KOÇ UNIVERSITY
COLLEGE OF ENGINEERING

COMP 341: INTRODUCTION TO ARTIFICIAL INTELLIGENCE

Final Exam

FALL 2019, 28/12/2019
DURATION: 150 MINUTES

Name: Solutions

ID: 00110001 00110000 00110000

- This exam contains 11 pages including this cover page and 8 questions. Check to see if any pages are missing. Put your initials on the top of every page, in case the pages become separated.
- By submitting this exam, you **agree** to fully comply with Koç University Student Code of Conduct, and accept any punishment in case of failure to comply. If you do not submit this exam on time, you will receive 0 credits.
- The exam is **closed book** and **closed notes**. You are **not** allowed to use any additional material including any electronic equipment such as computers and mobile phones.
- You are expected to be able provide clear and concise answers. Gibberish will not receive any credit. Your answers need to be readable by a human, illegible writing is not gradable.
- Read each question carefully and make sure to follow instructions. The written instructions take precedence over an answer that the instructor might give you during the exam, unless the instructor makes a class wide announcement.
- Do not write in the table below.

Question:	1	2	3	4	5	6	7	8	Total
Points:	8	6	12	12	18	16	15	19	106
Score:									

1. (8 points) True or False :

False In particle filtering, particles never get zero weights.

True Viterbi Algorithm is used to find the most likely hidden state sequence given an observation sequence.

True Regularization helps with overfitting.

False It is better to split the data into test and training sets after the pre-processing and feature extraction steps.

True Markov Decision Processes are used to model problems involving uncertain action outcomes.

False Value iteration is a model-free method.

False In Q-learning, the agent must follow its own policy.

True Given good features, approximate Q-learning helps with generalization.

2. (6 points) Direct Policy Evaluation: You observed the following episodes from an **undiscounted** MDP with two states A and B as below (the numbers denote the reward you receive):

$$(A, +2) \rightarrow (A, +1) \rightarrow (B, -2) \rightarrow (A, +2) \rightarrow (B, -1) \rightarrow terminate$$
$$(B, -2) \rightarrow (A, +2) \rightarrow (B, -1) \rightarrow terminate$$

Estimate the value function using direct evaluation and fill in the table below. Make sure to show your work below the table.

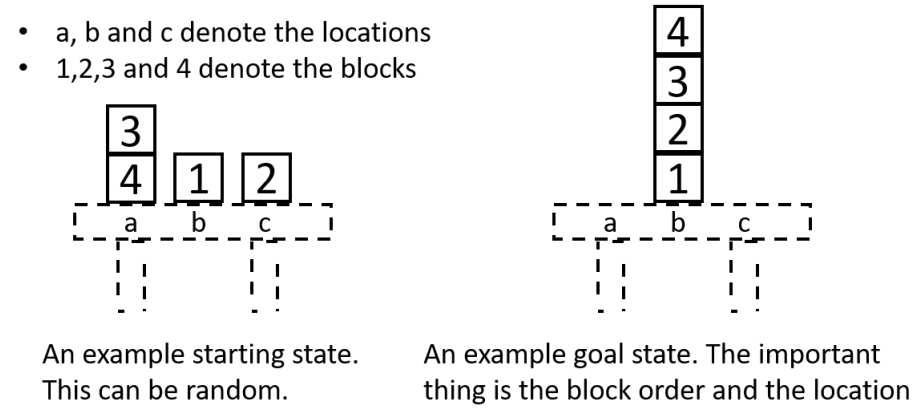
$V(A)$	$V(B)$
1	-1

Average the following for A:
First Episode: (+2+1-2+2-1,+1-2+2-1,+2-1) = (2,0,1)
Second Episode: (+2-1) = (1)
$V(A) = (2 + 0 + 1 + 1)/4 = 1$

Average the following for B:
First Episode: (-2+2-1,-1) = (-1,-1)
Second Episode: (-2+2-1,-1) = (-1,-1)
$V(A) = (-1 - 1 - 1 - 1)/4 = -1$

If you did it in a "first-visit" manner then
V(A) = (2+1)/2 = 1.5
V(B) = (-1+-1)/2 = -1

3. (12 points) Imagine a problem where there are n numbered blocks on top of a table. Blocks can be stacked on top of each other.There are also k discreet locations on the table. The goal is to stack the blocks such that they are ordered from the smallest to the largest on any of the locations. The starting block positions can be random. An example for $n = 4$ and $k = 3$ is given below. You can only move one



block at a time. You cannot move a block if there is another block on top of it. If you were to formulate this as a search problem (make sure to provide a general formulation and not something specific to the above example):

(a) (4 points) What would be your state representation? What would be your goal test for this representation?

