Exam 3 Review

CS440/ECE448, Spring 2021

Exam date: Friday, March 5, 1:00pm

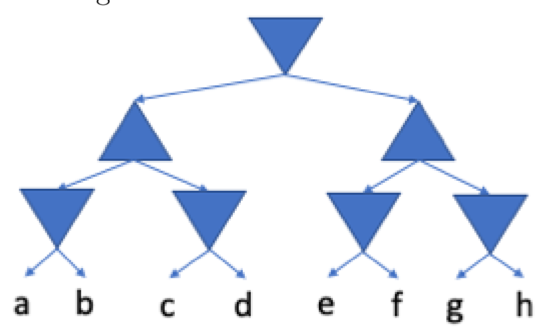
Question 1

What are the main challenges of adversarial search as contrasted with single-agent search? What are some algorithmic similarities and differences?

> **Solution:** The biggest difference is that we are unaware of how the opponent(s) will act. Because of this our search cannot simply consider my own moves, it must also figure out how my opponent will act at each level, thus effectively doubling the number of levels over which I have to search. Since the number of levels is the exponent in the computational complexity, this makes computational complexity much harder.

Question 2

Consider the minimax game tree shown below. Decisions by MAX are represented as upward-pointing triangles; decisions by MIN are represented as downward-pointing triangles; small letters denote outcomes of the game:



The values of each of the outcomes, to the MAX player, are as shown in the following table:

	Outcome							
	a	b	c	d	e	f	g	h
Value to the MAX player:	8	3	1	7	2	5	6	4

(a) What are the values of the two MAX nodes?

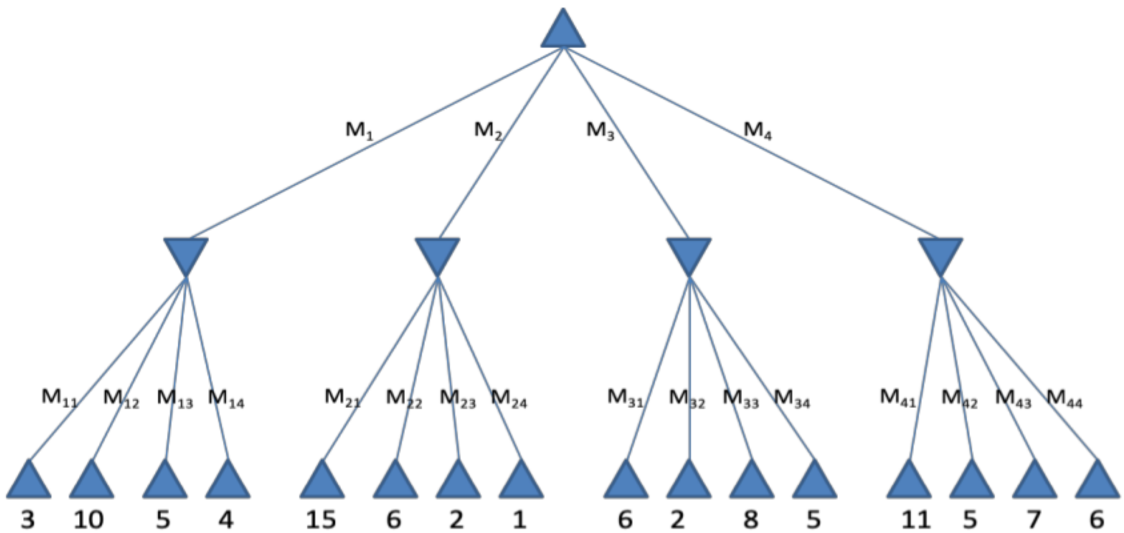
> **Solution:** Value of the max nodes are 3 and 4.

(b) Of the eight outcomes, which one(s) would be pruned by an alpha-beta search?

Solution: Only the node d is pruned.

