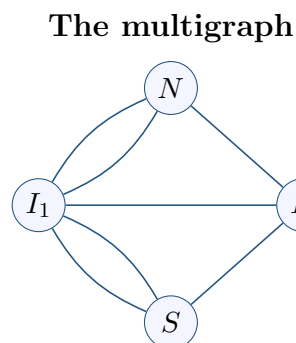
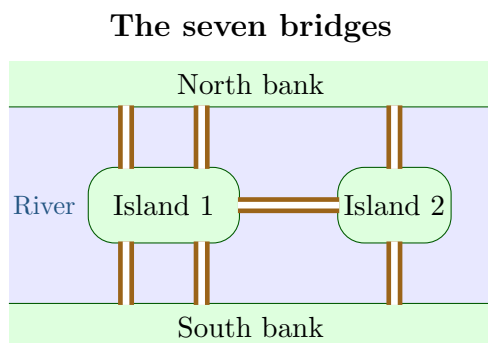


# Lecture 14: Graphs and Eulerian Circuits

Lecturer and scribe: Karthik Gajulapalli

## 1 The Seven Bridges of Königsberg

Euler's famous bridge problem begins with four pieces of land: a north bank, a south bank, and two islands. Seven bridges connect them.



There are two bridges from each bank to Island 1, one bridge from each bank to Island 2, and one bridge between the islands.

**Euler's question.** Can we take a walk that crosses *every bridge exactly once*?

We may revisit a piece of land, but we may not reuse a bridge. The starting and ending points are not prescribed. A particularly natural version asks for a walk that also returns to its starting point.

Trying routes quickly becomes frustrating: a choice that looks useful now may leave an inaccessible bridge at the end. How could we show that *no* choice of route works, without trying every possibility?

### 1.1 Keep Only the Connections

Recall our earlier pictures of graphs. Replace each piece of land by a **vertex** and each bridge by an **edge**. Distances, angles, and the shapes of the riverbanks do not affect the question. Only which pieces of land are connected matters.

The bridge graph is a **multigraph**: it has four vertices and seven edges, with distinct **parallel edges** joining the same pair of vertices. We must keep both bridges between a bank and Island 1: crossing one bridge does not use the other.

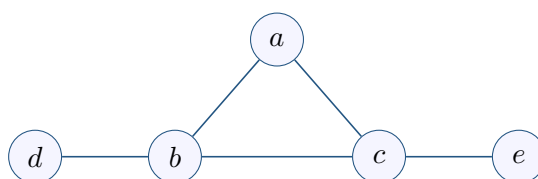
**The key observation.** Every time we enter a region and later leave it, we use a pair of bridges. This suggests that the *parity* of the number of bridges at each region should matter. We will turn this observation into a theorem, and the proof will also give an algorithm for finding a route whenever one exists.

## 2 Graphs and Vertex Degrees

All graphs in this lecture are **finite and undirected**. A graph  $G = (V, E)$  consists of a set  $V$  of **vertices** and a collection  $E$  of **edges**, each joining two vertices. Vertices joined by an edge are **adjacent**; an edge is **incident** to its endpoints.

A **simple graph** has no loops and no parallel edges. Its edges are unordered pairs of distinct vertices:  $\{u, v\} = \{v, u\}$ . A **multigraph** may have several distinct edges with the same endpoints, as in the bridge problem. A **loop** is an edge from a vertex to itself. Our drawings have no loops, but the degree and Eulerian-circuit results also allow them.

### 2.1 An Example



Here

$$V = \{a, b, c, d, e\}, \quad E = \{\{a, b\}, \{a, c\}, \{b, c\}, \{b, d\}, \{c, e\}\}.$$

The drawing represents the connections; moving vertices or bending edges does not change the graph. A crossing of two drawn edges is *not* a vertex unless it is explicitly marked as one.

### 2.2 Degree

The **degree** of a vertex  $v$ , written  $\deg(v)$ , is the number of edge ends incident to  $v$ . Each parallel edge contributes separately, and a loop contributes *two*. In a simple graph, this is exactly the number of neighbors of  $v$ .

For the graph above,

$$\begin{array}{c|ccccc} v & a & b & c & d & e \\ \hline \deg(v) & 2 & 3 & 3 & 1 & 1 \end{array} \quad \text{and} \quad \sum_{v \in V} \deg(v) = 10 = 2|E|.$$

Degree is a property of a *vertex*; a graph need not have a single common degree. A vertex of degree 0 is called **isolated**.

In the bridge graph, let  $N, S, I_1, I_2$  denote the two banks and islands. Then

$$\deg(N) = 3, \quad \deg(S) = 3, \quad \deg(I_1) = 5, \quad \deg(I_2) = 3.$$

All four degrees are odd, and their sum is  $14 = 2 \cdot 7$ .