A 2D $k \times nk$ integer array, where the first dimension represents the table locations and the second dimension represents the blocks for each location (3 points). Note that this handles stacks as well. Goal test would be to check whether the order is achieved by looking at each location (1 point). Other answers, such as using linked lists for each location, are possible.

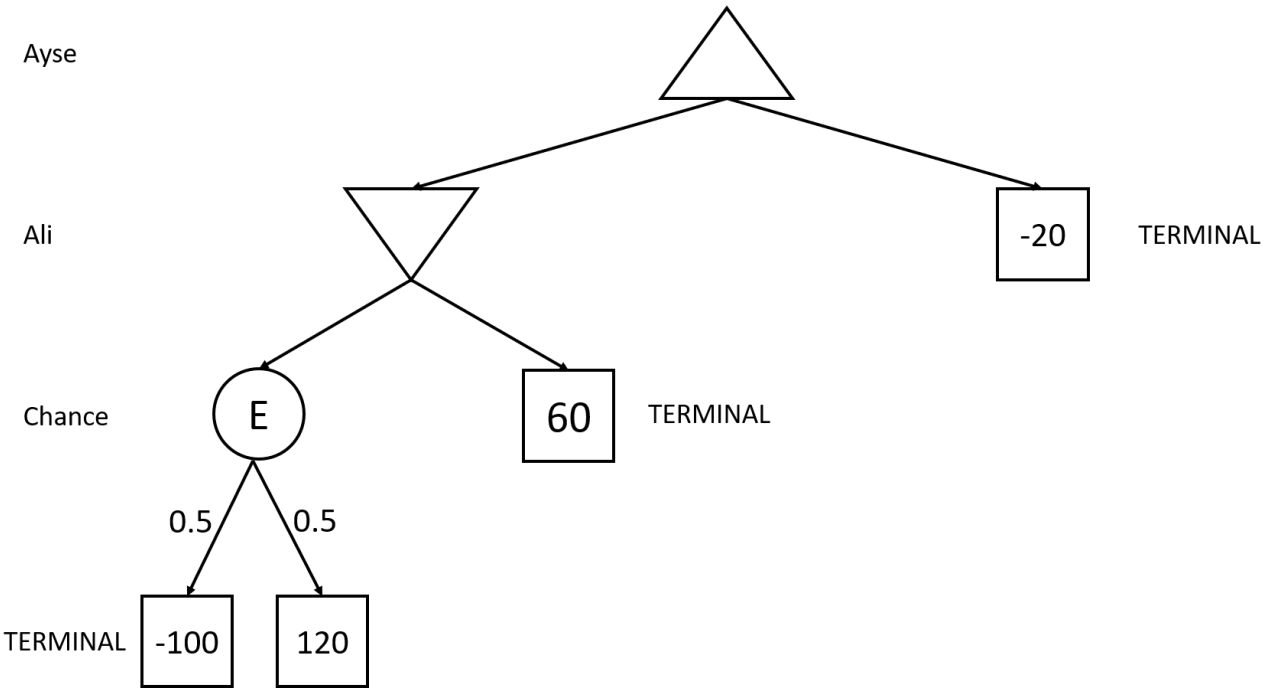
(b) (8 points) What algorithm would you chose to solve this problem? If any, what heuristic would you use and why? Would this heuristic favor solution efficiency or computational efficiency?

A* search if given a heuristic would get 3 points. A* without a heuristic, Uniform Cost Search, Breadth First Search, Iterative Deepening Depth First Search would get 2 points. Depth First Search with a justification for computational efficiency get 2 points. Just DFS would get 1 point. Some A* Heuristics:

- Number of blocks out of place: 1 point by itself. Admissible justification 2 points. Easy to compute additional 1 point.
- In addition to the above, +2 for each block that is not directly above the one it is supposed to. Note that to move a block to its appropriate place in this situation, you must at least make 2 moves. Such a heuristic, along with justification gets at least 2 points.
- Sum of Manhattan Distances to Place: 1 point by itself. Inadmissible but potentially faster to find a (non-optimal) solution justification 1 point. Easy to compute additional 1 point.

