KOÇ UNIVERSITY
College of Engineering

COMP 341: INTRODUCTION TO ARTIFICIAL INTELLIGENCE

Final

Fall 2023, 19/01/2024
Duration: 150 Minutes

Name: Solutions

ID: 00110001 00110000 00110000

- This exam contains 16 pages including this cover page and 10 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	9	10	Total
Points:	6	8	12	8	8	10	10	15	14	9	100
Score:											

1. (6 points) True or False :

   True ______ Training accuracy in classifcation problems can be 100%.

   False ______ In machine learning, direct model parameters (e.g. regression weights) are calculated to reduce overfitting.

   True ______ In value iteration, policy may converge before the values.

   True ______ Learning a policy for a problem modeled by a MDP results in a reflex-agent.

   False ______ Knowing the reward function is enough to extract a policy from TD learning.

   False ______ We do not need to know the discount factor of an MDP for approximate Q-learning.

2. (8 points) Answer the questions about Machine Learning (ML) below.

   (a) (2 points) Give two examples of how we would utilize ML within our Agent-Based AI formulation.

   Learning transition models, heuristics (more RL), policy from demonstrations, mid/high level features from examples

   (b) (3 points) Can we solve **underfitting** issues in ML by extracting the correct features? Explain your answer (This is not asking about how to do feature selection.)

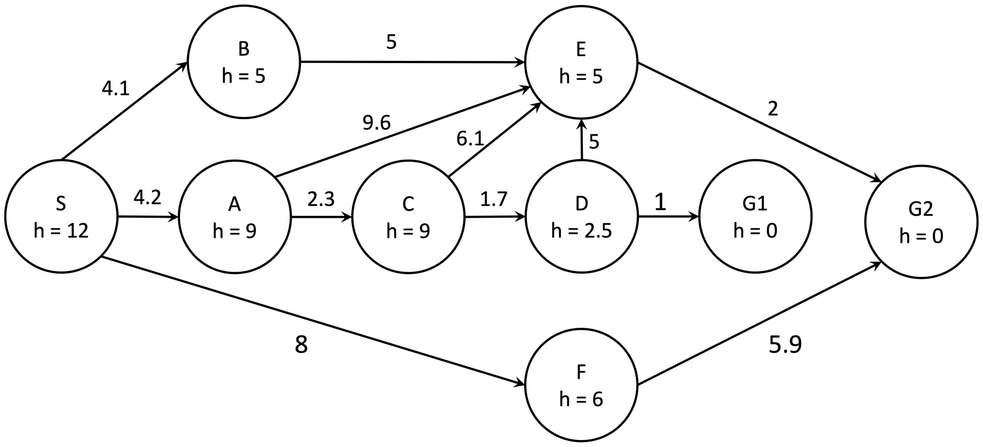
   More informative features, less need of complex models

   (c) (3 points) Can we solve **overfitting** issues in ML by extracting the correct features? Explain your answer (This is not asking about how to do feature selection.)

   More relevant features, less noise to overfit

3. (12 points) You are given the directed and weighted graph below, where **S** is the start state, and **G1** and **G2** are the goal states. The arcs represent transitions and directions, with their cost given next to them. The numbers inside the nodes are their heuristic value.

   You are asked to write the visiting/expanding order (order of popping from the frontier) of the nodes and the resulting solution paths with the given algorithms. The neighbors are added alphabetically to the frontier. The alphabetical order is: A,B,C,D,E,F,G1,G2,S. For algorithms using priority queues, break the ties alphabetically and assume that the priority values are updated when an existing node with a higher priority is being pushed.



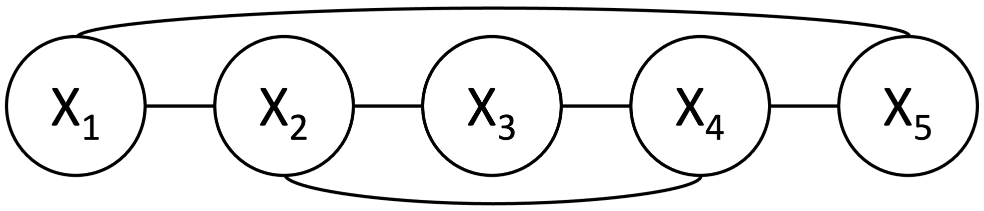
   (a) (4 points) Depth First Search:
   Popped Node Order:
   Stack based: S,F,G2
   Recursive: S,A,C,D,E,G2
   Resulting Path:
   Stack based: S-F-G2
   Recursive: S-A-C-D-E-G2

   (b) (4 points) Breadth First Search:
   Popped Node Order:
   S,A,B,F,C,E,G2
   Resulting Path:
   S-F-G2

   (c) (4 points) Greedy Search:
   Popped Node Order:
   S,B,E,G2
   Resulting Path:
   S-B-E-G2

4. (8 points) Miscellaneous.

