CS4341 Introduction to Artificial Intelligence. A Term 2017

SOLUTIONS Exam 1 - September 08, 2017

Solutions by Mike Sokolovsky, Sam Ogden, Ahmedul Kabir, and Prof. Ruiz
Department of Computer Science
Worcester Polytechnic Institute

Problem I. Intelligent Agents [10 Points]

1. [3 points] Define “rational agent”.

A rational agent is one that makes logical decisions to choose the course of action that will achieve the best outcome (or best expected outcome) and acts upon them.

2. [3 points] Do you think that the Turing test is a good way to judge a rational agent? Justify your answer.

The Turing test is not a good test for a rational agent, since the Turing test judges whether the actions or behavior of the agent are human-like, not whether they are rational.

3. [4 points] Consider the following robotic agent Rob: Rob’s job is to find the exit from any given room. Rob has cameras that allow it to “see” the objects in front of it. Rob has wheels that enable it to move in different directions. Rob also has arms with which it can try to move objects out of its way. Is Rob a reflex agent or is it a goal-based agent? Explain.

Goal-based, because Rob has a definitive goal of reaching the exit. His tools help him achieve that goal. A reflex agent would simply act based on its input stimuli without a goal in mind.

Problem II. Search [30 Points]

1. [10 points] Give a complete problem formulation for the following problem. Choose a formulation that is precise enough to be implemented. In particular, describe the initial state, the set of actions, the goal test (i.e., the criteria to determine if a state is a goal state) and the cost of each action in a given state. (Note: you don't have to solve the problem, just formulate it.)

   You have three jugs, measuring 12 gallons, 8 gallons, and 3 gallons, and a water faucet. You can fill the jugs up or empty them out from one another or onto the ground. You need to measure out exactly one gallon.

   State Space: all configurations of the three jugs with different amounts of water in each jug.

   Initial State: 1 empty 12-gallon jug, 1 empty 8 gallon jug, 1 empty 3 gallon jug

   Actions:

   - Action 1: Fill a jug from the faucet until the jug is full
   - Action 2: Dump water from one jug A into another jug B until B is full
   - Action 3: Dump all the water from one jug onto the ground

   Goal: A jug has 1 gallon of water in it and the other jugs are empty

   Cost: A reasonable approach would be to make the cost of each action a unit cost so that all actions have the same cost. Another possibility is to define the cost of an action to be equal to the number of gallons of water transferred by the action. Furthermore, one could think of making the cost of Action 3 (dumping water onto the group) higher to try to minimize wasting water.

2. [20 points] In this problem, you are asked to select the best search strategy for each of the situations given below. You can choose any of the informed or uninformed search methods discussed in class. Justify your answer briefly but decisively.

   a) All the arc (= action) costs are equal. No heuristic information is given. Time is not a problem, and you have plenty of space. But you must find an optimal solution.

      [2 points] Your choice of search method: Breadth-First Search (BFS)

      [3 points] Justification for choice:

   If all arc costs are the same, then a shallowest solution is an optimal solution. Since BFS finds a shallowest solution, then it will be an optimal solution. BFS can take a lot time and space/memory if the search tree is wide and the solution is deep, but these are not issues in this case.

   Note: Iterative deepening and uniform search could be also choices for this problem, but there is no need to use them if a more basic algorithm (BFS) can solve the problem. Iterative deepening would take more time (though it would save some space), and uniform search would need to keep track of cost unnecessarily.

b) Cost and heuristic information are given. You need to find an optimal solution and you want to save as much time as possible.

[2 points] Your choice of search method: A*

[3 points] Justification for choice:

Since cost and heuristic information are given, A* will find an optimal solution, assuming that the heuristic information is admissible and consistent. Furthermore, A* is the fastest search method that finds an optimal path.

c) The search space is large and the depth of the solution is not known. No cost and no heuristic information are given. You need to find the shallowest solution but you don't have much memory/space available.

[2 points] Your choice of search method: Iterative Deepening

[3 points] Justification for choice:

Iterative Deepening will find the shallowest solutions first if the depth limit is increased by 1 at each iteration. Also, Iterative Deeping will save memory/space. [Note that Breadth-First Search would also find the shallowest solution but would take a lot of memory/space, which is an issue in this case.]

d) Heuristic information is given. The search space is finite, but quite large. You want to find a good solution fast but don't have enough memory to keep more than one path at the time.

