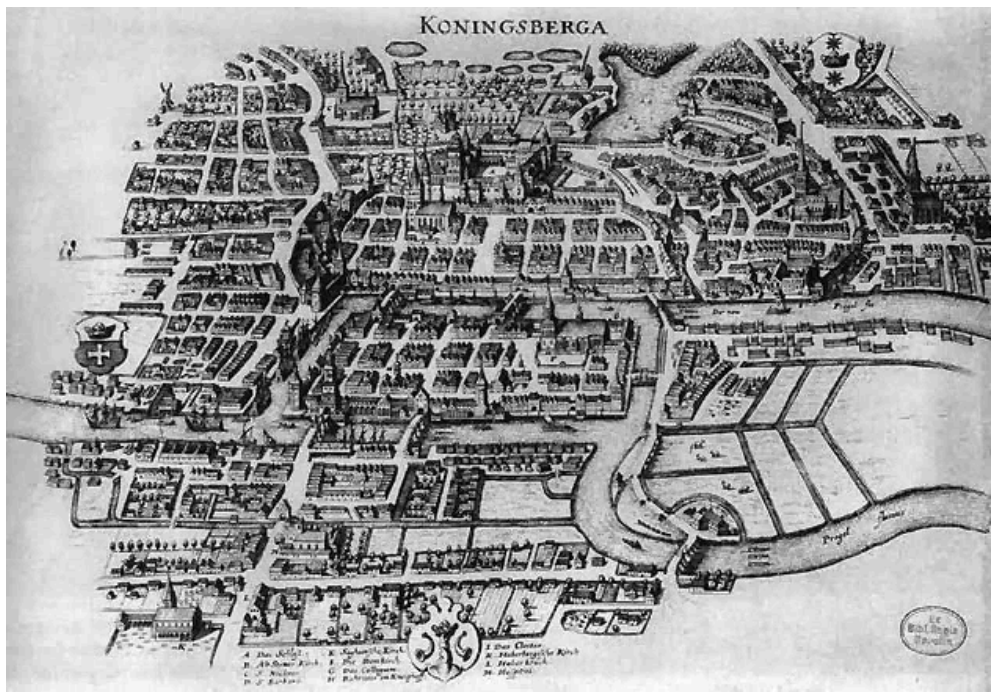
Module 01: Königsberg Bridge Puzzle

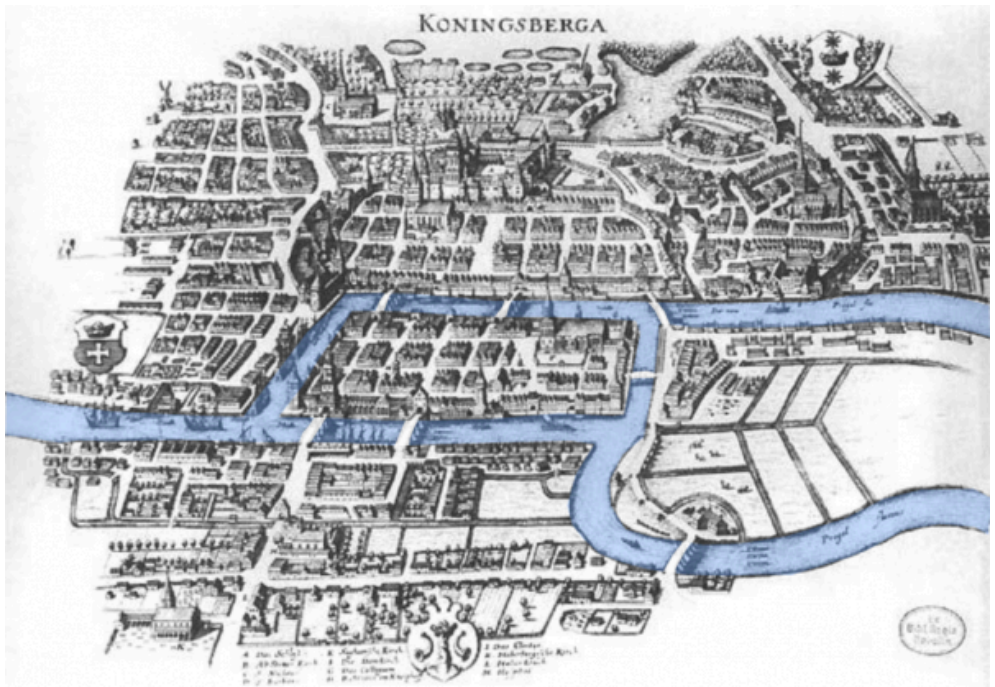
Advanced Topics in Network Science

Sadamori Kojaku

skojaku@binghamton.edu



The Königsberg Bridge Puzzle 🌉



- 18th century puzzle in Königsberg, Prussia (now Kaliningrad, Russia) 🇩🇪