   (a) (4 points) You are given a Constraint Satisfaction Problem (CSP). In the below graph, the nodes represent variables (X_i) and the arcs represent inequality constraints of this CSP. The domains (with members R_i) of the variables are given in the table. Apply the constraint propagation algorithm (AC3) given the initial domains and write the resulting domains to the table.



	X_1	X_2	X_3	X_4	X_5
Init	R_2, R_3	R_1	R_1, R_2, R_3	R_1, R_3	R_2, R_3
Answer	R_3	R_1	R_2	R_3	R_2

0.5 for X3, X4 R1 removal, 1 point per remaining removals, -0.5 point per wrong removal

   (b) (4 points) In simplified terms, integrated circuits are made up of components (chips, resistors, capacitors etc.) and connections. We want to have correct connections, the connections routes should not cross each other and everything should fit in a given shape (or area). Would you pick simple hill-climbing/descending, simulated annealing or genetic algorithms to solve this problem? Explain your reasoning. You do not need to formulate the problem in detail but do so if it helps your reasoning.

   Open ended. Want to see how students argue. I would pick in the order (i) simulated annealing, (ii) genetic algorithms and (iii) hill-climbing. The last one is obvious due to local minima reasons and would get only 2 points if the students give some reasons (e.g. speed of solution). Genetic algorithms are globally optimal but it is hard to formulate them. Merging two solutions have the potential to be both beneficial and not. Simualted annealing would be preffered since incremental changes would work better given a decent start point and it is also globally optimal given the right circumstances.

5. (8 points) You made measurements of three Boolean random variables, A, B and C. However, you lost the data labels, but not the sample counts. As a result, you gave each variable a new name; X_1, X_2 and X_3. You want to figure out the labels again, using the sample counts, i.e. figure out which X_i corresponds to which random variable (among A, B, C). In addition to the counts, you know that:

   - A and B are independent
   - True value for A, is more common than the true value of B.

   Given the sample counts and the additional information, is it possible to find the original labels (i.e., can you find a mapping between the original random variables and X_i's)? If yes, sketch your idea below. If not, argue why. Note that we are not asking you to do any computations.

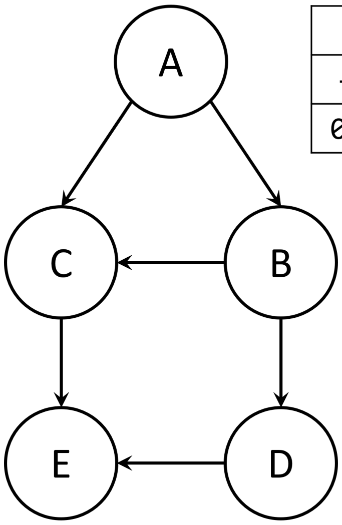
   According to the first bulletpoint: $\forall A = a, B = b : P(A = a, B = b) = P(A = a)P(B = b)$. Thus, we need to calculate $P(X_1), P(X_2), P(X_3)$, $P(X_1, X_2), P(X_1, X_3), P(X_2, X_3)$ and check the equality for all of X_1, X_2, X_3. (4 points)

   Note that the equality check will need to be "approximate" due to noise in real life measurements, e.g. check $|P(X_1)P(X_2) - P(X_1, X_2)| < \epsilon$. (2 points)

   Once we find the pair that satisfies the independence relation, we can look at their T counts or probabilities to decide on which one is A and which one is B. We should already have this information since we calculated $P(X_i)$. (1 points)

   The remaining one is C (1 points).

6. (10 points) Answer the following questions. Use the Bayesian Network below for parts (a) and (b).