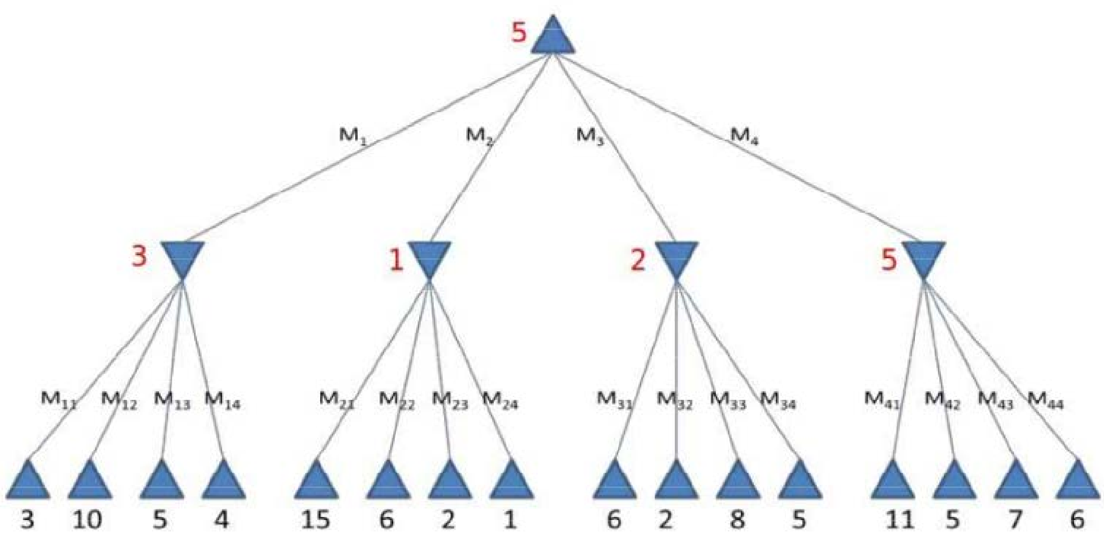
Question 3

Consider the following game tree (MAX moves first):



(a) Write down the minimax value of every non-terminal node next to that node.

Solution:

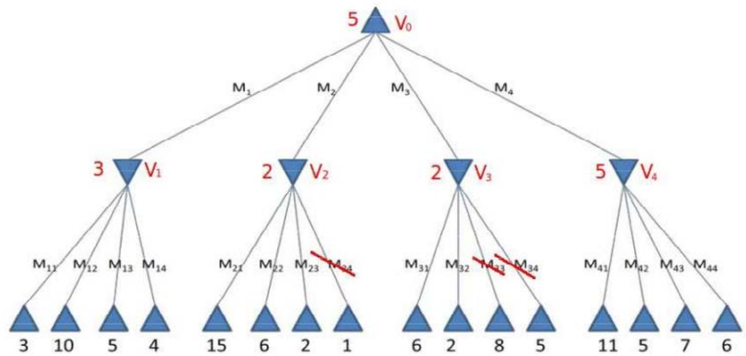


(b) How will the game proceed, assuming both players play optimally?

Solution: The game will choose the max value on depth 1, taking route M_4. It will then take the minimum value on depth 2 that is child of the chosen node, and hence take M_{42}.

(c) Cross out the branches that do not need to be examined by alpha-beta search in order to find the minimax value of the top node, assuming that moves are considered in the non-optimal order shown.

Solution:



(d) Suppose that a heuristic was available that could re-order the moves of both max (M_1, M_2, M_3, M_4) and min $(M_{11}, \ldots, M_{44})$ in order to force the alpha-beta algorithm to prune as many nodes as possible. Which max move would be considered first: M_1, M_2, M_3, or M_4? Which of the min moves $(M_{11}, \ldots, M_{44})$ would have to be considered?

Solution: The first max move to be considered would be V_4, because it allows us to set the highest α. Only 7 of the min moves would be considered: M_{41} through M_{44} would have to be considered to determine that $\alpha = 5$, and then (if the heuristic magically sorts moves in order for us), we would consider M_{32}, M_{24}, and M_{11}, find that all of them have values below α, and prune away their parents.

