CS 340 Home Work # 2 Solution

CLO 2

Total marks: 70

Q 1. (15 Marks) Your goal is to navigate a robot out of a maze. The robot starts in the center of the maze facing north. You can turn the robot to face north, east, south, or west. You can direct the robot to move forward a certain distance, although it will stop before hitting a wall.

(a) Formulate this problem. How large is the state space?

(b) In navigating a maze, the only place we need to turn is at the intersection of two or more corridors. Reformulate this problem using this observation. How large is the state space now?

(c) From each point in the maze, we can move in any of the four directions until we reach a turning point, and this is the only action we need to do. Reformulate the problem using these actions. Do we need to keep track of the robot's orientation now?

(d) In our initial description of the problem we already abstracted from the real world, restricting actions and removing details. List three such simplifications we made.

a. We'll define the coordinate system so that the center of the maze is at $(0, 0)$, and the maze itself is a square from $(-1, -1)$ to $(1, 1)$.

Initial state: robot at coordinate $(0, 0)$, facing North.
Goal test: either $|x| > 1$ or $|y| > 1$ where (x, y) is the current location.
Successor function: move forwards any distance d; change direction robot it facing.
Cost function: total distance moved.

The state space is infinitely large, since the robot's position is continuous.

b. The state will record the intersection the robot is currently at, along with the direction it's facing. At the end of each corridor leaving the maze we will have an exit node. We'll assume some node corresponds to the center of the maze.

Initial state: at the center of the maze facing North.
Goal test: at an exit node.
Successor function: move to the next intersection in front of us, if there is one; turn to face a new direction.
Cost function: total distance moved.

There are $4n$ states, where n is the number of intersections.

c. Initial state: at the center of the maze.
Goal test: at an exit node.
Successor function: move to next intersection to the North, South, East, or West.
Cost function: total distance moved.

We no longer need to keep track of the robot's orientation since it is irrelevant to predicting the outcome of our actions, and not part of the goal test. The motor system that executes this plan will need to keep track of the robot's current orientation, to know when to rotate the robot.

d. State abstractions:

(i) Ignoring the height of the robot off the ground, whether it is tilted off the vertical.

(ii) The robot can face in only four directions.

(iii) Other parts of the world ignored: possibility of other robots in the maze, the weather in the Caribbean.

Action abstractions:

(i) We assumed all positions we safely accessible: the robot couldn't get stuck or damaged.

(ii) The robot can move as far as it wants, without having to recharge its batteries.

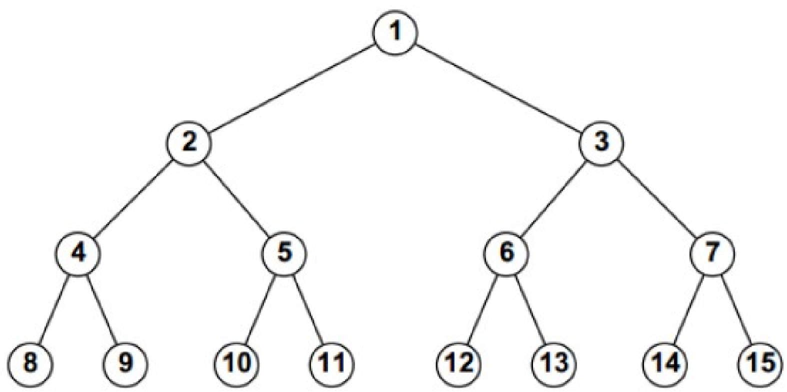
(iii) Simplified movement system: moving forwards a certain distance, rather than controlled each individual motor and watching the sensors to detect collisions.

Q2. (10 marks) Consider a state space where the start state is number 1 and each state k has two successors: numbers 2k and 2k + 1.

(a) Draw the portion of the state space for states 1 to 15.