P(A)	
+a	-a
0.4	0.6

	P(B|A)	
A	+b	-b
+a	0.2	0.8
-a	0.4	0.6

	P(D|B)	
B	+d	-d
+b	0.1	0.9
-b	0.7	0.3

		P(C|A,B)	
A	B	+c	-c
+a	+b	0.4	0.6
+a	-b	0.7	0.3
-a	+b	0.8	0.2
-a	-b	0.9	0.1

		P(E|C,D)	
C	D	+e	-e
+c	+d	0.2	0.8
+c	-d	0.7	0.3
-c	+d	0.6	0.4
-c	-d	0.1	0.9

(a) (2 points) Suppose you want to calculate $P(A|-b,-e)$ using sampling and the latest sample is $(-a,-b,+c,-d,-e)$. You randomly picked C to be sampled next. Write down the distribution (not the table, just the expression) that you are going to use to sample C. Simplify it as much as you can (you won't need to go too deep).

We want $P(C|+a,+b,+d,-e)$ (0.5 points for this). There are multiple ways to get this, in the end we will have:

$$\begin{aligned} P(C|-a,-b,-d,-e) &= \frac{P(-a)P(-b|-a)P(C|-a,-b)P(-d|-b)P(-e|C,-d)}{\sum_C P(-a)P(-b|-a)P(C|-a,-b)P(-d|-b)P(-e|C,-d)} \\ &= \frac{P(C|-a,-b)P(-e|C,-d)}{\sum_C P(C|-a,-b)P(-e|C,-d)} \end{aligned}$$

If you have $C=-c$ anywhere, it is automatically a 0.

(b) (3 points) Now calculate the conditional probability table associated with your answer to the previous question (Hint: This is not a complicated inference question).

C	$P(C\|-a,-b)P(-e\|C,-d)$	$P(C\|-a,-b,-d,-e)$
$+c$	$P(+c\|-a,-b)P(-e\|+c,-d)=0.9\cdot 0.3=0.27$	$0.27/(0.27+0.09)=3/4$
$-c$	$P(-c\|-a,-b)P(-e\|-c,-d)=0.1\cdot 0.9=0.09$	$0.09/(0.27+0.09)=1/4$

(c) (3 points) Given the below samples obtained from sampling, estimate $P(A|-b,-e)$.

A	B	C	D	E	Counts
$+a$	$-b$	$+c$	$+d$	$-e$	21
$+a$	$-b$	$+c$	$-d$	$-e$	5
$+a$	$-b$	$-c$	$-d$	$-e$	3
$+a$	$-b$	$-c$	$+d$	$-e$	2
$-a$	$-b$	$+c$	$+d$	$-e$	16
$-a$	$-b$	$+c$	$-d$	$-e$	2
$-a$	$-b$	$-c$	$+d$	$-e$	1