There are other heuristics. Grade will depend on the justification and admissibility. If your heuristic looks to be very good, you may get bonus points!

4. (12 points) Ayse and Ali are playing an adversarial game as shown in the game tree below. Ayse (the MAX player) and Ali (the MIN player) are both rational and they both know that their opponent is also a rational player. The game tree has one chance node E whose outcome can be either $E = -100$ or $E = +120$ with equal probability. The tree is given below:



Each player's utility is equal to the amount of money he or she has. The value x of each leaf node in the game tree means that Ali will pay Ayse x units of money after the game, so that Ayse's and Ali's utilities will be x and $-x$ respectively.

(a) (3 points) Suppose neither Ayse nor Ali knows the outcome of E before playing. What is Ayse's expected utility? Is this a fair game?

Value of $E = -100 \cdot 0.5 + 120 \cdot 0.5 = 10$. Thus Ali would chose left. Then Ayse would also chose left. Thus Ayse's expected utility is 10. This is not a fair game since Ayse will win more money in the long run (non-zero expected utility).

(b) (1 point) Ceren, a good friend of Ayse's, has access to E and can secretly tell Ayse the outcome of E before the game starts (giving Ayse the true outcome of E without lying). However, Ali is not aware of any communication between Ayse and Ceren, so he still assumes that Ayse has no access to E. What is Ayse's expected utility if Ceren tells her that $E = -100$?

-20 since that Ali would chose left to win 100. Ayse would rather pay 20 than 100 thus she would chose right

(c) (1 point) (Continuing from part b) What is Ayse's expected utility if Ceren tells her that $E = +120$?

120 since that Ali would still chose left and the outcome would be 120.

(d) (1 point) (Continuing from part c) What is Ayse's expected utility if Ceren secretly tells her the outcome of E before playing?

Since both are equally likely (0.5 probability for E's outcomes), Ayse's utility in this case would be: $-20 \cdot 0.5 + 120 \cdot 0.5 = 50$

(e) (2 points) We define the value of *private information* $V_A^{pri}(X)$ of a random variable X to a player A as the difference in player A's expected utility after the outcome of X becomes a private information to player A, such that A has access to the outcome of X, while other players have no access to X and are not aware of A's access to X. Can we assert whether or not $V_A^{pri}(X) \geq 0$? Justify your answer.

Yes. There is no deception so we know that the private information is correct. Also, player A can always choose to ignore this information and act in the same way as if he/she doesn't know this information, thus player A is guaranteed to obtain at least the same utility as before.

(f) (1 point) What is, $V_{Ayse}^{pri}(E)$, the value of private information of E to Ayse in the game tree above?

Follow the definition: 50 - 10 = 40

(g) (1 point) Ahmet also has access to E, and can make a public announcement of E (announcing the true outcome of E without lying), so that both Ayse and Ali will know the outcome of E and are both aware that their opponent also knows the outcome of E. Also, Ayse cannot obtain any information from Ceren now. What is Ayse's expected utility if Ahmet announces that $E = -100$?

Remove the chance effect and follow the minimax algorithm to get -20

(h) (1 point) (Continuing from part g) What is Ayse's expected utility if Ahmet announces that $E = +120$?

Remove the chance effect and follow the minimax algorithm to get 60

(i) (1 point) (Continuing from part h) What is Ayse's expected utility if Ahmet announces the outcome of E before playing?

Equally likely thus 20.

5. (18 points) Grandpac is hunting ghosts in a grid using his sonar. He cannot directly see the ghosts but he can make noisy observations about their locations. With years of practice, he has an idea of how ghosts might move. The world is such that the time progresses in discreet steps. Furthermore Grandpac and ghosts move in turns at each time step, Grandpac going first.

   (a) (4 points) Which representation approach among Static Bayesian Networks (BN), Hidden Markov Models (HMM) and Markov Decision Processes (MDP) should Grandpac chose to find the most likely location of a ghost at a given time step T? Formulate the problem based on your selection without going into details.

