CSE 415 Winter 2020 Assignment 4

Last name: Janarthanan First name: Sanjeev

Due Friday night February 7 via Gradescope at 11:59 PM. You may turn in either of the following types of PDFs: (1) Scans of these pages that include your answers (handwriting is OK, if it's clear), or (2) Documents you create with the answers, saved as PDFs. When you upload to GradeScope, you'll be prompted to identify where in your document your answer to each question lies.

Do the following five exercises. These are intended to take 20-25 minutes each if you know how to do them. Each is worth 20 points.

1 Blind Search

Sudoku is a popular type of puzzle that has many variations. One variation that is usually easier to solve than standard Sudoku is called Sudoku X 6 . The goal is to fill in the blank squares to get an array in which each of the rows, columns, two main diagonals, and six "regions" has each of the numbers 1 through 6 in it. The regions are 2 by 3 blocks, shown with bold outlines in the pictures below. (Standard Sudoku has 3 by 3 blocks.)

	1			4	
	2			5	
	3			6	

3	4	2	5	1	6
6	1	5	2	4	3
5	6	4	1	3	2
1	2	3	6	5	4
4	5	6	3	2	1
2	3	1	4	6	5

(a) (5 points) Describe a possible state representation for this puzzle, using either English, pseudocode or Python.

A list of 6 lists with each inner list containing 6 entries. Each inner list would represent a region of the board. If a square is blank it will be represented by None.

(b) (10 points) Provide pseudocode for a method `successor(s)` that takes a state s and returns a list of its successors.

```
def successor(s):
    moves = possible_moves(s) #function defined elsewhere
    successors = []
    for move in moves:
        new_state = state_transform(s, move) #func defined
        successors.append(new_state)
    return successors
```

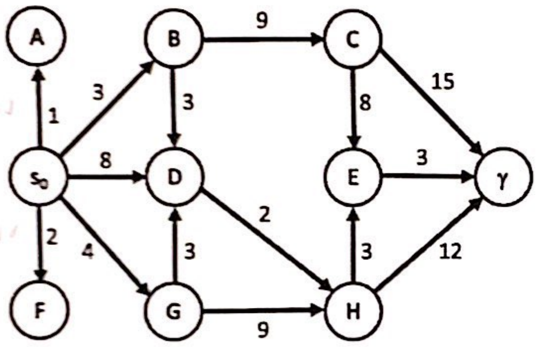
(c) (5 points) Provide pseudocode for a method `is_goal(s)` that takes a state s and returns True of s is a goal state.

```
def is-goal(s):
    for region in s:
        cur_nums = []
        for square in region:
            if square == None or square in cur-nums:
                return false
            cur_nums.append(square)
    #repeat process for rows and columns
    return True
```

2 Heuristic Search

(a) (5 points) Describe some of the challenges and trade-offs that need to be considered when selecting a heuristic.

One needs to consider the consistency and admissiblity of the heuristic. Additionally, the heuristic must be positive. If you want a faster heuristic you can choose to compromise on it being admissible or consistent. If the heuristic is too low it will expand too many nodes.



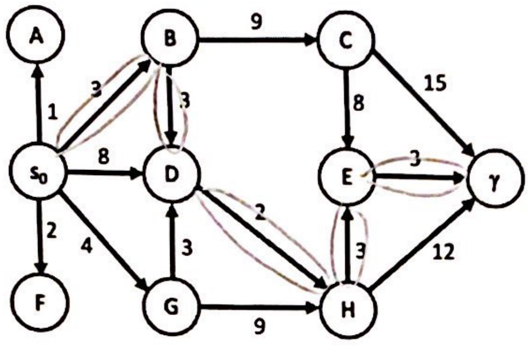
state (s)	s_0	A	B	C	D	E	F	G	H	γ
heuristic $h_1(s)$	14	15	4	10	3	2	16	10	5	0
heuristic $h_2(s)$	14	15	10	10	7	2	16	10	5	0
heuristic $h_3(s)$	14	15	12	10	7	2	16	10	5	0

(b) (5 points) Which heuristics (h_1, h_2, h_3) shown above are admissible? h_1, h_2

(c) (5 points) Which heuristics (h_1, h_2, h_3) shown above are consistent? h_2

(d) (5 points) Which of the 3 heuristics shown above would you select as the best heuristic to use with A* search, and why? Refer to consistency/admissibility in your justification.

h_1 and h_2 are the only admissible heuristics, and h_2 is the only consistent heuristic. As such, h_2 is the best heuristic to use for A* search. It will find the shortest path.



