**Course 5: Deep Learning**

**Problem Solving:**

Problem solving refers to the process of finding a solution to a given problem. In the context of artificial intelligence, problem solving involves identifying an effective approach or technique to solve a particular problem. In engineering and mathematics, problem solving is often seen as an optimization process, where the objective is to find the best possible solution.

**Steps of Problem Solving:**

* **Problem Definition:** This step involves clearly understanding and defining the problem at hand. It includes identifying the goals, constraints, and any specific requirements of the problem.

**Example:** Suppose you want to design a route planning system. The problem definition would involve understanding the start and end points, any intermediate stops, and any constraints such as time limitations or avoiding certain routes.

* **Problem Analysis:** In this step, you analyze the problem to gain a deeper understanding of its underlying structure, dependencies, and potential challenges. It helps in identifying the key components and relationships within the problem.

**Example:** Analyzing the route planning problem might involve examining the road network, considering traffic patterns, and understanding the availability of alternative routes.

* **Selection of a Solving Technique:** Once the problem is defined and analyzed, you need to choose an appropriate technique or approach to solve the problem. This can involve search algorithms, knowledge representation methods, abstraction techniques, or other problem-solving strategies.

**Example:** For the route planning problem, you might consider using search algorithms to find the optimal path or heuristic methods to quickly find a near-optimal solution.

**Solving Problems by Search:**

Solving problems by search is a common approach used in artificial intelligence. It involves systematically exploring the problem space to find a solution. The problem space consists of all possible states and the operators that can transform one state into another.

**Steps of Solving Problems by Search:**

Solving problems by search is a common approach in AI, where the problem is decomposed into a set of actions that can be taken to achieve the desired objectives. Each action in the search process results in a change of the problem's state.

The problem definition in **search-based problem solving** typically includes the following components:

* Search Space: The search space represents all the possible states of the problem. It can be explicitly constructed by enumerating all possible states or implicitly defined using data structures and operators.
* Initial State: One or more initial states from which the search process starts.
* Final State(s): One or more states that represent the desired solution or goal of the problem.
* Paths: Paths are sequences of successive states that connect the initial state to a final state.
* Rules (Actions): The set of rules or operators that define how a state can be transformed into another state. These rules are often represented by successor functions that generate the next state given a current state.

**Example 1: Puzzle Game with 8 Pieces**

Let's consider a puzzle game with 8 pieces, where the search space consists of different board configurations. The initial state is a random configuration, and the final state is a configuration where all the pieces are sorted in a specific manner. The rules define the movements of the white pieces within the table. The objective is to find an optimal sequence of white moves to reach the final state.

**Example 2: Queen's Problem**

In the Queen's problem, the search space consists of different board configurations for a game with N queens. The initial state is a configuration without any queens, and the final state is a configuration where N queens are placed such that no two queens can attack each other in one move. The rules involve putting a queen on the board, with conditions that ensure the queen is not attacked by any other queen. The solution is an optimal placement of the queens.

When **solving problems by search**, several factors should be considered:

* Decomposition: Determine whether the problem can be decomposed into sub-problems that are independent or have interdependencies.
* Predictability: Understand the predictability of the possible state space. Can the states be explicitly constructed, or are they generated based on some rules or functions?
* Solution Type: Define whether you need to find any solution or an optimal solution. Optimal solutions minimize costs or maximize benefits.
* Representation: Decide whether the solution is represented by a single state or a sequence of successive states (path).
* Knowledge: Determine if any prior knowledge or constraints can be used to limit the search or guide the solution identification process.
* Interaction: Assess whether the problem requires human interaction or can be fully automated.

**Search Strategies:**

* Search strategies are techniques or algorithms used to guide the search process in problem solving. Different search strategies have different characteristics, such as computational complexity, completeness, and optimality.

**Search strategies** determine how the search space is explored to find a solution. Different search strategies have different exploration mechanisms and can be selected based on the problem's characteristics. Here are some common search strategies:

* **Breadth-First Search (BFS):** Explores all neighboring states of the current state before moving to the next level of the search tree. BFS guarantees finding the optimal solution, but it may require a large amount of memory.
* **Depth-First Search (DFS):** Explores as far as possible along each branch before backtracking. DFS may find a solution quickly but does not guarantee optimality.
* **Uniform Cost Search (UCS):** Explores states with the lowest cost first, based on a cost function. UCS guarantees finding the optimal solution but can be computationally expensive.
* **Iterative Deepening Search (IDS):** Performs a series of depth-first searches with increasing depth limits until a solution is found. IDS combines the advantages of BFS and DFS.

**A heuristic search algorithm** that evaluates the cost of a path (g(n)) and an estimate of the remaining cost to the goal (h(n)). It uses a heuristic function (often denoted as f(n) = g(n) + h(n)) to guide the search process and find the optimal solution efficiently.

These are just a few examples of search strategies, and many variations and hybrid approaches exist. The choice of a search strategy depends on factors such as computational resources, problem characteristics, desired solution quality, and available domain knowledge.