   Representation: HMMs with sonar as emissions and "idea of ghosts" as the transition function. Prior probs can be uniform.

   (b) (2 points) Hunting a ghost gives Grandpac 20 points. He must use his "Proton Pack" on the ghosts to hunt them but this special weapon costs 10 points to use. Suppose that he has estimated the position of a ghost at the grid location (x, y) with probability p. What is his expected utility as a function of p? For what value of p the Grandpac should use his weapon on the ghost? If he never or always should, state it so.

   $20p - 10$, use when $p > 0.50$.

   (c) (12 points) Ghastly is tired of Grandpac hunting his ghost friends and decides to intervene. When Ghastly is around, ghosts can no longer move and the grid becomes slippery such that Grandpac cannot always move where he wants to. Grandpac and Ghastly can see each other on the grid. Ghastly acts againts Grandpac. The turn taking nature of the world stays the same with Grandpac going first. How should Grandpac represent this problem? What algorithm(s) can he use to solve it? (You are free to chose anything from the topics that we have covered or come up with your own approach)

   This was an open-ended question with many possible answers. There are a few potential directions to select. You should have paid attention to:

   - Ghosts are immobile: This affects the state OR the reward function (if MDP route is selected). Assume known (1 point), low uncertainty due to stationary ghosts and many measurements Grandpac will take (2 points) or if reward, unknown to be discovered (2 points).
   - Ghastly and Grandpac can see each other: This affects the state. Implying no uncertainty for this part. (1 point)
   - Slippery Grid: Action uncertainty for Grandpac (2 points). May assume the same for Ghastly as well but do not have to.
   - Adversarial Nature: Need to somehow plan for multiple agents. (2 points)
   - Zero-Sumness: Can assume to be zero-sum with Grandpac's utility since Grandpac wants to hunt and Ghastly wants to protect ghosts. (1 point)

   Mentioning these somehow already gets you 8 points. The remaining is for the method:

   - Adversarial search: Expectiminimax for actions. Maybe roll the ghost uncertainty into the search or assume known. Up to 3 points based on formulation
   - MDPs: Define a zero-sum MDP with minimax value functions instead of just max. Up to 4 points. Need to assume known ghost places
   - Reinforcement Learning: Similar to above but making it online and having the ghosts be part of the unknown reward function. Up to 5 points.
   - Partially Observable MDPs (POMDP): MDPs where there is state and action uncertainty. In addition need to define a minimax formulation. Up to 8 points.

6. (16 points) We have five planes: A, B, C, D, and E and two runways: international and domestic. We would like to schedule a time slot and runway for each aircraft to either land or take off. We have four time slots: $\{1; 2; 3; 4\}$ for each runway, during which we can schedule a landing or take off of a plane. We must find an assignment that meets the following constraints:

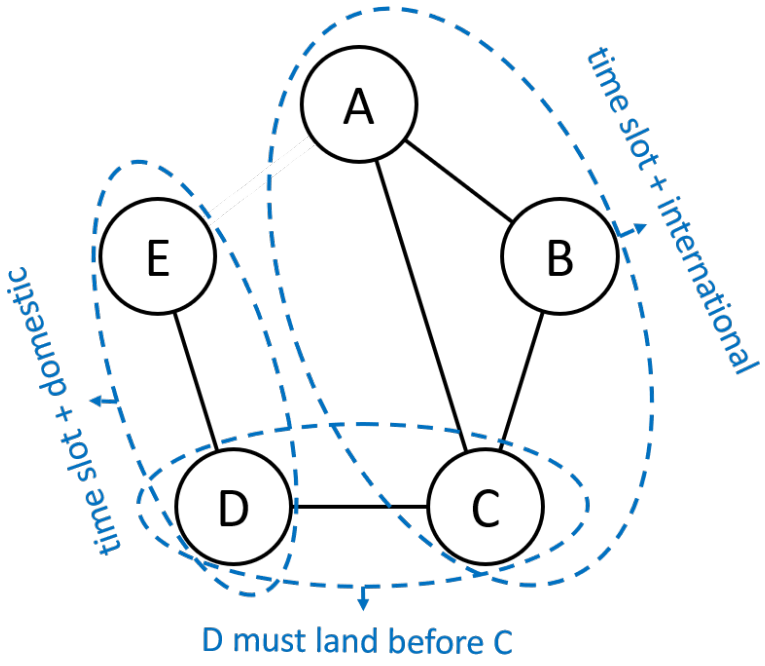
   - Plane B has lost an engine and must land in time slot 1.
   - Plane D can only arrive at the airport to land during or after time slot 3.
   - Plane A is running low on fuel but can last until at most time slot 2.
   - Plane D must land before C takes off, because some passengers must transfer from D to C.
   - Planes A, B, and C cater to international flights and can only use the international runway.
   - Planes D and E cater to domestic flights and can only use the domestic runway.
   - No two aircrafts can reserve the same time slot for the same runway.