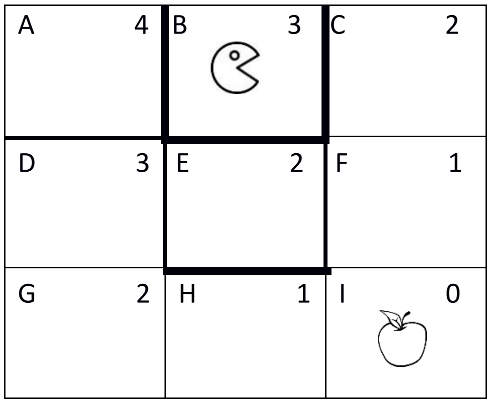
[2 points] Your choice of search method: Hill-Climbing without backrracking

[3 points] Justification for choice:

Hill-climbing uses heuristic information. Hill-climbing without backtracking only keeps the current path and does local search, so it uses less memory than other approaches like Greedy search. Also, hill-climbing tends to find good solutions), if it finds one, assuming that the heuristic information is good.

Problem III. Informed Search [35 Points]

Consider the very simplified version of Pacman shown below. The game world is a 3 X 3 grid where each cell is named by a letter from A to I. The Pacman agent is initially in cell B, and wants to get to the cell where the apple is so that it can eat the apple. Pacman doesn't know where the apple is, but it is able to detect the presence of the apple when (and only when) it is in the same cell with the apple. Also, Pacman has a heuristic function to estimate how close each cell is to the goal. On the top right corner of each cell, the heuristic cost is given (by the way, this heuristic function was calculated using the Manhattan distance from each cell to the goal). From any cell, the agent can make horizontal (left/right) or vertical (up/down) moves to an adjacent cell UNLESS the border between the two cells is marked by a thick line (borders between B & C and E & H) or the move would take Pacman outside the grid. The cost of an allowed move between two cells is 1 EXCEPT for moves between two cells separated by a fence marked with zigzag lines (borders between A & D and E & F) for which the move cost is 5.

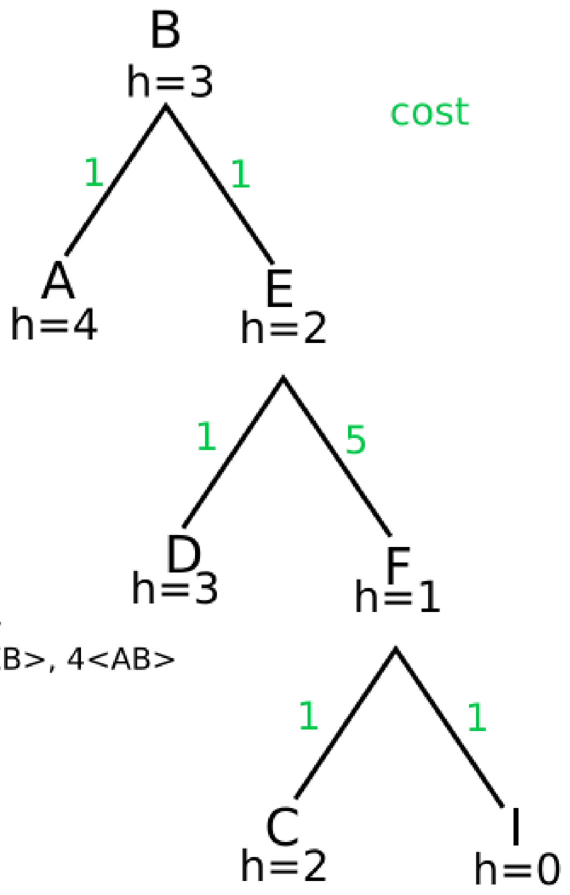


For this given problem, answer the following questions:

1. [5 points] Is the given heuristic function admissible? Justify our answer.

Yes, the heuristic function is admissible. The Manhattan distance uses a cost of 1 for each cell traversed in the shortest path between a cell and the goal. Since the actual cost of going from one cell to an adjacent cell in this problem is greater than or equal to 1, the Manhattan distance will not overestimate the actual cost of going between a cell and the end goal.

2. [12 points] Find a path to reach the goal using the **greedy best first search** algorithm. Show your work. In particular, draw the search tree, show the order in which the nodes will be expanded, show the state of the queue in each step, and calculate the final cost. Break any ties using the alphabetical order of the cell names.

