Chapter 2 - Search Problems

A **search problem** is a situation where an AI system must explore different possible solutions to find the best or most optimal outcome. The AI does this by systematically searching through possible choices, considering constraints, and selecting the most efficient path to reach a goal.

In real life, search problems are everywhere. Whether it's finding the fastest way to work, solving a puzzle, or navigating through an unfamiliar area, AI uses search algorithms to efficiently solve these problems. Different search algorithms exist, each suited to specific types of problems.

Key Components of a Search Problem

Every search problem consists of three main parts:

1. **Initial State** – The starting point of the problem (e.g., your current location in a city when looking for directions).
2. **Actions & Transitions** – The possible moves or decisions that can be made (e.g., turning left or right at an intersection).
3. **Goal State** – The desired outcome or solution (e.g., arriving at your destination).

A search algorithm processes these components to determine the best or most efficient path from the initial state to the goal state.

Examples of Search Problems in Everyday Life

1. Finding the Fastest Route (GPS Navigation)

- When you use Google Maps or Waze, the app searches for the best driving route by analyzing multiple factors: distance, road conditions, and live traffic updates.
- **Algorithm Used:**
  - **Dijkstra's Algorithm** finds the shortest path between locations.
  - _*A* Search_* improves efficiency by considering both distance and estimated time.

2. Playing Tic-Tac-Toe (Game AI)

- An AI opponent in a tic-tac-toe game searches through all possible moves and their outcomes before deciding on the best move.
- **Algorithm Used:**
  - **Minimax Algorithm** ensures the AI makes the best possible move while minimizing the opponent's chances of winning.

3. Solving a 15-Puzzle (Logical Puzzle)

- The 15-puzzle is a sliding tile game where a player must move tiles into the correct order by shifting them within a grid.
- AI can solve this puzzle by searching for the shortest sequence of moves leading to the solved state.
- **Algorithm Used:**
  - _*A* Search Algorithm_* finds the most efficient series of moves.

4. Finding Your Way Through a Maze (Autonomous Navigation)

- A robot, like a robotic vacuum cleaner, must navigate through obstacles in a room to clean efficiently or reach a charging station.
- **Algorithm Used:**
  - **Breadth-First Search (BFS)** explores all possible paths layer by layer.
  - **Depth-First Search (DFS)** explores one path deeply before backtracking.

Exercises

1. Fastest Route to Work

Imagine you are commuting to work and have three different routes:

- Route A is the shortest in distance but has heavy traffic.
- Route B is slightly longer but has fewer traffic lights.
- Route C is the longest but goes through scenic roads with no traffic.

  Question: If an AI navigation system had to choose the best route, which factors would it consider? Which search algorithm (Dijkstra's, A*, etc.) would be most effective?

2. Searching for a Lost Phone

You lost your phone somewhere in your house. You can use two different strategies:

1. **Depth-First Search (DFS)**: Search one room entirely before moving to the next.
2. **Breadth-First Search (BFS)**: Quickly scan all rooms before doing a detailed search.

Question: Which strategy would be more effective and why? Consider scenarios where one method might be better than the other.

Search Problem Terminologies

In AI, **search problems** involve an agent that explores different possibilities to find a solution. To better understand how AI searches for solutions, we must define the fundamental terms used in search problems.

---

1. Agent

An **agent** is an entity that interacts with its environment by perceiving information and making decisions to achieve a goal. The agent can be a person, an AI program, a robot, or any system capable of taking actions.

Real-Life Examples:

- **Self-Driving Car**: The car perceives traffic lights, road signs, and pedestrians, then decides when to stop, turn, or accelerate.
- **Chess AI**: The AI "sees" the board and determines the best move based on the current configuration.
- **Robot Vacuum Cleaner**: The vacuum detects walls, furniture, and dirt, deciding where to clean next.

---

2. State

A **state** represents a specific situation or condition of the agent in its environment at a given moment. It includes all necessary information that defines the current situation.

Real-Life Examples:

- **Self-Driving Car**: A state could be the car's location, its speed, and whether the traffic light is red or green.