- City had 7 bridges connecting 2 islands and mainland 🏙️
- **Challenge**: Find a route that crosses each bridge exactly once 🚶

Your turn! 

Find a route that crosses each bridge exactly once

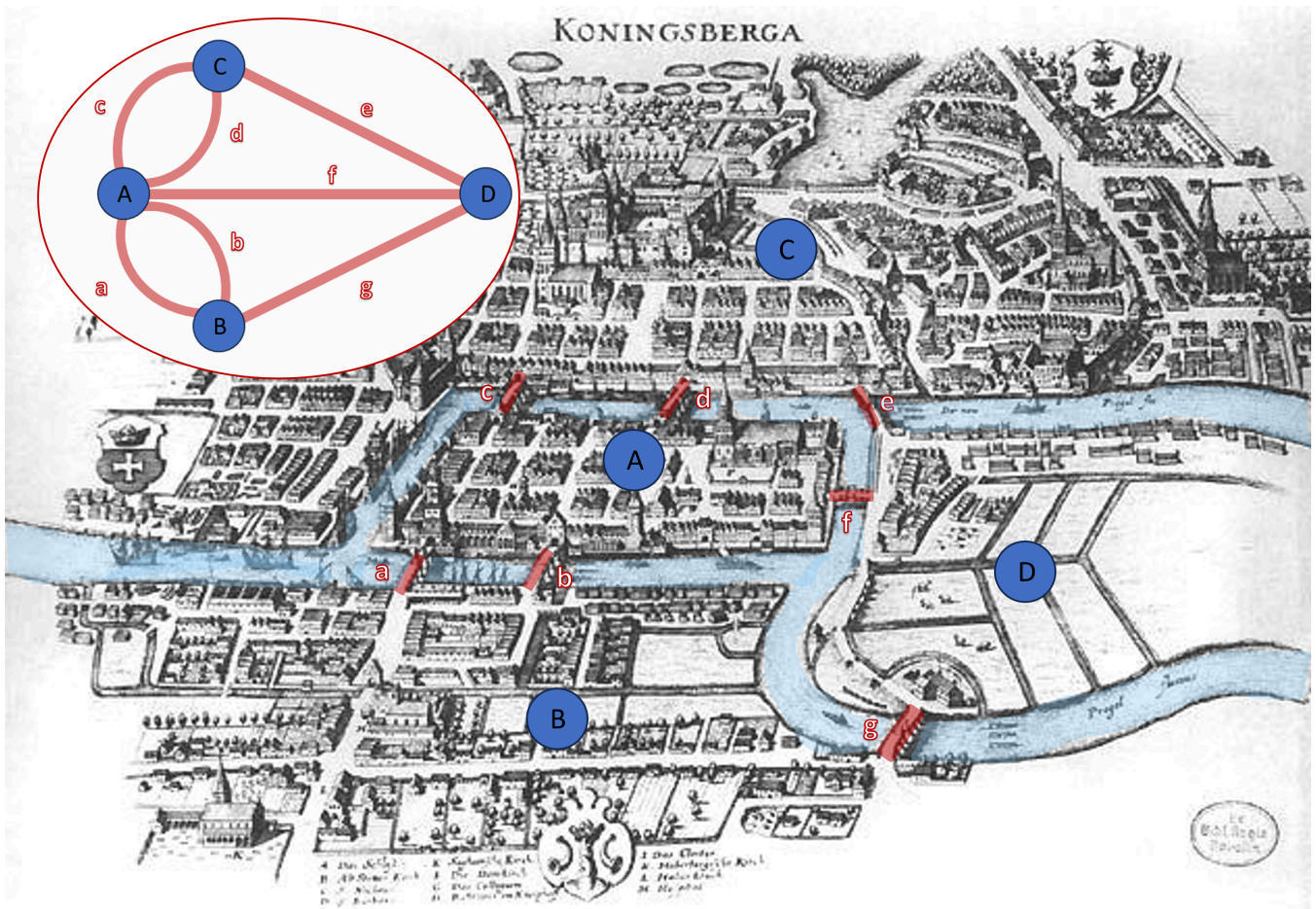
Take 10 mins - Think about your strategy through the pen & paper exercise