(b) Suppose the goal state is 11. List the order in which nodes will be visited for breadth first search, depth-limited search with limit 3, and iterative deepening search.

a.



b. Breadth-first: 1 2 3 4 5 6 7 8 9 10 11
Depth-limited: 1 2 4 8 9 5 10 11
Iterative deepening: 1; 1 2 3; 1 2 4 5 3 6 7; 1 2 4 8 9 5 10 11

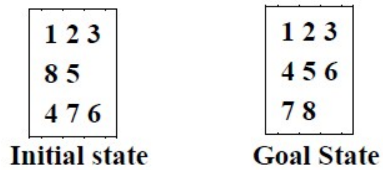
Q3. (5 Marks) Consider a heuristic path algorithm as a best-first search in which the evaluation function is $f(n) = (2 - w)g(n) + w\,h(n)$. What kind of search does this perform for w = 0, w = 1, and w = 2?

w=0 => Uniform Cost Search, when
w=1 => A* Search
w=2 => Greedy best-first Search.

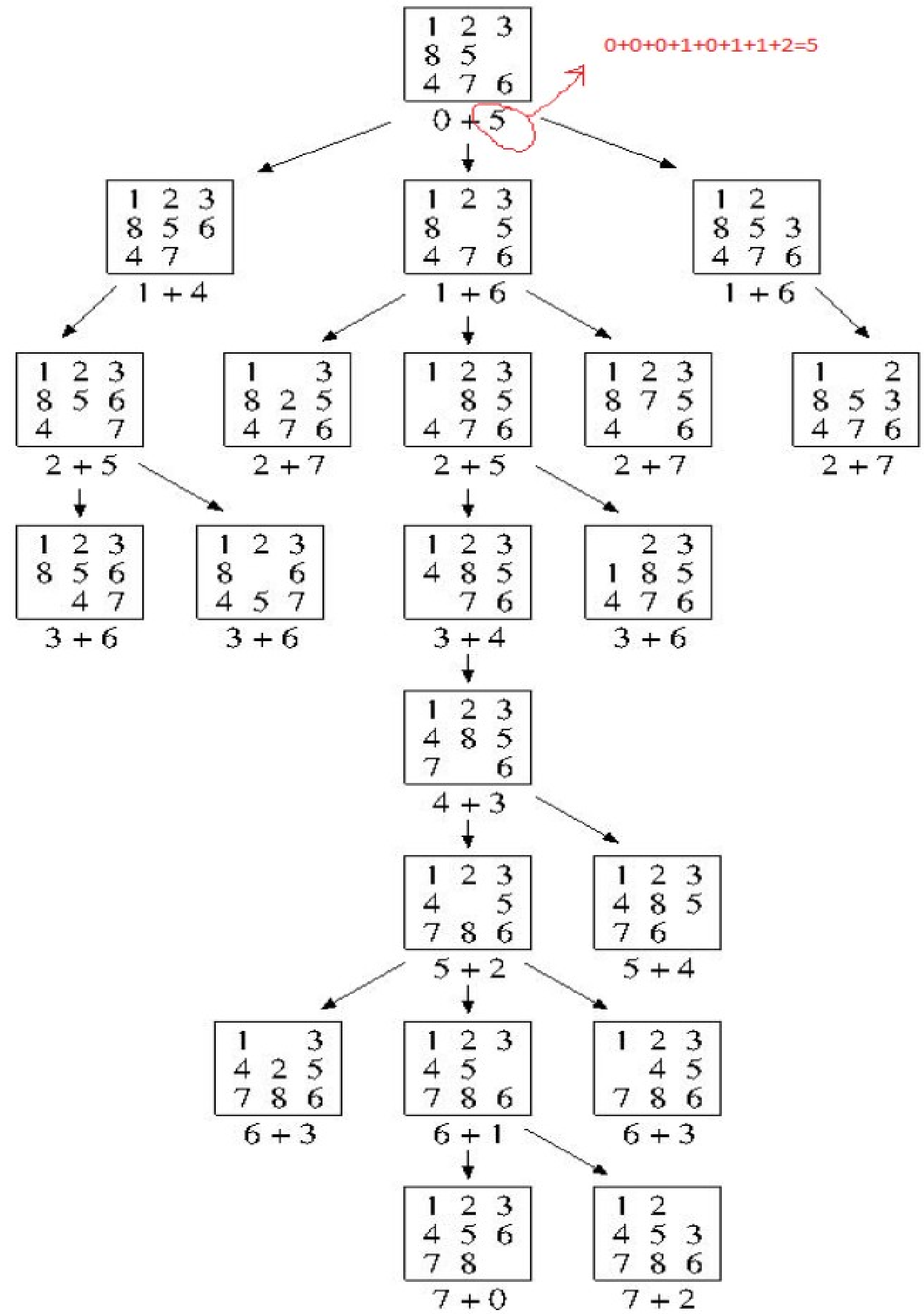
Q4. (5 Marks) What are the pros (if any) and cons (if any) to using A* versus Uniform Cost Search? Explain; consider both time and space.

