CS 4700: Foundations of Artificial Intelligence **(Due: 2/17/20, 11:59pm)**

Homework #2: Search Algorithms SOLUTIONS

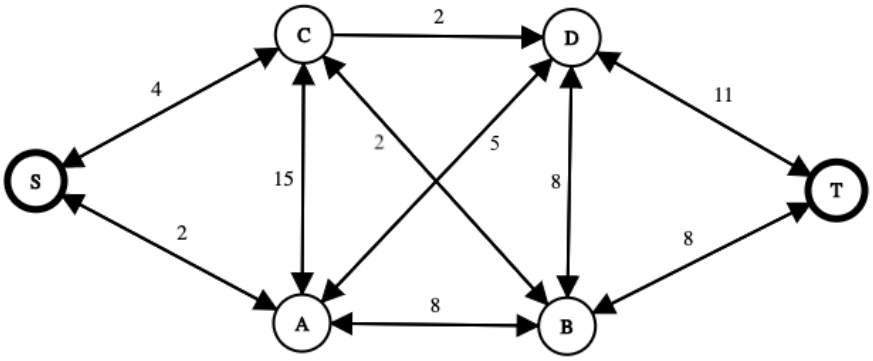
35 Points Total

Instructor: Haym Hirsh *Name:* Student name, *Netid:* NetId

Course Policy: Read all the instructions below carefully before you start working on the assignment, and before you make a submission.

- Please include your name and NetIDs on the first page. We recommend typesetting your submission in LaTeX, and an Overleaf template is linked here.
- As part of the typesetting requirement, all graphs must be computer-generated (no stylus drawn graphs will be accepted). We recommend using TikZ, this online tool, or Powerpoint/Google Slides/Keynote to draw any graphs.
- Homeworks must be submitted via Gradescope by the due date and time.
- Late homeworks are accepted until **2/19/20 at 11:59pm** for a 50% penalty per course policy.
- All sources of material outside the course must be cited. The University Academic Code of Conduct will be strictly enforced.

Problem 1: A* Search (12 points)



Consider the search space depicted above for a hypothetical search problem. S is the initial state and T is the goal state. The cost of each edge has been labeled on the graph.

A (4 points): Compute the shortest path and its cost using uniform cost search (Djikstra's algorithm).
Solution: S - C - B - T, 14

B (4 points): Consider the following table representing $h(s)$ for each state s in the space. What path does A* search find using this $h(s)$, and what is its cost?

State s	A	B	C	D
$h(s)$	10	16	9	9

Solution: S - C - D - T, 17
start from S:
f(A) = 2+10 =12
f(C) = 4+9 = 13
expand A (because f(a)=12 is the smallest so far):
f(C) = (2+15)+9=26
f(D) = (2+5)+9=16
f(B) = (2+8)+16=26
expand C (because f(c)=13 is the smallest so far)
f(D) = (4+2)+9=15
f(B) = (4+2)+16=22
f(A) = (4+15)+10=29
expand D (because f(d)=15 is the smallest so far)
f(B) = (6+8)+16 = 30
f(T) = (6+11) = 17
Path: S-C-D-T

C (4 points): Compare your results for (a) and (b). If they are the same, explain why. If they are different, provide a specific reason.
Solution: (a) and (b) are not the same because the heuristic is inadmissible.

Problem 2: DFS, BFS, IDS (13 points)

Consider the following puzzle called "Moving Magic Square". It is played on a 3×3 table containing each of the numbers 1 to 9. The number 9 is the "movable number". You can move 9 in four directions (up/down/left/right), and swap 9 with the number in that direction. The initial state is shown in Table 1. As the player, we want to move 9 to reach a final state such that the sum of the three numbers on every row, column, and diagonal is 15. There are multiple states that satisfy this condition, and you can stop your answer when you find the first satisfied state. (Hint: You can define the operations in order of up/down/left/right, which might be helpful for fewer steps to reach the satisfied state.)