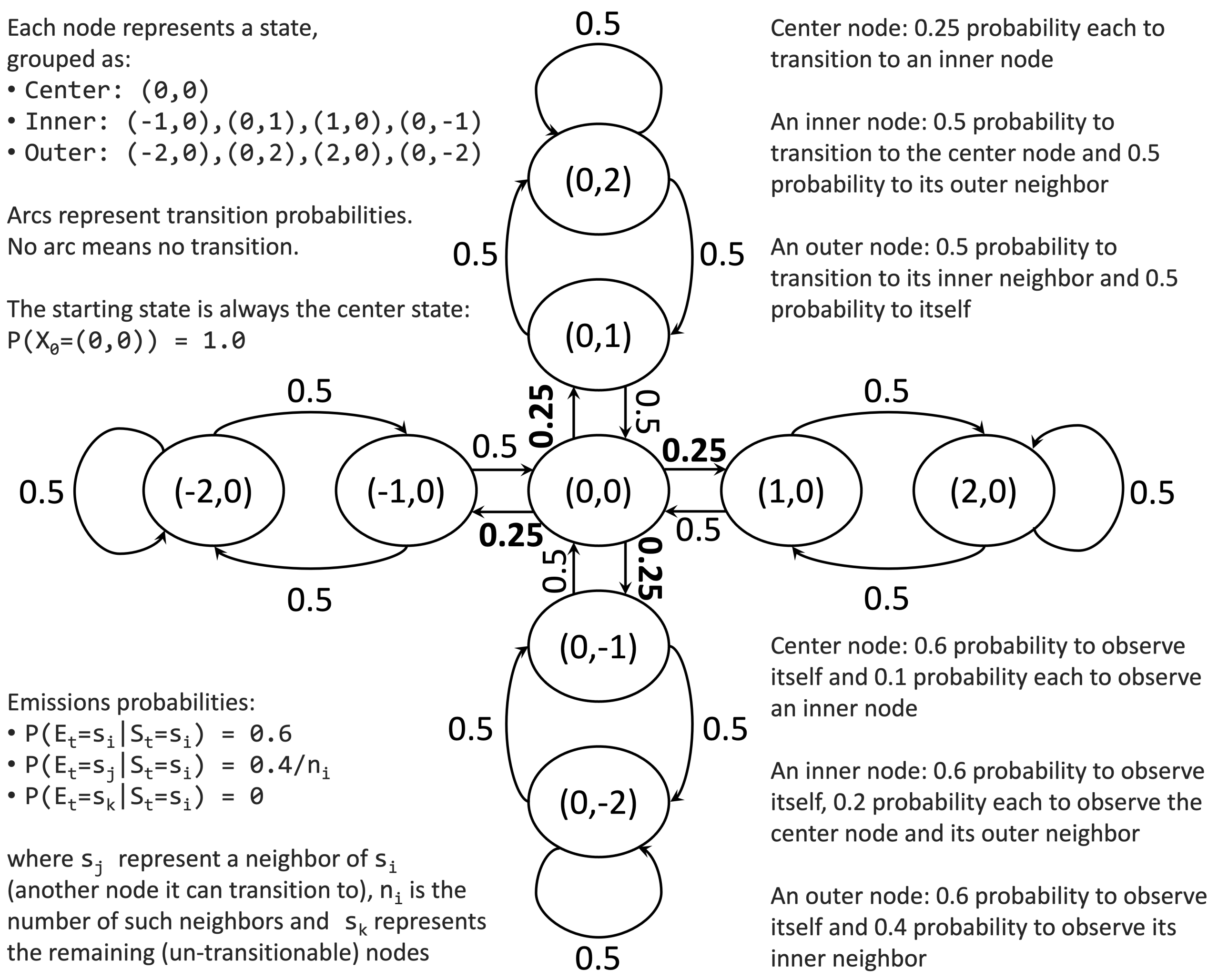
This is a very easy question. The only trick is to not try to calcualte their weights since we obtained them through Sampling. We can see that there are 50 samples total. We have 31 samples with $A = +a$ and 19 with $A = -a$. Thus:

$P(A = +a|-b,-e) = 31/50 = 0.62$ and $P(A = -a|-b,-e) = 19/50 = 0.38$.

(d) (2 points) Can we use approximate inference in Decision Networks, especially to calculate value of information? If so, how and for what? If not, why not?

Yes, there is nothing stopping us. We need to perform inference steps to calculate expected utilities. Approximate inference can be used here. The only caveat is that we may not be able to get 0 results for VPIs due to approximation error (last sentence bonus 1 point).

7. (10 points) Answer the questions based on the given Hidden Markov Model (HMM) below. This time we are going to use particle filtering to estimate state distributions.



Each node represents a state, grouped as:
- `Center: (0,0)`
- `Inner: (-1,0),(0,1),(1,0),(0,-1)`
- `Outer: (-2,0),(0,2),(2,0),(0,-2)`

Arcs represent transition probabilities. No arc means no transition.

The starting state is always the center state: $P(X_0=(0,0)) = 1.0$

Center node: 0.25 probability each to transition to an inner node

An inner node: 0.5 probability to transition to the center node and 0.5 probability to its outer neighbor

An outer node: 0.5 probability to transition to its inner neighbor and 0.5 probability to itself

Emissions probabilities:
- $P(E_t=s_i|S_t=s_i) = 0.6$
- $P(E_t=s_j|S_t=s_i) = 0.4/n_i$
- $P(E_t=s_k|S_t=s_i) = 0$

where s_j represent a neighbor of s_i (another node it can transition to), n_i is the number of such neighbors and s_k represents the remaining (un-transitionable) nodes

Center node: 0.6 probability to observe itself and 0.1 probability each to observe an inner node

An inner node: 0.6 probability to observe itself, 0.2 probability each to observe the center node and its outer neighbor

An outer node: 0.6 probability to observe itself and 0.4 probability to observe its inner neighbor

(a) (2 points) We start with 4 particles; $(-2,0),(-1,0),(0,0),(0,1)$. We make the observation $E=(-1,0)$. Fill the below table with the weight of each particle and show your work.