Evaluating the heuristic in A* can take extra time, but if the heuristic is good (informed) it can cut down the number of expanded states a lot (which helps running time and space)

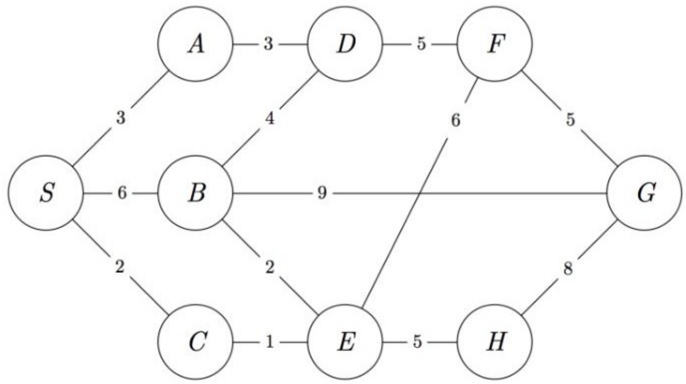
Q5. (15 Marks) Consider the following initial and goal states of 8-puzzle:



Trace the A* Search algorithm using the Total Manhattan Distance heuristic, to find the shortest path from the initial state shown above, to the goal state.



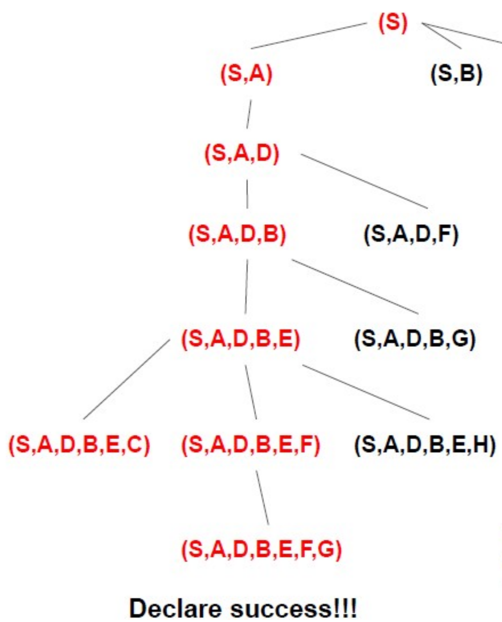
Q6. (20 Marks) For each of the following search algorithms, find the nodes expanded and the path returned (break ties alphabetically, e.g. S->A->D precedes S->C->E->B): Edges are bi-directional. Use DFS, BFS, UCS and Greedy BFS