6	9	8
7	1	3
2	5	4

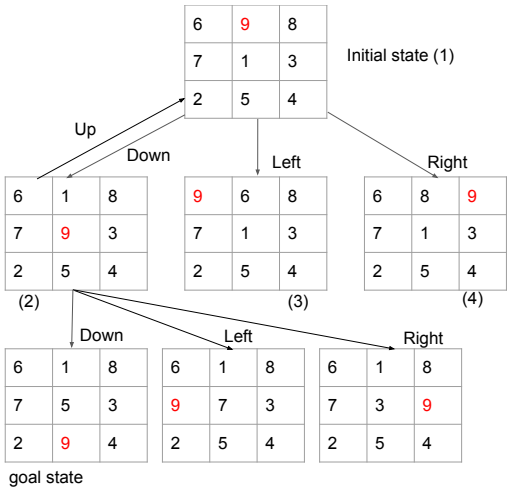
Table 1: Initial state

A (2 points): Formulate this as a search problem. What are the states, operators, initial state, and goal condition?
Solution:

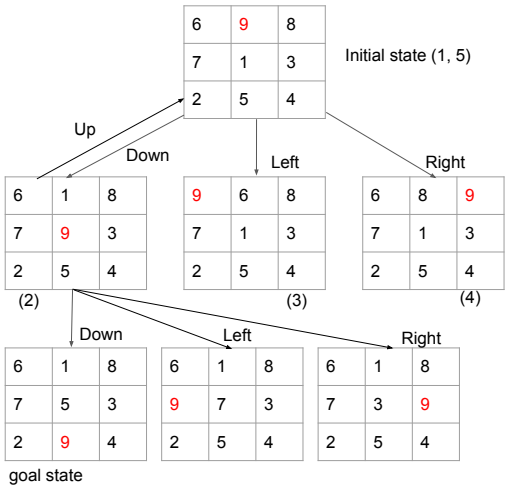
- the states: a 3×3 table with number 1 to 9
- operators: move 9 towards up/down/left/right direction, and *swap* 9 with the number in that direction
- initial state: Table 1
- goal condition: the sum of three numbers on every row, every column and every diagonal is 15

B (2 points): Is this state space a graph or a tree?
Solution: A graph.

C (3 points): Draw the portion of the state space that would be generated by the Breadth-First Search (BFS) procedure and mark the order in which each state is expanded with the numbers 1, 2, 3... Do not create multiple copies of states that are identical, and identify the states that are goal states.
Solution:
the successors of (2) should not be marked in the order because they are not expanded.



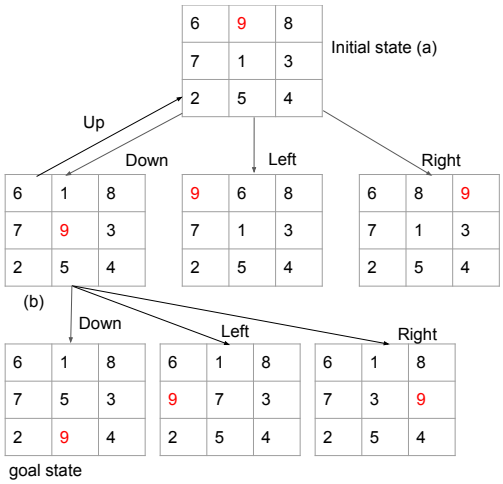
version2: if the algorithm doesn't check repeated status.



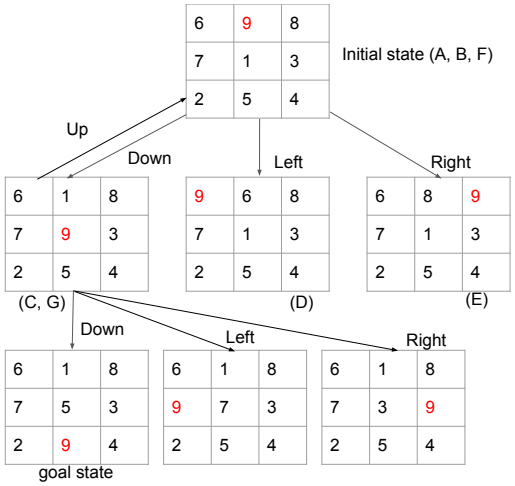
D (3 points): Draw the portion of the state space that would be generated by the Depth-First Search (DFS) procedure and mark the order in which each state is expanded with lower-case letters a, b, c... Do not create multiple copies of states that are identical, and identify the states that are goal states.

