

**PRINCIPLES OF**

**ARTIFICIAL INTELLIGENCE**

(19CSE451)

Case Study-1

**Snake and Ladder Game**

**Team:**

|  |  |
| --- | --- |
| Kosuri Bhavya Naga Sai | CB.EN.U4CSE22328 |
| Krishnam Abhina | CB.EN.U4CSE22331 |

**Problem Statement:**

Snake and Ladder Problem

To create an AI agent for the Snake and Ladder game, the goal is to find the minimum number of throws required to win the game(to reach the last square (100) ).Each board state is analysed by the agent, prioritizing ladders for forward movement and avoiding snakes that send the player backward.

**Problem Explanation:**

1. Graph Representation:
   * The Snake and Ladder board is modeled as a directed graph, where each square on the board corresponds to a vertex in the graph.
   * From any vertex v, there are directed edges to v+1,v+2,v+3,v+4,v+5andv+6, reflecting the possible moves based on the outcome of a dice throw.
2. Handling Ladders and Snakes:
   * If moving to any of these vertices leads to a square with a ladder, the edge is redirected to the top of the ladder.
   * If it leads to a square with a snake, the edge is redirected to the tail of the snake.
3. Shortest Path Problem:
   * The objective is to find the shortest path from the starting vertex (0) to the final vertex (99), which is the winning position.
   * This is effectively a shortest path problem in a directed graph, and we will use the Breadth-First Search (BFS) algorithm to solve it.

**Algorithm (CSP/Searching) for Solving:**

1. **Breadth-First Search (BFS):**  
   BFS is ideal for optimizing Snake and Ladder gameplay by exploring all possible moves level by level, ensuring the shortest path (minimum moves) is found. Each board position is treated as a node, with dice rolls as transitions between nodes. BFS systematically checks all move sequences, ensuring the agent finds the optimal path to reach the last square (100) while efficiently utilizing ladders and avoiding snakes.

**References:**

1. **Artificial Intelligence: A Modern Approach** by Stuart Russell and Peter Norvig – This book provides a detailed explanation of search algorithms like BFS, UCS, and A\*.
2. **GeeksforGeeks – Solving Snake and Ladder Problem Using BFS**: Provides a step-by-step breakdown of applying BFS to solve the Snake and Ladder game efficiently.
3. **AI Algorithms for Games by University of Alberta**: Offers insights into game-based AI algorithms that are applicable to board games like Snake and Ladder.

**Identified Generative AI Tool for Presentation:**

1. **Gamma AI** is a powerful generative AI tool for creating presentations. It offers AI-driven design suggestions and automated layout generation based on the content provided. It can assist in creating visually appealing slides for presenting AI concepts and solutions.

**BFS Algorithm:**

1. **Graph Initialization**:
   * Create an adjacency list representation of the directed graph using the edges defined by the moves allowed on the board.
2. **Add Edges**:
   * For each square (vertex) from 0 to 99, add directed edges to the next 1 to 6 squares.
   * If moving to any of these squares leads to a ladder or a snake, redirect the edge to the corresponding destination (top of the ladder or tail of the snake).
3. **BFS Implementation**:
   * Initialize a queue to facilitate BFS.
   * Keep track of discovered vertices to avoid revisiting them.
   * Enqueue the starting vertex (0) with an initial distance of 0.
   * While the queue is not empty:
     + Dequeue the front vertex and check if it is the destination (99).
     + If it is the destination, return the distance (minimum moves).
     + For each adjacent vertex, if it has not been discovered:

.Mark it as discovered and enqueue it with an incremented distance.

1. **Return Result**:
   * If the destination is reached during BFS, return the total number of moves taken to reach it.

**Snake & Ladder Board:**