State	h(n)
S	12
A	11
B	9
C	5
D	8
E	3
F	5
H	6
G	0

DFS

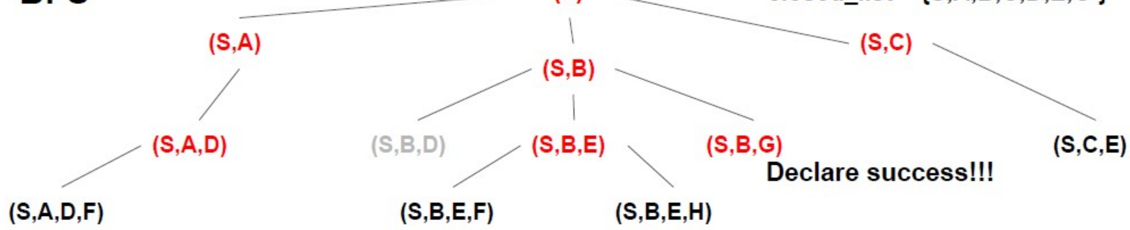
closed_list = {S, A, D, B, G, C, F, G }



Nodes expanded: 8
Path returned: (S,A,D,B,E,F,G)

BFS

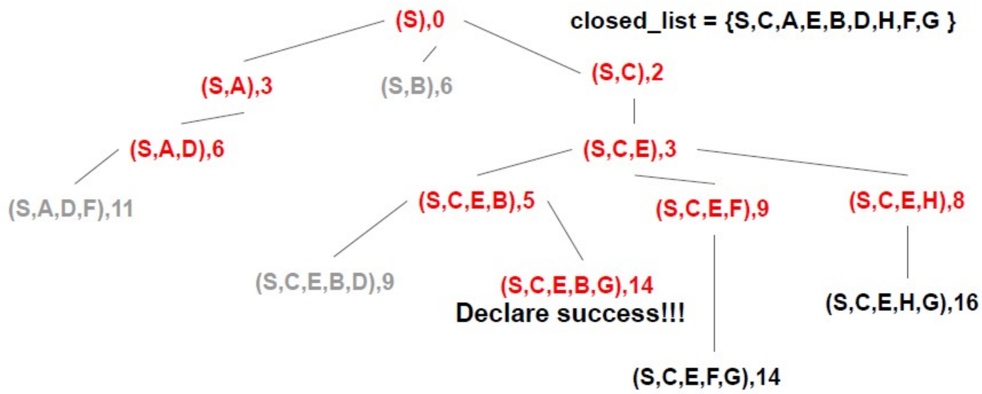
closed_list = {S,A,B,C,D,E,G }



Nodes expanded: 7
Path returned: (S,B,G)

UCS

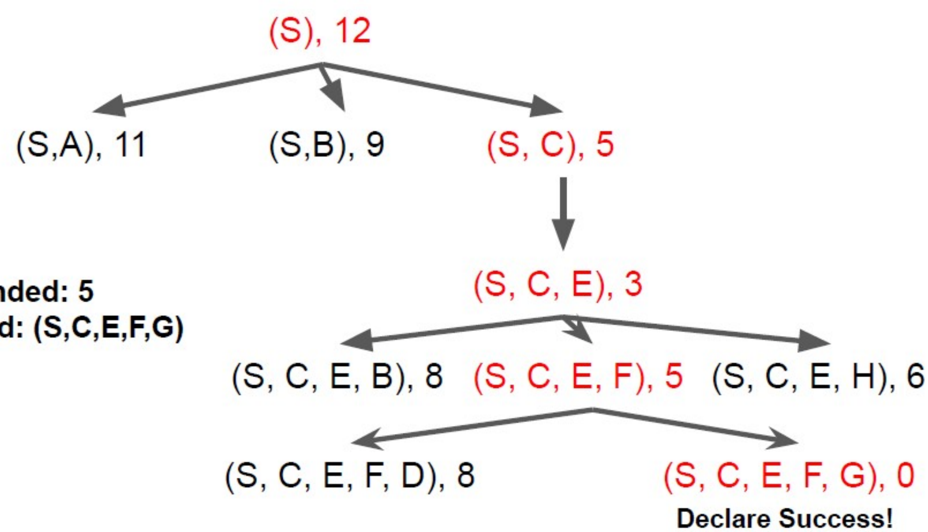
closed_list = {S,C,A,E,B,D,H,F,G }



Nodes expanded: 9
Path returned: (S,C,E,B,G)

Greedy

closed_list = {S,C,E,F,G }



Nodes expanded: 5
Path returned: (S,C,E,F,G)