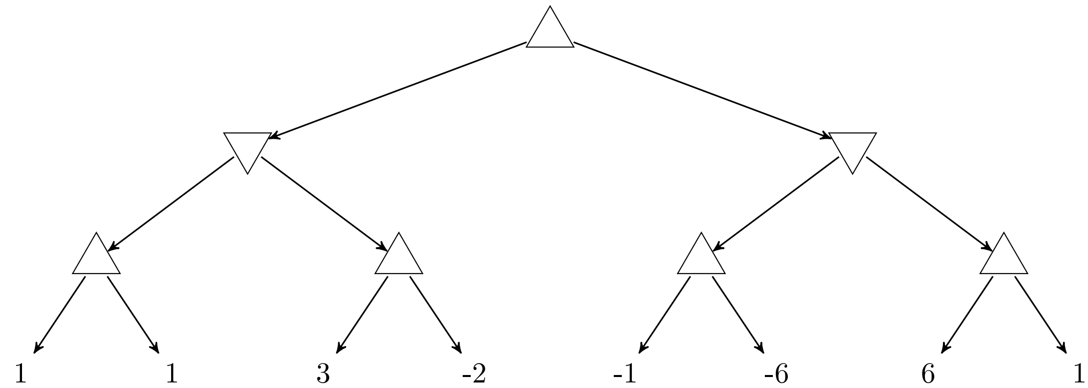
Question 4

How can randomness be incorporated into a game tree? How about partial observability (imperfect information)?

Solution: Randomness is incorporated using the expectiminimax algorithm, in which max tries to maximize the expected score, min tries to minimize the expected score. Partial observability is incorporated using a minimax state tree in which neither player knows for sure which state they're in; the max player chooses an action that maximizes the minimum payoff over all of the states he might be in.

Question 5

Two players, MAX and MIN, are playing a game. The game tree is shown below. Upward-pointing triangles denote decisions by MAX; downward-pointing triangles denote decisions by MIN. Numbers on the terminal nodes show the final score: MAX seeks to maximize the final score, MIN seeks to minimize the final score.



(a) Write the minimax value of each nonterminal node (each upward-pointing or downward-pointing triangle) next to it.

> **Solution:** From top to bottom, left to right, the values are 1, 1, -1, 1, 3, -1, 6.

(b) Suppose that the minimax values of the nodes at each level are computed in order, from left to right. Draw an X through any edge that would be pruned (eliminated from consideration) using alpha-beta pruning.

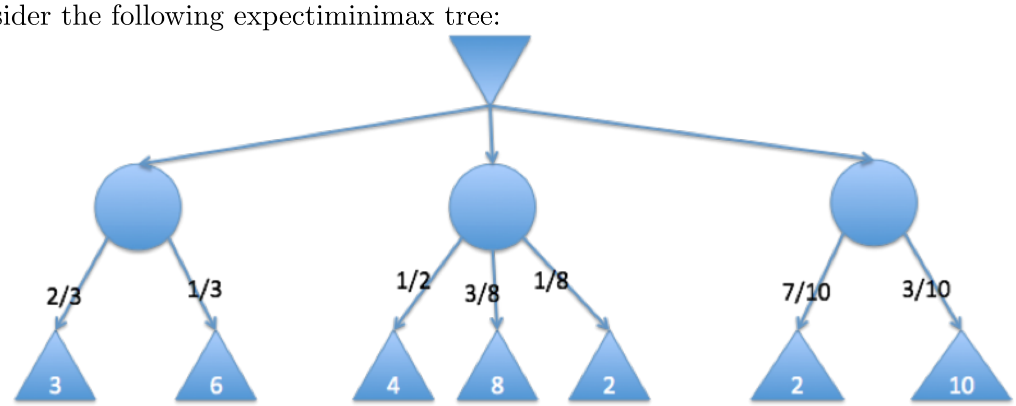
> **Solution:** The 4th edge at the bottom level, and the 4th edge at the middle level, would both be pruned.

(c) In this game, alpha-beta pruning did not change the minimax value of the start node. Is there any deterministic two-player game tree in which alpha-beta pruning changes the minimax value of the start node? Why or why not?

> **Solution:** No. Alpha-beta pruning only prunes branches that have no effect on the start node.

Question 6

Consider the following expectiminimax tree:



Circle nodes are chance nodes, the top node is a min node, and the bottom nodes are max nodes.

(a) For each circle, calculate the node values, as per expectiminimax definition.

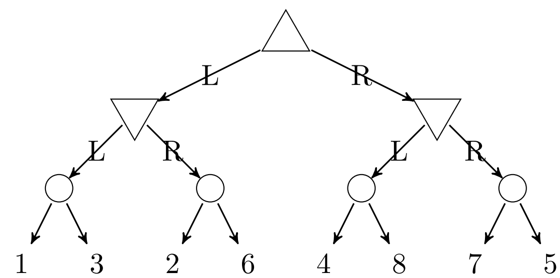
Solution: From left to right: 4, 5.25, 4.4.

(b) Which action should the min player take?

Solution: The first action.

