EAI 320

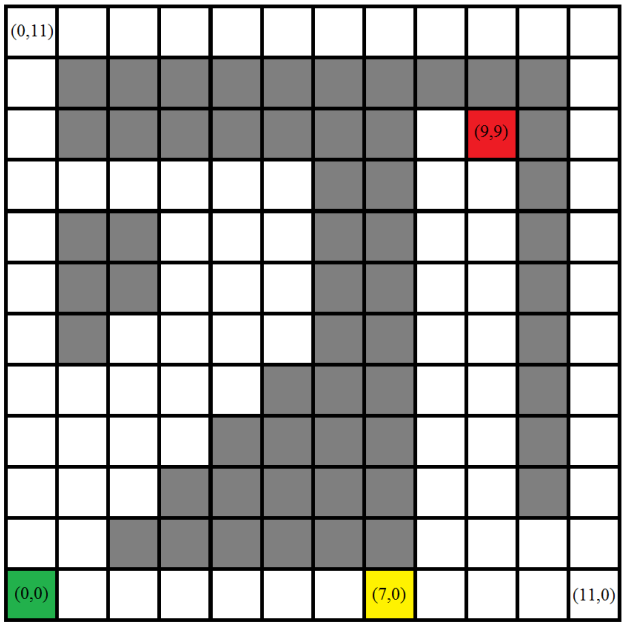
Practical Assignment 2

17 February 2016

Compiled by Hugo Coppejans

Problem

You have been assigned to design an AI agent for a computer game. The goal of the agent is to find a path from a starting point to a specified goal point. The environment is given as follows:



The green block is the starting point. The red block is the goal point. The grey blocks are walls. The agent cannot navigate to or over these blocks. The yellow block is used for question 2. Ignore it in question 1.

The agent can move vertically, horizontally and diagonally, one block at a time. This implies that, with exception to locations adjacent to the environment edges and the walls, the agent can move in 8 possible directions.

In python, this environment could be represented by the following list:

```
[[0,0,0,0,0,0,0,0,0,0,0,0],
[0,0,0,0,0,1,1,1,0,1,1,0],
[0,1,0,0,0,0,1,1,0,1,1,0],
[0,1,1,0,0,0,0,0,0,1,1,0],
[0,1,1,1,0,0,0,0,0,1,1,0],
[0,1,1,1,1,0,0,0,0,1,1,0],
[0,1,1,1,1,1,1,1,1,1,1,0],
[1,1,1,1,1,1,1,1,1,1,1,0],
[0,0,0,0,0,0,0,0,0,0,1,0],
[0,0,0,0,0,0,0,0,0,0,1,0],
[0,0,1,1,1,1,1,1,1,1,1,0],
[0,0,0,0,0,0,0,0,0,0,0,0]]
```

This is a tree search problem. The root node describes the state where the agent is located at (0,0). The tree is expanded by moving the agent. The next layer in the tree thus consists of the locations (0,1), (1,1) and (1,0). Each of these nodes can then be expanded. (Dont try grow the whole tree). Read through all the questions before you start.

Question 1

Question 1a

Implement the uniform cost search to find the path from (0,0) to (9,9). Use the Euclidean distance to assign a cost for moving from a current square its neighbour. This will result in a cost of 1 to move vertically or horizontally and a cost of sqrt(2) to move diagonally. The path cost is an accumulation of the cost of each move as you traverse down the tree.

Question 1b

Implement the greedy best search to find the path from (0,0) to (9,9) using the Euclidean distance as a heuristic.

Question 1c

Implement the A* search to find the path from (0,0) to (9,9). Use the heuristic used in question 1b and the path cost used in question 1a.

Question 1d

Document the results. In your results, include a plot consisting of

1. the path that is found,
2. the nodes that were explored (expanded) and
3. the walls

(The plot can be constructed using markers with different colours and sizes)

Question 1e

Provide a discussion which compares the methods in terms of which finds the shortest path and why.

Question 2

Assume that the yellow block at (7,0) is a wall. Compare the results of the uniform cost search, the greedy best search algorithm and the A* algorithm for this case.

Question 3

Discuss the properties of the implemented algorithms as follows:

Question 3a

Completeness: Do the algorithms find the solution? Under what circumstance would the algorithms fail to find a solution?

Question 3b

Time: In your code, count the number of branches in the search tree. Each time a node is expanded, you take a set of actions to form branches to new nodes. You need to count how many times you take an action in total.

Question 3c

Space: In your code, note how large the open set (or frontier queue see hint 4) grows in each algorithm. Explain how this could become a problem in worlds with larger state spaces.

Question 3d

Optimality: Discuss under which circumstances each algorithm is optimal.

HINTS

1. *When expanding a node, you consider the actions you can take. You can move to one of the 8 surrounding blocks if you are not located at the environment edge or a wall. You could write a function that returns the set of neighbours for a given location.*

2. *When you consider expanding a node, check if its value in the environment matrix is 1. If it is, ignore the node.*

3. *You need to have a memory structure consisting of the nodes that need to be investigated (open set) and the nodes that have been investigated (closed set). Additionally, you need to keep track of node parents for the path reconstruction when you find a solution. See* `https://en.wikipedia.org/wiki/A*_search_algorithm` *for an example of pseudo code.*

4. *The three algorithms are identical other than the fact that they use different cost functions. (Uniform cost search uses g(n), greedy best first search uses h(n) and A* uses g(n)+h(n)).*

5. *Your program should consist of three functions:*

    - *A search function (Can do greedy, uniform cost or A*)*
    - *A heuristic function (Implements Euclidean distance)*
    - *A successor function (returns all the states you could move too)*

Deliverables

- Write a technical report on your finding for this assignment.
- Include your code in the digital submission as an appendix, but leave it out for the hardcopy submission.

Instructions

- All reports must be in PDF format and be named report.pdf.
- Place the software in a folder called SOFTWARE and the report in a folder called REPORT.
- Add the folders to a zip-archive and name it EAI320_prac1_studnr.zip.
- All reports and simulation software must be e-mailed to *EAI320.UP@gmail.com* no later than 16:00 on 23 February 2016. No late submissions will be accepted.
- Place a hard copy of your report in the box in front of Eng 3 7-25 before the deadline.
- Submit your report online on ClickUP using the TurnItIn link.

Additional Instructions

- Do not copy! The copier and the copyee (of software and/or documentation) will receive zero for both the software and the documentation. Z-e-r-o.
- For any questions of appointments email me at *EAI320.UP@gmail.com*
- Make sure that you discuss the results that are obtained. This is a large part of writing a technical report.

Marking

Your report will be marked as follow:

- 60% will be awarded for the full implementation of the practical and the subsequent results in the report. For partially completed practicals, marks will be awarded as seen fit by the marker.
- 40% will be awarded for the overall report. This includes everything from the report structure, grammar and discussion of results. The discussion will be the bulk of the marks awarded.