pen-and-paper worksheet

Euler's Revolutionary Approach 🧠

What if we ignore all the physical details? 🙂

What do you see in this transformation? *What's the key insight?*



The Breakthrough Insight ✨

Euler realized: Only connections matter, not physical details!

- Landmasses → **dots** (nodes)
- Bridges → **lines** (edges)

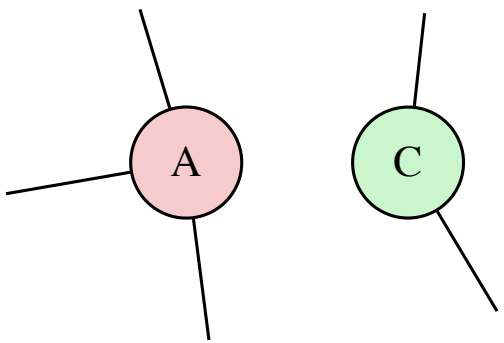
Abstraction is the a cornerstone of Network Science

This was the birth of graph theory and network science!

Euler's Reasoning 🧮

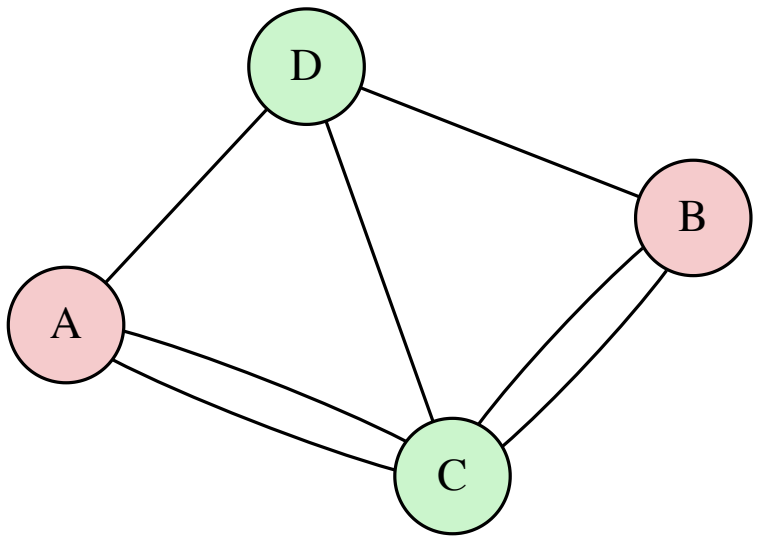
If you walk through a node, what happens?

- Even number of edges: Enter/leave perfectly ✅
- Odd number: One edge "left over" ❌
- Question: Where can those "leftover" edges be?
- Only at start/end points!



Euler's theorem says:

- 0 odd nodes → We can start and end at the same node while passing through all edges exactly once ✅
- 2 odd nodes → We can start and end at different nodes while passing through all edges exactly once ✅
- More than 2 → Not possible to pass through all edges exactly once ❌



Königsberg Bridge Problem - Seven bridges connecting four land areas

What's your verdict?