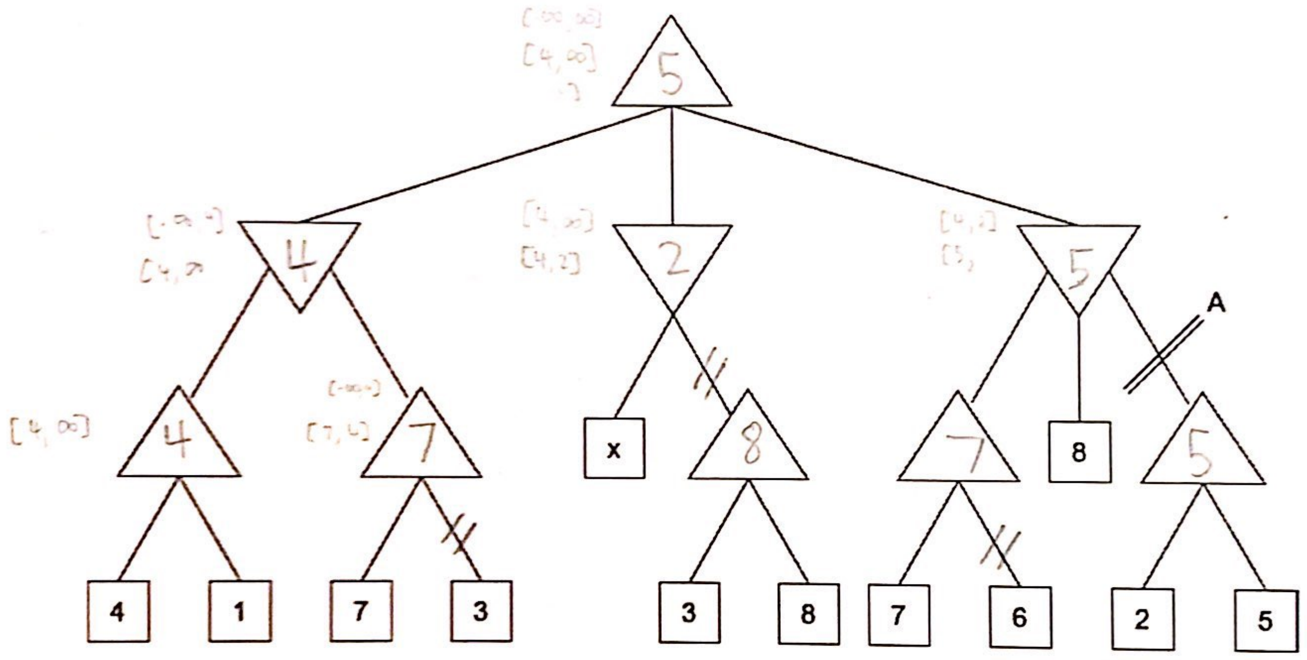
state (s)	s_0	A	B	C	D	E	F	G	H	γ
heuristic $h_3(s)$	6	8	5	5	3	1	8	5	2	0

(e) (5 points) Referring back to the graph again, trace out the path that would be followed in an A* search, given the heuristics provided above. As you trace the path, complete the table below, indicating which nodes are on the open and closed lists, along with their 'f' values:

	Open	Closed
Starting A^* search	[s_0, 6]	empty
s_0	[B, 8], [G, 9], [A, 9], [F, 10], [D, 11]	[S_0, 6]
B	[D, 6], [A, 9], [G, 9], [F, 10] [C, 14]	[S_0, 6], [B, 8]
D	[H, 4], [G, 9], [A, 9], [F, 10], [C, 14]	[S_0, 6], [B, 8] [D, 6]
H	[E, 4], [G, 9], [A, 9], [F, 10], [γ, 12], [C, 14]	[S_0, 6], [B, 8], [D, 6], [H, 4]
E	[γ, 3], [G, 9], [A, 9], [F, 10], [C, 14]	[S_0, 6], [B, 8], [D, 6], [H, 4], [E, 4]
γ	[G, 9], [A, 9], [F, 10], [C, 14]	[S_0, 6], [B, 8], [D, 6], [H, 4], [E, 4], [γ, 3]

3 Adversarial Search

In the following minimax game tree, Δ represent maximizing nodes and ∇ represent minimizing nodes while $\square$ nodes represent static evaluations of the states. Note that these values are not known to the search algorithm until it visits and explicitly evaluates the state. For the purposes of this question, the nodes in the tree should be processed from left to right. You may ignore the double slash (//) pruned branch labeled "A" until part c.