Solution:

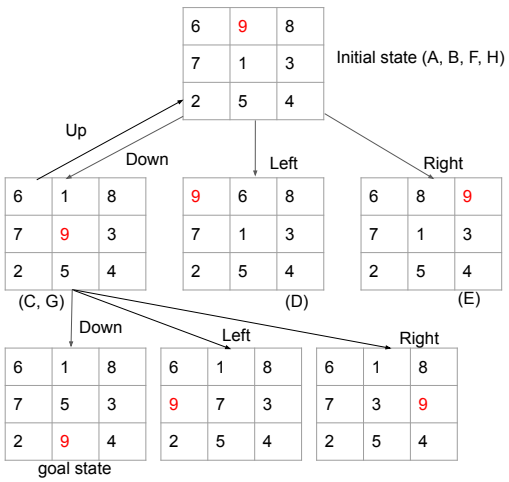
The algorithm needs to check repeated status and does not expand repeated status. If the algorithm doesn't check repeated status, DFS would switch between first two states.



E (3 points): Draw the portion of the state space that would be generated by the Iterative Deepening Search (IDS) procedure and mark the order in which each state is expanded with upper-case letters A, B, C... Do not create multiple copies of states that are identical, and identify the states that are goal states.
Solution:
state (D) and (E) are expanded with DBDFS but the successors of them are not added to the open list because bound=0, and their successors should not be shown in the search structure.



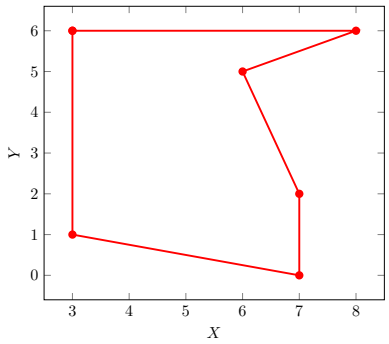
version2: if the algorithm doesn't check repeated status.



Problem 3: Hill Climbing Heuristics (10 points)

Hill-climbing methods have generated the best results obtained thus far for many very large Travelling Salesman Problems (TSPs). The TSP problem statement is the following: "Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city and returns to the origin city?". The 6 points in the diagram below define a hypothetical TSP comprised of 6 cities, and shows one sample path that is a possible solution to this TSP.

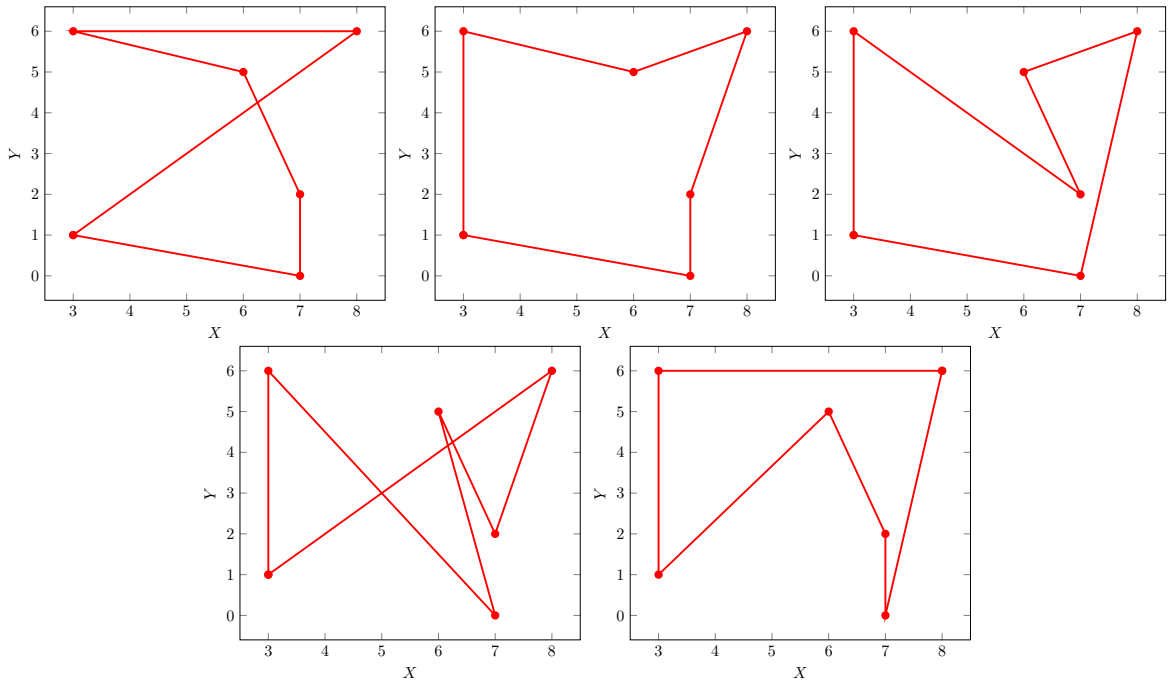
To solve such problems using hill climbing, states are an ordered list of cities specifying the order in which each city is visited. Thus, for example, in the diagram below the state is ((3,1), (3,6), (8,6), (6,5), (7,2), (7,0)). The last vertex in the list is then connected back to the first vertex in the list. The hill climbing operator we will consider here takes a state and swaps two of the cities in the list, thereby generating a new path that visits all the cities. The evaluation function $f(s)$ will simply be the length of the path given in s.