$(-2,0)$	$(-1,0)$	$(0,0)$	$(0,1)$
0.4	0.6	0.1	0.0

Trivial, we just plug in the emission probabilities:

$$P(E_t=(-1,0)|X_t=(-2,0)) = 0.4$$
$$P(E_t=(-1,0)|X_t=(-1,0)) = 0.6$$
$$P(E_t=(-1,0)|X_t=(0,0)) = 0.1$$
$$P(E_t=(-1,0)|X_t=(0,1)) = 0.0$$

If E and X ar reversed, 0 points. Also 0 points for no work or explanation. 0.5 points per correct otherwise

(b) (4 points) You want to sample the next four particles with probabilities proportional to their total weights, as calculated in the previous part. You are given the following uniform random samples, [0.5, 0.25, 0.6, 0.95]. Assuming the intervals for sampling are formed with the order $(-2,0),(-1,0),(0,0),(0,1)$, what are the states of the next 4 particles? Show your work.

For sampling, we need to normalize the weights (0.4+0.6+0.2+0=1.1), calculate the cumulative distribution and sample:

X	Weights	Normalized	Cumulative
(-2,0)	0.4	4/11	4/11
(-1,0)	0.6	6/11	10/11
(0,0)	0.2	1/11	11/11
(0,1)	0.0	0.0	11/11

The next step is to pick the particles based on where the uniform random samples fall. 0.5 yields (-1,0), 0.25 yields (-2,0), 0.6 yields (-1,0) and 0.95 yields (0,0).

We end up with 2 particles with state (-1,0), 1 particle with state (-2,0) and 1 particle with state (0,0). 1 point per correct. 0.5 point per correct but wrong order.

(c) (4 points) You want to sample next points based on the underlying model dynamics (ie. you want to elapse time), starting from the previous part. You are given the following uniform random samples, [0.4, 0.7, 0.25, 0.9]. Make sure the order of intervals for sampling are clear (left to up to you!). What are the states of the next 4 pre-observation particles? Show your work.

The transition probabilities and cumulative disttributions for the relevant states:

X_t	X_{t+1}	$P(X_{t+1}\|X_t)$	Cumulative
(-2,0)	(-2,0)	0.5	0.5
(-2,0)	(-1,0)	0.5	1.0
(-1,0)	(-2,0)	0.5	0.5
(-1,0)	(0,0)	0.5	1.0
(0,0)	(-1,0)	0.25	0.25
(0,0)	(0,1)	0.25	0.5
(0,0)	(1,0)	0.25	0.75
(0,0)	(0,-1)	0.25	1.0

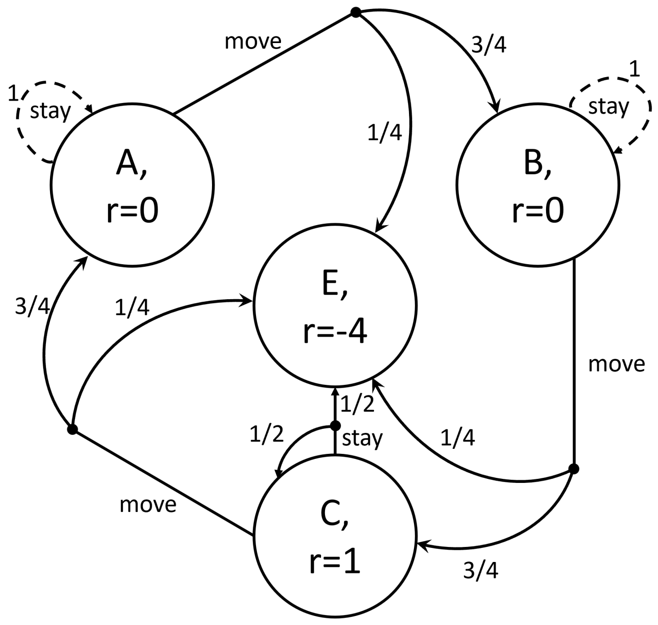
- For the first particle with state (-2,0), we resample (-2,0)
- For the second particle with state (-1,0), we resample (0,0)
- For the third particle with state (-1,0), we resample (-2,0)
- For the fourth particle with state (0,0), we resample (0,-1)

Depending on the way this question setup, different answers are possible. We are only going to look for consistency. 1 point per consistent and correct result. Inconsitencies will be graded as 0.

