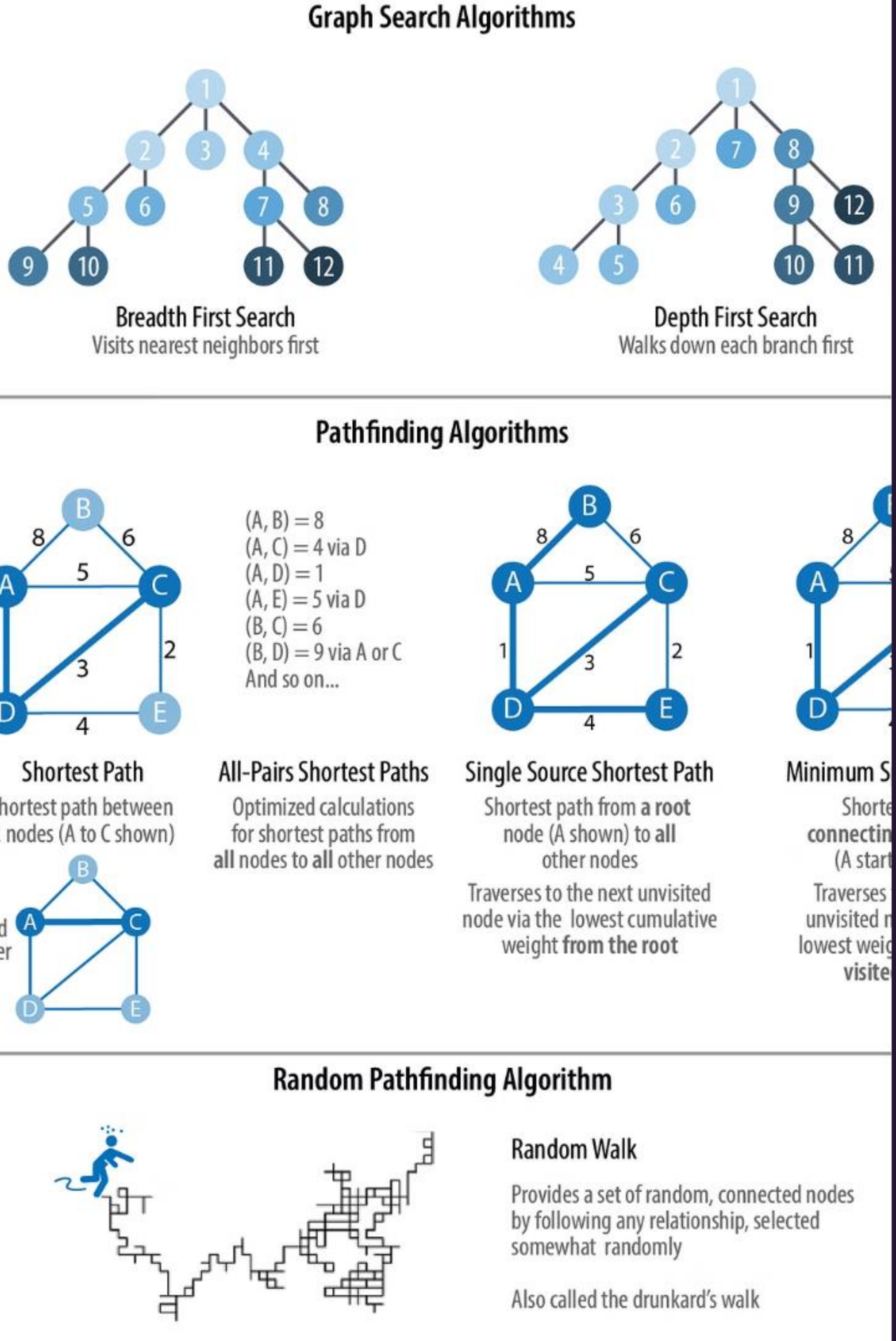
Introduction to A* Algorithm



Visits nearest neighbors first

Walks down each branch first

hortest path between nodes (A to C shown)

Optimized calculations for shortest paths from all nodes to all other nodes

Shortest path from a root node (A shown) to all other nodes

Traverses to the next unvisited node via the lowest cumulative weight from the root