Euler Path Theorem

An Euler path exists if and only if:

1. **The graph is connected** (can reach any node from any other)
2. **Either:**
    - All nodes have even degree (Euler circuit), OR
    - Exactly two nodes have odd degree (Euler path)

Königsberg verdict: 4 odd-degree nodes → **IMPOSSIBLE!**

Aftermath

The story takes a sobering turn during World War II. In 1944, Königsberg was heavily bombed by Allied forces, and later captured by the Soviet Union. Two of the seven historic bridges were destroyed in the bombardment.

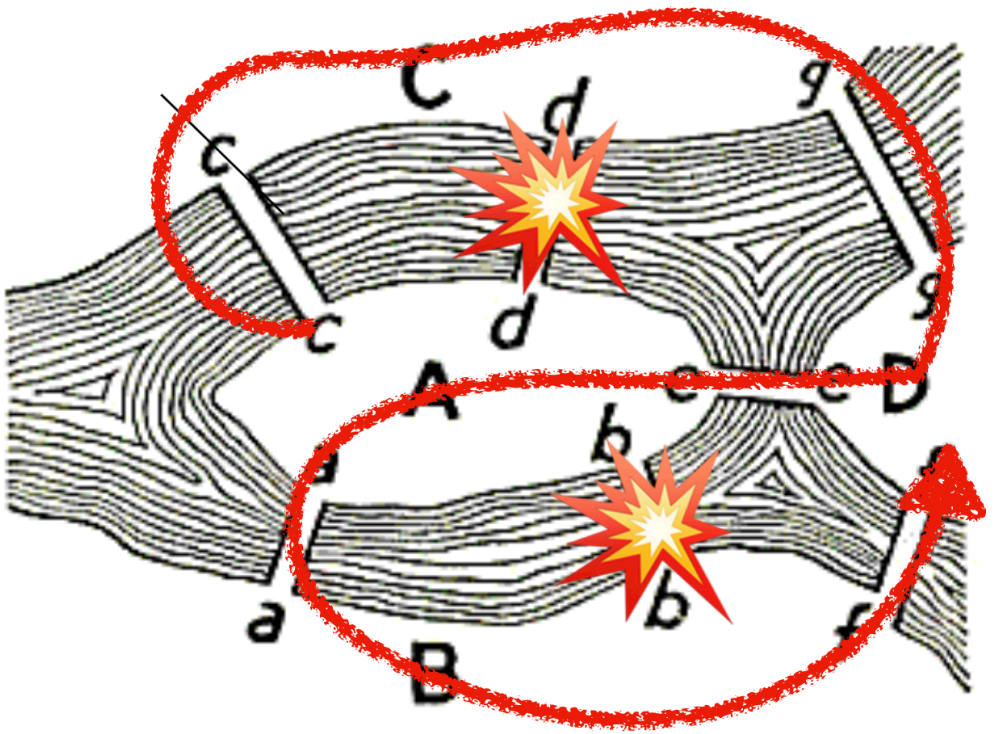


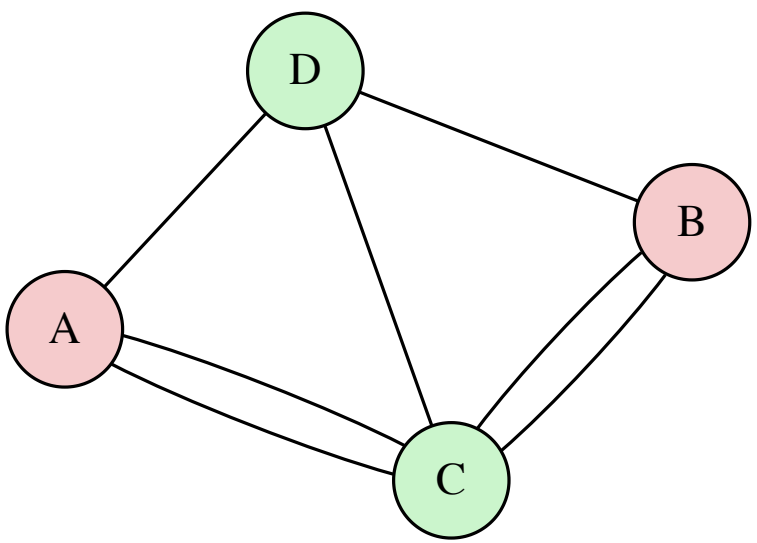
FIGURE 98. *Geographic Map: The Königsberg Bridges.*

