CSCI 3202
Final Exam
Fall 2020

Write **clearly** and **in the box**:

| |
|---|
| **Name: Sahib Bajwa** |
| **Student ID: 107553096** |
| **Section number: 001** |

- **RIGHT NOW**! Include your name, student ID and section number on the top of your exam. If you're handwriting your exam, write this information at the top of the first page!
- You may use the textbook, your notes, lecture materials, and Piazza as recourses. Piazza posts should not be about exact exam questions, but you may ask for technical clarifications and ask for help on review/past exam questions that might help you. You may not use external sources from the internet or collaborate with your peers.
- You may use a calculator.
- If you print a copy of the exam, clearly mark answers to multiple choice questions in the provided answer box. If you type or hand-write your exam answers, write each problem on their own line, clearly indicating both the problem number and answer letter. Start each new problem on a new page.
- Mark only one answer for multiple choice questions. If you think two answers are correct, mark the answer that **best** answers the question. No justification is required for multiple choice questions. For handwriting multiple choice answers, clearly mark both the number of the problem and your answer for each and every problem.
- For free response questions you must clearly justify all conclusions to receive full credit. A correct answer with no supporting work will receive no credit.
- The Exam is due to Gradescope by midnight on Monday, December 10.
- When submitting your exam to Gradescope, use their submission tool to mark on which pages you answered specific questions.

| Problem | Max Points |
|---|---|
| Short Response | 20 |
| Markov Models | 20 |
| Bayes' Nets | 20 |
| Learning | 20 |
| MDP | 20 |
| Total | 100 |

[2] (1) **[20 points] Short responses**. Provide justification when asked.

1A) **[5 points]** Which of the following five algorithms are guaranteed to find an *optimal* solution to a search problem for the shortest path cost on a given finite graph? **Circle** all that apply.

Breadth-First Search; Depth-First Search Greedy Best-First;

A* with *any* heuristic; [Uniform-Cost Search]

I cant print the exam, my only answer is Uniform-Cost Search (the only one I would circle). None of the others provide an optimal solution to the search problem. A* search is only optimal given a proper heuristic.

1B) **[5 points]** What is the major difference between active and passive learning? Provide an example of each and explain why those examples highlight the difference between the types of learning.

In passive learning, the agent will learn by evaluating the outputs of actions. This is done by using a already determined policy. In active learning, the agent will be able to choose between multiple policies before it makes a decision/action.

An example of passive learning can be seen with an agent that is finding a path that counts the distance from two cities a and b. A passive learning agent would have a predetermined policy and would determine what algorithm to use by evaluating the output of algorithms used to complete the task.

This example shows that in passive learning, the agent will evaluate a policy based on the outputs of the policy used to get from city a to city b.

An example of active learning can also be seen with an agent that is finding a path that counts the distance from two cities a and b. An active learning agent would evaluate what policy it is using at each step or node in the path. If for example the active learning agent sees that we are moving farther way from city b that when we started at city a, it can switch policies since that may be a negative reward.

This example shows that in active learning, the agent has the ability to choose between policies while trying to find the path from city a to city b.

1C) **[5 points]** In our course discussion of the $\varepsilon$-greedy algorithm, we take the original or best action with probability $1 - \varepsilon$ and take a random action drawn from a uniform distribution with probability $\varepsilon$. If we can tune or adjust $\varepsilon$ over the course of a search problem, would we initialize it with a low or high value at the start of training? What about at the end of the training epochs? Answer both prompts and justify your choices.

We would want to initialize $\varepsilon$ with a higher value at the start of training. At the end of training epochs, we would want $\varepsilon$ to be low.

We want $\varepsilon$ to be initialized high so that we can explore a lot at the beginning of the algorithm. Exploring allows us to see possible choices that we can make.

As training epochs go on, $\varepsilon$ needs to get lower. This is because as $\varepsilon$ gets lower, then we are 'closing in' on the correct choices from when we explored earlier.

1D) **[5 points]** True or False, and **Justify**. In $Q$-learning all samples must be from the optimal policy to find optimal (correct) $q$-values.