Several Equations For MDP and RL part (semantics and context are not given on purpose)

$$
\begin{aligned}
V^{\pi}(s) &= \sum_{s'} P(s'|s,\pi(s))(R(s,\pi(s),s') + \gamma V^{\pi}(s')) \\
Q^{\pi}(s,a) &= \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V^{\pi}(s')) \\
V^{\pi}(s) &= \max_a Q^{\pi}(s,a) \\
V^{*}(s) &= \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V^{*}(s')) \\
\pi^{*}(s) &= \arg\max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V^{*}(s')) \\
Q^{*}(s,a) &= \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V^{*}(s')) \\
\pi^{*}(s) &= \arg\max_a Q^{*}(s,a) \\
V_{t+1}(s) &= \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_t(s')) \\
V^{\pi}_{t+1}(s) &= \sum_{s'} P(s'|s,\pi(s))(R(s,\pi(s),s') + \gamma V^{\pi}_t(s')) \\
\pi_{t+1}(s) &= \arg\max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V^{\pi_t}(s')) \\
V(s) &\leftarrow V(s) + \alpha(r + \gamma V(s') - V(s)) \\
Q(s,a) &\leftarrow Q(s,a) + \alpha(r + \gamma \max_{a'}(Q(s',a')) - Q(s,a)) \\
w &\leftarrow w + \alpha(r + \gamma \max_{a'}(Q(s',a')) - Q(s,a))f(s,a)
\end{aligned}
$$

8. (15 points) Answer the questions based on the MDP given below. Read the explanations carefully!



- States: {A, B, C, E}
- Actions: stay and move
- Transition probabilities for the actions are given on the figure for A,B and C.
- **E is a terminal state**
- The rewards are written into the states. They are obtained while **transitioning into** the state! (r(s'))
- Discount Factor: $\gamma = 0.5$

(a) (6 points) You are given a policy that always executes the *move* action. Perform one step of iterative policy evaluation and fill in the table below. Make sure to show your work below the table. Note that the shown rewards are obtained when transitioning into the state. All values start at 0 initially.

Iteration	$V^\pi(A)$	$V^\pi(B)$	$V^\pi(C)$
0	0	0	0
1	-1.0	-0.25	-1.0

Since the reward is dependent only on the next state, our iterative update is:
$V^\pi_{k+1}(s) = \sum_{s \in S} P(s'|s,\pi(s))(R(s') + \gamma V^\pi_k(s'))$
Let's apply this (the value of the terminal state is 0):

$$\begin{aligned}
V^\pi_1(A) &= P(B|A, move)(R(B) + \gamma V^\pi_0(B)) + P(E|A, move)(R(E) + \gamma V^\pi_0(E)) \\
&= 0.75(0 + 0.5 \cdot 0) + 0.25(-4 + 0.5 \cdot 0) = -1.0 \\
V^\pi_1(B) &= P(C|B, move)(R(C) + \gamma V^\pi_0(C)) + P(E|B, move)(R(E) + \gamma V^\pi_0(E)) \\
&= 0.75(1 + 0.5 \cdot 0) + 0.25(-4 + 0.5 \cdot 0) = -0.25 \\
V^\pi_1(C) &= P(A|C, move)(R(A) + \gamma V^\pi_0(A)) + P(E|C, move)(R(E) + \gamma V^\pi_0(E)) \\
&= 0.75(0 + 0.5 \cdot 0) + 0.25(-4 + 0.5 \cdot 0) = -1.0
\end{aligned}$$

(b) (6 points) Perform two steps of value iteration and fill in the table below. Make sure to show your work below the table. Note that the shown rewards are obtained when transitioning into the state. You can re-use your calculations from the previous step. All values start at 0 initially.

Iteration	$V(A)$	$V(B)$	$V(C)$
0	0	0	0
1	0	0	-1.0
2	0	0	-1.0

We follow the same steps as the previous part for both actions and take the maximum one. We can directly look at the calculations of the first part for the *move* action.

Iteration 1:
$V(A) = 0$ with *stay* being the max action. (for *move*, it is -1).
$V(B) = 0$ with *stay* being the max action. (for *move*, it is -0.25).
For C:
stay: $0.5(1 + 0.5 \cdot 0) + 0.5(-4 + 0.5 \cdot 0) = -1.5$, which is smaller than -1 for the move action, thus $V(C) = -1.0$

Iteration 2:
For A and B, nothing changes since they were 0 to start with.
For C:
move: $0.75(0 + 0.5 \cdot 0) + 0.25(-4 + 0.5 \cdot 0) = -1$
stay: $0.5(1 + 0.5 \cdot -1.0) + 0.5(-4 + 0.5 \cdot 0) = -1.75$
Thus $V(C) = -1.0$

(c) (3 points) What is the policy extracted from the values calculated at part(b)? You can use the calculations from previous parts. What can you say about the optimality of this policy?