### 3 Three Degree Lemmas

**Lemma 1** (Handshake lemma). *For every finite undirected graph,*

$$\sum_{v \in V} \deg(v) = 2|E|.$$

*Proof.* Count the ends of all edges in two ways. Counting at each vertex gives  $\sum_{v \in V} \deg(v)$ . Counting edge by edge gives  $2|E|$ , since each edge has two ends. A loop has both ends at the same vertex and still contributes two.  $\square$

The name comes from a party: if an edge records a handshake between two people, then each handshake contributes one to each person's count.

**Lemma 2.** *Every finite undirected graph has an even number of vertices of odd degree.*

*Proof.* Let  $O$  be the set of vertices of odd degree. By the handshake lemma,

$$2|E| = \sum_{v \notin O} \deg(v) + \sum_{v \in O} \deg(v).$$

The left side and the first sum are even, so the second sum is even. A sum of  $|O|$  odd integers has the same parity as  $|O|$ . Hence  $|O|$  is even.  $\square$

In particular, a graph cannot have exactly one odd-degree vertex. The bridge graph has four, which is consistent with the lemma.

**Lemma 3.** *Every finite **simple** graph with  $n \geq 2$  vertices has at least two vertices of the same degree.*

*Proof.* Each degree belongs to  $\{0, 1, \dots, n-1\}$ . However, degrees 0 and  $n-1$  cannot both occur: a vertex of degree  $n-1$  is adjacent to every other vertex, so there cannot be an isolated vertex.

Thus the degrees all lie in  $\{0, \dots, n-2\}$  or all lie in  $\{1, \dots, n-1\}$ . Either way,  $n$  vertices have at most  $n-1$  possible degree values. The pigeonhole principle gives two vertices with the same degree.  $\square$

**Why simplicity matters.** With one edge between  $x, y$  and two parallel edges between  $y, z$ , the degrees are 1, 3, 2, all different. The first two lemmas allow parallel edges; the third does not.

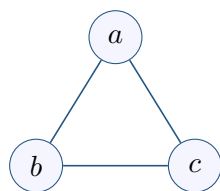
## 4 Eulerian Circuits

A **walk** follows edges from vertex to vertex. A **trail** is a walk with no repeated edge; vertices may repeat. A **path** has no repeated vertex. A graph is **connected** if every pair of vertices can be joined by a path.

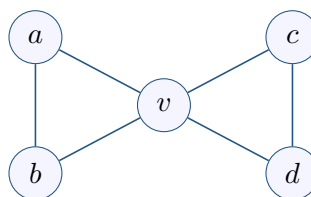
A **closed trail** starts and ends at the same vertex. An **Eulerian circuit** is a closed trail that uses every edge of the graph exactly once. A graph with an Eulerian circuit is called **Eulerian**.

**Edges, not vertices, must be used exactly once.** We may pass through a vertex several times. In a multigraph, we must distinguish the parallel edges when specifying the route.

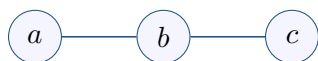
(a) A triangle



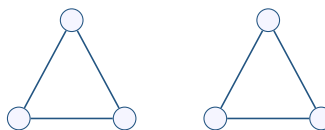
(b) Two triangles sharing a vertex



(c) A path



(d) Two disjoint triangles



- (a) The triangle has Eulerian circuit  $a, b, c, a$ .
- (b) The two triangles sharing  $v$  have Eulerian circuit  $v, a, b, v, c, d, v$ . The vertex  $v$  is visited repeatedly.
- (c) The path  $a, b, c$  uses every edge once but is not closed. There is no Eulerian circuit: after reaching either endpoint, returning would require reusing its only edge.
- (d) The two disjoint triangles have degree 2 at every vertex, but no single walk can use edges in both components.

The examples suggest two requirements: all degrees must be even, and the edges must belong to one connected part of the graph. For connected graphs, even degrees will turn out to be sufficient.

## 5 Euler's Theorem and the Construction

**Theorem 1** (Euler). *A finite connected undirected graph has an Eulerian circuit if and only if every vertex has even degree.*

The theorem allows parallel edges and loops. For the connected graph with one vertex and no edges, we regard the length-zero walk as an Eulerian circuit.

### 5.1 An Eulerian Circuit Forces Even Degrees

*Proof of the forward direction.* Suppose  $G$  has an Eulerian circuit. Read the circuit cyclically: every arrival at a vertex is paired with the next departure from that vertex. At the starting vertex, the final arrival is paired with the initial departure.

Because the circuit uses every edge exactly once, these pairs account for every incident edge end exactly once. Thus the number of incident edge ends at each vertex is twice the number of arrival–departure pairs there, and is even. A loop contributes an arrival and a departure at the same vertex, so it also contributes two. The degree-zero case is even as well.  $\square$

For the converse, we need to *build* a circuit. The idea is to find a closed trail, then insert more closed trails until every edge has been used.

