CS670/470 Individual Assignment 2

TA: Yong Zhuang, UMass-Boston 9/12/2017

1 Educational Goal

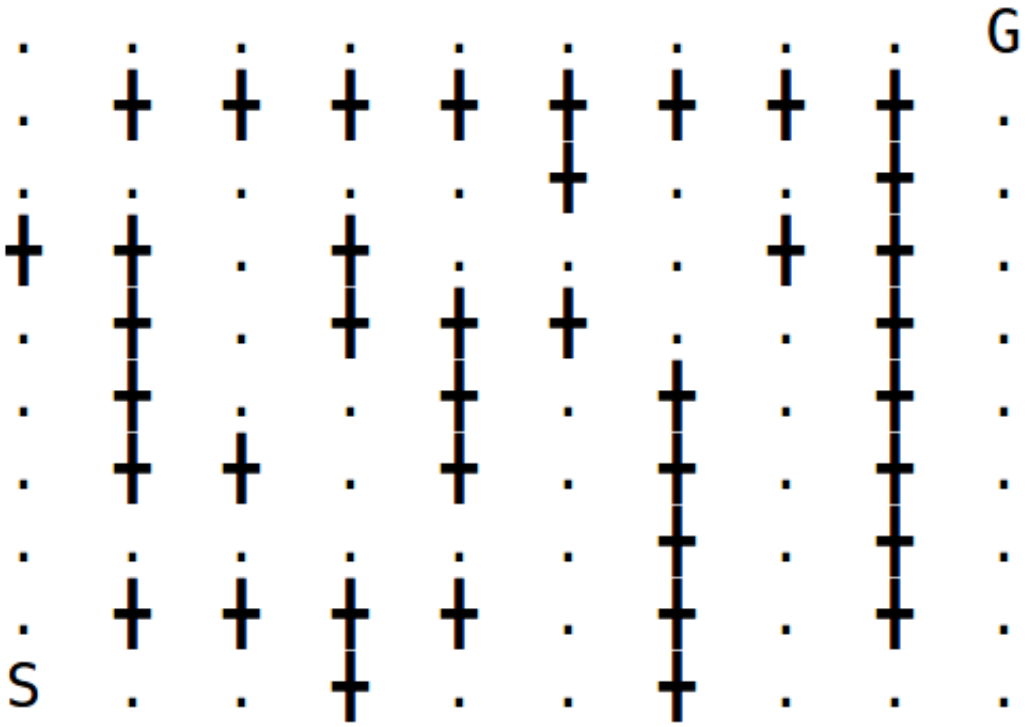
Practice Informed Search Algorithms.

2 Details

Project goal: Finding the path through a maze using Greedy Best-First Search and A^* search.
Due Date: 4:00 pm, October 12, 2017
Programming language: Python 2.7.

3 Maze

Size: 10×10
Start: Point S is the starting state located at $(0, 9)$
Goal: Point G is the goal state located at $(9, 0)$
In the figure below, dot (.) represents a path. cross (+) represents a wall.



4 Example

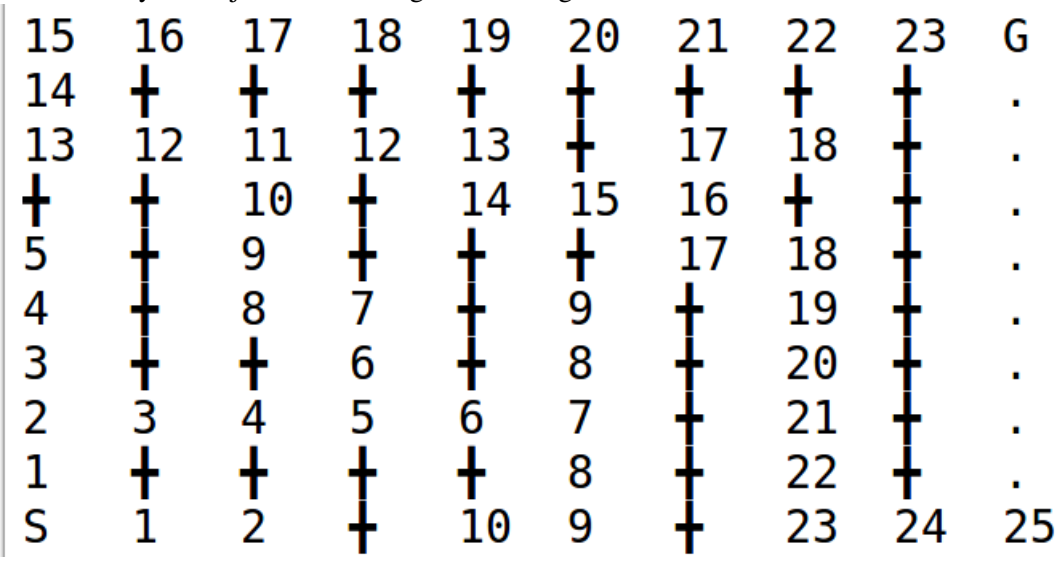
To help you implement the two informed search algorithm Greedy Best-First Search and A^* search, I prepared the sample code of the Dijkstra Search algorithm for you. So in this assignment, you should implement the two required informed search algorithms using the sample code.