Question 7

Consider a game with eight cards ($c \in \{1, 2, 3, 4, 5, 6, 7, 8\}$), sorted onto the table in four stacks of two cards each. MAX and MIN each know the contents of each stack, but they don't know which card is on top. The game proceeds as follows. First, MAX chooses either the left or the right pair of stacks. Second, MIN chooses either the left or the right stack, within the pair that MAX chose. Finally, the top card is revealed. MAX receives the face value of the card (c), and MIN receives $9 - c$. The resulting expectiminimax tree is as follows:



Assume that the two cards in each stack are equally likely. What is the value of the top MAX node?

Solution: Propagating backward using expectiminimax, we find that the value of the top node is 6.

Question 8

Give an example of a coordination game and an anti-coordination game. For each game, write down its payoff matrix, list dominant strategies and pure strategy Nash equilibria (if any).

Solution: The stag hunt is a coordination game. The payoff matrix is

	Player 1: Cooperate	Player 1: Defect
Player 1: Cooperate	2, 2	0,1
Player 2: Defect	1,0	1,1

The game of Chicken is an anti-coordination game. The payoff matrix is

	Player 1: Chicken	Player 1: Drive
Player 1: Chicken	0, 0	-1,1
Player 2: Drive	1,-1	-10,-10

Question 9

Consider the following game:

	Player A: Action 1	Player A: Action 2
Player B: Action 1	A=3 B=2	A=0 B=0
Player B: Action 2	A=1 B=1	A=2 B=3

(a) Find dominant strategies (if any).

Solution: A dominant strategy is defined as a strategy whose outcome is better for the player regardless of the strategy chosen by the other player. Let's first look for dominant strategies for A: Suppose B chooses Action1. A gets 3 if it chooses Action1 or 0 if it chooses Action2. So it shoould choose Action1. Now suppose B chooses Action2. A gets 1 if it chooses Action1 or 2 if it chooses Action2. So it should choose Action2. Thus there is no dominant strategy for A. Let's look at B: Suppose A chooses Action1. B gets 2 if it chooses Action1 or 1 if it chooses Action2. So it should choose Action1. Now suppose A chooses Action2. B gets 0 if it chooses Action1 or 3 if it chooses Action2. So it should choose Action2. Thus there is also no dominant strategy for B.

(b) Find pure strategy equilibria (if any).

Solution: A Nash Equilibrium is a set of strategies such that no player can get a bigger payoff by switching strageties, provided the other player sticks with the same strategy. There are two: (A: Action1, B: Action1) or (A: Action2, B: Action2).

Question 10

Suppose that both Alice and Bob want to go from one place to another. There are two routes R1 and R2. The utility of a route is inversely proportional to the number of cars on the road. For instance, if both Alice and Bob choose route R1, the utility of R1 for each of them is 1/2.

(a) Write out the payoff matrix.

Solution:

	Alice R1	Alice R2
Bob R1	A:0.5, B:0.5	A:1, B:1
Bob R2	A:1, B:1	A:0.5, B:0.5

(b) Is this a zero-sum game?

> **Solution:** No.

(c) Find dominant strategies, if any. If there are no dominant solution, explain why not.

> **Solution:** There is no dominant solution. The best strategy for each player depends on the strategy of the other player.

(d) Find pure strategy equilibria, if any. If there are no pure strategy equilibria, explain why not.

> **Solution:** There are two: (Alice=R1,Bob=R2) and (Alice=R2,Bob=R1).

(e) Find the mixed strategy equilibrium.

> **Solution:** Alice chooses R1 with probability p, and R2 with probability $1-p$. p must be chosen so that Bob's reward is independent of the action he takes.
>
> - Bob's Reward(R1)= $0.5p + (1-p) = 1 - 0.5p$
> - Bob's Reward(R2)= $p + 0.5(1-p) = 0.5 + 0.5p$
>
> Setting the two rewards equal, we find $p = 0.5$.

Question 11

The "Battle of the Species" game is defined as follows. Imagine a cat and a dog have agreed to meet for the evening, but they forgot whether they were going to meet at a frisbee field or an aquarium. The dog prefers the frisbee field and the cat prefers the aquarium. The payoff for each one's preferred activity is 4 and the payoff for the non-preferred activity is 3 assuming the cat and the dog end up at the same place. If they end up at different places, each gets a 1 if they are at their preferred place, and 0 if they are at their non-preferred place.

(a) Give the normal form (matrix) representation of the game.