### 5.2 Hierholzer's Algorithm

Assume  $G$  is connected and every degree is even. Mark every edge unused. If there are no edges, return the length-zero circuit. Otherwise:

1. **Build a first closed trail.** Start at any vertex  $s$ . Repeatedly follow any unused incident edge, marking it used, until no unused edge is available at the current vertex. Call the resulting trail  $T$ .
2. **Find an insertion point.** If unused edges remain, choose a vertex  $v$  on  $T$  incident to an unused edge.
3. **Build another closed trail.** Starting at  $v$ , follow unused edges until no further unused edge is available. Call this new trail  $C$ .
4. **Splice and repeat.** At an occurrence of  $v$  in  $T$ , insert the whole of  $C$ , then resume  $T$ . Repeat Steps 2–4 until every edge is used, and return  $T$ .

We will prove that each new trail ends where it began and that an insertion point always exists while unused edges remain.

## 6 The Class Example

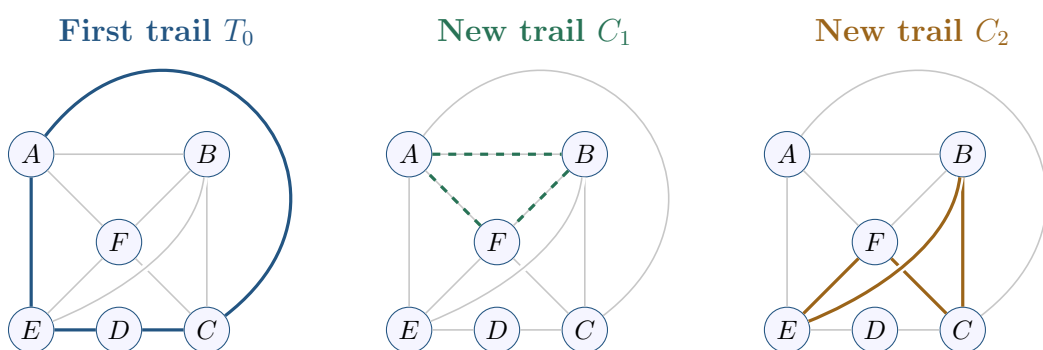
The class graph is a simple graph with vertices  $A, B, C, D, E, F$  and eleven undirected edges:

$$E = \{AB, AE, AF, AC, BF, BE, BC, EF, FC, ED, CD\}.$$

Here  $AB = BA$  denotes the unordered pair  $\{A, B\}$ . The graph is connected, and every degree is even:

| $v$       | $A$ | $B$ | $C$ | $D$ | $E$ | $F$ |
|-----------|-----|-----|-----|-----|-----|-----|
| $\deg(v)$ | 4   | 4   | 4   | 2   | 4   | 4   |

$$\sum_v \deg(v) = 22 = 2 \cdot 11.$$



Each panel highlights one trail in the full graph. Unmarked crossings are not vertices.

**First trail.** Start at  $D$  and choose

$$T_0 = D, E, A, C, D.$$

Both edges at  $D$  are now used, so the traversal stops. Seven edges remain unused; this closed trail is not yet an Eulerian circuit.

**First splice, at  $A$ .** The unused edges at  $A$  give the closed trail  $C_1 = A, B, F, A$ . There are then no unused edges at  $A$ . Insert  $C_1$  into  $T_0$ :

$$T_1 = D, E, \underbrace{A, B, F, A}_{\text{insert } C_1 \text{ at } A}, C, D.$$

**Second splice, at  $B$ .** The unused edges are  $BE, EF, FC, BC$ . Starting at  $B$  gives the closed trail  $C_2 = B, E, F, C, B$ . Insert it at  $B$  in  $T_1$ :

$$T_2 = D, E, A, \underbrace{B, E, F, C, B}_{\text{insert } C_2 \text{ at } B}, F, A, C, D.$$

This is an Eulerian circuit: it starts and ends at  $D$  and uses

$$DE, EA, AB, BE, EF, FC, CB, BF, FA, AC, CD,$$

each exactly once. Vertices may recur; edges may not.

## 7 Why the Construction Works

*Proof of the reverse direction.* Suppose  $G$  is connected and every vertex has even degree. The case with no edges is immediate, so assume that  $G$  has an edge. We verify the steps of Hierholzer's algorithm.

**1. A traversal cannot get stuck away from its start.** Consider any graph in which every degree is even. Start at  $s$  and follow unused edges until stuck; this process stops because there are finitely many edges. If it ends at  $w \neq s$ , then at  $w$  there is one more arrival than departure. The trail has therefore used an odd number of incident edge ends at  $w$ . Since the total degree at  $w$  is even, at least one incident edge is still unused. This contradicts being stuck. Hence the trail ends at  $s$  and is closed.

