CS 188
Spring 2022

Introduction to
Artificial Intelligence

Written HW 1 Sol.

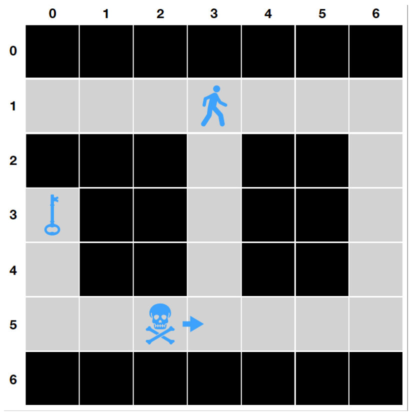
Solutions for HW 1 (Written)

Q1. [70 pts] Uninformed Search and Heuristics

Consider the following simplified version of the classic Atari video game, *Montezuma's Revenge*: It is played on the board illustrated below. An agent (represented by the person icon in cell (1,3)) wishes to grab the key (in cell (3,0)). A skull starts in cell (5,2) and moves to the right by one cell after each action is executed until it ends up in the rightmost cell, at which point it starts moving to the left, and repeats this pattern back and forth.

The agent can be facing either left or right. There are 10 possible actions for the agent: 2 turning actions ($face\_left$, $face\_right$) and 8 moving actions ($left, right, up, down, left\_up, left\_down$, $right\_up, right\_down$). The agent can move up or down while facing either direction, but can move sideways or diagonally only if facing in that direction. For example, if the agent is facing right but tries to move $left\_up$, the agent will not move and nothing will happen. Furthermore, if the agent is already facing $left$ and a $face\_left$ action is taken, nothing happens.

Lastly, the agent cannot move into a cell **currently occupied** by the skull, or a wall.



(a) Answer the following questions for the Montezuma's revenge board above:

(i) [7 pts] Let N be the number of possible cell locations that the agent can be in, and let M be the number of possible cell locations that the skull can be in. Recall that for "pacman pathing", the representation of the state was (x, y) where x was the row and y was the column of pacman's position.
Describe the minimal state space representation of a state in the state space for this game and give an expression for the size of the state space.

Representation of the state space: $(x, y, facing\_direction, y\_skull, skull\_direction)$

Size of the state space: $4 \times N \times M$

Explanation of each term in the size of the state space: There are N possible positions for the agent, 2 directions that the agent can face, and $2M$ possible positions+direction for the skull. One other way to think about why $2M$ is that the cycle of skull's movement is $2M$ (M moving to the right, and M moving to the left. Note that you could have also interpreted the problem in a way such that the skull cycle is of length $2(M - 1)$, and that is acceptable as well.

(ii) [3 pts] What is the goal test?
Is our current location $(x_H, y_H) = (3, 0)$?

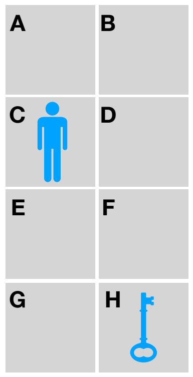
(b) Now, consider the simplified board below, which has **no skull** and **no facing-direction for the agent** (i.e., the agent can move in any of the 8 directions as long as it remains in the board). For the three following graph search algorithms, perform the search procedure yourself (**please show your work**) and provide answers to the questions below regarding the nodes expanded during the search as well as the final path found by the algorithm.

On this board, assume that a diagonal move has a cost of 3, whereas moving left, right, up, or down has a cost of 1. Do notice the difference in costs, and recall which algorithms use this cost information and which algorithms do not.

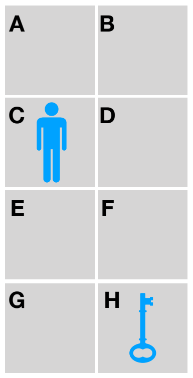
Remember that the search procedure should begin at the agent's starting position (C). To break ties when adding nodes of equal cost to the frontier, follow alphabetical order; for example, if you are considering nodes B and E with the same cost, then add node B to the frontier ahead of node E.

