ensia The National School of Artificial Intelligence المدرسة الوطنية العليا للذكاء الاصطناعي

Introduction to Artificial Intelligence

Lab 6 (Week 6) -Search Algorithms: Part 2 - More on Uninformed Search

2023 - 2024

10 March 2024

Objectives

- Implementation of the Depth-Limited Search (DLS) and Iterative Deepening Search (IDS) algorithms for the Travel Planning problem.
- Implementation of the Uniform Cost Search (UCS) algorithm for the Travel planning problem.

Overview:

In LAB 5, you effectively implemented the general graph search algorithm, successfully solving both the provided Eight Puzzle problem and the previously defined Travel Planning problem using some Uninformed search strategies (BFS and DFS).

In this current lab, your task is to extend the Travel Planning problem from LAB 5 by incorporating additional information. Additionally, you are required to implement and test three strategies—Depth-Limited Search, Iterative Deepening Search, and Uniform Cost Search—with the newly formulated problem.

Your mission

Your mission in this lab is to perform the following tasks using the Graph Search Algorithm implemented:

Task 1

In this task, you are presented with an extended version of the Travel Planning problem, incorporating new information. In this modified scenario, the objective remains to find the path from an initial state to a chosen goal state. The new attributes in this problem include the actual distance between neighboring states and the coordinates (latitude and longitude) of each state.

The connections between states are as follows:

—"Algiers" :
Coordinates: 36.7528, 3.0420,
Neighbors: "Oran": 450, "Constantine": 320, "Tizi Ouzou": 80, "Bechar": 700, "Setif": 200.

—"Oran":
Coordinates: 35.6969, −0.6331,
Neighbors: "Algiers": 450, "Tlemcen": 200, "Mascara": 150, "Skikda": 550, "Bejaia": 400.

—"Constantine":
Coordinates: 36.3650, 6.6147,
Neighbors: "Algiers": 320, "Annaba": 210, "Setif": 120, "Ghardaia": 600.

—"Tizi Ouzou":
Coordinates: 36.7117, 4.0456,
Neighbors: "Algiers": 80, "Bejaia": 150, "Bouira": 100, "Adrar": 800.

—"Tlemcen":
Coordinates: 34.8888, −1.3153,
Neighbors: "Oran": 200, "Sidi Bel Abbes": 120, "Mascara": 180, "Guelma": 350.

—"Annaba":
Coordinates: 36.9060, 7.7465,
Neighbors: "Constantine": 210, "Guelma": 100, "Skikda": 90, "El Oued": 450.

—"Bechar":
Coordinates": 31.6304, −2.2687,
Neighbors: "Algiers": 700, "Adrar": 100, "Tindouf": 300.

—"Setif":
Coordinates": 36.1869, 5.4175,
Neighbors: "Algiers": 200, "Constantine": 120, "Ghardaia": 300, "El Oued": 500.

—"Bejaia":
Coordinates: 36.7508, 5.0564,
Neighbors: "Oran": 400, "Tizi Ouzou": 150, "Bouira": 250, "Tindouf": 600.

—"Mascara":
Coordinates: 35.3984, 0.1401,
Neighbors: "Oran": 150, "Tlemcen": 180, "Guelma": 400, "El Oued": 550.

—"Guelma":
Coordinates: 36.4629, 7.4267,
Neighbors: "Tlemcen": 350, "Annaba": 100, "Mascara": 400, "Tindouf": 700.

—"Skikda":
Coordinates: 36.8796, 6.9036,
Neighbors: "Oran": 550, "Annaba": 90, "El Oued": 350, "Tindouf": 800.

—"Ghardaia":
Coordinates: 32.4882, 3.6733,
Neighbors: "Constantine": 600, "Setif": 300, "Adrar": 400.

–"Bouira":
Coordinates: 36.3724, 3.9007,
Neighbors: "Tizi Ouzou": 100, "Bejaia": 250, "Tindouf": 650.

–"Adrar":
Coordinates: 27.8617, −0.2917,
Neighbors: "Bechar": 100, "Tizi Ouzou": 800, "Ghardaia": 400.

–"El Oued":
Coordinates: 33.3564, 6.8631,
Neighbors: "Annaba": 450, "Mascara": 550, "Skikda": 350.

–"Tindouf":
Coordinates: 27.6706, −8.1476,
Neighbors: "Bechar": 300, "Bejaia": 600, "Guelma": 700, "Skikda": 800, "Bouira": 650.

Considering these constraints you are asked to:

- Modify the Travel Planning class from LAB 5 to consider the newly added attributes (This involves adjusting the existing methods to account for the coordinates and distances between cities).

Task 2

For this second task, you are given pseudo algorithms for Depth Limited Search and Iterative Deepening Search presented in Algorithms 1 and 2 respectively.

Algorithm 1: Depth Limited Search algorithm

Input: $Problem, Max\_depth$;
Output: $Solution$;
Call DFS algorithm with Max_depth ;
Explore a node only if it has depth less than Max_depth ;

Algorithm 2: Iterative Deepening Search algorithm

Input: $Problem, Max\_depth$;
Output: $Solution$;
Initialize solution to None ;
Initialize Max_depth;
while $solution == None$ **do**
    solution = Depth_Limited_Search(Problem, Max_depth);
    Increment Max_depth ;
end

Your goal is to :

- Implement the corresponding functions (or methods) utilizing the Graph Search General Algorithm implemented in LAB 5.
- Solve the Travel Planning problem Using both Strategies.
- Display the following: the cost of the solution node, the number of steps (explored nodes) leading to the solution node and the depth of the solution node.

Task 3

For the third task, your objective is to implement the Uniform Cost Search Algorithm. This algorithm considers the actual distance between two cities as a cost and searches for a solution path using these distances. During each iteration of the search process, it selects the node to expand, the one on the frontier with the minimal cost from the root.

Moreover, you are given an improved Node class that incorporates a ___gt___() magic method, enabling the algorithm to compare nodes based on their **cost**.

```
class Node:
    def __init__(self, state, parent=None, action=None, cost=0):
        self.state = state
        self.parent = parent  # node
        self.action = action  # action performed to get to this node
        self.cost = cost      # (incremented with each newly expanded node)
        if parent is None:    #root node
            self.depth = 0    # level in the graph 0 for the root node
        else:
            self.depth = parent.depth + 1 # parent level + 1

    def __hash__(self):
        if isinstance(self.state, list):
            return hash(tuple(map(tuple, self.state)))
        return hash(self.state)

    def __eq__(self, other):
        return isinstance(other, Node) and self.state == other.state

    def __gt__(self, other):
        ..................          # Fill in the gap
```

In line with the provided instructions, perform the following tasks:

- Fill in the missing components of the given `Node` class.
- Modify the Graph General Search algorithm to incorporate the Uniform Cost Search strategy.
- Solve the Travel Planning problem in Task 1 using the Uniform Cost Search strategy.
- Display the following scores:
  - The number of steps (explored nodes) from the initial to goal node.
  - The cost (distance) from the initial to goal node.
  - The depth of the goal node.

Discussion

- On the basis of a comparison of the above strategies using the given metrics, discuss which strategy you find better and why.