> **Solution:**

	Dog: Frisbee	Dog: Aquarium
Cat: Frisbee	C:3,D:4	D:0,C:0
Cat: Aquarium	C:1,D:1	C:4,D:3

(b) Find dominant strategies (if any). Briefly explain your answer.

> **Solution:** None. A dominant strategy is a strategy that maximizes the player's payoff regardless of what the other player does. In this case, if one player chooses frisbee, the other one should choose frisbee, and if one chooses aquarium, the other one should choose aquarium. Therefore, there is no dominant strategy.

(c) Find pure strategy equilibria (if any). Briefly explain your answer.

> **Solution:** (Dog: frisbee; Cat: frisbee); (Dog: aquarium; Cat: aquarium). From either of these two states, no player can get a bigger payoff from changing actions unilaterally.

Question 12

When we apply the Q-learning algorithm to learn the state-action value function, one big problem in practice may be that the state space of the problem is continuous and high-dimensional. Discuss at least two possible methods to address this.

> **Solution:**
>
> 1. Discretize the state space.
> 2. Design a lower-dimensional set of discrete features to represent the states.
> 3. Use a parametric approximator (e.g., a neural network) to estimate the Q function values and learn the parameters instead of directly learning the state-action value functions.

Question 13

What is the optimal policy defined by the Bellman equation?

> **Solution:**
>
> $$\pi^*(s) = \underset{a}{\operatorname{argmax}} \sum_{s'} P(s'|s,a)U(s')$$

Question 14

Simplified GridWorld	Column Number	
	1	2
Row 1	-0.04	-0.04
Row 2	-1	1
Row 3	0	-0.04

Consider a simplified version of GridWorld, shown above. Each position is a state, i.e., $s = (\text{row}, \text{column})$, where row $\in \{1,2,3\}$ and column $\in \{1,2\}$. The grid above shows the reward, $R(s)$, associated with each state. The robot starts in state with $s = (3,1)$; if it reaches either state $s = (2,1)$ or $s = (2,2)$, the game ends.

The transition probabilities are simpler than the ones used in lecture. Let the action variable, a, denote the state to which the robot is trying to move. The robot must choose to try to move to one of its neighboring squares; it cannot choose to remain still, and it cannot choose to aim itself toward a wall. For example, from square $(3,1)$, it can only choose $a \in \{(2,1),(3,2)\}$

If the robot tries to move to any state that is a neighbor of the state it currently occupies, then it either succeeds ($s' = a$ with probability 0.8), or else it remains in the same state ($s' = s$ with probability 0.2). To put the same transition probabilities in the form of an equation, we could write:

$$P(s'|s,a) = \begin{cases} 0.8 & \text{if } s' = a \\ 0.2 & \text{!if } s' = s \end{cases}$$

Use $U^{(t)}(s)$ to denote the estimated utility of state s after t rounds of value iteration. Assume that $U^{(0)}(s) = 0$ and $U_1(s) = R(s)$.

(a) After the second round of value iteration, with discount factor $\gamma = 1$, what are the values $U^{(2)}(s)$ for each of the six states?

> **Solution:** $U^{(2)}(s) = R(s) + \gamma \max_a \sum_{s'} P(s'|s,a)U^{(1)}(s')$, so

Simplified GridWorld	Column Number	
	1	2
Row 1	-0.08	0.752
Row 2	-1	1
Row 3	-0.032	0.752

(b) After how many rounds of value iteration (at what value of t) will $U^{(t)}$(START), the value of the starting state, become positive for the first time?

> **Solution:** $t = 3$: $U^{(3)}(s) = 0.8(0.752)$ is a positive number.

Question 15

In a Markov Decision Process with finite state and action sets, model-based reinforcement learning needs to learn a larger number of trainable parameters than model-free reinforcement learning.

√ **True**

◯ False

Explain:

> **Solution:** Model-based learning needs to learn $P(s'|s,a)$, a set of $N_S^2 N_a$ parameters, where N_s is the number of states, N_a the number of actions. Model-free learning needs to learn $Q(s,a)$, a set of only $N_s N_a$ trainable parameters.

Question 16

After t iterations of the "Value Iteration" algorithm, the estimated utility $U(s)$ is a summation including terms $R(s')$ for the set of states s' that can be reached from state s in at most $t-1$ steps.