By convention, the depth of the starting node is 0, and the length of a path is the number of edges it contains.

Finally, when listing the order/number of nodes expanded, do not include nodes which are taken off the frontier but discarded immediately due to already having been expanded.



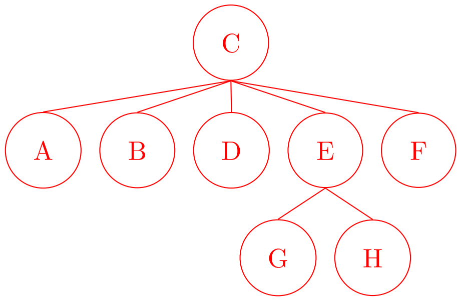
(i) [8 pts] **Breadth-first graph search**
Frontier data structure: FIFO.
Recall that BFS selects the *shallowest* unexpanded node in the frontier for expansion, which will be the *oldest* node in the frontier.



Order of nodes expanded: C, A, B, D, E, F, G, H (can omit H)
Number of nodes expanded: 8 (or 7 if omit H)
Path returned: C → E → H
Length of path: 2
Cost of path: 4
What are the *depths* $d(A), d(B), \ldots d(H)$?

State s	A	B	C	D	E	F	G	H
$d(s)$	1	1	0	1	1	1	2	2

Solution Explained: frontier data structure follows FIFO.

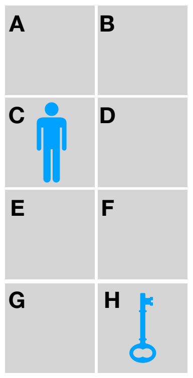


1. expanded = { C } frontier = { A B D E F }
2. expanded = { C A } frontier = { B D E F }
3. expanded = { C A B } frontier = { D E F }
4. expanded = { C A B D } frontier = { E F }
5. expanded = { C A B D E } frontier = { F G H }
6. expanded = { C A B D E F } frontier = { G H }
7. expanded = { C A B D E F G } frontier = {H }
8. expanded = { C A B D E F G H } Goal reached

(ii) [8 pts] **Uniform-cost graph search**

Frontier data structure: priority queue (make sure you update/reorder the whole frontier after each addition)

Recall that UCS keeps track of the lowest cost, $g(v)$, to get from the start node to the node v.



Order of nodes expanded: C, A, D, E, B, F, G, H (can omit H)

Number of nodes expanded: 8 (or 7 if omit H)

Path returned: C $\rightarrow$ D $\rightarrow$ F $\rightarrow$ H

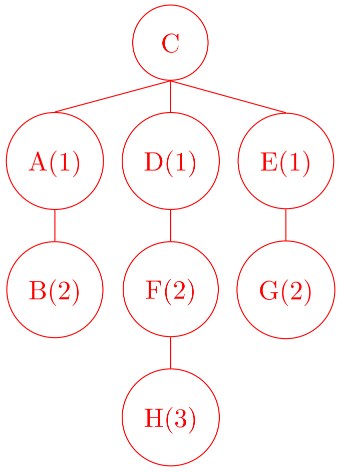
Length of path: 3

Cost of path: 3

What are $g(A), g(B), \ldots, g(H)$?

State s	A	B	C	D	E	F	G	H
$g(s)$	1	2	0	1	1	2	2	3

Solution Explained: PQ $\rightarrow$ need to reorder frontier based on the g-cost every time.



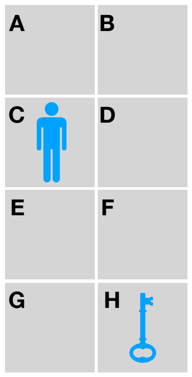
Note nodes already expanded will not be added to the frontier and node already in frontier will be updated with the smaller cost.