   (a) (8 points) Formulate this as a CSP in terms of variables, domains and constraints. Express the constraints using mathematical or logical notation rather than with words. Draw the constraint graph.

   Variables are the planes: A,B,C,D,E (1 point)
   Domain is a tuple made up of the time slot and the runway: $(\{1, 2, 3, 4\}, \{domestic, international\})$ (1 point)
   Constraints: (0.75 point per correct constraint. -0.5 if a wrong constraint is given)

   - B[0] = 1
   - D[0] $\geq$ 3
   - A[0] $\leq$ 2
   - D[0] < C[0]
   - A[1] = B[1] = C[1] = *international*, D[1] = E[1] = *domestic*
   - A $\neq$ B $\neq$ C $\neq$ D $\neq$ E

   Constraint Graph: (1.5 point)



(b) (4 points) What are the domains of the variables after enforcing arc-consistency? (Cross out values that are no longer in the domain.)

A	~~1~~	2	~~3~~	~~4~~
B	1	~~2~~	~~3~~	~~4~~
C	~~1~~	~~2~~	~~3~~	4
D	~~1~~	~~2~~	3	~~4~~
E	1	2	~~3~~	4

A:1, B:1, C:1,2,3, E:3 are 1/3, others are 1/4. Additional 1/4 for getting everything correct. -1/4 for each mistake

(c) (4 points) Arc-consistency can be rather expensive to enforce, and we believe that we can obtain faster solutions using only forward-checking on our variable assignments for this problem. Using the Minimum Remaining Values (MRV) heuristic, perform backtracking search on the graph, breaking ties by picking lower values and characters first. List the (variable; assignment) pairs in the order they occur (including the assignments that are reverted upon reaching a dead end). Enforce unary constraints before starting the search.

(B, 1, international), (A, 2, international), (C, 3, international), (C, 4, international), (D, 3, domestic), (E, 1, domestic):

Step 1: B has 1 remaining value as 1 which we pick. Forward checking (FC) removes 1 from A and C.

Step 2: A has 1 remaining value as 2 which we pick. FC removes 2 from C.

Step 3: C has 2 remaining values, 3 and 4. Pick 3 since it is lower. FC removes 3 and 4 from D, making it impossible solve. So we backtrack.

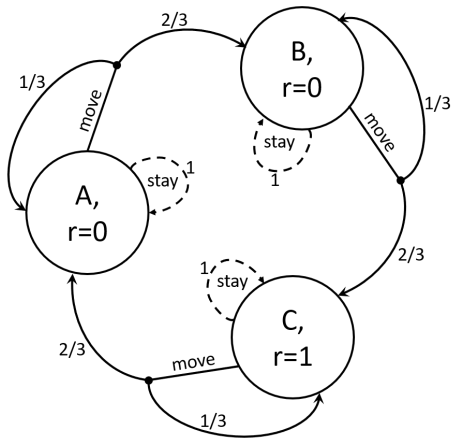
Step 4: Pick 4 for C. FC removes 4 from D.

Step 5: D has 1 remaining, 3 which we pick. FC removes 3 from E.

Step 6: Pick 1 for E since it is the lowest available. We are done.

All or nothing grading, make sure the assignments are correct.