- **Chess AI**: A state could be the exact arrangement of pieces on the board at a particular moment.
- **Robot Vacuum**: A state could be the current position of the vacuum and the areas that are already cleaned.

---

3. Initial State

The **initial state** is where the agent starts its search for a solution. It is the starting point of the problem.

Real-Life Examples:

- **Self-Driving Car**: The car starts at your house before beginning the journey.
- **Chess AI**: The initial state is the standard chessboard setup before any moves are made.
- **Robot Vacuum**: The vacuum starts from its charging dock before it begins cleaning.

---

4. Goal

The **goal** is the desired state that the agent is trying to reach. Once the agent reaches this state, the search problem is solved.

Real-Life Examples:

- **Self-Driving Car**: The goal is to reach the correct destination using the fastest or safest route.
- **Chess AI**: The goal is to checkmate the opponent’s king.
- **Robot Vacuum**: The goal is to clean all areas of the room before returning to the charging dock.

---

5. Actions

Actions are the possible moves or decisions the agent can take in a given state to reach the goal. The agent chooses from these actions based on the problem's rules and constraints.

Real-Life Examples:

- **Self-Driving Car**: Actions include accelerating, braking, turning left, turning right, or stopping.
- **Chess AI**: Actions involve moving a piece to a valid position based on the game's rules.
- **Robot Vacuum**: Actions include moving forward, turning left, turning right, or stopping to avoid obstacles.

---

6. Transition Model

The **transition model** defines how the agent moves from one state to another based on the chosen action. It determines the new state that results from performing an action in the current state.

Real-Life Examples:

- **Self-Driving Car**: If the car is at an intersection and chooses the action "turn left," the transition model updates the car's position accordingly.
- **Chess AI**: If the AI moves a bishop diagonally, the transition model updates the board to reflect the new piece position.
- **Robot Vacuum**: If the vacuum chooses "move forward," the transition model updates its location. If an obstacle is detected, the transition model may force the vacuum to stop or change direction.

Summary Table

Term	Definition	Example (Self-Driving Car)	Example (Chess AI)	Example (Robot Vacuum)
Agent	Entity that perceives and acts	The car navigating roads	The AI deciding moves	The vacuum navigating a room
State	Current situation of the agent	Car's location, speed, and traffic	Chessboard configuration	Vacuum's position and cleaned areas
Initial State	Where the agent starts	Car at home before trip	Standard chess setup	Vacuum at charging dock
Goal	Desired outcome	Reaching the destination	Checkmate opponent	Clean all rooms

Term	Definition	Example (Self-Driving Car)	Example (Chess AI)	Example (Robot Vacuum)
Actions	Possible moves the agent can make	Turn, accelerate, stop	Move a piece	Move, turn, stop
Transition Model	How actions change the state	Turning left updates car's location	Moving a bishop updates board	Moving forward updates vacuum's position

---

Exercises

1. Grocery Shopping with an AI Assistant

Imagine you have an AI assistant helping you complete a shopping list at a supermarket.

- **Agent**: The AI assistant guiding you.
- **State**: Your current position in the supermarket and items in your cart.
- **Initial State**: You enter the supermarket with an empty cart.
- **Goal**: Collect all items on your shopping list.
- **Actions**: Move to different aisles, pick up an item, check the list.
- **Transition Model**: If you pick up an apple, the shopping list updates, and you move to the next needed item.

  Question: How would the AI decide the best route through the store to collect all items efficiently?

---

2. Finding a Lost Pet

Your pet cat has gone missing, and you need to search for it in your neighborhood.

- **Agent**: You, searching for the cat.
- **State**: Your current location and whether you have found clues.
- **Initial State**: You start at home with no knowledge of where the cat is.
- **Goal**: Find the cat.
- **Actions**: Search the backyard, ask neighbors, check under cars.

- **Transition Model**: If a neighbor says they saw the cat in a park, your search location updates.

  Question: What strategy would be most efficient for finding your pet—searching nearby first (Breadth-First Search) or checking hiding spots deeply (Depth-First Search)? Why?