Question:

In the previous question, we learned the condition for the possibility to cross each bridge exactly once.

Now, let's also add a new condition: *Return to the starting point*.

How does this change the condition for the possibility?



Königsberg Bridge Problem - Seven bridges connecting four land areas

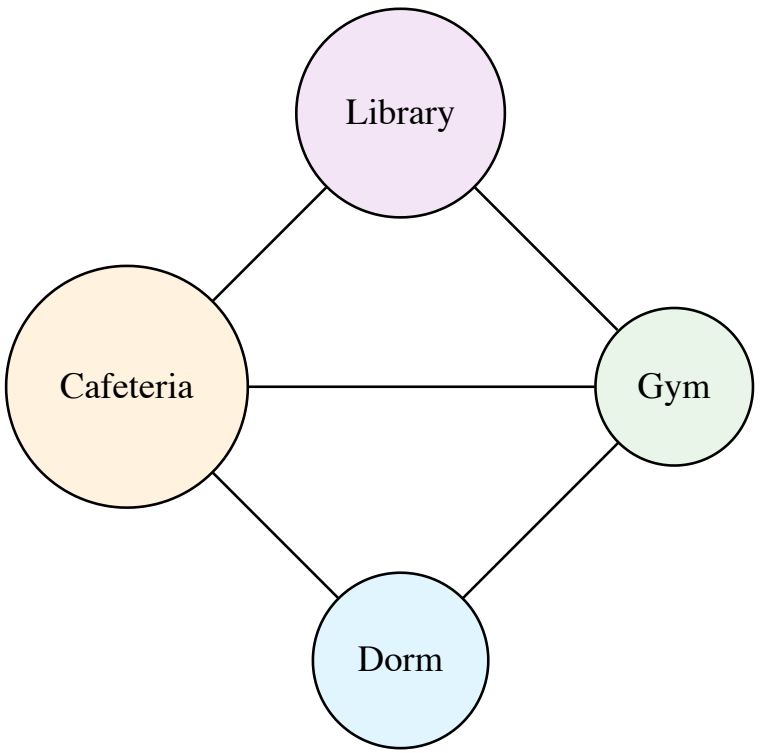
Answer:

1. Graph is connected, AND
2. Either:
    - All nodes have even degree
    - ~~Exactly two nodes have odd degree~~

This is called an Euler circuit.

Why we cannot have odd-degree nodes?

- **Walk**: Any sequence of connected nodes
- **Path**: Walks without repeated nodes
- **Circuit**: Paths that return to the starting point
- **Trail**: Paths without repeated edges
- **Cycle**: Trails that return to the starting point



Question: Are trails always paths? Are paths always trails?

Examples

Question:

What are the real-world examples of the paths and trails?