7. (15 points) Dynamic Programming: Answer the questions based on the MDP below



States: {A, B, C}
Actions and Transition Probabilities:
- stay: stays in the current state with probability 1
- move: moves to the next state with 2/3 probability, stays in the current state with 1/3 probability

Rewards: R(A) = 0, R(B) = 0, R(C) = 1
Discount Factor: $\gamma = 0.6$

(a) (12 points) Perform two steps of value iteration and fill in the table below. Make sure to show your work below the table. An equation is given as a helper.

$$V^*(s) = R(s) + \gamma \max_a (\sum_{s'} P(s'|s,a)V^*(s'))$$

Iteration	$V(A)$	$V(B)$	$V(C)$
0	0	0	1
1	0	0.4	1.6
2	0.16	0.72	1.96

Iteration 1: First expression in the max is for the move action and the other is for the stay action. I moved the $R(s)$ and γ inside the max for convenience

$$\begin{aligned}
V^\pi(A) &= \max(0 + 3/5 \cdot (1/3 \cdot 0 + 2/3 \cdot 0, 0 + 3/5 \cdot (1 \cdot 0)) \\
&= \max(0,0) = 0, \text{both actions yield the same outcome} \\
V^\pi(B) &= \max(0 + 3/5 \cdot (1/3 \cdot 0 + 2/3 \cdot 1, 0 + 3/5 \cdot (1 \cdot 0)) \\
&= \max(2/5, 0) = 2/5, \textit{move} \text{ action is better} \\
V^\pi(C) &= \max(1 + 3/5 \cdot (1/3 \cdot 1 + 2/3 \cdot 0, 1 + 3/5 \cdot (1 \cdot 1)) \\
&= \max(6/5, 8/5) = 8/5, \textit{stay} \text{ action is better}
\end{aligned}$$

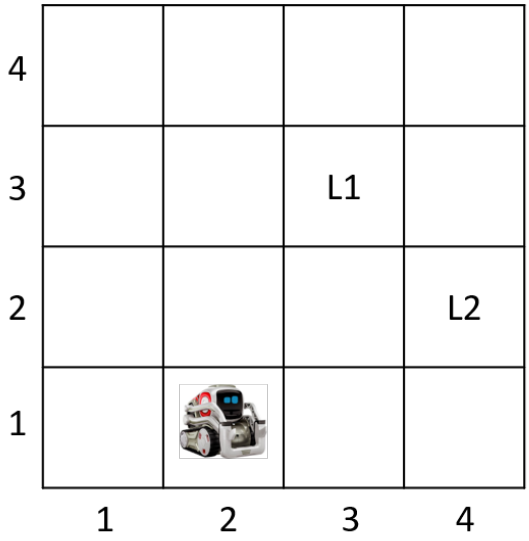
Iteration 2:

$$\begin{aligned}
V^\pi(A) &= \max(0 + 3/5 \cdot (1/3 \cdot 0 + 2/3 \cdot 2/5, 0 + 3/5 \cdot (1 \cdot 0)) \\
&= \max(4/25, 0) = 4/25 = 0.16, \textit{move} \text{ action is better} \\
V^\pi(B) &= \max(0 + 3/5 \cdot (1/3 \cdot 2/5 + 2/3 \cdot 8/5), 0 + 3/5 \cdot (1 \cdot 2/5)) \\
&= \max(18/25, 6/25) = 18/25 = 0.72, \textit{move} \text{ action is better} \\
V^\pi(C) &= \max(1 + 3/5 \cdot (1/3 \cdot 8/5 + 2/3 \cdot 0, 1 + 3/5 \cdot (1 \cdot 8/5)) \\
&= \max(33/25, 49/25) = 49/25 = 1.96, \textit{stay} \text{ action is better}
\end{aligned}$$

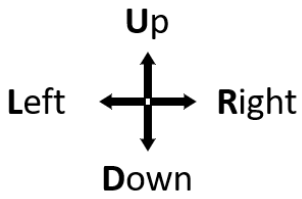
2 points per correct value. 1 point if there is a mathematical error.

(b) (3 points) What is the policy extracted from the calculated values?
Look at the argmax for iteration 2.
$\pi(A) = move$, $\pi(B) = move$, $\pi(C) = stay$, 1 point per correct answer

8. (19 points) Value Function Approximation. The robot given below is trying to explore the area and find safe routes to resources. The state of the robot is the grid it is in. Robot can move in four cardinal directions. The landmarks, L1 and L2, signify that there is a resource close-by. The locations of these landmarks are known to the robot (L1 $= (x_{l1}, y_{l1})$ and L2 $= (x_{l2}, y_{l2})$).



