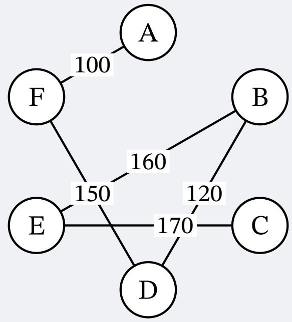


Figure 236: Final Minimum Spanning Tree

Key Terms

- acyclic
- tree
- forest
- path graph or linear graph
- star tree
- root
- starlike tree
- caterpillar tree
- lobster tree
- spanning tree
- minimum spanning tree

Key Concepts

- A brute force algorithm always finds the ideal solution but can be impractical whereas a greedy algorithm is efficient but usually does not lead to the ideal solution.
- A Hamilton cycle of lowest weight is a solution to the traveling salesperson problem.
- The brute force method finds a Hamilton cycle of lowest weight in a complete graph.
- The nearest neighbor method is a greedy algorithm that finds a Hamilton cycle of relatively low weight in a complete graph.

Videos

- The Problem in *Good Will Hunting* by Numberphile
- Spanning Trees in Graph Theory
- Use Kruskal's Algorithm to Find Minimum Spanning Trees in Graph Theory

Formulas

- The number of edges in a tree graph with n vertices is $n - 1$. A connected graph with n vertices and $n - 1$ edges is a tree graph.

Projects

Everyone Gets a Turn! – Graph Colorings

Let's put your knowledge of graph colorings to work! Your task is to plan a field day following these steps.

1. Select between seven and ten activities for your field day. You can look online for ideas.
2. Create a survey asking for the participants to select the three to five events in which they would most like to participate. Survey between seven and ten people.
3. Use the results of your survey to create a graph in which each vertex represents one of the events. A pair of vertices will be adjacent if there is at least one participant who would like to participate in both events.
4. Find a minimum coloring for the graph. Explain how you found it and how you know the chromatic number of the graph.
5. Use your solution to part d to determine the minimum number of timeslots you must use to ensure that everyone has the opportunity to participate in their top three events.
6. Find the complement of the graph you created. Explain what the edges in this graph represent.

   A Beautiful Day in the Neighborhood – Euler Circuits

   Let's apply what you have learned to the community in which you live. Using resources such as your county's property appraiser's website, create a detailed graph of your neighborhood in which vertices represent turns and intersections. Represent a large enough part of your community to include no fewer than 10 intersections or turns. Then use your graph to answer the following questions.

   1. Label the edges of your graph.
7. Determine if your graph is Eulerian. Explain how you know. If it is not, eulerize it.
8. Find an Euler circuit for your graph. Give the sequence of vertices that you found.
9. What does the Euler circuit you found in part c represent for your community?
10. Describe an application for which this Euler circuit might be used.

    Dream Vacation – Hamilton Cycles and Paths

    Where in the world would you like to travel most: the Eiffel Tower in Paris, a Broadway musical in New York city, a bike tour of Amsterdam, the Tenerife whale and dolphin cruises in the Canary Islands, the Giza Pyramid in Cairo, or maybe the Jokhang Temple in Tibet? Let's plan your dream vacation!

    1. Which four destinations are at the top of your bucket list?

11. Draw a complete weighted graph with five vertices representing the four destinations and your home city, and the weights representing the cost of travel between cities.
12. Use a website (such as Travelocity) to find the best airfare between each pair of cities. List the airlines and flight numbers along with the prices. Include cost for ground transportation from the nearest airport if there is no airport at the destination you want to visit.
13. Use the nearest neighbor algorithm to find a Hamilton cycle of low weight beginning and ending in your hometown. What is the weight of this circuit and what does it represent?
14. Use the brute force method to find a Hamilton cycle of lowest weight beginning and ending in your hometown. What is the weight of this circuit? Is it the same or different from the weight of the Hamilton cycle you found in Exercise 4?
15. Suppose that instead of returning home, you planned to move to your favorite location on the list, but you wanted to stop at the other three destinations once along the way. Where would you move? List all Hamilton paths between your hometown and your favorite location.
16. Find the weights of all the Hamilton paths you found in Exercise 6.