1. expanded = { C } frontier= { A(1) D(1) E(1) B(3) F(3) }
2. expanded = { C A } frontier = { D(1) E(1) B(2) F(3) }
3. expanded = { C A D } frontier = { E(1) B(2) F(2) }
4. expanded = { C A D E } frontier = { B(2) F(2) G(2) H(4) }
5. expanded = { C A D E B } frontier = {F(2) G(2) H(4) }
6. expanded = { C A D E B F } frontier = { G(2) H(3) }
7. expanded = { C A D E B F G } frontier = { H(3)}
8. expanded = { C A D E B F G H } Goal reached

(iii) [8 pts] **A* graph search (with Manhattan distance to the goal as the heuristic)**

Frontier data structure: priority queue (make sure you update/reorder the whole frontier after each addition)

Recall that A* computes $f(v)$ for the nodes v that it expands, with $f(v) = g(v) + h(v)$ where $g(v)$ is the lowest cost to reach v from the start node and $h(v)$ is an estimate of the distance from v to the goal.



Order of nodes expanded during the search: C, D, E, F, G, H (can omit H)

Number of nodes expanded during the search: 6 (or 5 if omit H)

Path returned by the search: C $\rightarrow$ D $\rightarrow$ F $\rightarrow$ H

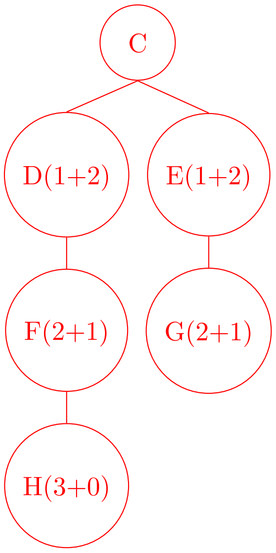
Length of path returned by the search: 3

Cost of path returned by the search: 3

What are $f(A), f(B), \ldots, f(H)$? Note that the h(v) is found by calculating the Manhattan distance from v to goal.

State s	A	B	C	D	E	F	G	H
$f(s)$	1+4	2+3	0+3	1+2	1+2	2+1	2+1	3+0

Solution Explained:

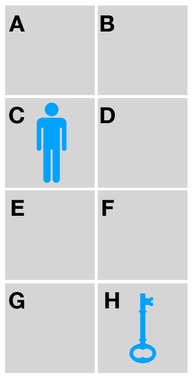


Note: nodes already expanded will not be added to the frontier and node already in the frontier will update with the smaller cost.
Cost: $f(v) = g(v) + h(v)$.

1. expanded = { C }   frontier= { D(1+2) E(1+2) F(3+1) A(1+4) B(3+3) }
2. expanded = { C D }   frontier = { E(1+2) F(2+1) A(1+4) B(2+3)}
3. expanded = { C D E }   frontier = { F(2+1) G(2+1) H(4+0) A(1+4) B(2+3) }
4. expanded = { C D E F }   frontier = { G(2+1) H(3+0) A(1+4) B(2+3) }
5. expanded = { C D E F G }   frontier = {H(3+0) A(1+4) B(2+3) }
6. expanded = { C D E F G H }   Goal reached

(c) [10 pts] Given your answers above, what are the qualitative differences between the results achieved by BFS, UCS, and A*? Which one finds the shortest path (in number of steps)? Which one finds the optimal path (in cost)?

BFS is able to find the most direct path (in number of steps).
UCS takes into account the cost of transitions and finds the path of optimal/least cost to the key. However, it has to expand nodes in the opposite direction of the key as it does not take into account any problem-specific information aside from path cost.
A* search finds the same optimal path as UCS but because of the heuristic, is able to do so while expanding fewer nodes.



(d) For the same board and setting as part (b), give an example for each of the following types of heuristics. Please briefly explain why the example you chose satisfies the requested properties.

(i) [3 pts] Admissible and consistent.
Note: You can use a heuristic that we have frequently used in this class, or you can just assign any set of numbers to the states that qualifies as an admissible and consistent heuristic.

