CS 70 Discussion 2B

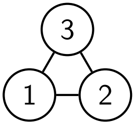
September 13, 2024

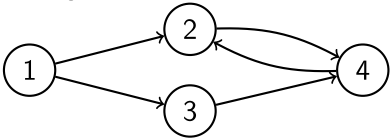
Graphs

Graph definitions (some requirements are specific to this class):

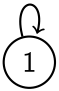
- **Vertices/Nodes** (typically denoted as circles/squares)
- **Edges** (either **directed** or **undirected**, and are denoted as lines between two nodes)
- Maximum of one edge between each pair of nodes (max one in each direction for directed graph)
- No edge from a node back to itself (**self-loop**)

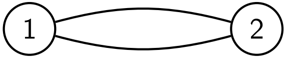
Examples of valid graphs:

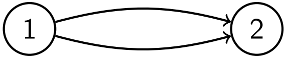




Examples of invalid graphs:







Graphs (Cont.)

Degree: Number of edges connected to a particular vertex

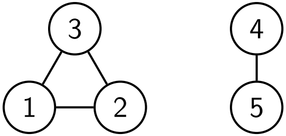
- **Handshake Lemma**: $\sum_{v \in V} \deg(v) = 2|E|$ (E is the set of all edges in the graph, V is the set of all vertices in the graph, and deg is a function that gives us the degree of a specific vertex)
- Directed graphs: **in-degree** (number of edges going into a vertex) and **out-degree** (number of edges going out of a vertex)

Types of graph traversals:

- **Walk**: Start at some node and move between nodes using edges
- **Path**: Same as a walk, but cannot revisit edges or vertices
- **Tour**: Same as a walk, but starts and ends at the same vertex
- **Cycle**: Same as a tour, but cannot revisit edges or vertices (except for the starting/ending vertex)

Graphs (Cont.)

Connected Component: A set of vertices in a graph where every vertex in the set can visit any other vertex in the set through some path (ex: the connected components of the below graph are $\{1, 2, 3\}$ and $\{4, 5\}$)



Eulerian Tour/Walks

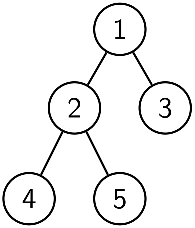
- **Eulerian Tour**: A tour that visits every edge in the graph exactly once
- **Eulerian Walk**: A walk that visits every edge in the graph exactly once
- **Condition 1**: An Eulerian tour exists iff all degrees in the graph are even:
  - An Eulerian tour can be started from any vertex!
  - There is an algorithm that can find an Eulerian tour in a graph quickly (if it exists)
- **Condition 2**: An Eulerian walk exists iff all degrees in the graph are even except for exactly 2 odd-degree vertices:
  - The Eulerian walk must start at one of the odd-degree vertices
  - There is an algorithm that can find an Eulerian walk in a graph quickly (if it exists)

Trees

There are three *equivalent* definitions for a tree:

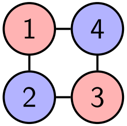
- A connected graph with no cycles
- A connected graph with $n-1$ edges (n is number of vertices)
- A connected graph where removing any edge splits graph into multiple connected components

Ex:



Graph Coloring

Vertex Coloring Problem: Color all vertices of a graph using *k* colors such that no adjacent vertices have the same color. Ex:



Edge Coloring Problem: Color all edges of a graph using *k* colors such that no adjacent edges have the same color. Ex: