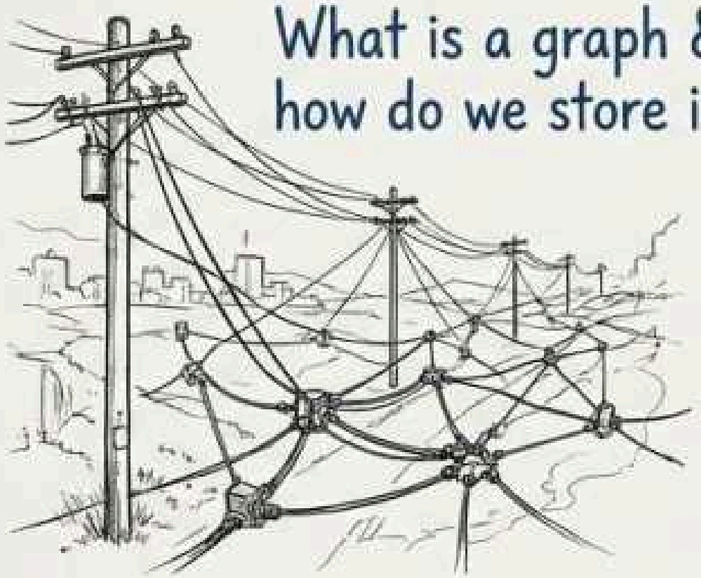
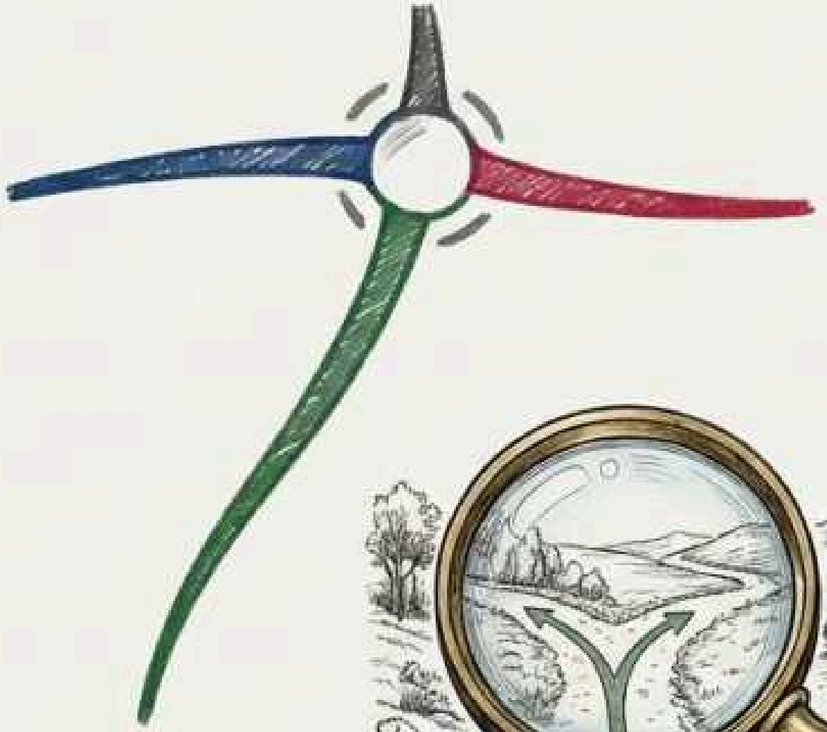
Three Core Questions of Graph Theory

¶ 4.1



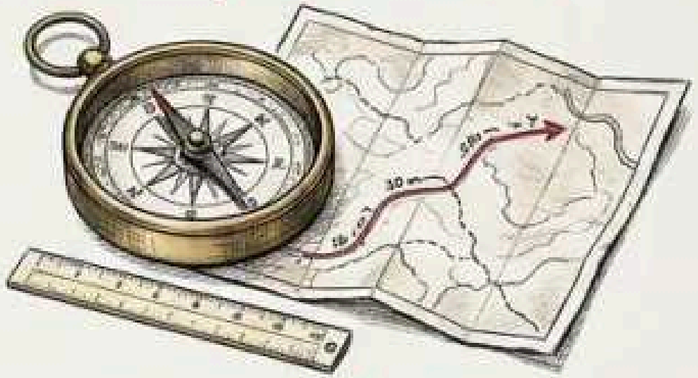
What is a graph & how do we store it?



How do we explore it? (Breadth-First / Depth-First)



How do we find shortest paths & cheapest networks?



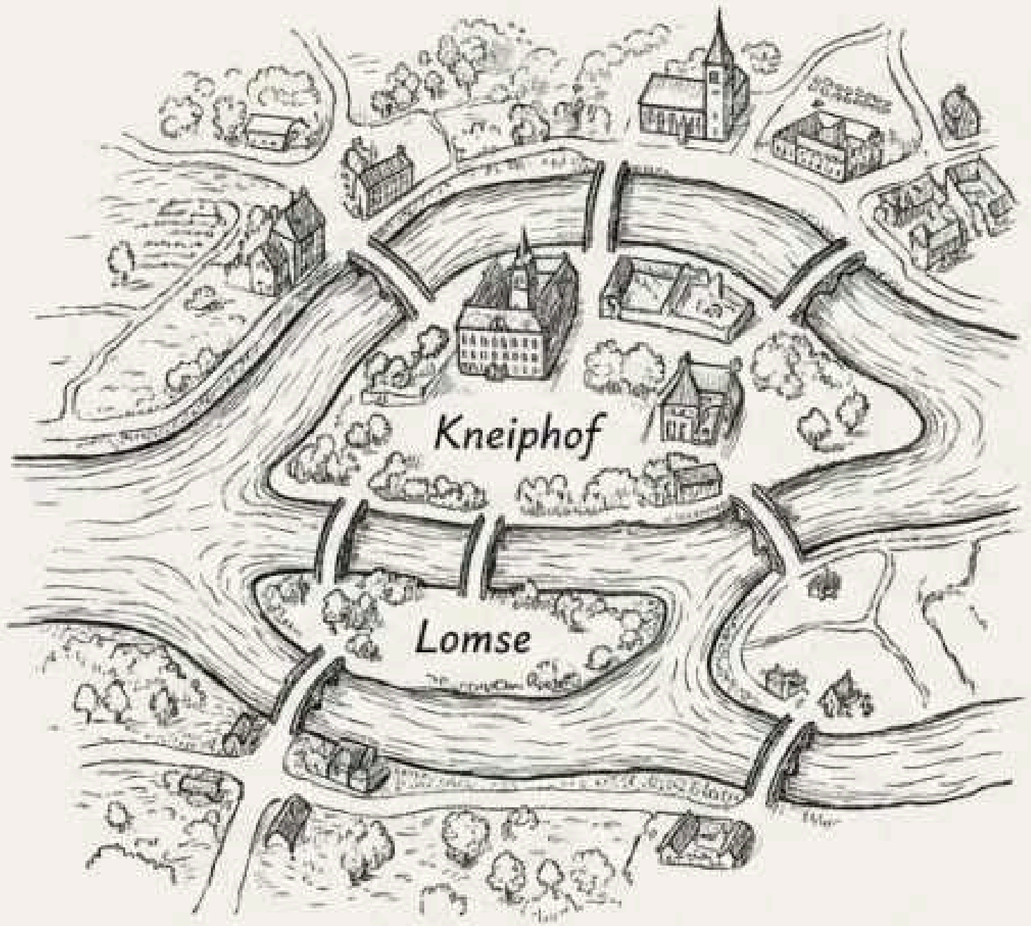
How do we find shortest paths & cheapest networks?

At a glance: Vocabulary, Storage, Search (BFS/DFS), Topological Sort, Shortest Paths, Spanning Trees, Union-Find.

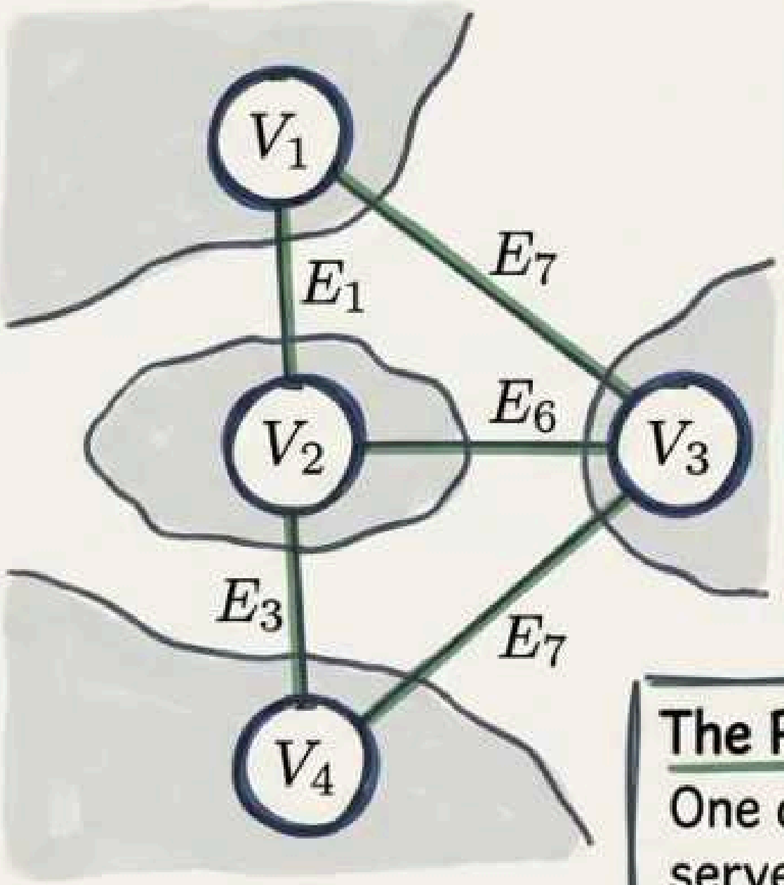
Gemini Notebook

The Universal Abstraction of Networks

¶ 4.2



Leonhard Euler, 1736



Core Definition:
A set of vertices joined by edges.

Edge Variations:
Undirected (friendships) vs. Directed (web links); Unweighted vs. Weighted (distance/cost).

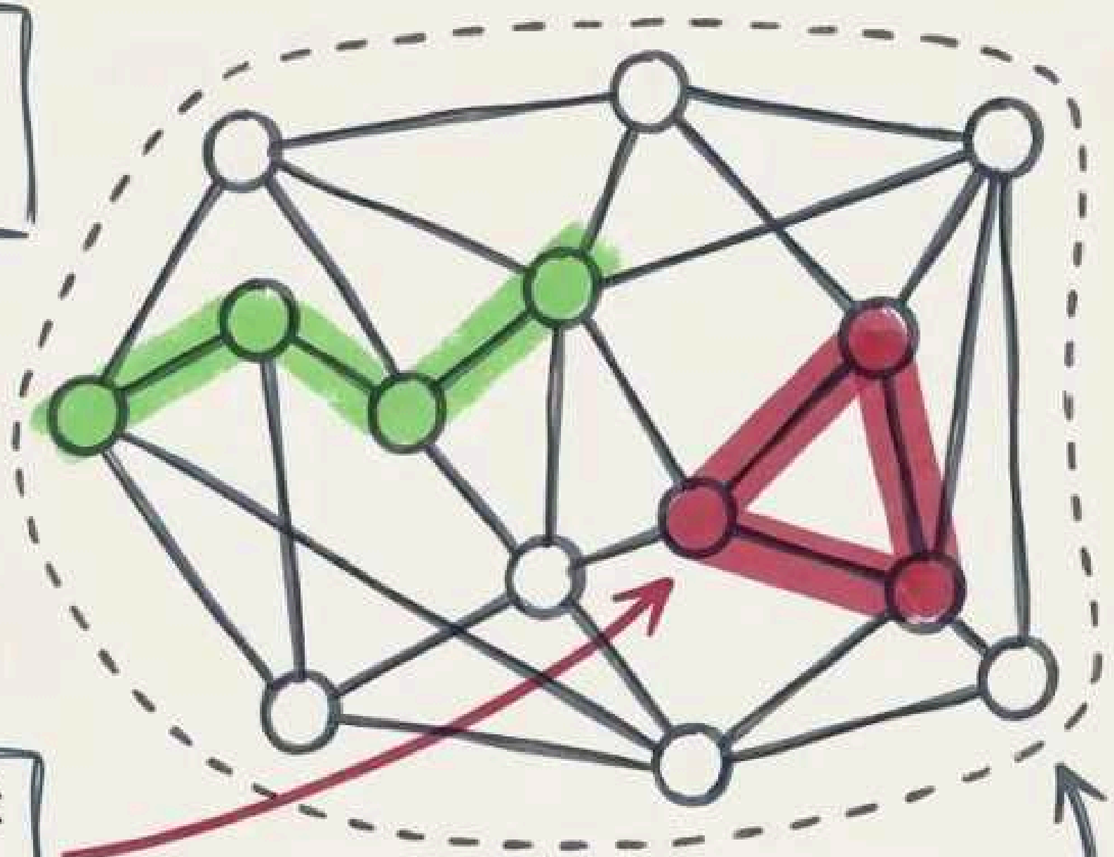
The Power of Abstraction:
One algorithmic framework serves road maps, social networks, the web, task dependencies, and circuits.

Gemini Notebook

The Core Vocabulary of Graph Theory

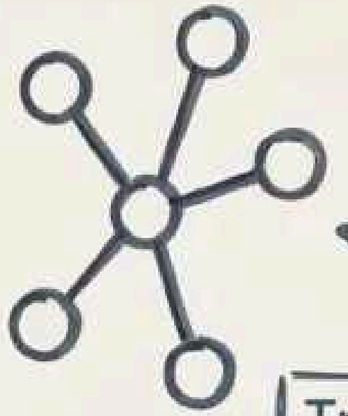
¶ 4.3

Path: A sequence of vertices joined by edges.



Cycle: A path that returns to its starting vertex.

Connected Graph: A path exists joining every pair of vertices.



Tree: A connected graph with strictly zero cycles.

$$V = |\text{Vertices}|, \; E = |\text{Edges}|$$

Gemini Notebook

Graph Storage: Matrix vs. List

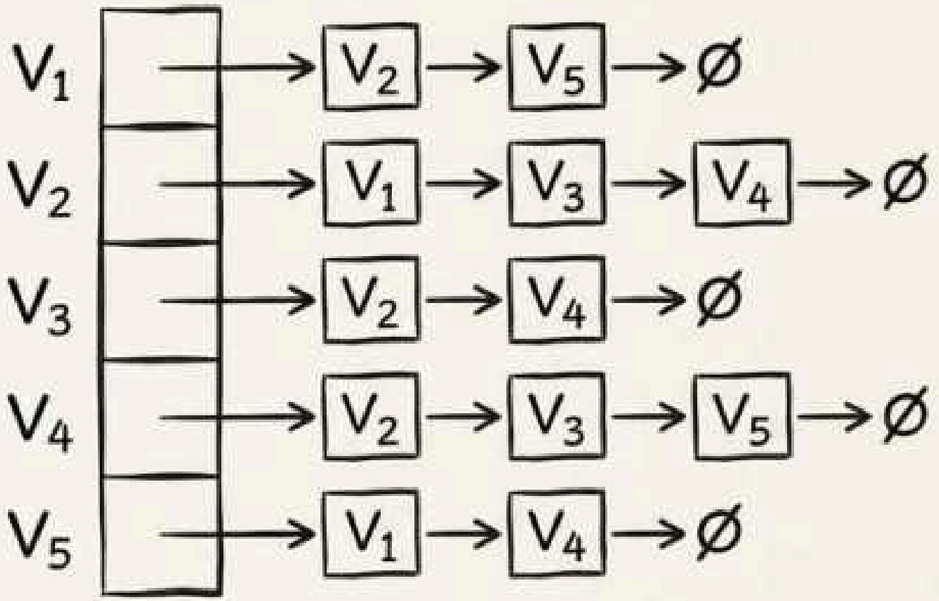
¶ 4.4

Adjacency Matrix

0	1	0	0	1
1	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0

- ✓ Edge test $O(1)$
- ✓ Memory requirement $O(V^2)$

Adjacency List

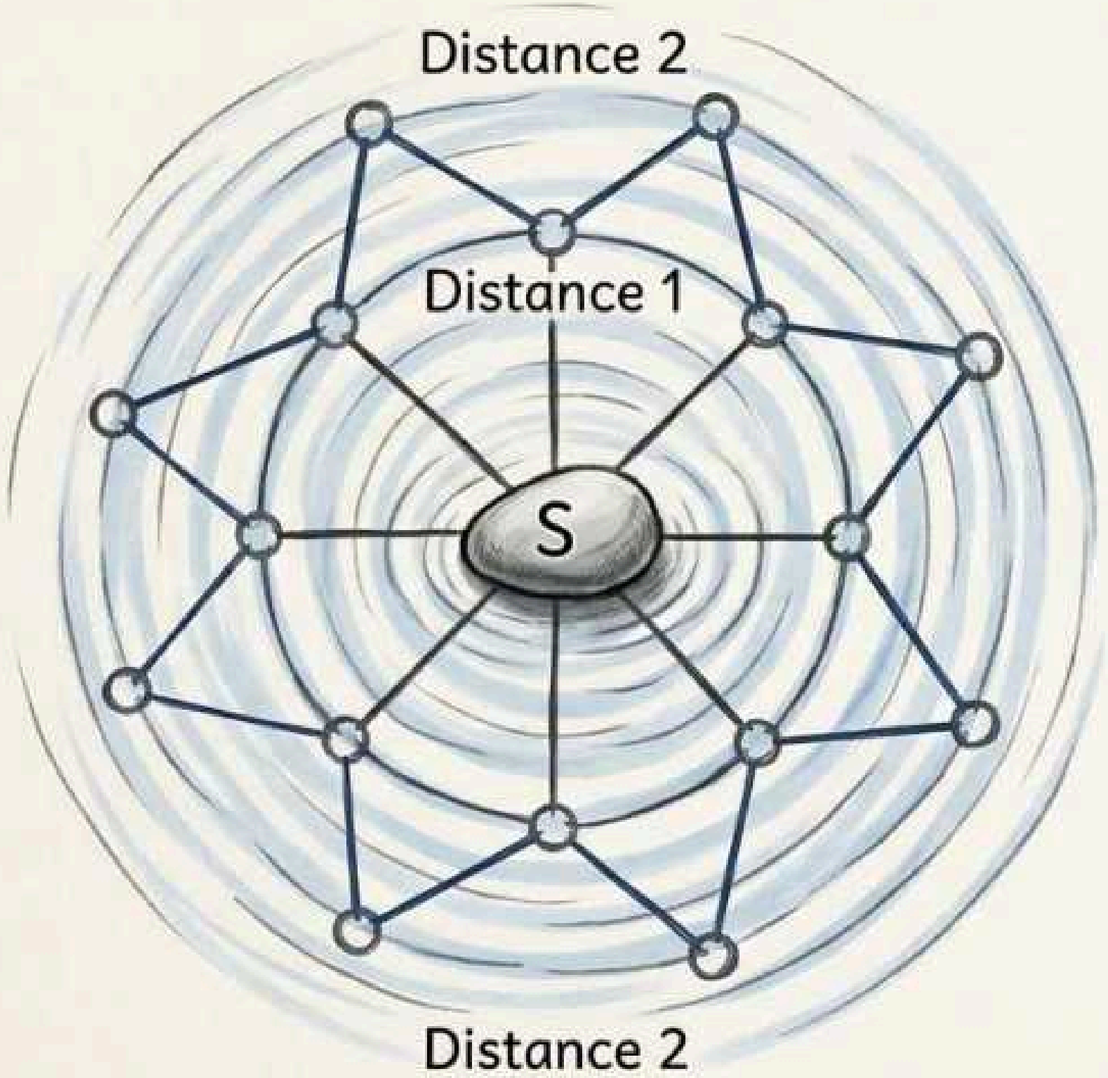


- ✓ Memory requirement $O(V + E)$

Most real-world graphs are sparse. Adjacency lists are the better choice for sparse graphs—the choice depends on the ratio of edges to vertices.

Gemini Notebook

Breadth-First Search (BFS): Concentric Exploration ¶ 4.5



Mechanism:

Uses a Queue.
Marks vertices as visited to prevent infinite loops.

Performance:

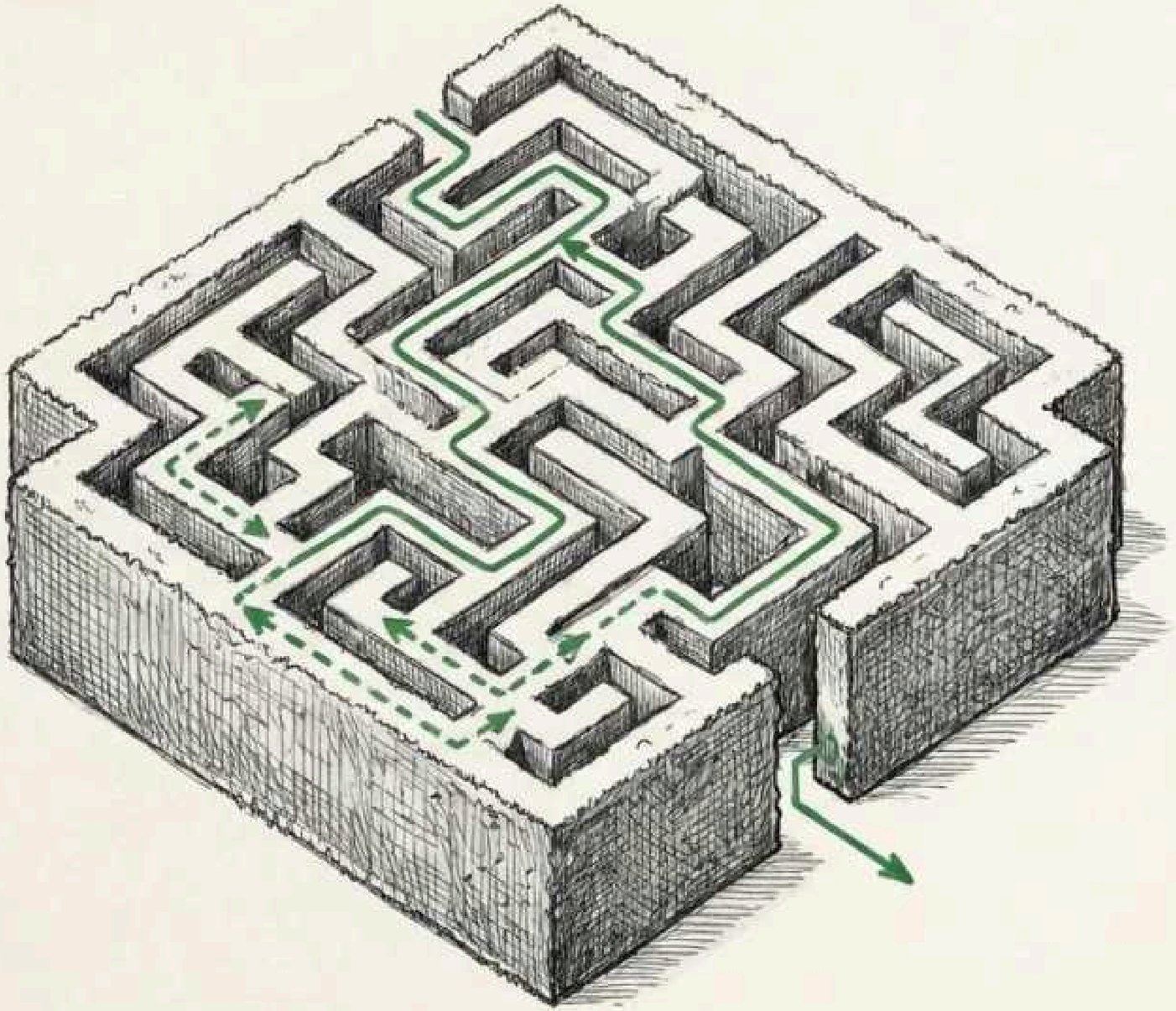
$$O(V + E)$$

Superpower:

In an unweighted graph, the first time it reaches a vertex is guaranteed to be the shortest path (fewest edges).

Gemini Notebook

Depth-First Search (DFS): Plunging and Backtracking ¶ 4.6



Mechanism:

Uses a Stack or Recursion.
Goes as deep as possible before reversing.

Performance:

$$O(V + E)$$

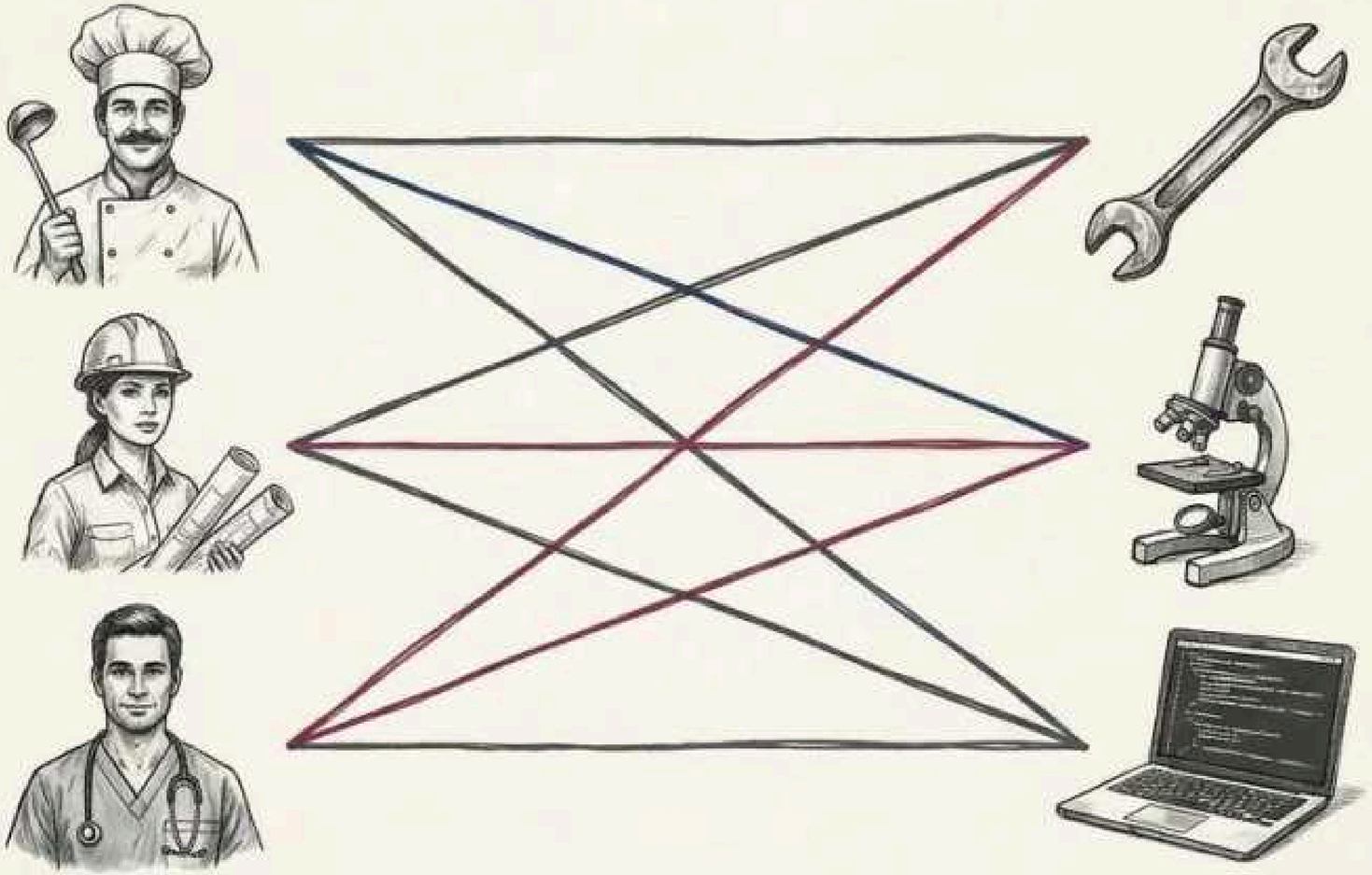
Applications:

Finding connected components, detecting cycles, solving mazes, and serving as the foundation for topological sorting.

Gemini Notebook

Bipartite Graphs and Matching Problems

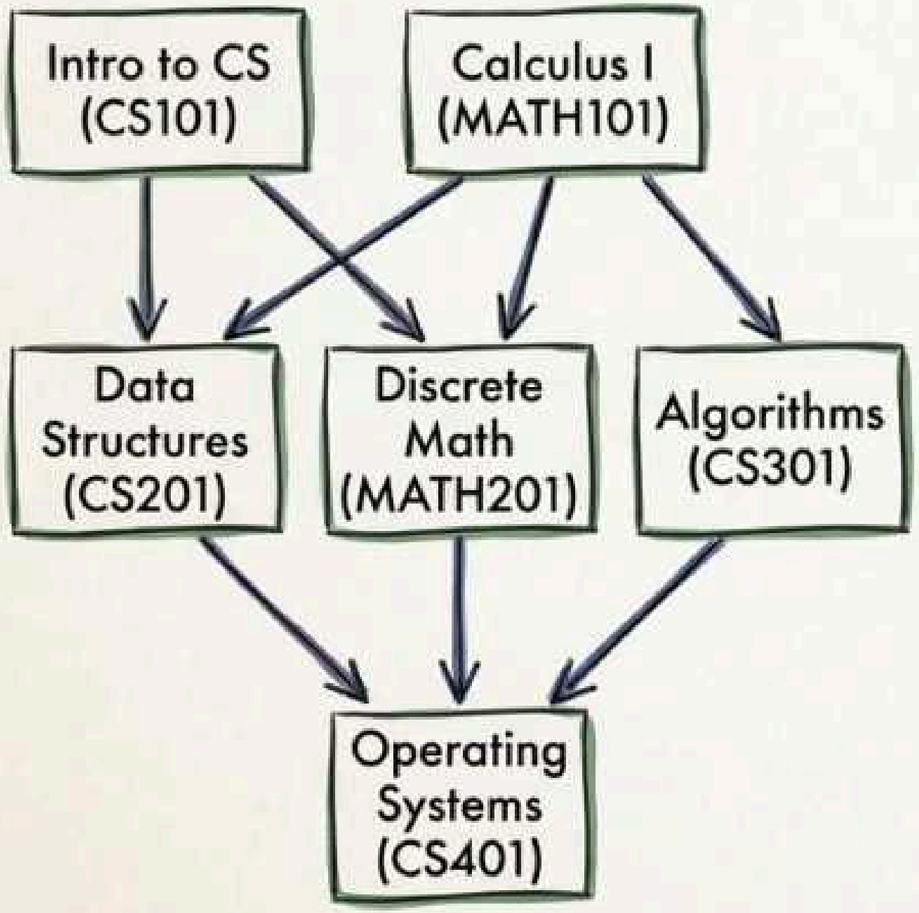
¶ 4.7

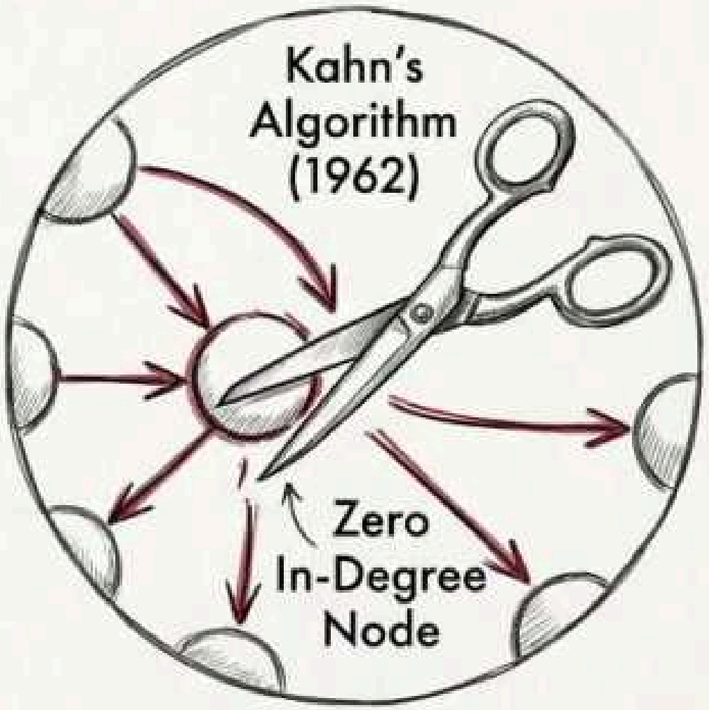


- Definition: Vertices split into two distinct groups with edges only going between them.
- The Rule: A graph is bipartite exactly when it has no cycles of odd length.
- The Test: Can be proven via two-coloring using Breadth-First Search.

Gemini Notebook

Topological Sorting for Directed Dependencies





Target Concept: Directed Acyclic Graphs (*DAGs*).

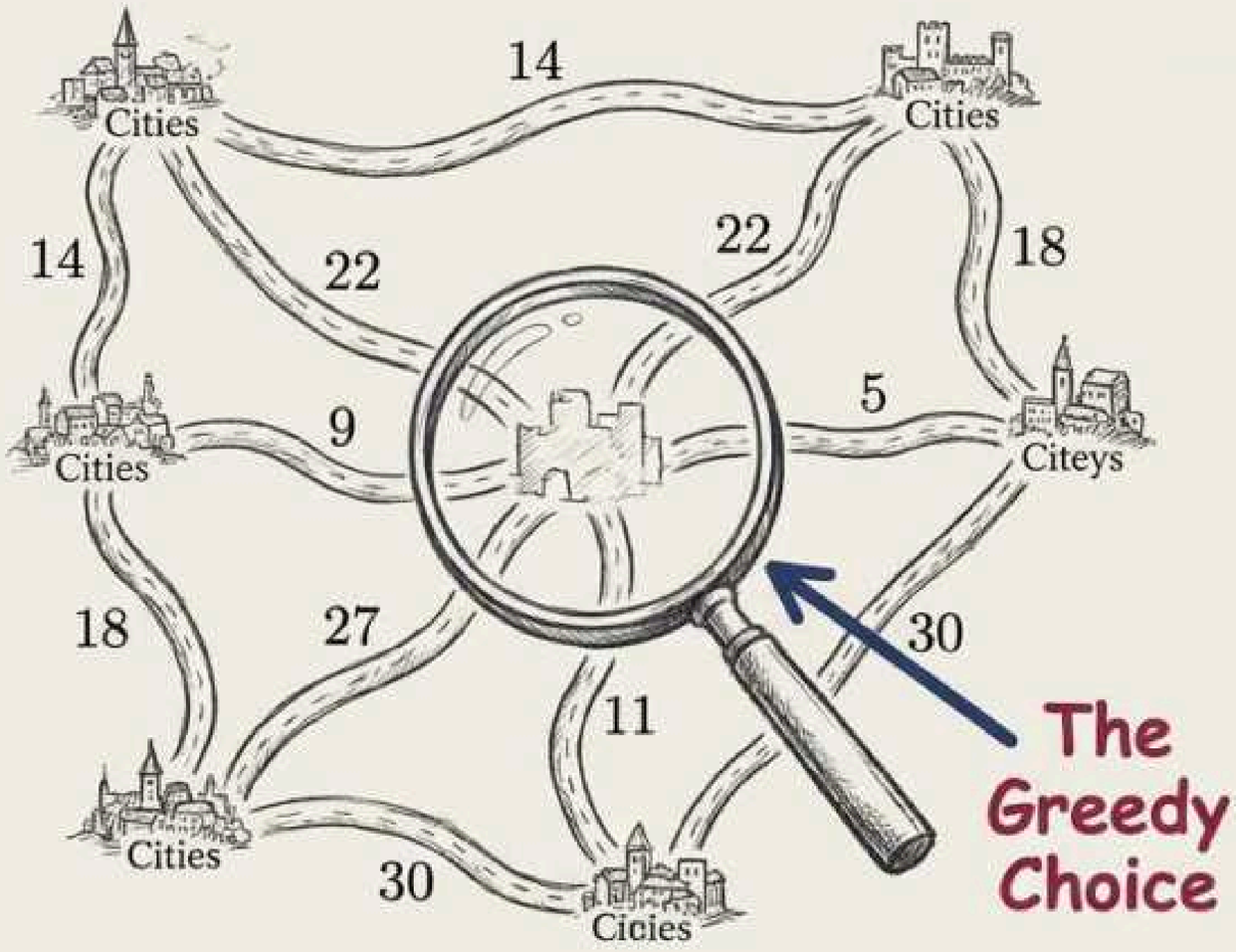
The Output: Orders vertices so that every single edge points forward.

The Limitation: If a cycle exists, no valid order exists (dependencies are deadlocked).

Gemini Notebook

Dijkstra's Algorithm for Shortest Paths

¶ 4.9



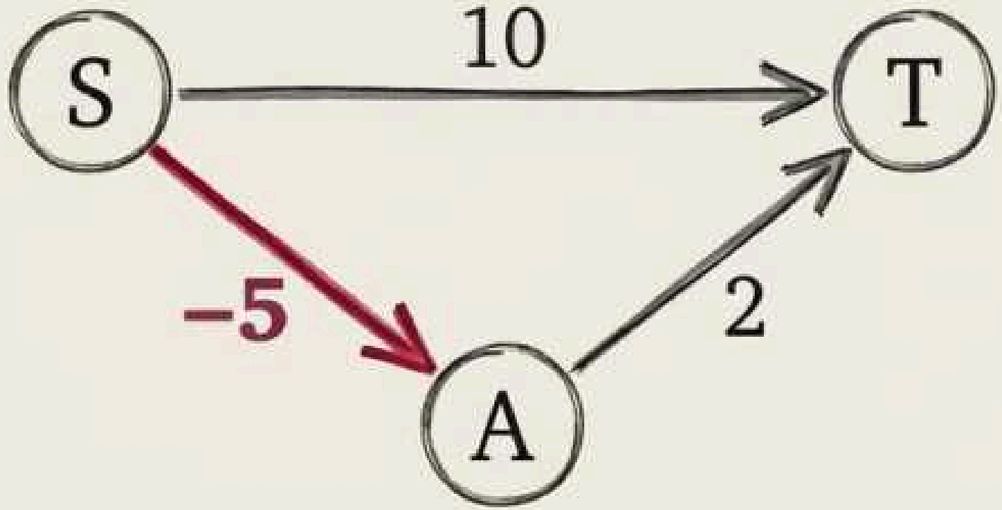
- The Problem: BFS only counts edges; weighted graphs require measuring total distance.
- The Mechanism: Uses a priority queue to continually commit to the unfinished vertex with the smallest tentative distance.
- Performance: Runs in $O((V + E) \log V)$ with a binary heap.

Gemini Notebook

The Relaxation Step and the Greedy Limit

¶ 4.10

If $d(u) + w(u, v) < d(v)$
then replace $d(v)$ with $d(u) + w(u, v)$



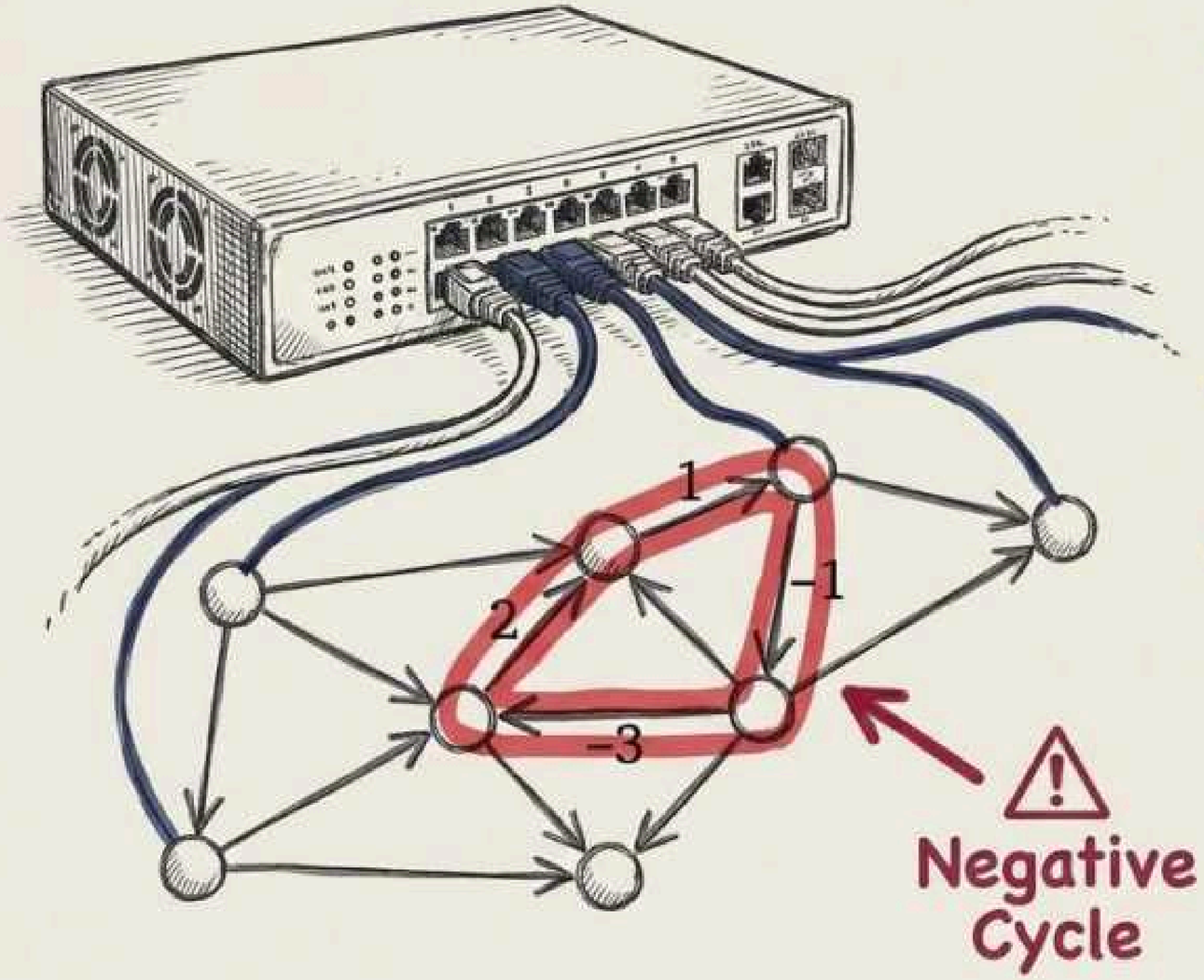
Relaxation: The act of replacing a known path with a newly discovered, shorter one.

The Constraint: Dijkstra is guaranteed correct only when there are no negative weights. Negative weights shatter the greedy assumption.

Gemini Notebook

Bellman-Ford and Negative Weights

¶ 4.11



- The Solution: Relaxes every edge $V - 1$ times to handle negative weights.
- Performance: $O(V \cdot E)$
- The Superpower: Detects negative cycles (where total weight falls below zero and no valid shortest path can exist). The foundation of internet distance-vector routing.

Gemini Notebook

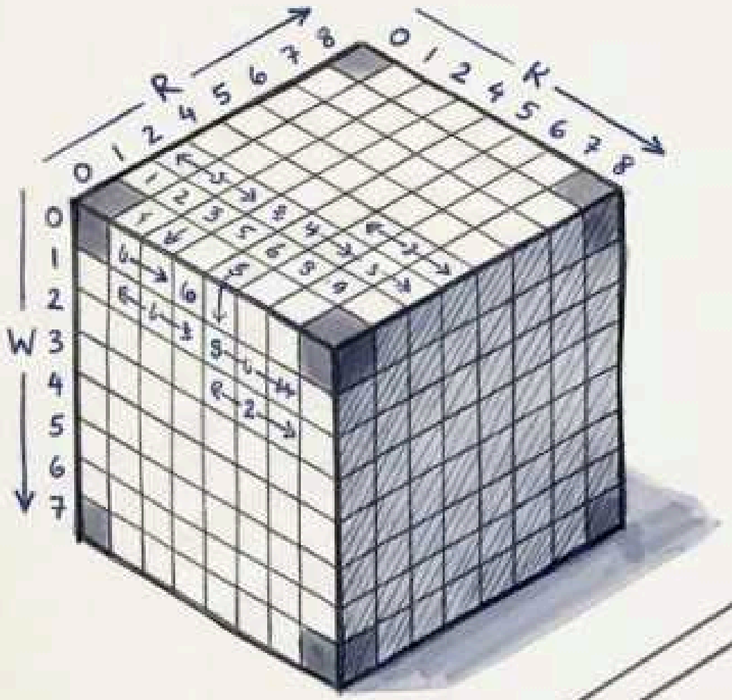
All-Pairs and Heuristic Routing

¶ 4.12

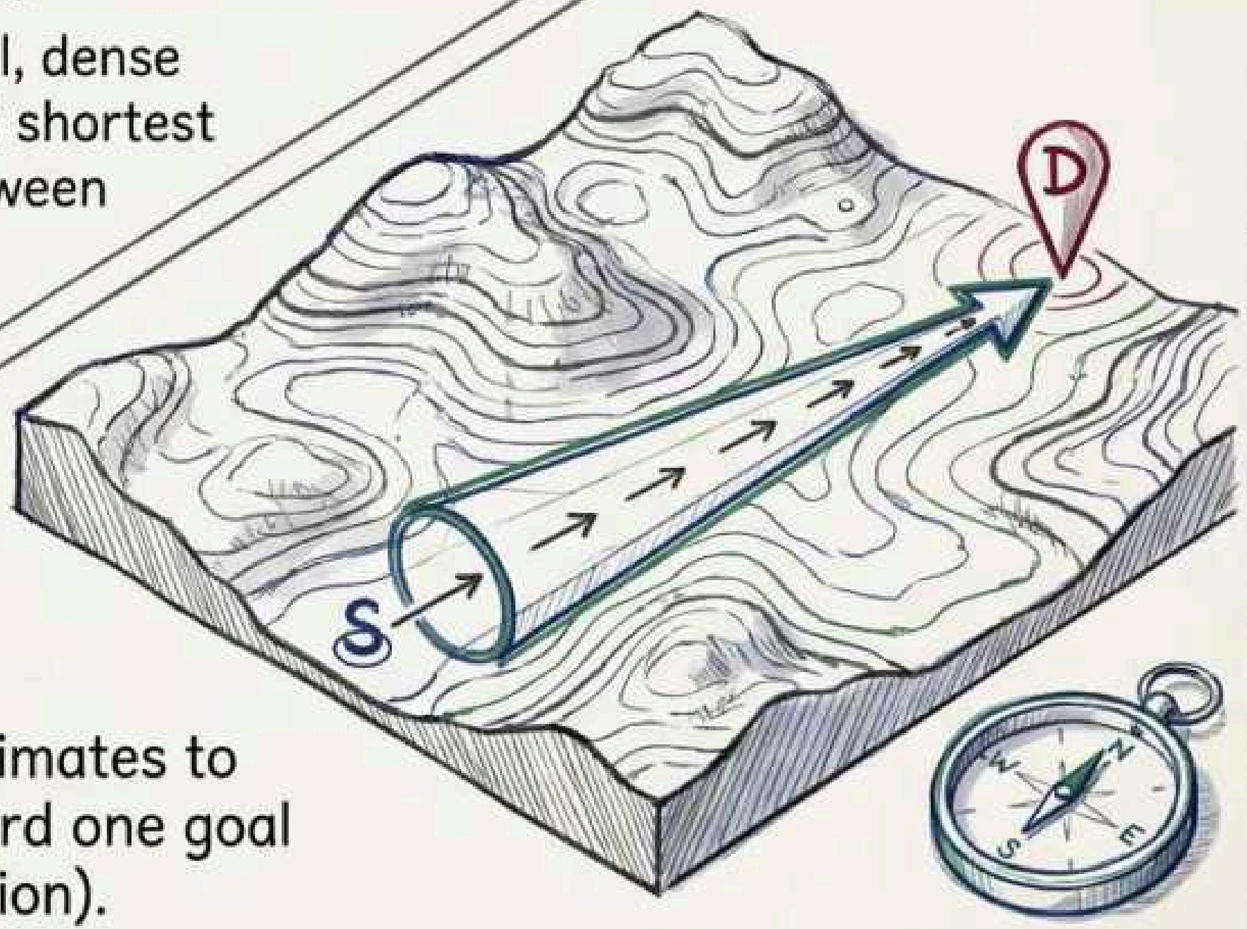
Floyd-Warshall

$$\mathcal{O}(V^3)$$

- Best for small, dense graphs to find shortest distances between all pairs.



A* Search

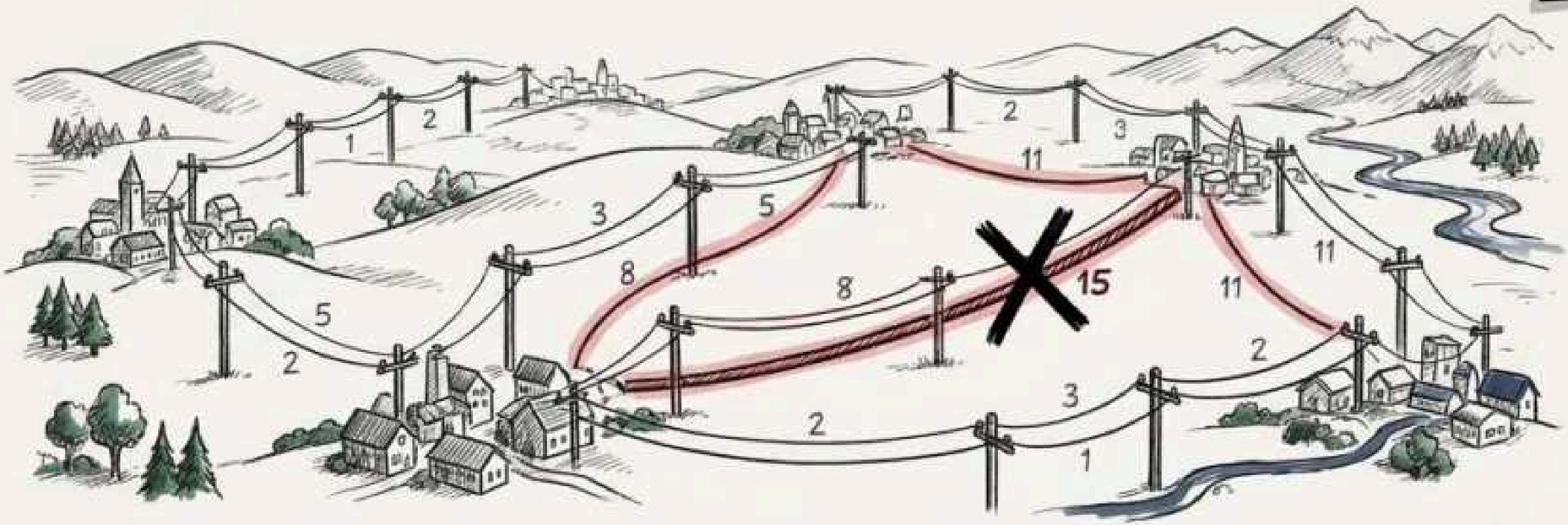


- Adds heuristic estimates to guide search toward one goal (Game AI, Navigation).

Gemini Notebook

Minimum Spanning Trees: Kruskal's Algorithm

¶ 4.13



The Goal:

Connect all vertices without any cycles at the absolute lowest total cost (e.g., wiring a network).

Kruskal's Approach (1956):

Sort all edges by weight.

Add each edge unless it creates a cycle.

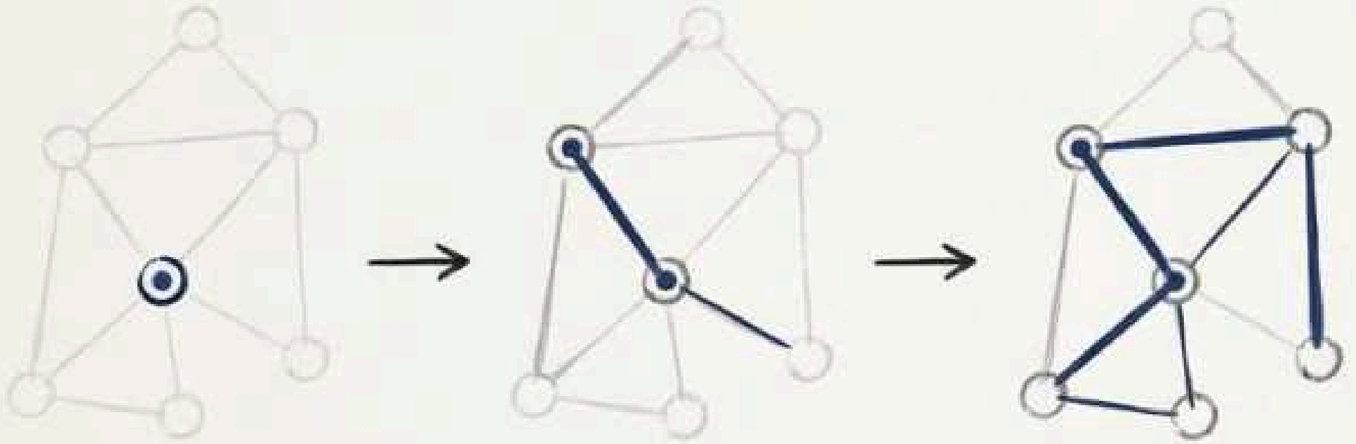
Core Principle:

The cheapest network never contains a cycle, because the costliest edge of any cycle can always be dropped.

Gemini Notebook

Prim's Algorithm: Organic Growth

Three-Step Growth Sequence:

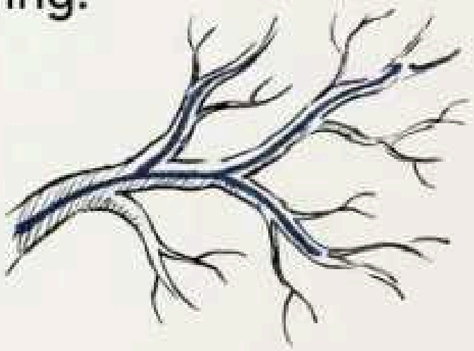


Start: Single Root

Step 1: Absorb Cheapest Adjacent Edge

Step 2: Expand Cohesive Tree

- **Mechanism:** Grows a single continuous tree from a starting vertex (unlike Kruskal's scattered edge assembly).
- **Performance:** Both Kruskal's and Prim's are greedy algorithms running in $O(E \log V)$ given optimal data structures.
- **Applications:** Network wiring and data clustering.

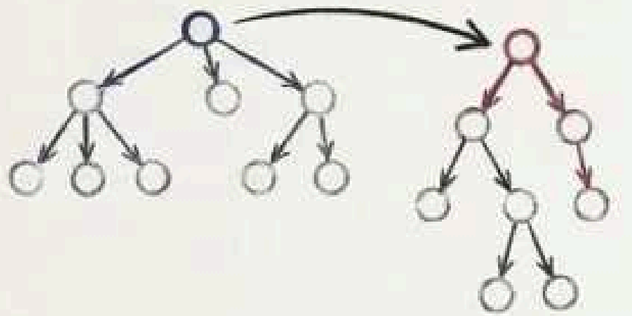


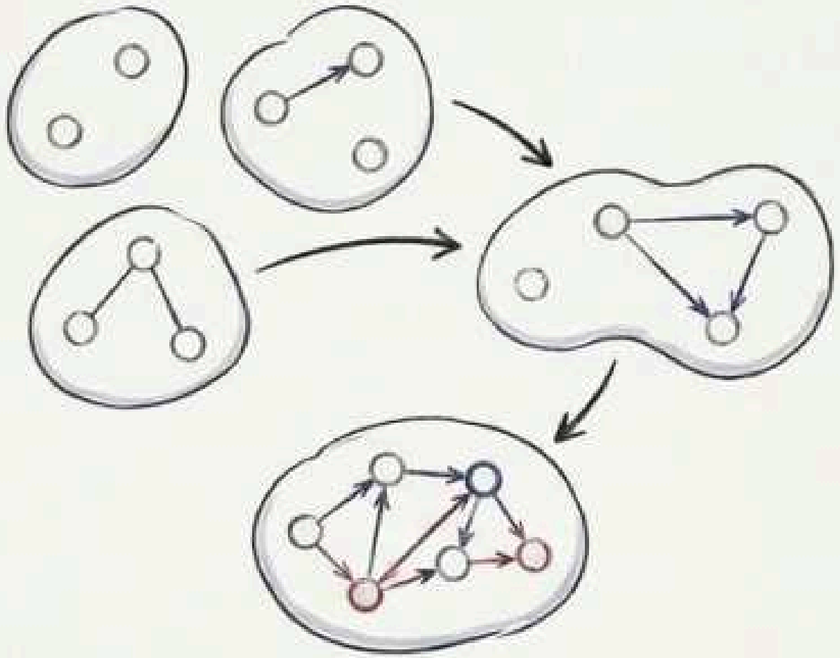
Gemini Notebook

The Union-Find Data Structure

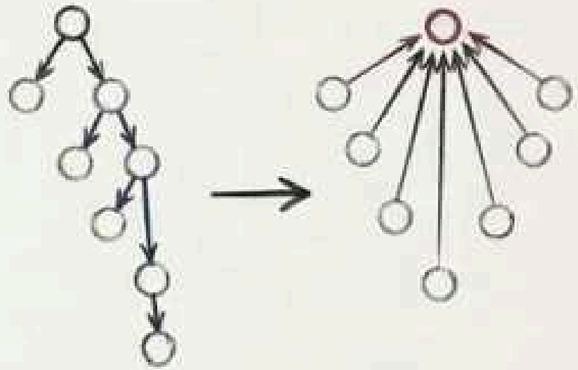
¶ 4.15

Union by Rank





Path Compression



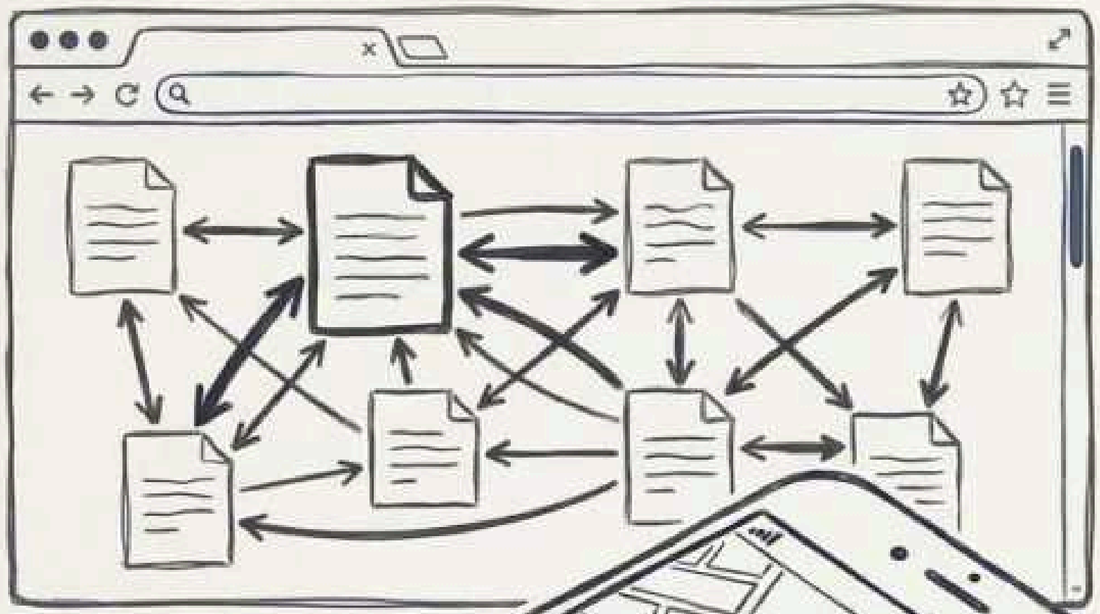
Purpose: Quickly answers if two vertices are already connected (essential for Kruskal's cycle detection).

Performance Magic: Operations take nearly constant time, analyzed using the extremely slow-growing inverse Ackermann function.

Gemini Notebook

Graphs Driving Modern Services

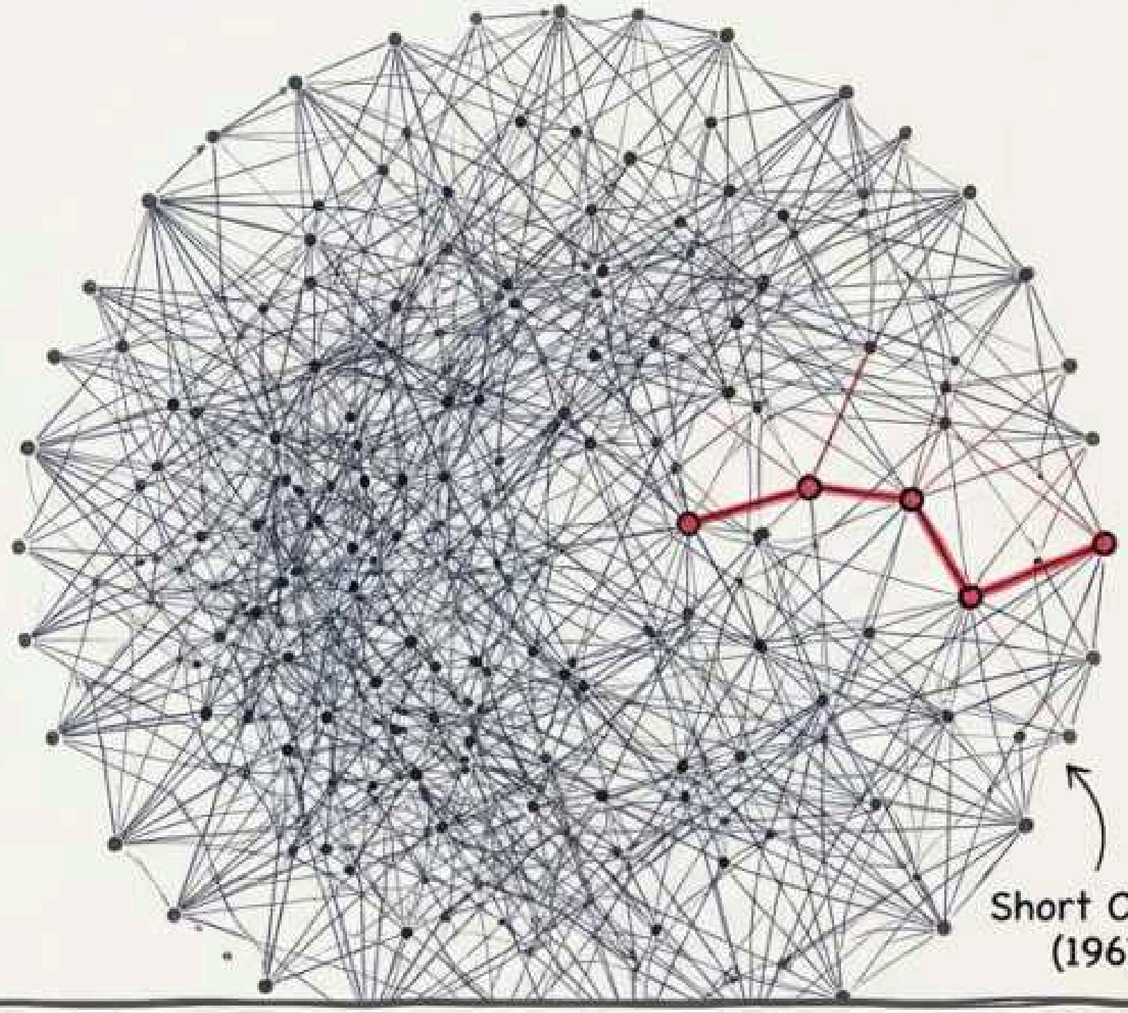
¶ 4.16



PageRank (1998)



GPS Navigation



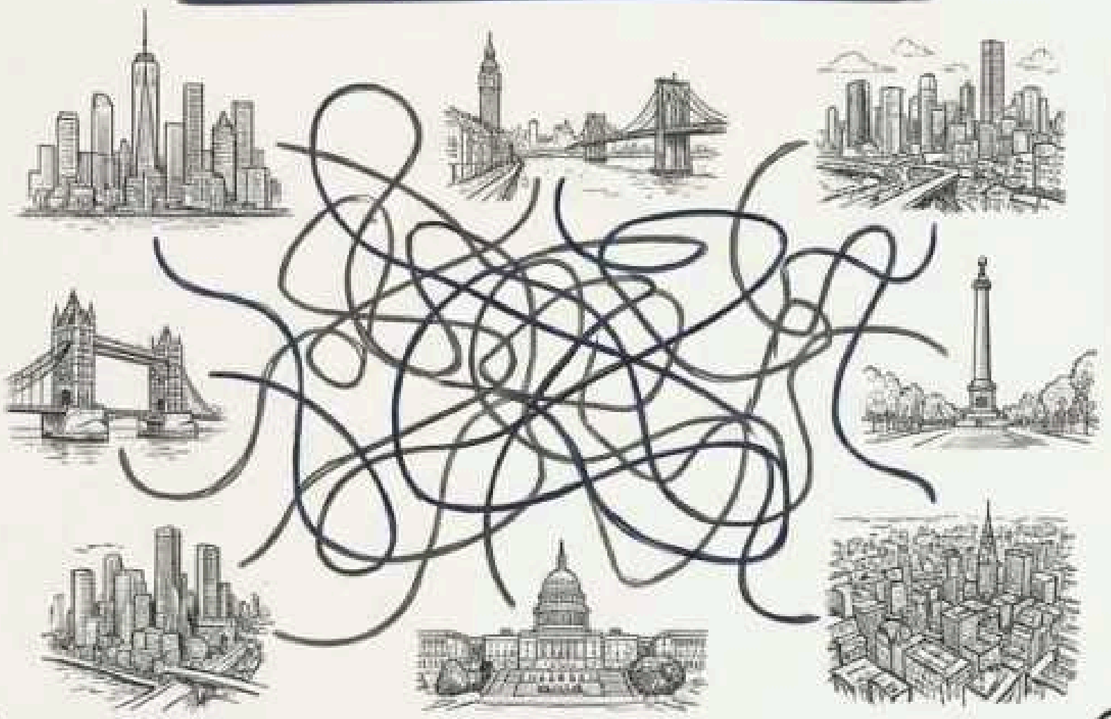
Short Chains (1967)

Modern civilization runs on graph theory: web search ranking, global navigation, and the structure of human society.

Gemini Notebook

Intractable Problems and the Limits of Speed ¶ 4.17

Traveling Salesman Problem



Graph Coloring

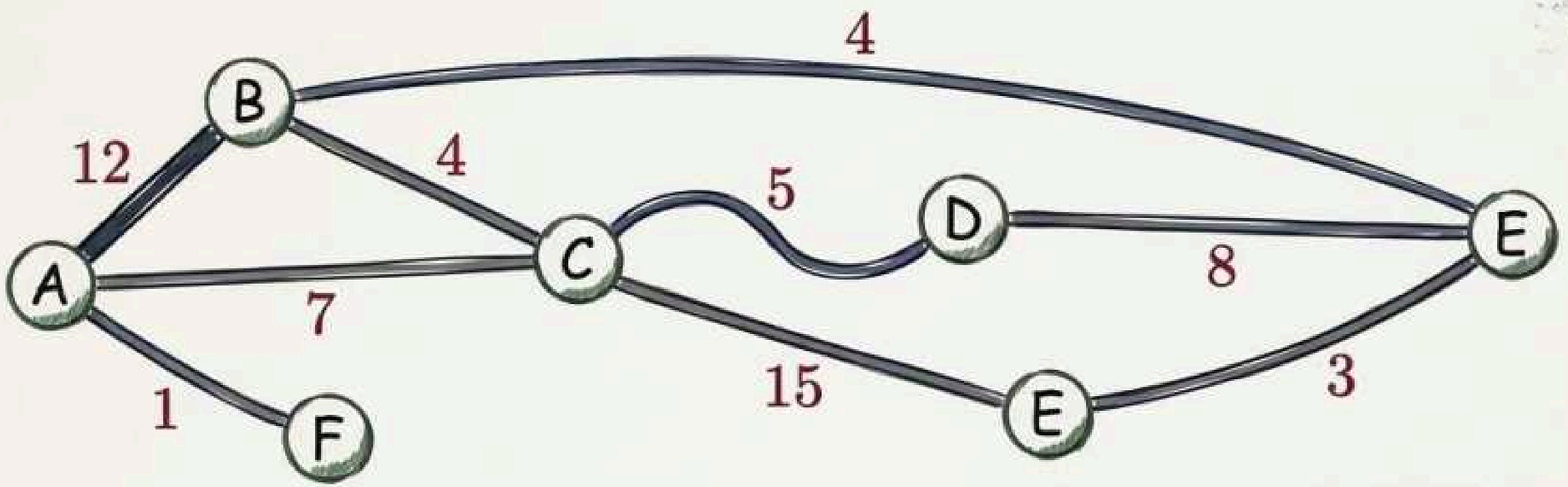


The Hard Truth: No fast algorithms are known to solve either problem exactly.

The Reality: Because these problems resist efficient solutions, real-world systems must accept approximate answers (Preview for Lecture 6).

Gemini Notebook

Closing Exercise: Topological vs. Weighted Search ¶ 4.18



BFS (Edges)	
City	Edge Count

Dijkstra (Distance)	
City	Total Distance

- The Task: Run Breadth-First Search from one city (e.g., City A). Then, run Dijkstra's algorithm by hand. List the distances as they settle.
- The Objective: Compare the answers to see where edge counts diverge from actual distances.
- Looking Ahead: Lecture 5: Searching and Sorting.

Gemini Notebook