Shorte connectin (A start

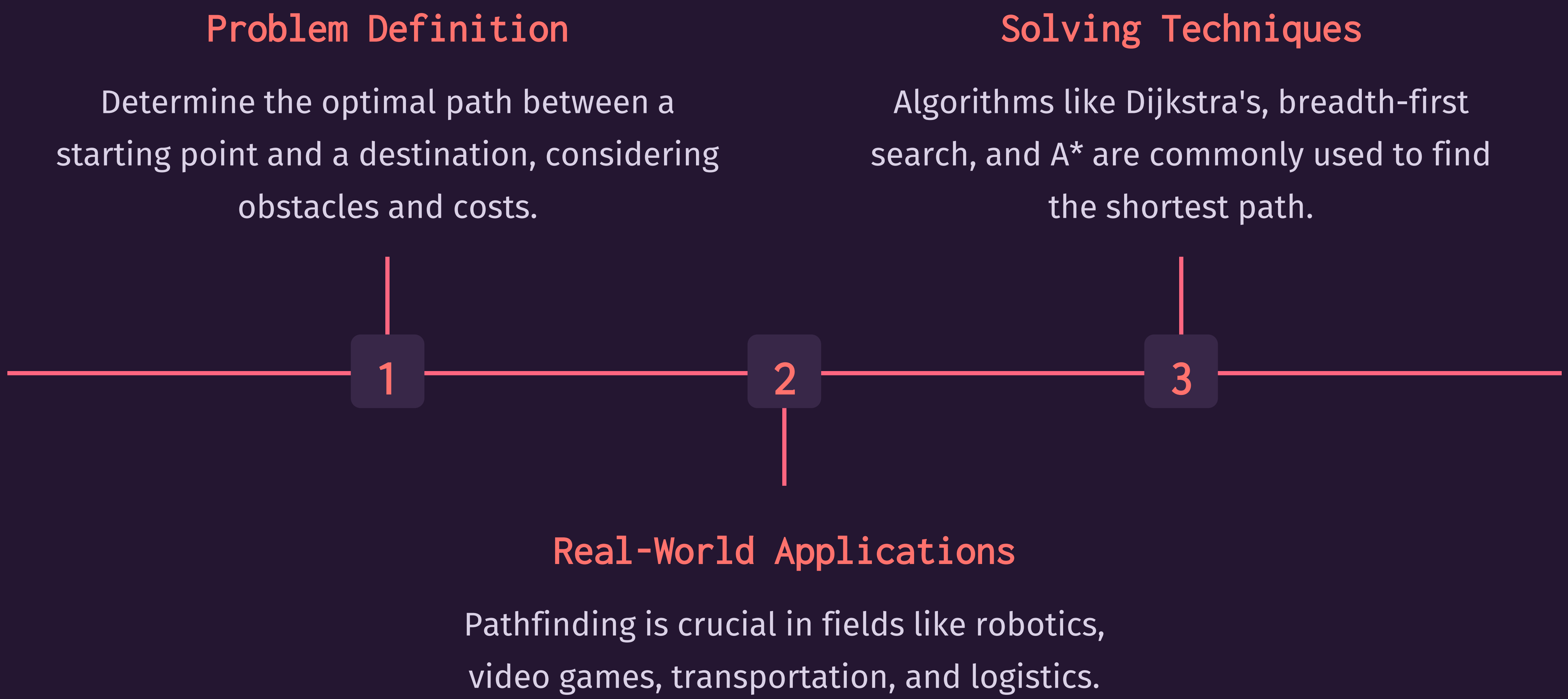
Traverses unvisited n lowest weig visite

Provides a set of random, connected nodes by following any relationship, selected somewhat randomly

Also called the drunkard's walk

The A* algorithm is a widely used pathfinding algorithm that efficiently finds the shortest path between two points. It utilizes a heuristic function to estimate the cost to the goal, allowing it to prioritize the most promising routes and avoid exploring unnecessary areas.

Pathfinding and Shortest Path Problems



Problem Definition

Determine the optimal path between a starting point and a destination, considering obstacles and costs.

Real-World Applications

Pathfinding is crucial in fields like robotics, video games, transportation, and logistics.

Solving Techniques

Algorithms like Dijkstra's, breadth-first search, and A* are commonly used to find the shortest path.

Heuristic Functions and their Importance

Defining Heuristics

Heuristic functions estimate the cost-to-go from the current node to the goal. They guide the A* algorithm towards the most promising paths.

Importance of Heuristics

Well-designed heuristics can significantly improve the efficiency of the A* algorithm, leading to faster and more accurate pathfinding.

Examples of Heuristics

Common heuristics include Euclidean distance, Manhattan distance, and weighted combinations of these.

The A* Algorithm Step-by-Step Explanation

1

Initialize

Start with the initial node and an open list of nodes to explore.

2

Evaluate

Calculate the f-cost (g-cost + h-cost) for each neighboring node.

3

Expand

Add the node with the lowest f-cost to the closed list and explore its neighbors.

4

Repeat

Continue this process until the goal node is reached or no more nodes can be explored.

Advantages and Disadvantages of A* Algorithm

1 Advantages

Finds the shortest path, is optimal, and can handle complex environments with obstacles.

2 Disadvantages

May require significant memory for large search spaces, and the performance depends on the quality of the heuristic function.

3 Trade-offs

The A* algorithm balances efficiency and optimality, making it a widely adopted choice for pathfinding problems.

Applications of A* Algorithm

Robotics

Used for robot navigation, path planning, and obstacle avoidance.

Video Games

Employed for non-player character (NPC) pathfinding and navigation.

Transportation

Utilized for route optimization in logistics, GPS navigation, and traffic management.

GIS

Applied in geographic information systems for finding optimal paths and routes.

Conclusion and Key Takeaways

Efficient Pathfinding

The A* algorithm is a powerful tool for finding the shortest path in various applications.

Heuristic Design

The choice of heuristic function is crucial for the algorithm's performance and accuracy.

Versatile Applications

A* algorithm is widely used in robotics, video games, transportation, and geographic information systems.

Continuous Improvement

Ongoing research and advancements in the A* algorithm aim to enhance its capabilities.

Thank you

- satvik verma (221030173)
- akash gupta (221030174)
- samarth sharma (221030183)