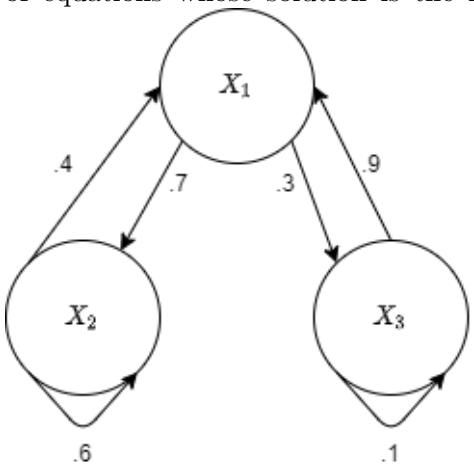
False

We can start with a policy, initial action-utilities for known states, and can still find optimal q-values. We update our state at the end of each sample, and doing this until the episode ends helps us converge to the optimal q-values. We didn't have to use the optimal policy to find the optimal q-values. (Nov 16 lecture pages 21 - 24).

4

2) **[20 points] Markov Models.**

2A) **[7 points]** Consider the following graph which represents a Markov model, with state transition probabilities for $X$ shows along each edge. **Set up** (you don't have to solve) a system of equations whose solution is the long-run distribution or *stationary distribution* for $X$:




$$\pi(x') = \sum_x q(x'|x)\pi(x)$$

2A) $X_1$ $X_2$ $X_3$

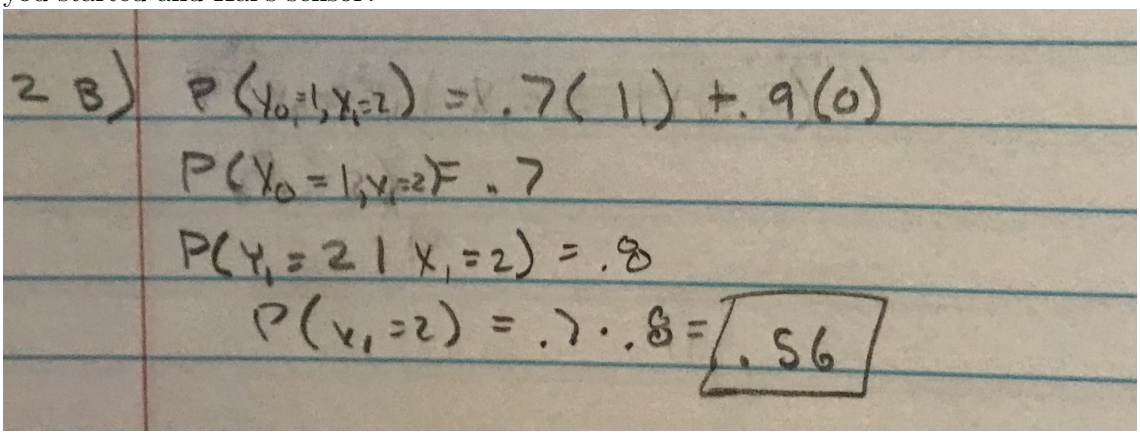
$$\begin{bmatrix} \pi(x_1) \\ \pi(x_2) \\ \pi(x_3) \end{bmatrix} = \begin{bmatrix} q(x_1|x_2)\pi(x_2) + q(x_1|x_3)\pi(x_3) \\ q(x_2|x_1)\pi(x_1) + q(x_2|x_2)\pi(x_2) \\ q(x_3|x_1)\pi(x_1) + q(x_3|x_3)\pi(x_3) \end{bmatrix}$$

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}' = \begin{bmatrix} .4(x_2) + .9(x_3) \\ .7(x_1) + .6(x_2) \\ .3(x_1) + .1(x_3) \end{bmatrix}$$

**Parts 2B and 2C refer to the following**: With the same 3 states for $X$ as above, suppose you're designing a smart house lighting AI that you've cleverly named Housing Automated Lighting (HAL). HAL's sole purpose is to turn the lights on in whichever room you're in. Your house has the same 3 rooms/states $X$ as above, and the graph above represents your probability of moving from one room to the next each hour of the day. At the *start* of the day, $(t = 0)$ you always begin in your bedroom marked $X_1$ above, which we'll represent as the event $X_0 = 1$.