A (2 points): How many states would this operator generate for the sample solution above?
Solution: $\binom{6}{2}$, which equals 15

B (3 points): Consider the subset of these states that were created by swapping (8,6) with one of the other vertices. Draw all of these states, and give the value $f(s)$ for each such state.

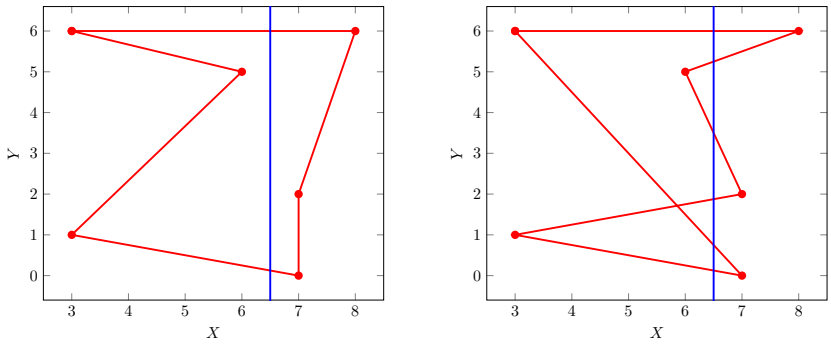
We suggest copying and pasting the source TikZ (labelled with "copy here" in the LaTeX source) from the starting state S and changing the coordinates to show all possible successor states.
Solution: Given the following successor states, there would be no move since the initial state, S, is more optimal than all the successor states.



Solutions:

1. $\{(3,1),(8,6),(3,6),(6,5),(7,2),(7,0)\}$; $f(s) = 24.51873$
2. $\{(3,1),(3,6),(6,5),(8,6),(7,2),(7,0)\}$; $f(s) = 20.64456$
3. $\{(3,1),(3,6),(7,2),(6,5),(8,6),(7,0)\}$; $f(s) = 26.26107$
4. $\{(3,1),(3,6),(7,0),(6,5),(7,2),(8,6)\}$; $f(s) = 31.66657$
5. $\{(8,6),(3,6),(3,1),(6,5),(7,2),(7,0)\}$; $f(s) = 26.24504$

C (5 points): Now lets say that we added a river at $X = 6.5$ and a cost, R, is added to the heuristic $f(s)$ every time the path crosses the river. Let n be the number of times a path crosses the river. What would the costs R_l and R_r (left and right, respectively) have to be in order for hill climbing to yield the following solutions? Note that R can be a negative value.



Solution: $R_l > 0.94$ and $R_r < -2.17$