√ **True**

◯ False

Explain:

Solution: Value iteration starts with $U(s) = 0$. Each iteration updates $U(s)$ by adding $R(s)$, plus the maximum over all actions of the expected utility $U(s')$ of the state s' that can be reached from state s in one step. In t iterations of this algorithm, one accumulates rewards from states that are up to $t - 1$ steps away.

Question 17

A cat lives in a two-room apartment. It has two possible actions: purr, or walk. It starts in room $s_0 = 1$, where it receives the reward $r_0 = 2$ (petting). It then implements the following sequence of actions: a_0 =walk, a_1 =purr. In response, it observes the following sequence of states and rewards: $s_1 = 2$, $r_1 = 5$ (food), $s_2 = 2$.

(a) The cat starts out with a Q-table whose entries are all $Q(s, a) = 0$, then performs one iteration of TD-learning using each of the two SARS sequences described above (one iteration/time step, for two time steps). Because the cat doesn't like to worry about the distant future, it uses a relatively high learning rate ($\alpha = 0.05$) and a relatively low discount factor ($\gamma = \frac{3}{4}$). Which entries in the Q-table have changed, after this learning, and what are their new values?

Solution:

- $t = 0$:

$$\begin{aligned} Q_{local} &= r_0 + \gamma \max_a Q(s_1, a) = 2 + 0 = 2 \\ Q(1, \text{walk}) &\leftarrow Q(1, \text{walk}) + \alpha(Q_{local} - Q(1, \text{walk})) \\ &= 0 + 0.05(2 - 0) = 0.1 \end{aligned}$$

- $t = 1$:

$$\begin{aligned} Q_{local} &= r_1 + \gamma \max_a Q(s_2, a) = 5 + 0 = 5 \\ Q(2, \text{purr}) &\leftarrow Q(2, \text{purr}) + \alpha(Q_{local} - Q(2, \text{purr})) \\ &= 0 + 0.05(5 - 0) = 0.25 \end{aligned}$$

So the changed values are $Q(1, \text{walk}) \leftarrow 0.1$ and $Q(2, \text{purr}) \leftarrow 0.25$.

(b) Instead of model-free learning, the cat decides to implement model-based learning. It estimates $P(s'|s, a)$ using Laplace smoothing, with a smoothing parameter of $k = 1$, using the two SARS observations listed at the start of this problem. What are the new values of $P(s'|s = 2, a = \text{purr})$ for $s' \in \{1, 2\}$?

Solution:

$$P(s'=1|s=2,a=\text{purr}) = \frac{1+\text{Count}(s_t=2,a_t=\text{purr},s_{t+1}=1)}{2+\sum_{s'}\text{Count}(s_t=2,a_t=\text{purr},s_{t+1}=s')} = \frac{1}{3}$$
$$P(s'=2|s=2,a=\text{purr}) = \frac{1+\text{Count}(s_t=2,a_t=\text{purr},s_{t+1}=2)}{2+\sum_{s'}\text{Count}(s_t=2,a_t=\text{purr},s_{t+1}=s')} = \frac{2}{3}$$

(c) After many rounds of model-based learning, the cat has deduced that $R(1) = 2$, $R(2) = 5$, and $P(s'|s, a)$ has the following table:

a:	purr		walk	
s:	1	2	1	2
$P(s'=1\|s,a)$	2/3	1/3	1/3	2/3
$P(s'=2\|s,a)$	1/3	2/3	2/3	1/3

The cat decides to use policy iteration to find a new optimal policy under this model. It starts with the following policy: $\pi(1) = \text{purr}$, $\pi(2) = \text{walk}$. Now it needs to find the policy-dependent utility, $U^\pi(s)$. Again, because the cat doesn't care about the distant future, it uses a relatively low discount factor ($\gamma = 3/4$). Write two linear equations that can be solved to find the two unknowns $U^\pi(1)$ and $U^\pi(2)$; your equations should have no variables in them other than $U^\pi(1)$ and $U^\pi(2)$.

Solution: The two equations are

$$U^\pi(1) = R(1) + \frac{3}{4}\sum_{s'} P(s'|1,\pi(1))U^\pi(s')$$
$$U^\pi(2) = R(2) + \frac{3}{4}\sum_{s'} P(s'|2,\pi(2))U^\pi(s')$$

Plugging in the given values of all variables, we have

$$U^\pi(1) = 2 + \frac{3}{4}\left(\frac{2}{3}U^\pi(1) + \frac{1}{3}U^\pi(2)\right)$$
$$U^\pi(2) = 5 + \frac{3}{4}\left(\frac{2}{3}U^\pi(1) + \frac{1}{3}U^\pi(2)\right)$$

(d) Since it has some extra time, and excellent python programming skills, the cat decides to implement deep reinforcement learning, using an actor-critic algorithm. Inputs are one-hot encodings of state and action. What are the input and output dimensions of the actor network, and of the critic network?