So that HAL may have enough information to track which room you're in, it's equipped with a sensor $Y$, which is a little noisy. When you're actually in room $i$, it senses that $Y = i$ with probability 80%, while it incorrectly diagnoses that you're in each one of the other rooms 10% of the time. In other words, at time $j$, the sensed room will be $P(Y_j = i|X_j = i) = 0.8$ for the room you're in and $P(Y_j = i|X_j \neq i) = 0.1$ for each other room.

2B) **[7 points]** You wake up as usual at time $t = 0$ and stumble out of bed to begin your day. It's now time $t = 1$. HAL *senses* you in the kitchen (room 2), or $Y_1 = 2$. What is the probability you're actually in the kitchen at time $t = 1$? In other words, what is $P(X_1 = 2)$ *given* where you started and Hal's sensor?

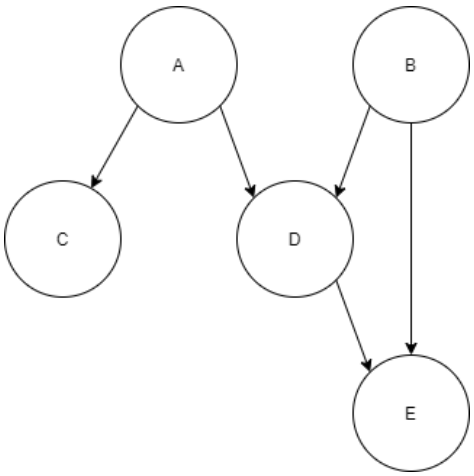



$P(X_1 = 2) = .56$ given we started at X1 and Hal sensed X2 at t = 1

2C) **[6 points]** After sensing that $Y_1 = 2$, HAL then *again* senses you in the kitchen, so $Y_2 = 2$. Should this increase or decrease the conditional probability that $P(X_1 = 2)$? You may attempt to answer with a fully-explained intuitive answer or perform all exact calculations if you wish.

This should increase the conditional probability that $P(X_1 = 2)$. There is a 60% chance that you stay in the kitchen given that you are already in the kitchen, meaning that if HAL was correct for t = 1, there is a greater chance that you stayed in the kitchen than left for t = 2. If HAL was incorrect about you being in the kitchen for t = 1, that would mean that you moved to X3, which not only has a lower chance of you moving there, but HAL would be wrong with an accuracy of 20%. Since there is no way to get from X3 to X2, this would mean that HAL was wrong again for t = 2 with an accuracy of 20%. I think this makes it pretty clear that since HAL sensing you in the kitchen for t = 2 (with an accuracy of 80%), this increases the conditional probability that $P(X_1 = 2)$.

3) **[20 points]** Consider the following Bayesian Network, where all variable nodes may only take on True/False as values.




$P(A = True) = 0.3$
$P(B = True) = 0.6$

CPT for C:

| $A$ | $P(C = True)$ |
|---|---|
| True | 0.6 |
| False | 0.3 |

CPT for D:

| $A$ | $B$ | $P(D = True)$ |
|---|---|---|
| True | True | 0.4 |
| True | False | 0.5 |
| False | True | 0.1 |
| False | False | 0.2 |

CPT for E:

| $B$ | $D$ | $P(E = True)$ |
|---|---|---|
| True | True | 0.2 |
| True | False | 0.3 |
| False | True | 0.5 |
| False | False | 0.9 |

**Show all work** for the following queries:

3A) **[7 points]** What is the probability that all five variables are simultaneously true?

$P(A = T, B = T, C = T, D = T, E = T) = P(A = T) * P(B = T) * P(C = T|A = T) * P(D = T|A = T, B = T) * P(E = T|, B = T, D = T)$
$= 0.3 * 0.6 * 0.6 * 0.4 * 0.2$
$= 0.00864$

3B) **[7 points]** What is the probability that $A$ is false *given* that all four other variables are true?

