CS470/570

Artificial Intelligence

Spring 2018

Project #1 Pathfinding

Due: Friday, Febuary 9th

Pathfinding is a search problem commonly encountered in computer strategy games. The goal in pathfinding is to find the least cost path between two points on a map. The map is divided into a discrete cells, usually squares or hexagons. Typically there is cost associated with moving into a cell, which depends on the terrain in a cell. E.g. moving into a forest square costs more than moving into a field. (You might consider why the cost is associated with moving into, rather than out of, a cell.)

In games pathfinding is typically an informed search; the agent knows approximately where the goal is and can calculate heuristics like the straight line distance from any point to the goal. In games the pathfinding environment is usually treated as fully observable. If parts of the map are unknown the agent may only plan a route to the edge of the observed region. If there are hidden obstacles, the agent typically picks a route ignoring them and must replan after finding them. (In some cases computer agents may be allowed to 'cheat' and observe more of the map than a human player would be allowed to.)

Although games are the most obvious application of pathfinding there are many other problems that require pathfinding. For example, determining the flow of water or oil between underground pools may require pathfinding with different types of soil and rock impeding the flow to a greater or lessor extent.

**Project:** Test a number of different search strategies for pathfinding.

**Details:**For this problem the map will use a rectangular grid. Agents will be able to move in four directions (up, down, left, right), but not diagonally.

Your program should be able to read a map file in the following format.  
Width Height  
StartX StartY  
GoalX GoalY  
map  
Where the map is an array of characters. Width and Height define the width and height of the map. StartX, StartY, GoalX, and GoalY define the start and goal locations respectively. Note as a C++ programmer I have defined these positions starting from 0, so 0,0 is in the top left-hand corner of the map. [Here](http://www2.cs.uidaho.edu/~cs470_570/s18/project1/map.txt)is a map file to use for your results. The characters are interpreted as follows:

|  |  |  |
| --- | --- | --- |
| Character | Meaning | Movement Cost |
| R | road | 1 |
| f | field | 2 |
| F | forest | 4 |
| h | hills | 5 |
| r | river | 7 |
| M | mountains | 10 |
| W | water | can't be entered |

After a successful search the program should do the following:

* Draw the path on the map.
* Denote all of the explored squares on the map.
* Denote the current open/frige list.
* Report the length of the path found.
* Report the **cost** of the path found.

This does not require a graphical output (although that's probably best). You could draw a text version of the map (as in the sample file) and underline explored squares, bold open squares, and highlight squares on the successful path.

**Algorithms:**You will need to test the following algorithms:

* Breadth first.
* Lowest cost.
* Greedy best first
* A\* with at least two different heuristics.

**Experiments:**For each algorithm you will need to do the following:

* Show that the algorithm works. In particular you should show, with an actual figure, both the cells explored by the algorithm and the route that the algorithm finds. Make sure that the figure clearly shows that the algorithm works. For example, if there is a mountain range that impedes movement show that the algorithm searches for paths around it.

**Write-up**: For each of the search algorithms you should have a figure showing the explored cells, open cells, and path and you should report the length and cost of the found solution.

Include a brief discussion section after the figures discussing the results. In particular how the algorithms compared in terms of the results and whether any of the algorithms appeard to work incorrectly.

Attach your code.