Solution: The actor network takes a state as input, thus its input dimension is 2 (if the input is a one-hot encoding of two states). It computes the probability that any given action is the best action, so its output dimension is 2 (if there are two possible actions). The critic takes, as input, an encoding of the state (two dimensions), and an encoding of the action (two dimensions, if the action is a one-hot encoding of two possible actions), for a total of 4 input dimensions. It computes, as output, a real-valued score $Q(s,a)$, which is a 1-dimensional (scalar) output.

Question 18

Simplified GridWorld	Column Number	
	1	2
Row 1	-0.04	0
Row 2	-0.04	-1
Row 3	1	-0.04

Consider a simplified version of GridWorld, shown above. Each position is a state, i.e., $s = (\text{row}, \text{column})$, where $\text{row} \in \{1,2,3\}$ and $\text{column} \in \{1,2\}$. The possible actions are the different directions in which the robot can attempt to move, i.e., $a \in \{\text{down}, \text{up}, \text{left}, \text{right}\}$. Assume that the reward for each state, $R(s)$, is known, and is shown in the map above, but that the transition probabilities $P(s'|s,a)$ are not known. The robot starts in state $s = (1,2)$; if it reaches either state $s = (2,2)$ or $s = (3,1)$, the game ends.

Assume that, from any state s, for any action a, the possible outcomes s' are $s' \in \{s, \text{NEIGHBORS}(s)\}$ (the robot might wind up back in the same state, or in one of the neigbors of the same state), but the probabilities of these outcomes are unknown. Note that the cardinality of the set $\text{NEIGHBORS}(s)$ depends on s: some states have 2 neighbors, some have 3.

The robot performs the following action, and observes the following outcome: $(s,a,s') = ((1,2), \text{Left}, (1,1))$. Given this one training observation, use Laplace smoothing, with a smoothing parameter of $k = 1$, to estimate the value of $P(s'|s,a)$ for this particular combination of (s,a,s').

Solution:

$$P(s' = (1,1)|s = (1,2), a = \text{Left})$$
$$= \frac{\#\text{ times }(s = (1,2), a = \text{Left}, s' = (1,1))\text{ observed} + k}{\#\text{ times }(s = (1,2), a = \text{Left})\text{ observed} + k \times \#\text{ distinct values of }s'}$$
$$= \frac{1+1}{1+3}$$

Question 19

Remember that the Actor-Critic algorithm trains two neural nets: an Actor neural net that computes $\pi_a(s) = P(a = \text{best action}|s)$, and a Critic neural net that computes $Q(s,a)$ =expected sum of all current and future rewards if action a is performed in state s. Consider a cat living in a two-room apartment ($s \in \{1,2\}$) with two possible actions ($a \in \{\text{purr}, \text{walk}\}$). Suppose that, after 3000 iterations of Actor-Critic learning, the cat has learned neural nets that generate the outputs shown in the following two tables:

	$\pi_a(s)$		$Q(s,a)$	
a	1	2	1	2
purr	0.95	0.68	0.41	0.04
walk	0.05	0.32	0.58	0.91

Based on these learned models, what are the values, $U(1)$ and $U(2)$, of states 1 and 2? Express your answer as a sum of products of real numbers; do not simplify.

Solution:

$$U(1) = (0.95)(0.41) + (0.05)(0.58)$$
$$U(2) = (0.68)(0.04) + (0.32)(0.91)$$

Question 20

Recall that demographic parity, predictive parity, and balanced error are defined as follows:
Demographic Parity:

$$p(\hat{Y}=1|A=a) = p(\hat{Y}=1|A=a') \ \forall a, a'$$

Predictive Parity:

$$p(Y=1|\hat{Y}=1, A=a) = p(Y=1|\hat{Y}=1, A=a') \ \forall a, a'$$

Balanced Error:

$$p(\hat{Y}=1|Y=1, A=a) = p(\hat{Y}=1|Y=1, A=a') \ \forall a, a'$$

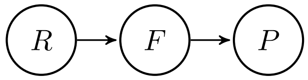
A particular state decides to use AI in order to decide who gets parole from jail. In order to guarantee that their algorithm is fair, they require that the probability that a prisoner is granted parole must be independent of race. Is this an example of demographic parity, predictive parity, or balanced error?