- Actions:



- State: (x,y) location of the robot, e.g. (2,1) in the figure
- L1 and L2: Known landmarks
- Discount: 1.0

The robot wants to use function approximation get the values of each state. It decides to use the following features, given the current state $s = (x, y)$.

- Current x-coordinate: $f_1(s) = x$
- Current y-coordinate: $f_2(s) = y$
- Manhattan Distance to L1: $f_3(s) = |x - x_{l1}| + |y - y_{l1}|$
- Manhattan Distance to L2: $f_4(s) = |x - x_{l2}| + |y - y_{l2}|$

Furthermore, it uses a linear function approximator:

$$\hat{V}(s, w) = w_1 f_1(s) + w_2 f_2(s) + w_3 f_3(s) + w_4 f_4(s) = w^T f(s)$$

The robot then observes the following transitions:

$$(2,1), -0.1 \rightarrow (2,2), -0.1 \rightarrow (2,3), +1$$

Answer the questions below:

(a) (3 points) Calculate the feature vectors of the observed states
$L1 = (3,3), L2 = (4,2)$

$$f((2,1)) = \begin{bmatrix} 2 \\ 1 \\ 3 \\ 3 \end{bmatrix}, \quad f((2,2)) = \begin{bmatrix} 2 \\ 2 \\ 2 \\ 2 \end{bmatrix}, \quad f((2,3)) = \begin{bmatrix} 2 \\ 3 \\ 1 \\ 3 \end{bmatrix}$$

(b) (12 points) Use the observed transitions to update the weights, starting from zero weights with the learning rate $\alpha = 0.2$.

$w = [0,\ 0,\ 0,\ 0]^T$

First Update: $w \leftarrow w + \alpha(r + w^T f(s_2) - w^T f(s_1))f(s_1)$

$$w \leftarrow \begin{bmatrix}0\\0\\0\\0\end{bmatrix} +0.2(-0.1+0-0) \begin{bmatrix}2\\1\\3\\3\end{bmatrix} \Rightarrow w = \begin{bmatrix}-0.04\\-0.02\\-0.06\\-0.06\end{bmatrix}$$

Second Update: $w \leftarrow w + \alpha(r + w^T f(s_3) - w^T f(s_2))f(s_2)$

$$w \leftarrow \begin{bmatrix}-0.04\\-0.02\\-0.06\\-0.06\end{bmatrix} +0.2(-0.1+[-0.04,\ -0.02,\ -0.06,\ -0.06]^T \begin{bmatrix}2\\3\\1\\3\end{bmatrix} -[-0.04,\ -0.02,\ -0.06,\ -0.06]^T \begin{bmatrix}2\\2\\2\\2\end{bmatrix}) \begin{bmatrix}2\\2\\2\\2\end{bmatrix}$$

$$= \begin{bmatrix}-0.04\\-0.02\\-0.06\\-0.06\end{bmatrix} + 0.2(-0.1-0.38+0.36) \begin{bmatrix}2\\2\\2\\2\end{bmatrix} \Rightarrow w = \begin{bmatrix}-0.088\\-0.068\\-0.108\\-0.108\end{bmatrix}$$

It is okay if you did it separately for each weight vector w_i, should get the same answer. 1.5 point per weight per step (4 dimensions and 2 steps = 8 answers). 0.5-1 point for calculation error if the path is correct. To grade the second step, assume that the 1st step's output is totally correct.

(c) (4 points) If the robot wanted to learn Q-values, how would you change the formulation? Would you add more features? What would you do to handle the addition of actions?

This may have multiple answers. Some options regarding features :

- Concatenating a one-hot vector of dimension 4 (number of actions) to the original feature set
- Adding the resulting Manhattan Distances (MD) to the landmarks if the specific action was taken at that state. This can even replace the original MD features
- Similar to above but instead binary or three valued (-1,0,1) to indicate the change in MD to landmarks.
- Something to do with the bearing of the action and the angle between the landmarks and the robot
- Removing the coordinates (must be in combination with more landmark relative action features, e.g. the previous 3 bullet points)

Otherwise we need to have exploration and exploitation trade-off strategy. Then the update rules are mostly similar.