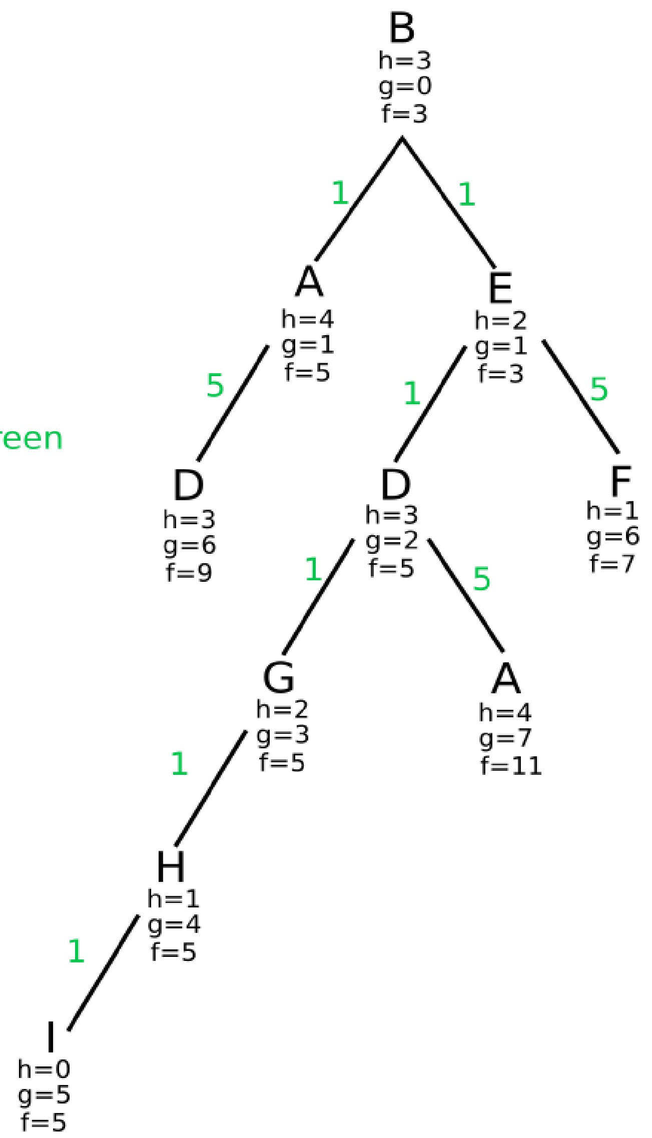


Expansion	Queue
B	3<B>
E	2<EB>, 4<AB>
F	1<FEB>, 3<DEB>, 4<AB>
I	0<IFEB>, 2<CFEB>, 3<DEB>, 4<AB>

Final Path	Final Cost
BEFI	1+5+1 = 7

3. [15 points] Do the same as III.2, this time for **A* search**.

h = heuristic
g = cost so far
f = g + h
path costs in green



Expansion	Queue
B	3<B>
E	3<EB>, 5<AB>
A	5<AB>, 5<DEB>, 7<FEB>
D	5<DEB>, 7<FEB>, 9~~<DAB>~~ Prune
G	5<GDEB>, 7<FEB>, 11<ADEB>
H	5<HGDEB>, 7<FEB>, 11<ADEB>
I	5<IHGDEB>, 7<FEB>, 11<ADEB>

Final Path
BEDGHI

Final Cost
1+1+1+1+1 = 5

4. [3 points] Did you get the same paths in III.2 and III.3? If so, why? Or did you get different paths? If so, why?

We got different paths because A* took cost into account whereas Greedy Search did not. A* took into account the high cost of traveling over fences. Also, A* always produces the optimal solution. Greedy did not produce an optimal solution in this case.