$P(A = F)$ given everything else is T is $= P(A = F)$ since A is not dependant on other nodes. Thus, $P(A = F)$given everything else is T is $= 0.7$

3C) **[6 points]** What is the probability that $C$ is true *given* that $D$ is true?

$P(C = T)$ given (D = T) $= P(C = T|A = T) + P(C = T|A = F)$ since C is only dependant on A, which is not dependant on anything.
Thus, $P(C = T)$ given (D = T) $= P(C = T|A = T) + P(C = T|A = F)$
$= 0.6 + 0.3$
$= 0.9$

4) **[20 points]** Suppose an agent exists on state space with three states, $X$, $Y$, and $Z$, which each states hold some features about the kittens and puppies currently playing with our agent. Within each state, the agent has two actions: pet the dog (denoted "dog") and pet the cat ("cat"). The agent chooses actions according to policy $\pi$, but does not know the underlying process that dictates state transitions. So it sets up an experiment, wherein it:

- Starts at a *random* state $s$, and chooses an action $a$.
- Observes the successor state $s'$ of that action and the rewards $r$ resulting from that transition.
- Updates $Q$-values for that state-action pair.

Suppose that the $Q-$values are all initialized to 0, the learning rate is fixed as $\alpha = \frac{1}{3}$, and there is no discount factor ($\gamma = 1$). The first 6 training episodes are shown at left below.

4A) **[12 points]** Run $Q-$learning updates on the table at left, updating the desired quantities to the right in the order of the training episodes:

| $s$ | $a$ | $s'$ | $r$ |
|---|---|---|---|
| X | "cat" | Y | 3 |
| Z | "dog" | Y | 3 |
| Y | "cat" | Z | -3 |
| X | "cat" | Y | 6 |
| Y | "dog" | X | 0 |
| X | "cat" | Z | 3 |

$Q_1(X, cat) = 0+(1/3)[3+1max(0)-0] = 1$

$Q_1(Z, dog) = 0+(1/3)[3+1max(0)-0] = 1$

$Q_1(Y, cat) = 0+(1/3)[-3+1max(0)-0] = -1$

$Q_2(X, cat) = 1+(1/3)[6+1max(Q1(Y,"cat"))-1] = 2.33$

$Q_1(Y, dog) = 0+(1/3)[0+1max(0)-0] = 0$

$Q_3(X, cat) = 1.67+(1/3)[3+1max(Q2(Z,"dog"))-1.67] = 2.4467$

4B) **[4 points]** After your training episodes, our agent constructs a policy $\pi$ that maximizes the estimated utility in a given state. What are the actions chosen by the agent in states $X$ and $Y$?
The actions chosen by the agent in state X is: "cat"
The action chosen by the agent in state Y is: "dog"

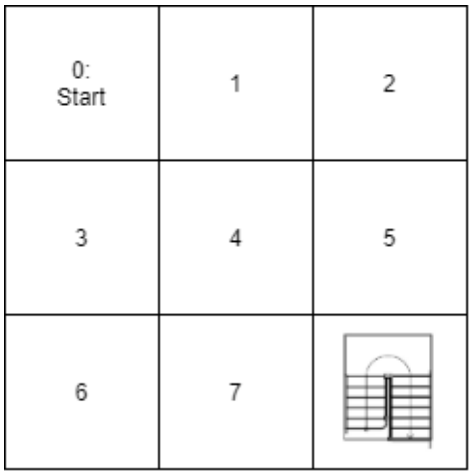
4C) **[4 points]** Suppose our agent decides to *estimate* the underlying state transitions from its empirical results in 4A, as it would in *adaptive dynamic* reinforcement learning. Without using any augmented counting such as Laplace smoothing, what would you estimate from the training episodes for:

$$P(s' = Y | s = X, a = \text{"}cat\text{"}) = \underline{Q2(x, cat) - Q1(x, cat) = 1.33. + 1.33 increase}$$

$$P(s' = Z | s = X, a = \text{“}cat\text{”}) = \underline{Q3(x, cat) - Q2(x, cat) = 0.1167. + 0.1167 increase}$$



5) **[20 points] MDP**. Consider the MDP at left below, where an agent starts in a dangerous dungeon. The agent has the standard actions of movement NSEW to non-wall locations. There is a reward of 1 for escaping the floor at the staircase, which represents a terminal state (tile that would be indexed 8). Suppose that the discount factor is $\gamma = 1$ (so no discounting) and there is no reward associated with any state other than the stairs.



