Introduction

In this lesson, you will learn about an *algorithm*, and another theorem due to Euler that settles the Königsberg bridges problem for any graph.

Goals

At the end of this assignment, a student should be able to:

- state and use a theorem by Euler about special cycles on graphs,
- discuss what an algorithm is, and
- implement an algorithm that finds Eulerian cycles.

A student might also be able to:

- Solve a challenge that generalizes the Königsberg bridges puzzle.

Reading and Questions for Graphs Day 07

A Necessary Condition for Eulerian Graphs

What would it look like if we could solve the Königsberg bridges problem?

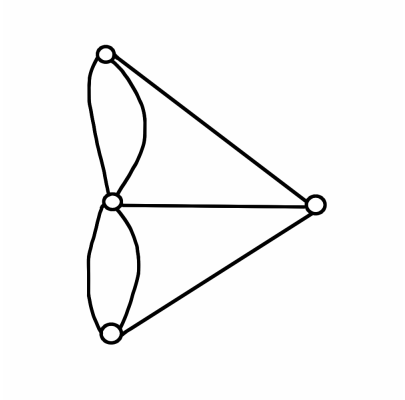

Figure 1: The Königsberg Bridges graph

How about for another graph? What would we need to be true for a graph for there to be an Eulerian cycle on that graph?

Well, an Eulerian cycle has to be a cycle, first of all. Imagine walking along this cycle. At each vertex, the cycle has to enter and leave the same number of times. But to be an *Eulerian* cycle, each edge has to be used exactly once. So, at each vertex there must be an even number of edges. That means that each vertex must have even degree.

Theorem. If a connected graph has an Eulerian cycle, then all of its vertices have even degree.

This is a useful thing to know. Already it helps us see that the Königsberg bridges problem doesn't have a solution: *all* of its vertices have odd degree. The people of Königsberg didn't even have a chance.

Exercise 1. Use Theorem to decide if the graphs $K_{3,3}$, K_4, and K_6 have an Eulerian cycle.

Exercise 2. Use Theorem to settle the Five Rooms Puzzle.

Exercise 3. Use Theorem to design two interesting graphs that don't have Eulerian cycles. Can you make it "not obvious?"

The kind of thing in Theorem is called a *necessary* condition. This is because if a graph is going to be Eulerian, it *has* to satisfy this condition, too. It is necessary.

A Sufficient Condition

Now, what can we say if a graph has the property that all of its vertices have even degree? It is necessary for the graph to be Eulerian. Is it also *sufficient*? That is, is the condition strong enough to not only rule out graphs which are not Eulerian, but also to guarantee that a graph is Eulerian?

It turns out to be true. The necessary condition is also sufficient. We can improve our theorem to this one:

Theorem (Leonhard Euler, 1736)**.** A connected graph has an Eulerian cycle if, and only when the degree of each of its vertices is even. That is, the two statements below are equivalent:

- The graph G is Eulerian.
- The for every vertex v of the graph G, the degree of v is even.

This result is widely credited with being the first result from graph theory.

Exercise 4. Use Theorem to decide if the graphs K_3 and K_5 are Eulerian or not. In general, for which values of n is the graph K_n an Eulerian graph?

Theorem is pretty amazing. Using it we can exactly answer the question of "is this graph Eulerian," not by looking for paths, but by just counting edges at each vertex.

Algorithms, and a Proof of Euler's Theorem

The best kind of solution to the (general) Königsberg bridge problem would be a sure-fire method for finding an Eulerian cycle. What would that entail? It would be nice if we could find a simple set of instructions, a recipe, for creating the Eulerian cycle. Those instructions should be really simple to follow, too, and we should be sure that they would always work.

Such a process is called an *algorithm*. There are algorithms everywhere in your life, whether you realize it or not. At some point you learned how to do the *long division* algorithm. If you are lucky, maybe you learned to do *Russian peasant multiplication* at some point.

Above, we saw an argument for why the condition on vertices is necessary. Now it is time to see an argument (a proof!) for why the condition on vertices is sufficient. The proof consists of an algorithm for for actually constructing the Eulerian cycle.

As you read this proof, please work through it for the example of the graph $G = K_5$.

Proof of Sufficiency. Suppose that G is a connected graph which has each of its vertices of even degree.

Label the vertices with counting numbers in any way you desire, but be sure to use an unbroken string of them:

$$1, 2, 3, \ldots, V-1, V,$$

where V is the total number of vertices.

1. Start at the vertex labeled 1.
2. Follow an unused edge which joints 1 to the smallest numbered vertex available. Mark this edge.
3. At this new vertex, follow whichever available unused edge leads to the smallest numbered vertex. Mark the edge as you cross it.
4. Continue until you come back to vertex 1.
5. If there are more edges attached to vertex 1 which have not been used, repeat the process above using only edges you have not previously marked.
6. Put these cycles together by attaching them at the vertex 1.
7. When all of the edges attached to vertex 1 have been used. Go to the smallest numbered vertex which has unused edges and do the process above for this vertex.
8. At some point, the process will stop, because each step involves decreasing the number of unused edges, so you eventually run out. When this happens, you will have decomposed the graph into some number of cycles. Because the graph is connected, each cycle must share a vertex with at least one other cycle. Choose such points and splice the cycles together.

□

Exercise 5. Follow the steps of the proof to try to find Eulerian cycles in the graphs you made in Exercise 3. Where does the process break?

Exercise 6. Why is it important that the graph have vertices of even degree? How might the algorithm break if you have a vertex of odd degree?

Exercise 7. Design a connected graph with at least seven vertices and a largish number of edges. Make sure that each vertex has even degree. Apply the algorithm to your graph to find an Eulerian cycle.

A New Challenge

Here is a variant of the Königsberg bridges puzzle: Suppose that you do not require your path to be a cycle. That is, we don't necessarily want to start and end at the same place. But we do want to traverse each edge exactly once. Can such a path be constructed?

In general, suppose G is a connected graph. What must be true about the graph in order to find a path in the graph which uses each edge exactly once? Find a necessary condition.

Is that condition sufficient, too? How much of the work and ideas above can be reused?