Graphs and it’s types

presented by

Bishal Giri
Abhijan Sapkota
Aayam Tiwari
Ishan Shrestha

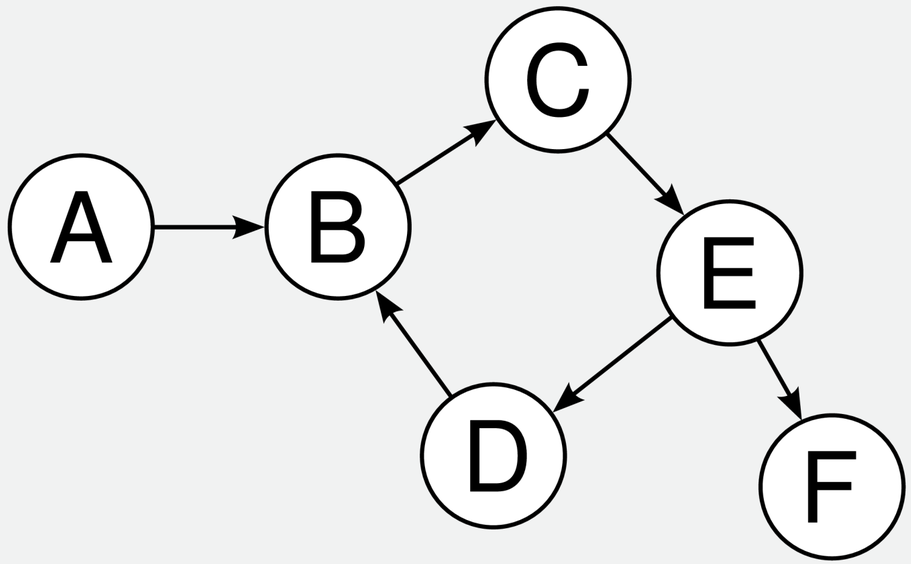
Overview

- Introduction
- Key Terminologies
- Types of Graphs
- Graph Applications
- Conclusion & Takeaways
- References
- Q&A / Discussion

Introduction

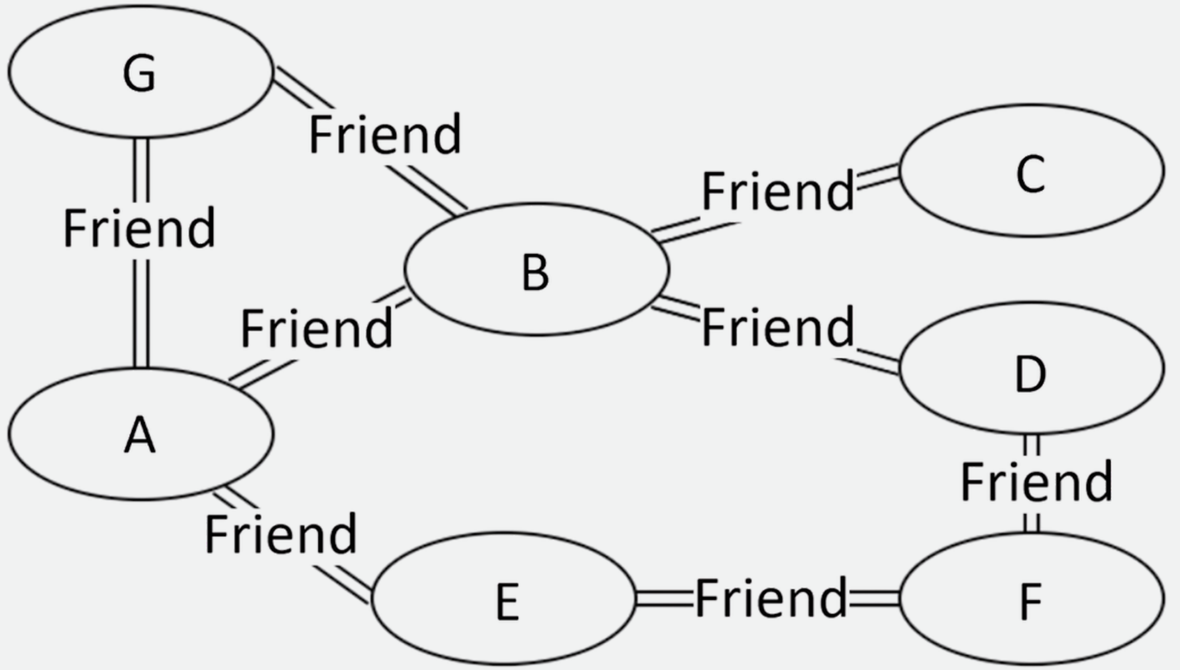
A graph is a non-linear data structure consisting of nodes (also called vertices) connected by edges.

- Directed/Undirected or Weighted/ Unweighted
- Models real-world connections (networks, routes, etc.)



Examples

- Social Networks
- Web Structure
- Airline Routes
- Neural Networks (AI)



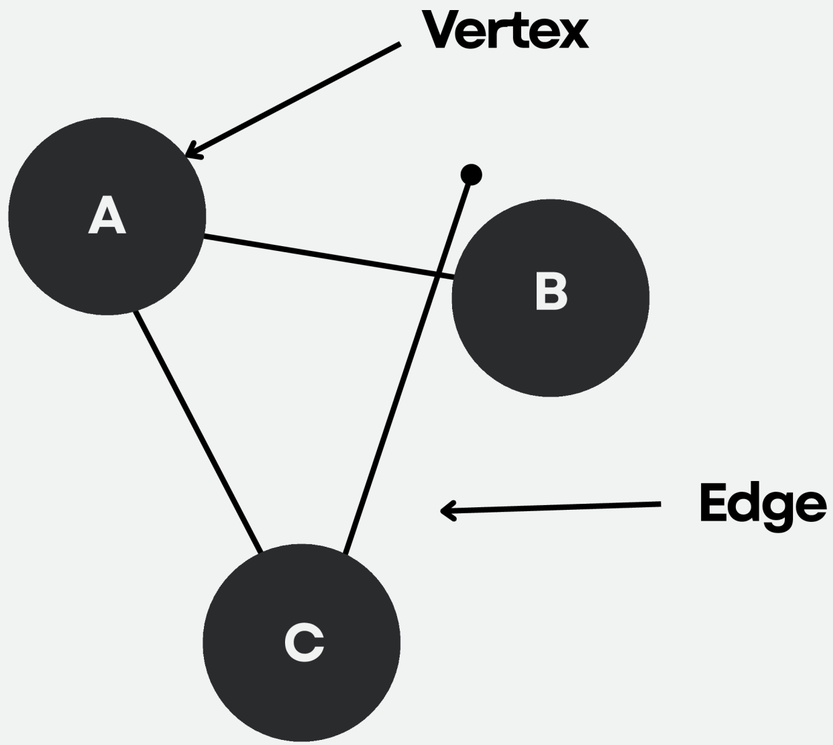
Graph Terminologies

Vertex (or Node)

- A fundamental point or entity in a graph.
- Represents an object, location, or piece of data.
- **Analogy:** Cities on a map.

Edge (or Arc/Link)

- A connection or relationship between two vertices.
- Indicates that two vertices are related in some way.
- **Analogy:** Roads connecting cities.



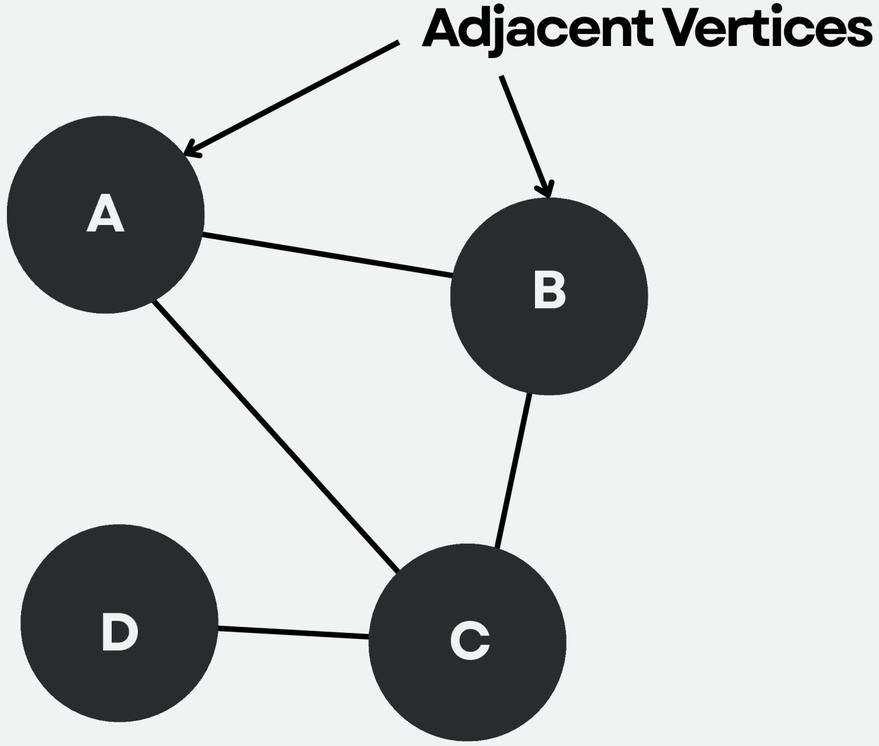
Graph Terminologies

Adjacent Vertices

- Two vertices are adjacent if they are directly connected by a single edge.
- Adjacency defines immediate neighbors in a graph.

Degree of a Vertex (for Undirected Graphs)

- The degree of a vertex is the total number of edges connected to it.
- **Example:** Vertex A: connected to 2 edges (Degree = 2).
- It tells us how many direct connections a vertex has.



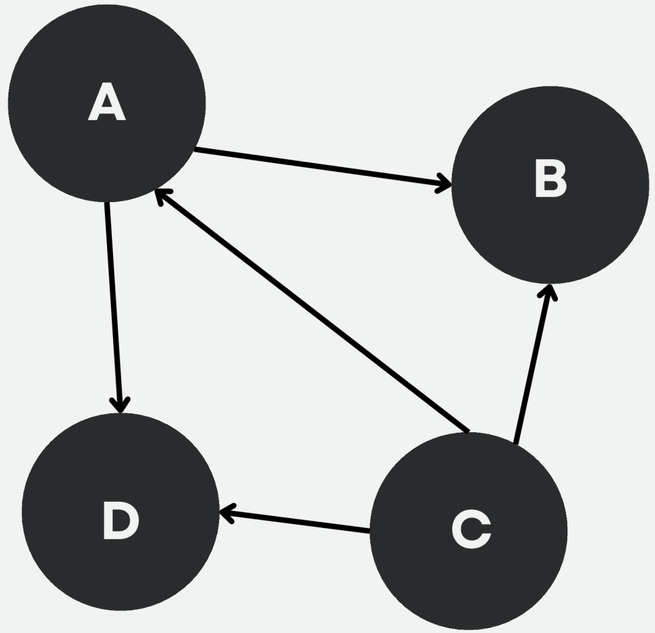
Graph Terminologies

In-degree (Directed Graphs)

- For a directed graph, the in-degree of a vertex is the number of incoming edges pointing towards it.
- How many other vertices point to this vertex.
- **Example:** Vertex A has arrows coming from C. (In-degree of A = 1)

Out-degree (for Directed Graphs)

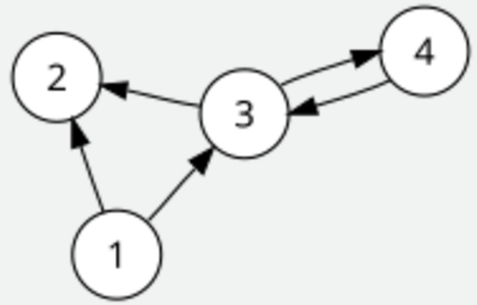
- For a directed graph, the out-degree of a vertex is the number of outgoing edges starting from it.
- How many other vertices this vertex points to.
- **Example:** Vertex A has arrows going to B and D. (Out-degree of A = 2)



Types of Graph

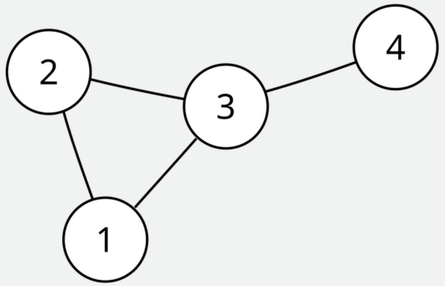
Directed Graph

- In a directed graph, the connection between two nodes is one-directional.
- It is a graph in which each edge has a direction to its successor.
- It is a graph with only directed edges.



Undirected Graph

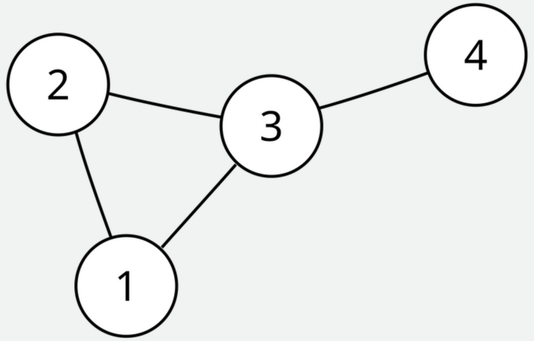
- In an undirected graph, all connections are bi-directional.
- It is a graph in which there is no direction on the edges.
- The flow between two vertices can go in either direction.



Types of Graph

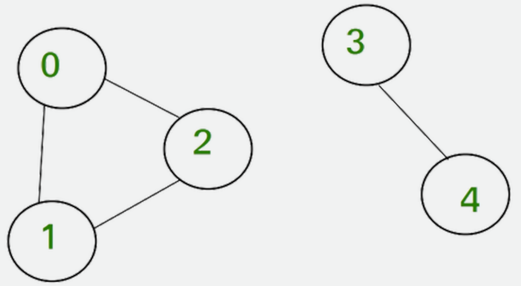
Connected Graph

- An undirected graph is called connected if there is a path between every pair of distinct vertices of the graph.



Disconnected Graph

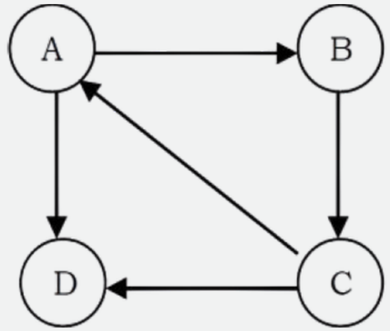
- An undirected graph that is not connected is called disconnected.



Types of Graph

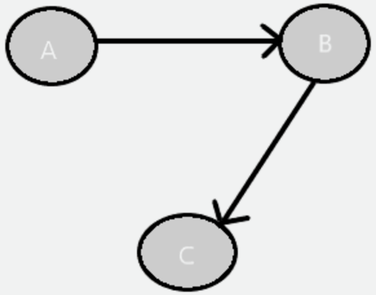
Strongly Connected Graph

- A directed graph is strongly connected if there is a path from A to B and from B to A whenever A and B are vertices in the graph.



Weakly connected Graph

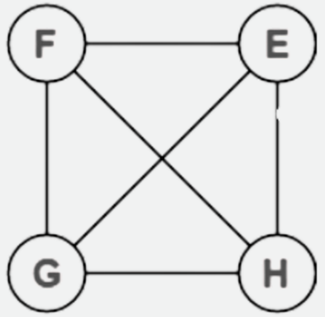
- A directed graph is weakly connected if there is a path between every two vertices in the underlying undirected graph. That is, a directed graph is weakly connected if and only if there is always a path between two vertices when the directions of the edges are disregarded.



Types of Graph

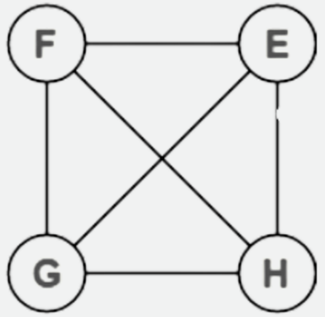
Complete Graph

- A graph in which any V node is adjacent to all other nodes present in the graph is known as a complete graph.
- An undirected graph contains the edges that are equal to edges=n(n-1)/2 where n is the no. of vertices present in the graph.



Regular Graph

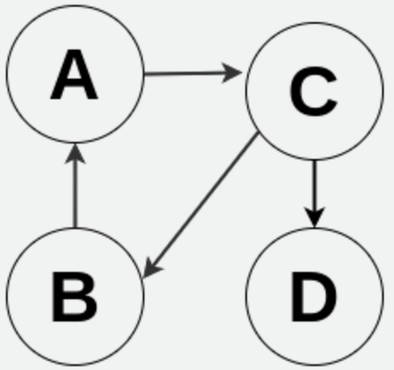
- It is a graph which nodes are adjacent to each other. i.e., each node is accessible from any other node.



Types of Graph

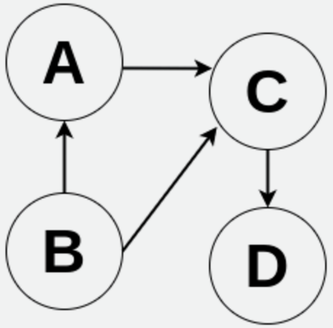
Cycle Graph

- A graph having cycle is called cycle graph.
- A closed simple path is called cycle.



Acyclic Graph

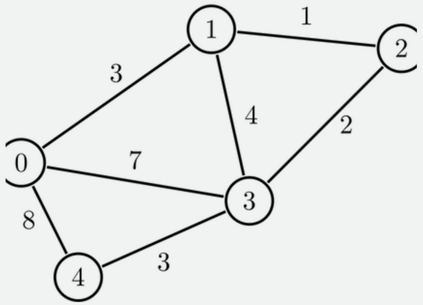
- A graph without cycle is called acyclic graphs.



Types of Graph

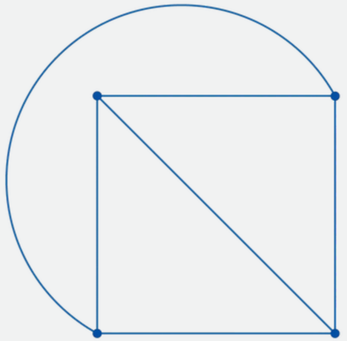
Weighted Graph

- Graphs that have a number assigned to each edge are called weighted graphs.
- In a weighted graph, each edge has an associated numerical value, called the weight of the edge. Edge weights may represent distances, costs, etc.
- Example: In a flight route graph, the weight of an edge represents the distance in miles between the endpoint airports



Planar Graph

- A graph is called planar if it can be drawn in the plane without any edges crossing, i.e., it can be drawn on the plane in such a way that its edges intersect only at their endpoints. In other words, it can be drawn in such a way that no edges cross each other.



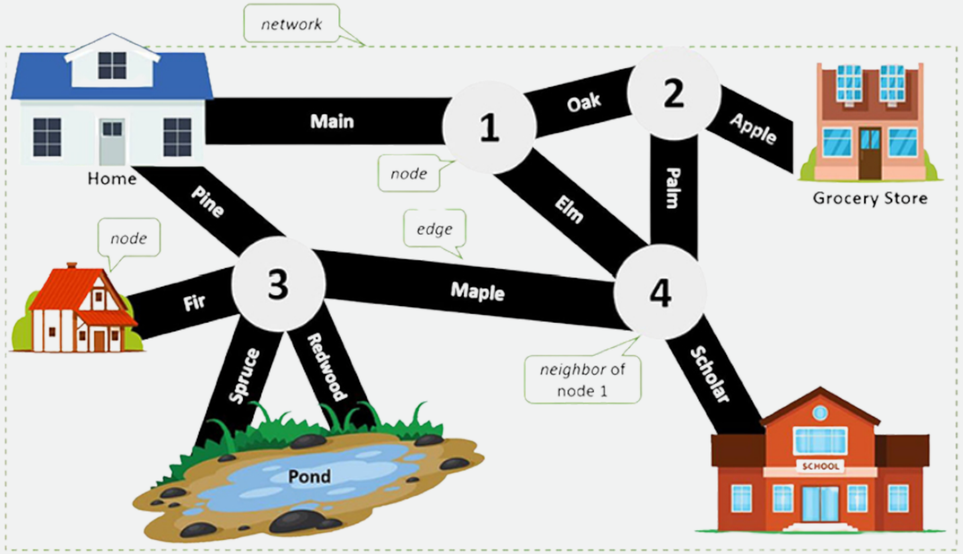
Application of Graphs

Graphs are a core data structure in Data Structures and Algorithms (DSA), and their applications are vast and essential across many domains. Some are explained below

- **Google Maps / GPS Navigation** – shortest path finding
- **Social Networks** – friend suggestions, community detection
- **Computer Networks** – routing algorithms, network topology
- **Web Page Ranking** – Google PageRank algorithm
- **Task Scheduling** – topological sort in DAGs
- **Compiler Design** – symbol dependency resolution
- **Transport Systems** – railway/flight connections

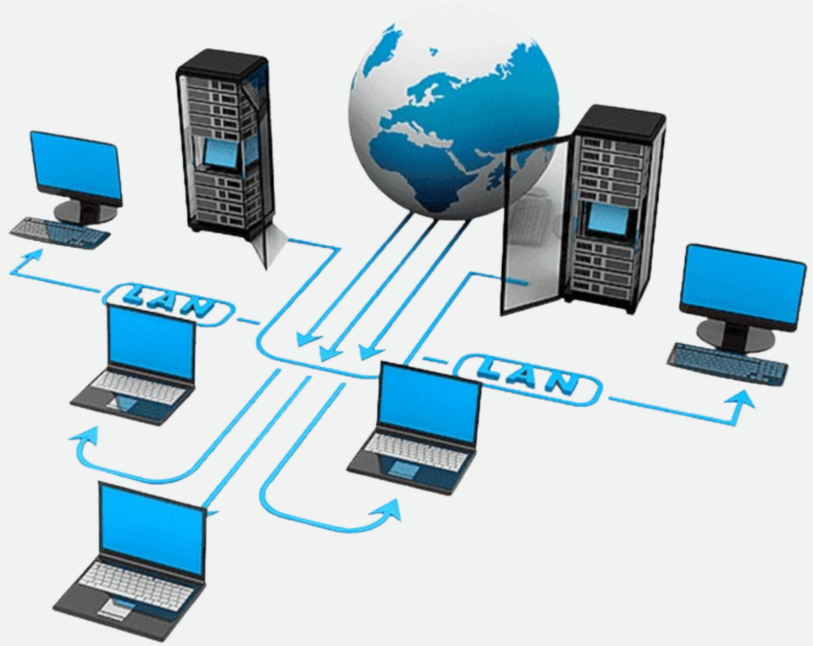
Dijkstra’s Algorithm -Finding the Shortest Path

- Used to find the shortest path in a graph
- Works on weighted graphs (with distances/time on edges)
- Used by Google Maps, network routers, and games
- Finds shortest paths from one node to all others



Computer Networks

- Devices (routers, switches, computers) are modeled as nodes, and communication links as edges.
- Graph algorithms like Dijkstra and Bellman-Ford are used in routing protocols such as OSPF and RIP.
- Helps determine optimal data paths, detect network loops, and ensure reliable transmission.
- Real-world use: Internet packet routing, data center traffic flow, wireless mesh networks.



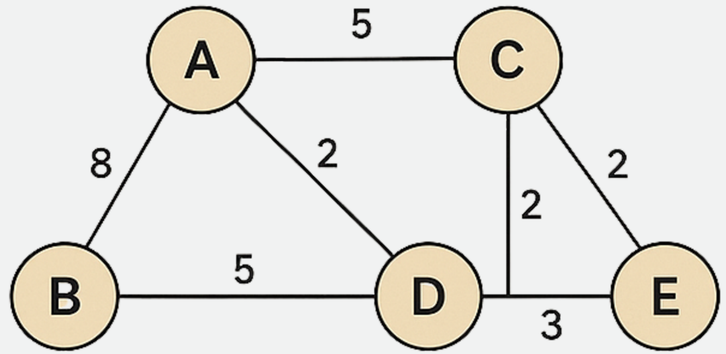
Task Scheduling

In task scheduling, each task is a node, and if one task must finish before another, we draw a directed edge between them. This forms a Directed Acyclic Graph (DAG). To find a valid order to complete all tasks, we use a graph algorithm called Topological Sort.

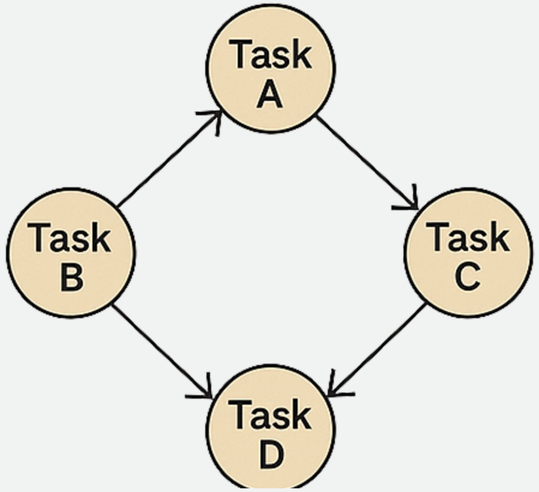
Transport Systems

- Airports or stations = nodes, routes = edges.
- Graphs help plan optimal paths, schedule trains/flights, and avoid congestion or collisions.

Transport Systems



Task Scheduling



Conclusion

- GRAPHS ARE FUNDAMENTAL NON-LINEAR DATA STRUCTURES.
- THEY MODEL REAL-WORLD CONNECTIONS AND RELATIONSHIPS (E.G., NETWORKS, ROUTES).
- DIVERSE GRAPH TYPES EXIST FOR VARIOUS APPLICATIONS.
- GRAPHS ARE ESSENTIAL FOR SOLVING COMPLEX PROBLEMS (E.G., NAVIGATION, SOCIAL NETWORKS, TASK SCHEDULING).

References

- https://www.geeksforgeeks.org/dsa/introduction-to-graphs-data-structure-and-algorithm-tutorials/
- https://www.w3schools.com/dsa/dsa_theory_graphs.php
- https://www.geeksforgeeks.org/dsa/applications-of-graph-data-structure/
- https://rajshah001.medium.com/graphs-and-real-life-application-28759b77b833

Any questions?