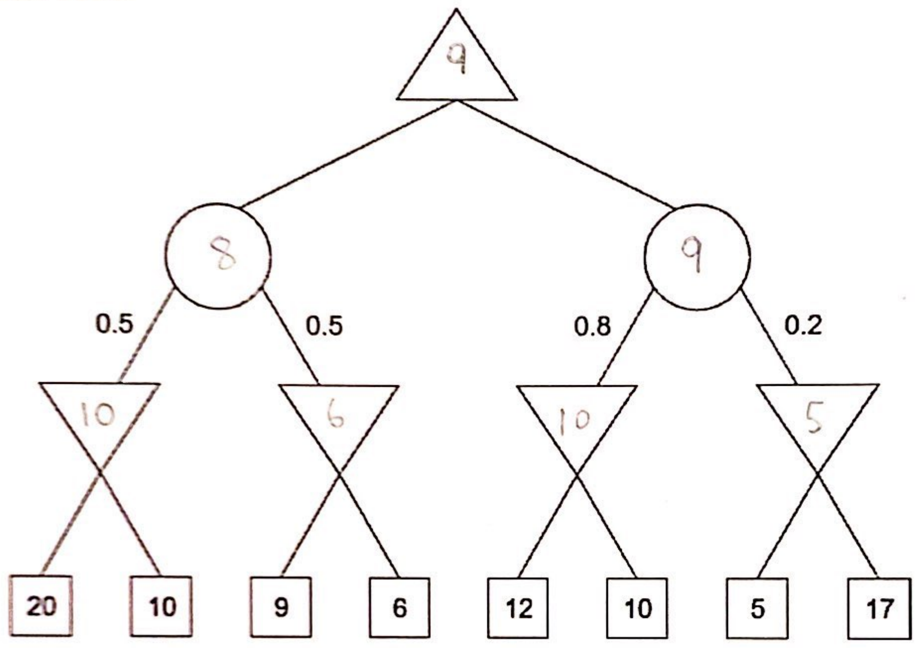
(a) (5 points) Let $x = 2$ in the static evaluation node containing the variable in the tree. Fill in the nodes in the tree with the correct values selected by the maximizing and minimizing players during the minimax algorithm.

(b) (5 points) Let $x = 2$ in the static evaluation node containing the variable in the tree. Apply alpha-beta pruning to the tree. Specifically, mark the appropriate edges in the tree with a double slash (//) to indicate which nodes were pruned using the algorithm.

(c) (5 points) What is the smallest value of x where branch "A" is pruned using alpha-beta?

$\boxed{8}$

Consider the following expectimax game tree. Note that the new ◯ nodes represent expectation nodes and the probability of their successors are denoted on the outgoing edges of these nodes.



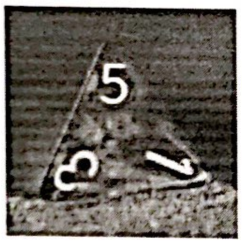
(d) (5 points) Fill in the nodes in the tree with the correct values selected by the maximizing and minimizing players during the expectimax algorithm.

4 Markov Decision Processes

Consider the following game. In each turn you have a choice of rolling a special die, or stopping the game. The die is biased - every time you roll, it produces 1, 3, 5 or 6 with equal probability. No other values are possible (It's a tetrahedral die.) At any point of time, you can either roll or stop if the total "score" (obtained by adding the values on the die from every rolling) is less than 7. If the "score" reaches or exceeds 7, you "go bust" and go to the final state, accruing zero reward.

When in any state other than the final state, you are allowed to take the stop action. When you stop, you reach the final state and your reward is the total "score" if it is less than 7.

Note: there is no direct reward from rolling the dice (or we could say that there is a reward but it's always 0). The only non-zero reward comes from explicitly taking the stop action. Discounting or not should not matter in the MDP for this game, but for the record, we assume no discounting (i.e., $\gamma = 1$).



1S
0
1
2
3
4
5
6
≥7

Figure 1: The value of a tetrahedral die like this, after a roll, is at the top, here 5, which should show equally well on any of the three faces that touch the top vertex.

(a) (6 points) Write down the states (in any order) and actions for this MDP. (Hint: there are 8 states in total and each should correspond to a numeric value except the initial and final states)

States	Actions
Initial	roll
1	roll, stop
2	:
3	:
4	:
5	:
6	:
Final	–

You start in the initial state where you roll. Then you roll until you choose to stop or your total score equals or is greater than 7.