4.1 Dijkstra Search

Dijkstra's algorithm greedily visits nodes in order of increasing distance from the initial vertex. It was conceived by computer scientist Edsger W. Dijkstra in 1956 and published three years later.
https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm.

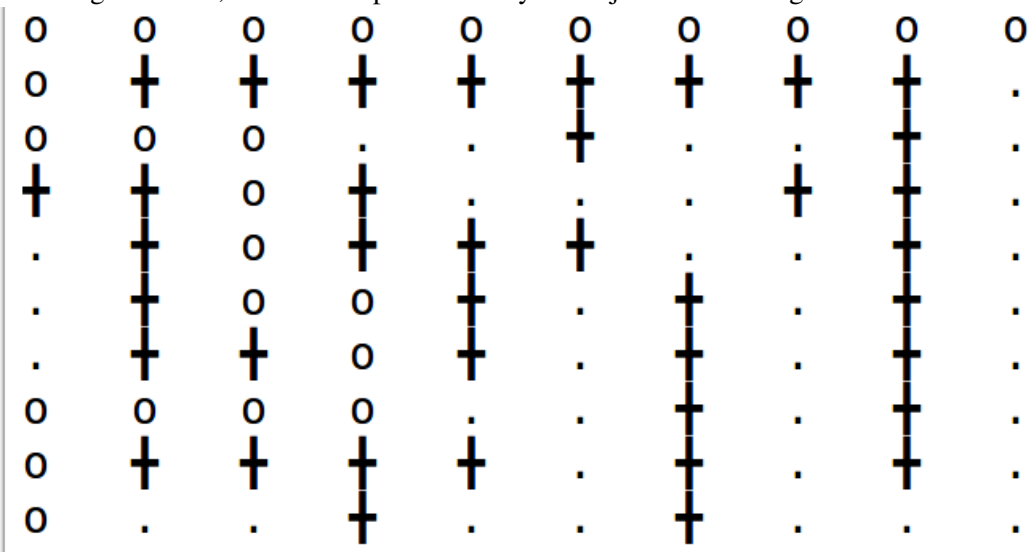
4.2 Map of the cost information

In the figure below, each number indicates the path cost from the starting state S to the current state calculated by the Dijkstra Search algorithm using the Manhattan distance.



4.3 Map of the path

In the figure below, the shortest path found by the Dijkstra Search algorithm is indicated by 0's.



4.4 Sample Code

Sample code is available at UMass Boston online blackboard.

5 Tasks

Let the movement cost from one cell to its 4 nearest neighbors(top, right, bottom, left) be equal to 1, and use Manhattan distance as the heuristic function.

Task 1. Greedy Best-First Search [30 points]: Review the sample code, then finish the implementation of Greedy Best-First Search, and draw the cost map and the path map same as those in sections 4.2 and 4.3.

Task 2. A^* search (Basic) [30 points]: Complete code to implement A* search, using the Manhattan distance to the goal at each cell as the h function. Draw the cost map and the path map same as those in sections 4.2 and 4.3.

Task 3. A^* search (Advanced) [20 points]: Design your own heuristic function to improve upon the efficiency of the A^* search algorithm, and draw the cost map and the path map same as those in sections 4.2 and 4.3. Do not manually set h values for particular cells and h cannot be the Manhattan distance either; h values must be assigned by a function. Only the 4 best solutions will get 100% of these 20 points.

Task 4. Summary [20 points]: Compare the results of 3 above algorithms and explain which one is the best and why.

6 Submission Requirements

Submit your code, and your report (including your summary of the task 4, and the cost maps and the path maps of the tasks 1 - 3) through your UMassOnline account.