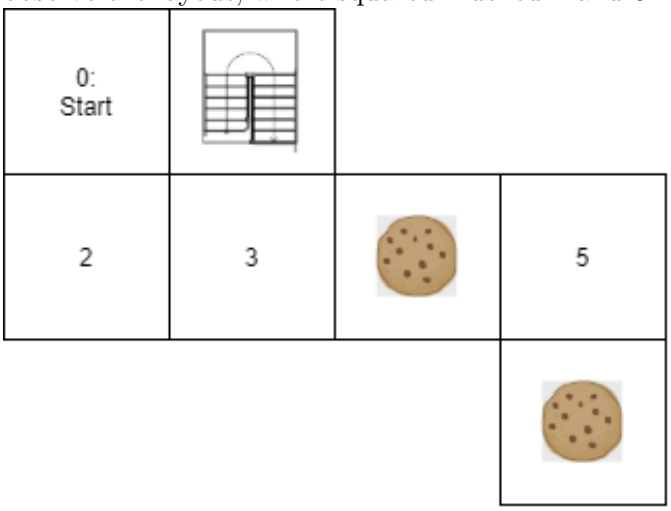

5a) **[8 points]** Complete the following table where each row represents a step of *value iteration*:

| $k$ | $U(0)$ | $U(1)$ | $U(2)$ | $U(3)$ | $U(4)$ | $U(5)$ | $U(6)$ | $U(7)$ | $U(8)$ |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | .2 | .4 | .6 | .4 | .6 | .8 | .6 | .8 | 1 |
| 2 | 0 | .4 | .6 | .4 | .6 | .8 | .6 | .8 | 1 |
| 3 | 0 | 0 | .6 | 0 | .6 | .8 | .6 | .8 | 1 |
| 4 | 0 | 0 | 0 | 0 | 0 | .8 | 0 | .8 | 1 |
| 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

5b) **[4 points]** Value iteration should have *converged* in the steps above, such that $U_k(s) = U_{k+1}(s)$ for all states. At which step did it do so?

This happened at step 4 (or step 5 if you consider step 1 to be step 0). This happens after 4 total moves, when we get to the stairs.

5c) **[8 points]** Suppose we reach the next level, and now the dungeon floors also are shockingly housing delicious cookies that provide a *one-time* reward of 10 each! You enter the level and observe the layout, where squared indexed 4 and 6 hold cookies:




List *all* elements of the state space and the optimal policies for each assuming that each state now has a (punishment) reward of $-0.05$ for remaining in the level. The discount factor is still 1. You may assume that the agent can never occupy a space with an uneaten cookie, but you should include states that could only be reached via the agent jumping over or teleporting past a cookie. The first state-optimal policy pair is provided below.

| Location(agent) | State(cookie #4) | State(cookie #6) | Action |
|---|---|---|---|
| 0 | eaten | eaten | East |
| 0 | eaten | not eaten | South |
| 0 | not eaten | not eaten | South |
| 2 | eaten | eaten | North |
| 2 | eaten | not eaten | East |
| 2 | not eaten | not eaten | East |
| 3 | eaten | eaten | North |
| 3 | eaten | not eaten | East |
| 3 | not eaten | not eaten | East |
| 5 | eaten | not eaten | South |
| 5 | eaten | eaten | West |

I am assuming that you cannot skip past the first cookie without eating it, thus we cannot have a situation where cookie 4 is uneaten and cookie 6 is eaten.
You do not need to justify how you arrived at each policy. **Have a great break!**