Solution: This is an example of demographic parity.

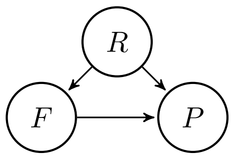
Question 21

The LSI-R is a survey instrument that many precincts used to decide whether or not to grant parole to a prisoner. The survey does not explicitly ask about race, but it asks questions that are causally dependent on race, such as "when was your first encounter with police?"

(a) Draw a Bayesian network representing the causal relationships among the variables R =race, F =first encounter with police, and P =granted parole as they were instantiated in the LSI-R.

Solution: 

(b) An analyst who has taken CS440, and therefore knows something about fairness, proposes that parole decisions should be explicitly based on race. They propose, specifically, that a prisoner's answer to the question "when was your first encounter with police?" should be interpreted in the context of his or her race. Draw a Bayesian network representing the causal relationships among the variables R =race, F =first encounter with police, and P =granted parole as they would be instantiated in this new proposed model.

Solution: 

Question 22

In a pinhole camera, a light source at (x, y, z) is projected onto a pixel at $(x', y', -f)$ through a pinhole at $(0, 0, 0)$. Write $\sqrt{(x')^2 + (y')^2}$ in terms of x, y, z, and f.

Solution: From the idea of similar triangles, we have

$$\frac{x'}{f} = -\frac{x}{z}, \quad \frac{y'}{f} = -\frac{y}{z}$$

from which we derive

$$\sqrt{(x')^2 + (y')^2} = \frac{f}{z}\sqrt{x^2 + y^2}$$

Question 23

Under what circumstances is a difference-of-Gaussians filter more useful for edge detection than a simple pixel difference?

Solution: A difference-of-Gaussians filter first smooths the input image (using a Gaussian smoother), then computes a pixel difference. The smoothing step can reduce random noise. Therefore, this procedure is more useful if the input image has some random noise in it.

Question 24

The real world contains two parallel infinite-length lines, whose equations, in terms of the coordinates (x, y, z), are parameterized as $ax + by + cz = d$ and $ax + by + cz = e$; in addition, both of these lines are on the ground plane, $y = g$, for some constants (a, b, c, d, e, g). Show that the images of these two lines, as imaged by a pinhole camera, converge to a vanishing point, and give the coordinates (x', y') of the vanishing point.

Solution: From the idea of similar triangles, we have

$$\frac{x'}{f} = -\frac{x}{z}, \quad \frac{y'}{f} = -\frac{y}{z}$$

From which we derive

$$x = \frac{-zx'}{f}, \quad y = \frac{-zy'}{f}$$

So the equations of the two lines are

$$-\frac{ax'}{f} - \frac{by'}{f} + c = \frac{d}{z}$$
$$-\frac{ax'}{f} - \frac{by'}{f} + c = \frac{e}{z}$$

As $z \to \infty$, the right-hand-sides of these two equations both go to zero, and the equations of both lines converge to

$$ax' + by' = cf$$

In addition, we have $y = g$, so $y' = -fg/z \to 0$, and therefore $x' = cf/a$. The coordinates are $(x', y') = (cf/a, 0)$.

Question 25

Consider the convolution equation

$$Z(x', y') = \sum_m \sum_n h(m, n) Y(x' - m, y' - n)$$

Where $Y(x', y')$ is the original image, $Z(x', y')$ is the filtered image, and the filter $h(m, n)$ is given by

$$h(m, n) = \begin{cases} \frac{1}{21} & 1 \leq m \leq 3, \quad -3 \leq n \leq 3 \\ -\frac{1}{21} & -3 \leq m \leq -1, \quad -3 \leq n \leq 3 \end{cases}$$

Would this filter be more useful for smoothing, or for edge detection? Why?

Solution: The sum of $h(m, n)$, over all m and n, is 0. So if it is filtering a constant-color region, the output would always be zero, regardless of the input color. So it's not very useful for smoothing.

Any given pixel of $Z(x', y')$ is the difference between the pixels $Y(x', y')$ to its left, minus those to its right. Since it's computing a difference, it would be useful for edge detection.