**Simplified AI Lecture 3 Summary: Solving Problems by Searching Algorithm**

**What Are Search Algorithms?**

* Search algorithms help AI find solutions to problems.
* They help the AI navigate from a starting point to a goal.
* Different search methods are used for different types of problems.

**Types of Search Algorithms**

There are two main types of search algorithms:

**1. Uninformed (Blind) Search**

* Has no background information about the problem.
* Number of steps and path cost are unknown
* Agent only knows when it reaches a goal
* Examples: BFS, DFS, UCS

**2. Informed (Heuristic) Search**

* Uses background information about the problem.
* Has estimates about how far the goal is.
* More efficient than uninformed search.
* Examples: Greedy Best-First, A\* Search

**Uninformed Search Strategies**

**Breadth-First Search (BFS)**

* Explores all nodes at the current level before moving to the next level.
* Uses a FIFO queue (First In, First Out)
* Complete algorithm (guarantees a solution if one exists).
* Searches level by level, starting from the root.

**Uniform-Cost Search (UCS)**

* Prioritizes the lowest-cost path when multiple paths exist.
* Visits the next node which has the least total cost from the root.
* Ensures optimality if no negative costs are present.
* Ensures completeness if states are finite with no zero-weight loops.
* UCS behaves like BFS when all transitions have equal costs.

**Depth-First Search (DFS)**

* Explores branch nodes deeply before backtracking
* Uses a LIFO stack (Last In, First Out)
* Goes as deep as possible along one path before backtracking
* May not find the shortest path

**Bidirectional Search**

* Runs two searches at the same time
* One search from the start state going forward
* Another search from the goal state going backward
* They meet in the middle at a common node
* This reduces the search space

**Informed Search Strategies**

**Greedy Best-First Search**

* Uses a heuristic function h(n) that estimates the cost to the goal.
* Always picks the node that seems closest to the goal.
* May not find the optimal solution.
* Can get stuck in loops or dead ends.
* Uses a priority queue for implementation.

**A\* Search**

* Combines the best of UCS and Greedy Best-First Search
* Uses the evaluation function f(n) = g(n) + h(n)
  + g(n) is the path cost from start to current node
  + h(n) is the estimated cost from current node to goal
* UCS keeps solution cost low
* Best-first helps find solution quickly
* A\* combines these approaches for an efficient search

**Important Concepts**

* Traversal: Visit each node once
* Search: Find a path between two nodes
* Complete algorithm: Guarantees finding a solution if one exists
* Optimal solution: The solution with the lowest cost
* Heuristic function: An estimate of the cost to reach the goal