We look at the argmax actions from part (b). For A and B, this is *stay* and for C, this is *move*, i.e., $\pi(A) = stay$, $\pi(B) = stay$, $\pi(C) = move$. This is the optimal policy since the policy did not change between the iterations.

9. (14 points) You are given an unknown MDP with three states $S = \{A, B\}$, two actions $a = x, y$ and **discount factor** $\gamma = 0.5$. The return of the terminal state is 0. You observe the following episodes (each transition is s, a, r):

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

Answer the questions based on these episodes.

(a) (6 points) Estimate the Q-Values with direct evaluation and fill in the table below. Show your work.

$Q(s,a)$	A	B
x	0.1875	0.25
y	0.875	1.0

Let's start by calculating returns with the given episodes and discount factor.

Episode 1:	Episode 2:
$G_4 = 1.0, \quad (A, x)$	$G_3 = 1.0, \quad (A, y)$
$G_3 = -1 + 0.5 \cdot G_4 = -0.5, \quad (B, y)$	$G_2 = 2 + 0.5 \cdot G_3 = 2.5, \quad (B, y)$
$G_2 = 1 + 0.5 \cdot G_3 = 0.75, \quad (A, y)$	$G_1 = -1 + 0.5 \cdot G_2 = 0.25, \quad (B, x)$
$G_1 = -1 + 0.5 \cdot G_2 = -0.625, \quad (A, x)$	

Let's use these returns to estimate Q-values (1.5 each):

$Q(A, x) = (1 - 0.625)/2 = 0.1875$
$Q(A, y) = (0.75 + 1)/2 = 0.875$
$Q(B, x) = 0.25$
$Q(B, y) = (-0.5 + 2.5)/2 = 1$

First-visit would change $Q(A, x)$.

(b) (2 points) What is the policy implied by the Q-values calculated at part (a)?
We take the argmax action by looking at the highest values along the columns and pick the corresponding row. $\pi(A) = y, \pi(B) = y$

(c) (6 points) Q-Learning: Fill in the table below by applying **Q-learning** updates using the first episode (repeated below), starting from all zeros and taking the learning rate as $\alpha = 0.2$. Fill in the table below with your answers and show your work.

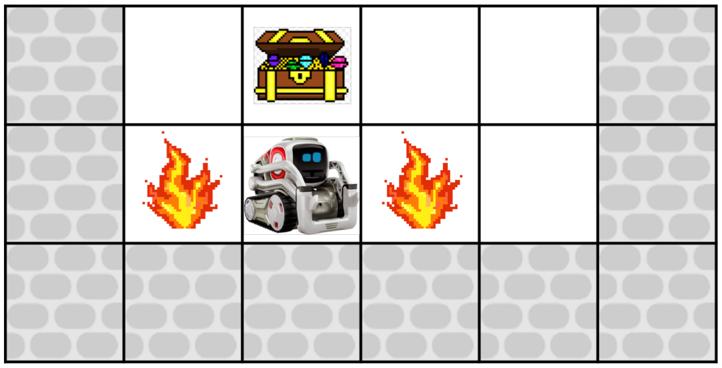
$$(A, x, -1) \rightarrow (A, y, +1) \rightarrow (B, y, -1) \rightarrow (A, x, +1) \rightarrow \textit{terminate}$$

Episode and Transition	$Q(s,a)$	Value after Update
E1T1	$Q(A,x)$	$0 + 0.2(-1 + 0.5 \cdot 0 - 0) = -0.2$
E1T2	$Q(A,y)$	$0 + 0.2(1 + 0.5 \cdot 0 - 0) = 0.2$
E1T3	$Q(B,y)$	$0 + 0.2(-1 + 0.5 \cdot 0.2 - 0) = -0.18$
E1T4	$Q(A,x)$	$-0.2 + 0.2(1 + 0.5 \cdot 0 - (-0.2)) = 0.04$

This was a repetitive question but it was good a exercise to see the difference between MC and TD.

Update rule: $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_t + \gamma \max_{a'}(Q(s_{t+1}, a') - Q(s_t, a_t))$, $Q(s_{t+1}, \cdot) = 0$ if s_{t+1} is terminal.

10. (9 points) We have a game where a robot tries to navigate a maze (represented by a grid) while avoiding fire pits and collecting treasure chests. An instance of a game is given below where the robot has fire pits to its both sides and a treasure chest above it. Robot's actions are $\{left, right, up, down, stop\}$ (self-explanatory). The robot stays in the same location if it tries to move to a wall. If the robot gets into a fire pit, the game ends.