- **Path**: A travel itinerary that visits each city exactly once
- **Trail**: A mail carrier's route that visits each street exactly once

Network Connectivity 🔗

Look at Euler's conditions again:

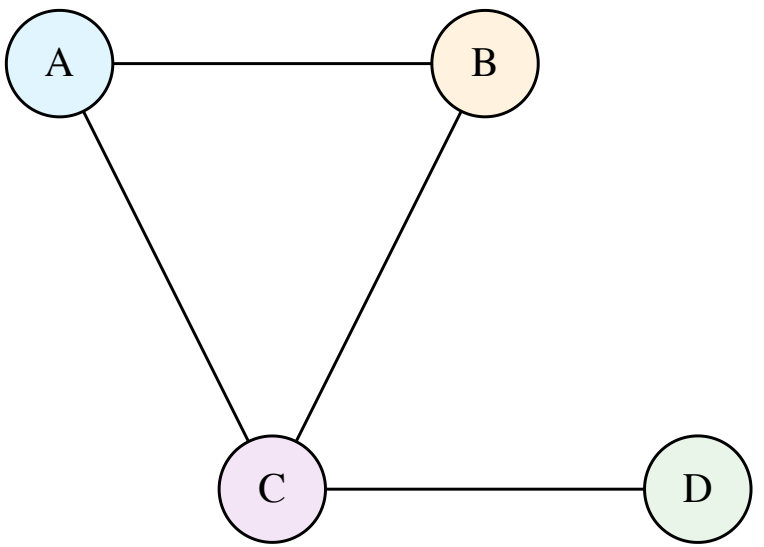
1. Either all nodes have even degree OR exactly two have odd degree, and
2. The graph is connected

What does "connected" mean?

What Does "Connected" Really Mean? 🤔

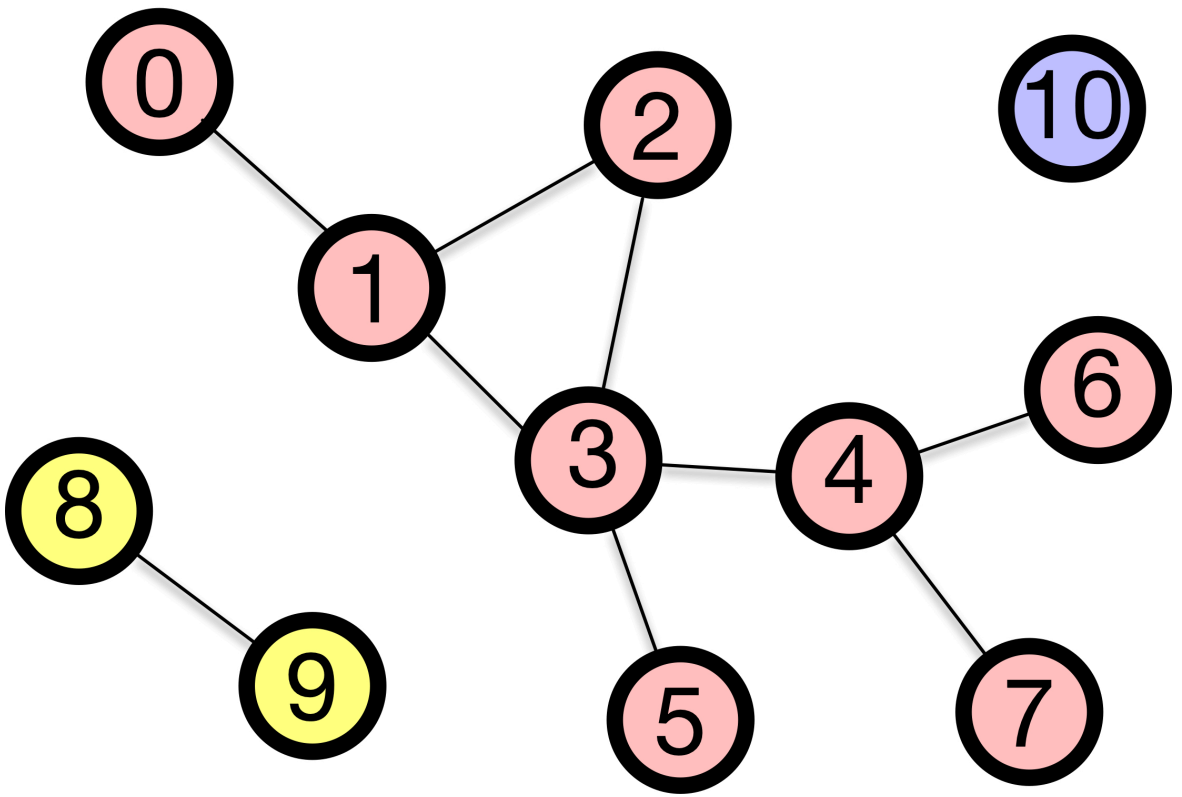
Definition: A network is **connected** if there is a path between every pair of nodes.