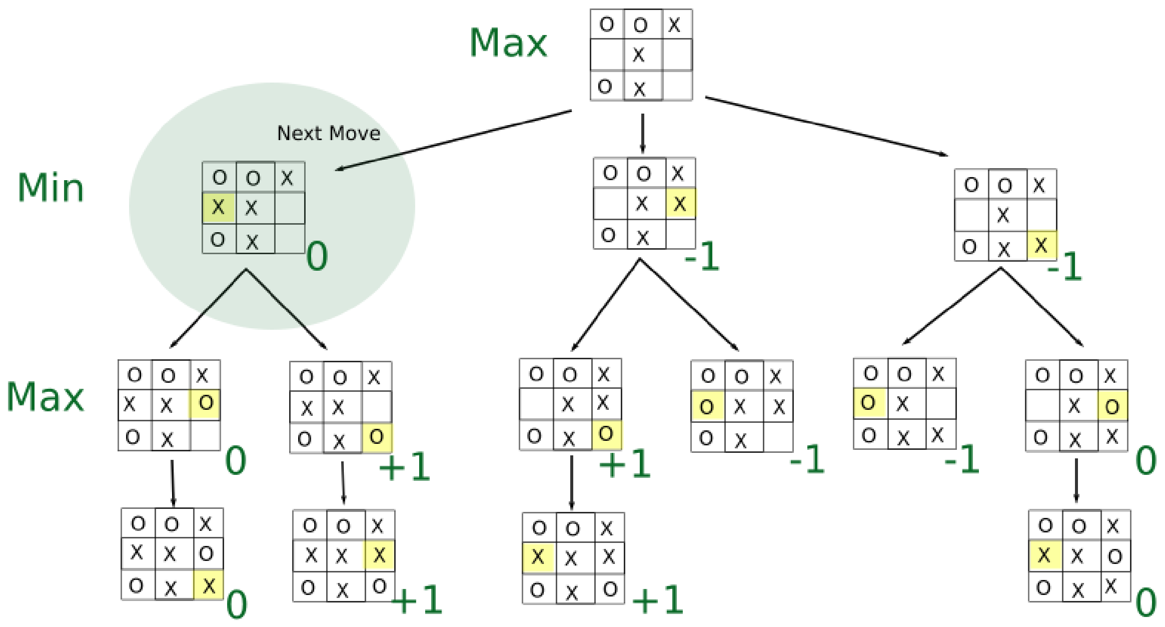
Problem IV. Adversarial Search [25 Points]

1. [13 points] Suppose that you are playing a game of tic-tac-toe on a 3X3 board with your friend. You are the MAX player, and you are playing with "X". The current board configuration is depicted below and it is your turn to move.

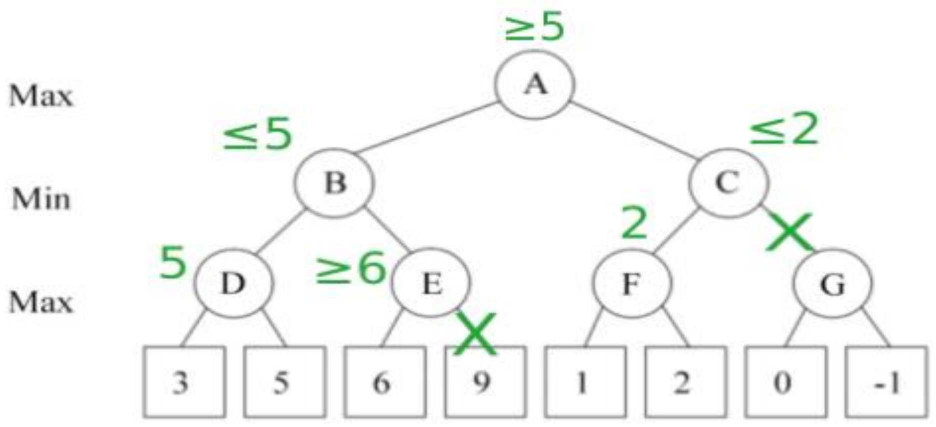
   Assume also that the utility function is given by:

$$Utility of a terminal board configuration = \begin{cases} +1, if\ 'X' wins \\ 0, if it is a draw \\ -1, if\ 'O' wins \end{cases}$$

   Use the **minimax** procedure to select your next move. Show the steps followed by minimax to expand the game tree, which is rooted at the current board configuration below. Make sure to show your work clearly on the minimax tree. Remember to mark on the search tree the move that minimax will output as your optimal next move.



2. [12 points] Suppose that you and your friend are now playing a different game. It is your turn to move (you are MAX), and the tree below represents your situation. The values of the utility function at the leaf nodes are shown in the tree. Use the **minimax** procedure together **with alpha-beta pruning** to select your next move. Mark clearly with an "X" the nodes/branches that don't need to be evaluated. When you decide to prune a node/subtree, explain why that pruning was possible.



We visit the nodes in Depth First Search order.

At node D we are looking for the maximum value so we pick 5 (out of 3 and 5).

Backtracking to B, we are looking for the minimum. We already have a potential 5 from node D. Thus, we only need to consider nodes less than 5. Looking at E, our first leaf is 6. Since E is a maximizing node, the final reported value will be ≥6. B has to minimize between D (5) and E(≥6). It will choose D and skip exploring E further.

Backtracking to A, we are maximizing between 5 (B) and whatever will be at C. Going to node F, we maximize between its children's values 1 and 2 and get a value of 2. Since this is passed to C, a minimization node, C will always be ≤ 2. Because A maximizes between B (5) and C (≤2), we can skip G.

Therefore we can prune the E leaf at 9 and the subtree rooted at G.