We want to train an agent with RL to play this game. Since the game is big, we want to apply approximate RL. In our MDP formulation, the state includes robot location and the maze information (location of fire pits, treasure chests, walls). We come up with the following features $f_d(s,a)$ and $f_t(s,a)$:

- $f_d(s,a) = A(s) + 2B(s,a)$
- $f_t(s,a) = C(s) + 2D(s,a)$

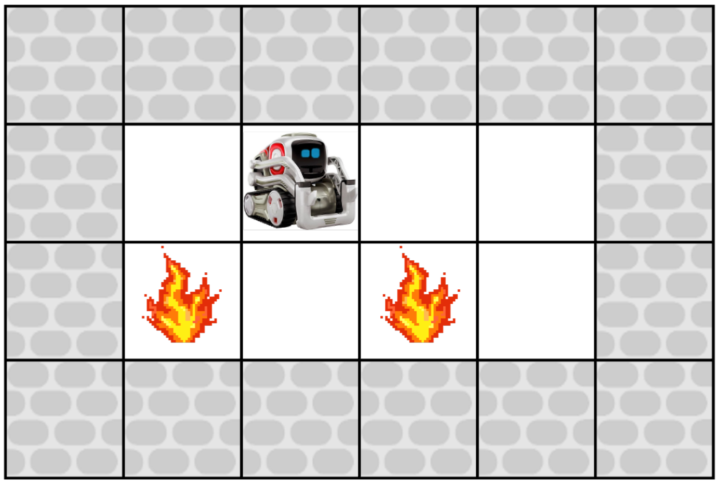
where

- $A(s)$: The number of fire pits within 1 step of state s
- $B(s,a)$: The number of fire pits the robot fall into after taking action a from state s
- $C(s)$: The number of treasure chests within 1 step of state s
- $D(s,a)$: The number of treasure chests taken after taking action a from state s

(a) (3 points) We are approximating the Q-values as $Q(s,a) = w_d f_d(s,a) + w_t f_t(s,a)$. Calculate $Q(s,a)$ for the $\{right, up, stop\}$ actions and the above game state given $w_d = -5$ and $w_t = 10$.
Let's calculate the features first. $A(s) = 2$ and $C(s) = 1$ are the same for all the actions. We calculate B, D for each action, followed by features, followed by the Q-values (0.5 for correct features, 0.5 for correct Q-values per action):

- *right*: $B(s,a) = 1, D(s,a) = 0 \rightarrow f_d(s,a) = 2 + 2 \cdot 1 = 4, f_t(s,a) = 1 + 2 \cdot 0 = 1$
  $\rightarrow Q(s, right) = -5 \cdot 4 + 10 \cdot 1 = -10$
- *up*: $B(s,a) = 0, D(s,a) = 1 \rightarrow f_d(s,a) = 2 + 2 \cdot 0 = 2, f_t(s,a) = 1 + 2 \cdot 1 = 3$
  $\rightarrow Q(s, up) = -5 \cdot 2 + 10 \cdot 3 = 20$
- *stop*: $B(s,a) = 0, D(s,a) = 0 \rightarrow f_d(s,a) = 2 + 2 \cdot 0 = 2, f_t(s,a) = 1 + 2 \cdot 0 = 1$
  $\rightarrow Q(s, stop) = -5 \cdot 2 + 10 \cdot 1 = 0$

(b) (6 points) The robot takes the action *up* from this state and it ends up in the below state, receiving a reward of 50. With discount factor $\gamma = 0.5$ and learning rate $\alpha = 0.2$, update the weights for each feature (starting from $w_d = -5$ and $w_t = 10$). Note that the treasure chest is gone (the robot picked it up!).



Update rule: $w \leftarrow w + \alpha(r + \gamma \max_{a'}(Q(s', a')) - Q(s, a))f(s, a)$.
We need to calculate $\max_{a'}(Q(s', a'))$ but we can see that all actions have 0 features and thus $\max_{a'}(Q(s', a')) = 0$.

$$w_d \leftarrow -5 + 0.2(50 + 0.5 \cdot 0 - 20)2 = 7$$
$$w_t \leftarrow 10 + 0.2(50 + 0.5 \cdot 0 - 20)3 = 28$$

2 points per max-Q for next state, 2 points each for update. If the feature values are forgotten, no partial points are given