Example: You can walk from any building to any other building if connected. A building is not connected if there is no route to it along the walkways.



Connected Component

Definition: A connected component is a maximal set of nodes where every node can reach every other node within that set.



Question: Is a single node a connected component?

In large real-world networks, what would you expect 🤔?

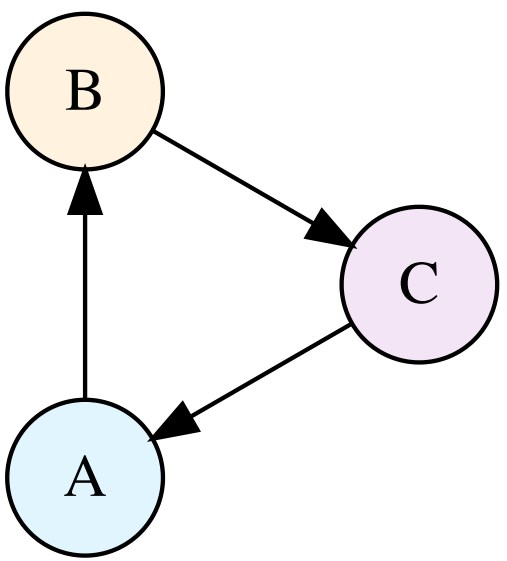
- Many tiny components of 2-3 nodes?
- One huge component containing most nodes?
- All components roughly equal size?

- Many networks contain a giant component that contains a significant fraction of all nodes in the network
- **Definition:** A giant component is a connected component where almost every node in the network is reachable from any other node in the component.

Context: What if edges have direction? (Think Twitter follows, webpage links)

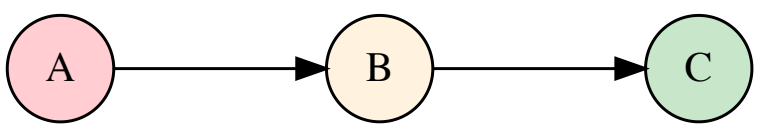
Strongly connected 💪

Every node can reach every other node following edge directions



Weakly connected 🤝

Connected if we ignore edge directions



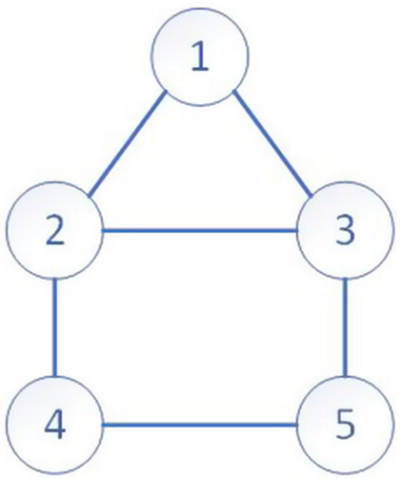
Question: Is every strongly connected component also weakly connected?

Coding Networks in Python 

Given any network, how would you represent it in a computer?