**2. The unused graph still has even degrees.** A closed trail uses an even number of incident edge ends at every vertex: arrivals and departures pair up. Deleting its edges therefore subtracts an even number from every degree. All degrees in the remaining graph are even. Consequently, Step 1 applies to every new traversal of unused edges, even if the unused graph is disconnected.

**3. Unused edges give an insertion point on  $T$ .** Suppose an unused edge remains. If one of its endpoints lies on  $T$ , we already have the required vertex. Otherwise, connectedness of the original graph gives a path from  $T$  to an endpoint of that edge. The first edge of this path that leaves the vertices of  $T$  must be unused: every used edge belongs to  $T$ , so both of its endpoints lie on  $T$ . Its endpoint on  $T$  is the required insertion point.

**4. Splicing preserves a closed trail and makes progress.** The new closed trail  $C$  begins and ends at the chosen vertex  $v$ . We can follow  $T$  to  $v$ , traverse  $C$ , and continue along  $T$ . The result is closed, and no edge repeats because  $C$  uses only edges that were unused by  $T$ . Each splice adds at least one edge.

Since  $G$  has finitely many edges, the process terminates. By Step 3, it can terminate only when all edges have been used. The final closed trail is an Eulerian circuit.  $\square$

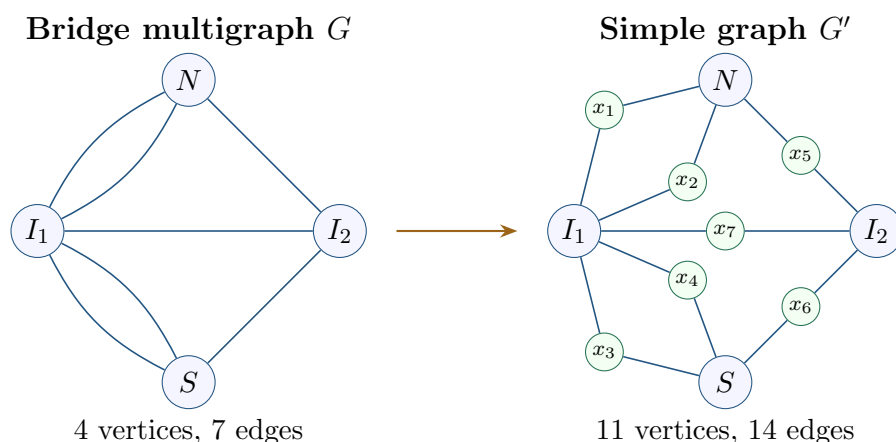
## 8 From Multigraphs to Simple Graphs

We can replace the bridge multigraph by a simple graph without changing whether a closed walk using every edge exactly once exists.

**The construction: subdivide each edge.** For every non-loop edge  $e$  joining  $u$  to  $v$ , introduce a *fresh* vertex  $x_e$  and replace  $e$  by the two-edge route

$$u - x_e - v.$$

Use a different new vertex for *each* edge, including parallel edges. Thus two edges between  $u, v$  become  $u, x_1, v$  and  $u, x_2, v$ . The resulting graph has no parallel edges: the new endpoints distinguish the replacement edges.



**Degrees and connectedness are preserved.** Each original vertex keeps its degree, and each new vertex has degree 2. Replacing edges by routes also preserves connectedness. For the bridge graph, the four original degrees remain 3, 3, 5, 3, and the seven new vertices all have degree 2.

**Why the reduction preserves closed edge-covering walks.** Given such a walk in the multigraph, replace every traversal of  $e = uv$  by  $u, x_e, v$  (or the reverse). This uses each replacement edge exactly once. Conversely, start a closed edge-covering walk in the new graph at an original vertex. Whenever it enters a new degree-2 vertex, it must leave along the other edge. Compress each whole replacement route back to its original edge. The result is a closed walk using every original edge exactly once. Thus the multigraph has an Eulerian circuit if and only if the resulting simple graph does.

**If loops are allowed.** Replace a loop at  $u$  by  $u, x_e, y_e, u$ , using two fresh vertices. This triangle has no parallel edges; it contributes two to  $\deg(u)$ , and both new degrees are 2. The same expansion and compression argument applies. Subdividing a loop only once would leave two parallel edges, so two new vertices are needed for this case.

## 8.1 Back to the Seven Bridges

The bridge graph is connected but has degrees 3, 3, 5, 3, so it has no Eulerian circuit. Could we cross every bridge once by allowing different starting and ending regions?

In any trail using all edges, every vertex other than its two possible endpoints has its arrivals paired with departures and therefore has even degree. Thus such a trail can have at most two odd-degree vertices. The bridge graph has four, so *even an open walk crossing every bridge exactly once is impossible*.