School of Computing Sciences

Module: CMP-(6040A/7028A) Artificial Intelligence

Assignment: Solution of the 8-Tile problem

Set by	: Pierre Chardaire e-mail: pc@uea.ac.uk
Date set	: 6 October 2020
Value	: 20%
Date due	: 10 November by 15:00
Returned by	: Week 11
Submission	: Blackboard

Learning outcomes

Apply techniques previously learned in the course to the solution of a search problem. Develop programming skills.

Specification

Overview

The aim of this coursework is to develop solutions for a classical AI search problem and to contrast the efficiency of the techniques implemented.

Description

In this coursework you are asked to implement Depth First Search with Iterative Deepening (DFID) and Iterative Deepening A* (IDA*) for the solution of the 8-Tile problem.

Using the state representation seen in class, the goal state will be:
`[2,2,[[1,2,3],[4,5,6],[7,8,0]]]`
corresponding to the configuration:

1	2	3
4	5	6
7	8	

You are asked to solve the following instances in the smallest number of moves:

```
[0, 0, [[0, 7, 1], [4, 3, 2], [8, 6, 5]]],
[0, 2, [[5, 6, 0], [1, 3, 8], [4, 7, 2]]],
[2, 0, [[3, 5, 6], [1, 2, 7], [0, 8, 4]]],
[1, 1, [[7, 3, 5], [4, 0, 2], [8, 1, 6]]],
[2, 0, [[6, 4, 8], [7, 1, 3], [0, 2, 5]]],
[0, 2, [[3, 2, 0], [6, 1, 8], [4, 7, 5]]],
[0, 0, [[0, 1, 8], [3, 6, 7], [5, 4, 2]]],
[2, 0, [[6, 4, 1], [7, 3, 2], [0, 5, 8]]],
[0, 0, [[0, 7, 1], [5, 4, 8], [6, 2, 3]]],
[0, 2, [[5, 4, 0], [2, 3, 1], [8, 7, 6]]],
[2, 1, [[8, 6, 7], [2, 5, 4], [3, 0, 1]]].
```

Note that the last instance is the hardest instance in the sense that no other instance of the problem requires more moves.

Both codes, for DFID and for IDA*, should output the following for each instance:

- the number of moves to solve it (i.e. the number of states minus 1 in the shortest path from the instance to the goal),
- the number of calls to your `move` procedure made during the search for a solution, and
- the computing time the search took.

The code for DFID may reuse the `move` method seen in the lectures and adapt the DFS code provided. Note that you may have to deal with the fact that lists are passed by reference (i.e. deep copy may be needed).

The code for IDA* should rely on an h function made of the sum of evaluation functions of the numbered tiles, where the evaluation of a tile is the Manhattan distance between its position in the current state and its position in the goal state (This is the minimum number of moves a tile has to make to get into its final position.).

Relationship to formative assessment

Extension of work on search methods seen in labs.

Deliverables

You are required to produce two independent Python programs to solve the 8-tile problem using

1. Iterative Deepening Depth First Search (program should be named `IDS.py`)
2. Iterative Deepening A* (program should be named `IDAstar.py`)

The codes must be clearly commented.

You must provide a comment at the start of each program that contains the results you were asked to output, that is the output of the solution method used in the program for each of the provided instances in the order they are listed in this document.

Resources

- Lecture notes.
- Labsheet solution on search.
- The Python tutorial: `https://docs.python.org/3.7/tutorial/index.html`

Marking scheme

Each of the two programs will be worth 50 marks, with 20 marks for the correctness of the code, 20 marks for its efficiency and 10 marks for its readibility (relevant comments are expected).