(b) (10 points) Give the full transition function $T(s, a, s')$. Here s is a current state, a is an action, and s' is a possible next state when a is performed in s. Assuming your states are s_0, s_1, s_2, s_3 etc., and actions are a_0, a_1 etc., some examples of how you should write the function are as follows:

$$T(s, a, s') = \langle\text{value}\rangle; s = s_0, s' \in \{s_1, s_2, s_3, \ldots\}$$

$$T(s_0, a_1, s_1) = \langle\text{value}\rangle$$

$T(\text{initial}, \text{roll}, s') = 1$, $s' \in \{1, 3, 5, 6\}$, roll $\in \{1, 3, 5, 6\}$
$T(1, \text{roll}, s') = 3/4$, $s' \in \{2, 4, 6\}$, roll $\in \{1, 3, 5\}$
$T(1, \text{roll}, \text{final}) = 1/4$; roll $\in \{6\}$
$T(2, \text{roll}, s') = 1/2$, $s' = \{3, 5\}$, roll $\in \{1, 3\}$
$T(2, \text{roll}, \text{final}) = 1/2$; roll $\in \{5, 6\}$
$T(3, \text{roll}, s') = 1/2$, $s' = \{4, 6\}$; $T(3, \text{roll}, \text{final}) = 1/2$
$T(4, \text{roll}, s') = 1/4$, $s' = \{5\}$; $T(4, \text{roll}, \text{final}) = 1/2$
$T(5, \text{roll}, s') = 1/4$, $s' = \{6\}$; $T(5, \text{roll}, \text{final}) = 3/4$
$T(6, \text{roll}, \text{final}) = 1$
$T(s, \text{stop}, \text{final}) = 1$, $s \in \{1, 2, 3, 4, 5, 6\}$

(c) (2 points) Give the full reward function $R(s, a, s')$.

$R(s, \text{roll}, s') = 0$, $s \in \{\text{initial}, 1, 2, 3, 4, 5, 6\}$, $s' \in \{1, 2, 3, 4, 5, 6, \text{final}\}$
$R(s, \text{stop}, \text{final}) = s$, $s \in \{1, 2, 3, 4, 5, 6\}$

(d) (2 points) What is the optimal policy? There is no need to perform value iteration or use any fancy math; just write your answer in words.

Once your total score reaches 4 or higher, you have a 3/4 chance of going bust if you roll again. As such, I think you should roll until you reach a total score of 4 or higher then stop.

5 Computing MDP State Values and Q-Values

Consider an MDP with two states s_1 and s_2 and transition function $T(s, a, s')$ and reward function $R(s, a, s')$. Let's also assume that we have an agent whose discount factor is $\gamma = 1$. From each state, the agent can take three possible actions, i.e., $a \in \{x, y, z\}$. The transition probabilities for taking each action and the rewards for transitions are shown below.

s	a	s'	$T(s,a,s')$	$R(s,a,s')$
s_1	x	s_1	1	0
s_1	x	s_2	0	0
s_1	y	s_1	0.5	4
s_1	y	s_2	0.5	1
s_1	z	s_1	0.4	0
s_1	z	s_2	0.6	10
s_2	x	s_1	0	0
s_2	x	s_2	1	0
s_2	y	s_1	0.9	2
s_2	y	s_2	0.1	8
s_2	z	s_1	1	1
s_2	z	s_2	0	0

Compute V_0 , V_1 and V_2 for states s_1 and s_2.

(a). $V_0(s_1) =$ 0
(b). $V_0(s_2) =$ 0
(c). $V_1(s_1) =$ 6
(d). $V_1(s_2) =$ 1.8
(e). $V_2(s_1) =$ 7.8
(f). $V_2(s_2) =$ 7.8

Now, compute Q_2 for states s_1 and s_2.
(g). $Q_2(s_1, x) =$ 7.8
(h). $Q_2(s_1, y) =$ 9.8
(i). $Q_2(s_1, z) =$ 13.8
(j). $Q_2(s_2, x) =$ 7.8
(k). $Q_2(s_2, y) =$ 9.6
(l). $Q_2(s_2, z) =$ 8.8

Finally, compute V_2 and Q_2 for both states, but with $\gamma = 0.5$.

(m). $V_2(s_1) =$ 6.9
(n). $V_2(s_2) =$ 4.8

(o). $Q_2(s_1, x) =$ 3.45
(p). $Q_2(s_1, y) =$ 5.45
(q). $Q_2(s_1, z) =$ 9.45
(r). $Q_2(s_2, x) =$ 2.4
(s). $Q_2(s_2, y) =$ 4.2
(t). $Q_2(s_2, z) =$ 3.4