Three ways to represent the same network:

1. **Edge Table** - List of connections
2. **Adjacency List** - Each node's neighbors
3. **Adjacency Matrix** - Grid of 1s and 0s



5 nodes, 6 edges

Edge Table: The Direct Approach 📋

Simply list every connection:

```
edges = [
    (0, 1),  # Node 0 connects to Node 1
    (0, 2),  # Node 0 connects to Node 2
    (1, 2),  # Node 1 connects to Node 2
    (1, 3),  # Node 1 connects to Node 3
    (2, 4),  # Node 2 connects to Node 4
    (3, 4)   # Node 3 connects to Node 4
]
```

- *How would you count the degree of node 1 from this list?*
- *How would you find the neighbors of node 1?*

Adjacency List: Neighborhood Map 

Each node knows its neighbors:

```
neighbors = {
    0: [1, 2],     # Node 0 connects to nodes 1,2
    1: [0, 2, 3],  # Node 1 connects to nodes 0,2,3
    2: [0, 1, 4],  # Node 2 connects to nodes 0,1,4
    3: [1, 4],     # Node 3 connects to nodes 1,4
    4: [2, 3]      # Node 4 connects to nodes 2,3
}
```

- *How would you count the degree of node 1 from this list?*
- *How would you find the neighbors of node 1?*

Adjacency Matrix: The Math Way 📐

Grid where entry $(i, j) = 1$ if connected:

```python
import numpy as np

matrix = np.array([
    [0, 1, 1, 0, 0],  # Node 0: connects to 1,2
    [1, 0, 1, 1, 0],  # Node 1: connects to 0,2,3
    [1, 1, 0, 0, 1],  # Node 2: connects to 0,1,4
    [0, 1, 0, 0, 1],  # Node 3: connects to 1,4
    [0, 0, 1, 1, 0]   # Node 4: connects to 2,3
])
```

- *How would you count the degree of node 1 from this matrix?*
- *How would you find the neighbors of node 1?*

Implementing Euler's Theorem 🧮

```python
def has_euler_path(adjacency_matrix):
    # Calculate degrees
    degrees = adjacency_matrix.sum(axis=1)

    # Count odd degrees
    odd_count = sum(1 for d in degrees if d % 2 == 1)

    # Euler's condition
    return odd_count == 0 or odd_count == 2
```

Do you agree with this?

The Missing Piece: Connectivity ⚖️

Revisit

An Euler path exists if and only if:

1. **The graph is connected** ← We forgot this!
2. **Exactly 0 or 2 nodes have odd degree**

Module 01 Review 🎯

- **Euler's legacy** (1736)
  - Abstraction over physical details
  - Focus on relationships → Birth of graph theory
- **Euler's theorem**
  - Euler path exists if and only if the graph is connected and exactly 0 or 2 nodes have odd degree
- **Key concepts**:
  - Path, walk, trail, circuit, cycle
  - Connected component, giant component, degree
- **Computational Representation**
  - Edge table, adjacency list, adjacency matrix

- **Walk**
  - Any sequence of connected nodes
- **Trail**
  - Walk without repeated edges
- **Path**
  - Walk without repeated nodes
- **Circuit/Cycle:**
  - Closed versions that return to start
- Is a path always a trail 🤔?
  - Yes. Path does not repeat edges.

- **Connected**
  - A network where there is a path between every pair of nodes
- **Connected component**
  - A maximal set of nodes where every node can reach every other node within that set
- **Giant component**
  - A connected component where almost every node in the network is reachable from any other node in the component
- **Strongly connected**
  - A network where every node can reach every other node following edge directions
- **Weakly connected**

- **Euler path**
  - A path that visits each edge exactly once
- **Euler circuit**
  - An Euler path that starts and ends at the same node
- What is the condition for the existence of an Euler circuit?
  - Graph is connected and all nodes have even degree.

Representation of Networks

Edge Table

```
edges = [
    (0, 1),
    (1, 2),
    (2, 3)
]
```

Best for: Storage, I/O

Adjacency List

```
neighbors = {
    0: [1],
    1: [0, 2],
    2: [1, 3]
}
```

Best for: Neighbor search

Adjacency Matrix

```
matrix = np.array([
    [0, 1, 0],
    [1, 0, 1],
    [0, 1, 0]
])
```

Best for: Math operations

My recommendation:

Use edge table for saving the network data. Use (sparse) adjacency matrices for analysis.

Coming up in Module 02:

Small world networks

Almost all 8 billion people on the planet are your friends of friends of friends of friends of friends of friends.