State s	A	B	C	D	E	F	G	H
Heuristic $h(s)$	4	3	3	2	2	1	1	0

Explanation: Manhattan Distance is admissible and consistent. Answers might vary

(ii) [4 pts] Admissible but inconsistent

State s	A	B	C	D	E	F	G	H
Heuristic $h(s)$	4	2	2	2	2	1	1	0

Explanation: Manhattan Distance, except reduced $h(B)$ and $h(C)$ (other solutions exist, such as reduced $h(A)$). This makes them remain an underestimate, but this causes $A \to B$ and $A \to C$ to be overestimated. True cost is 1 for these, but difference in heuristic cost says 2, so we are overestimating the transition cost of the edge. Answers might vary but need to overestimate at least one edge

(iii) [3 pts] Inadmissible and inconsistent

State s	A	B	C	D	E	F	G	H
Heuristic $h(s)$	5	2	2	2	2	1	1	0

Explanation: Manhattan Distance, except overestimated cost-to-goal for A, overestimated $A \to B$. Answers might vary but need to 1. overestimate cost-to-goal from one location, 2. overestimate at least one edge

(e) **(i)** [8 pts] For this new version of the game, your friend Nancy suggests taking the old game setting from part (b) and now adding the ability for the agent to perform a maximum of one "teleportation" action during the game. That is, on one of the agent's moves, it can choose to jump from its current state to any other non-goal state on the board, and the cost of teleporting is 1.

1. How does this new teleportation ability change the state space of the game from part (b), which was (x, y)? Does anything need to be added or removed?

Whether or not the teleportation step has been taken should be added to the state.

2. Nancy argues that in this new game, at least one previously consistent heuristic can become inconsistent. Is Nancy right?

[✓ Yes, and I will give an example below.]
○ No, and I will provide a proof below.

Note: we define heuristics for this problem as being a function of **only** the cell location: They cannot incorporate anything that did not exist in the old version of the game that we are comparing to.

If you believe Nancy is right, give an example of a heuristic that used to be consistent in the old game but is no longer consistent in this new game. Make sure to explain why it is no longer consistent (perhaps with a drawing of a board state and an explanation).

State	A	B	C	D	E	F	G	H
$h(s)$	4	3	3	2	2	1	1	0

The manhattan distance heuristic (written in the previous part) will no longer be consistent because the true cost of any transition can now be 1, due to teleportation. In this case, $h(a) - h(b)$ using the old heuristic values could definitely overestimate this true cost.

If you believe Nancy is wrong, provide an argument for why a heuristic that was consistent in the old game must also remain consistent in this new game. Be specific about your reasoning and use mathematical quantities such as heuristic costs of states $h(v)$ and true costs of actions $c(v, a, v')$.

(ii) [8 pts] For this new version of the board, your friend Ethan suggests adding the skull back to the old board setting from part (b), and having the skull move back and forth between the cells E and F.

1. How does the presence of this skull change the state space of the game from part (b), which was (x, y)? Does anything need to be added or removed?

The location of the skull needs to be added to the state.

2. Ethan argues that in this new board, at least one previously consistent heuristic can become inconsistent. Is Ethan right?

○ Yes, and I will give an example below.
[✓ No, and I will provide a proof below.]

Note: we define heuristics for this problem as being a function of **only** the cell location: They cannot incorporate the location of the skull, since that did not exist in the old version of the board that we are comparing to.

If you believe Ethan is right, give an example of a heuristic that used to be consistent on the old board but is no longer consistent on this new board. Make sure to explain why it is no longer consistent (perhaps with a drawing of a board state and an explanation.

State	A	B	C	D	E	F	G	H
$h(s)$								

If you believe Ethan is wrong, provide an argument for why a heuristic that was consistent in the old board must also remain consistent in this new board. Be specific about your reasoning and use mathematical quantities such as heuristic costs of states $h(v)$ and true costs of actions $c(v, a, v')$.

For a consistent heuristic in the old board, $h(v) - h(v') \leq c(v, a, v')$. In the new board, the cost of getting somewhere is strictly greater than or equal to the cost of getting there in the old board. Thus, $c_{new}(v, a, v') \geq c(v, a, v')$, which means $h(v) - h(v') \leq c_{new}(v, a, v')$ and thus the heuristic is still consistent in this new board.