COMP3211 Fundamentals of AI **Name:** _____
Fall 2020 Final
10/12/2020 **Stu ID:** _____
Time Limit: 170 Minutes

Instructions:

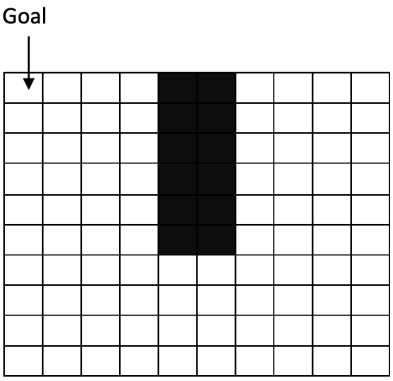
1. This exam contains 11 pages (including this cover page) and 9 questions.
2. Upload your answer to canvas as a pdf file (preferred) or a zip archive of image files (problem1.jpeg, problem2.jpeg,...) named by your student ID.
3. Observe the honor code - it's open book but no discussions or outside help are allowed.
4. Sign on the class zoom and turn on the video but mute the audio.
5. The exam starts at 12:30pm and ends at **3:20pm**. You will then have 10 minutes to upload your answers to canvas.

Grade Table (for teacher use only)

Question	Points	Score
Simple Agents	8	
Perceptron Learning and GSCA Rule Learning	12	
MDP	12	
Uncertainty	10	
Propositional Logic	10	
FOL	12	
Search	10	
Games and RL	12	
Concepts	14	
Total:	100	

Question 1 [8 points]

Consider the robot with the same specification as our boundary-following robot discussed in class: eight sensors $s_1, ..., s_8$ and four actions ($North$, $South$, $East$, and $West$). Now suppose the environment is a 10x10 grid with an obstacle inside as shown below:



Suppose we want the robot to go to the **top left** corner, wherever its initial position is. We know that this cannot be done by a reactive (stimulus-response) agent (one of our midterm question).

1.1 We can let the robot remember the past k actions and sensor readings. What $k \leq 8$ will be sufficient for it to achieve this goal, wherever its initial starting position is? By achieving the goal we mean that the robot will eventually stop at this position. It is okay for it to visit this position more than once before it stops (*nil* action). Please briefly justify your answer informally. Of course, you can also justify it formally by giving, for example, a production system for achieving this goal, but this is not necessary. Notice that you do not need to come up with a minimal k. But you will not get any point if you just say, for example, $k = 8$ without any satisfactory explanation.

1.2 If you can add more sensors to the robot, what other sensors we can think of to make this goal achievable by a reactive agent? Again briefly justify your answer.

Solution:

- $k = 6$ will work: once the state is $s_1 = s_2 = s_3 = s_7 = s_8 = 1$, it can head south 6 times, and if $s_8 = 0$ at the end, then it knows it was the wrong position and can move east.
- One example is with a sonar: if the longest distance to the wall is more than 9, then it's the goal position.

Other reasonable answers can be accepted.

Question 2 [12 points]

Consider the following data set:

ID	x_1	x_2	x_3	x_4	OK
1	1	0	1	0	Yes
2	0	0	1	0	Yes
3	1	0	0	0	No
4	1	1	0	1	No
5	0	1	1	1	No

where x_1, x_2, x_3 and x_4 are some features that should not concern us here.

2.1 Use these five instances to train a single perceptron using the error-correction procedure. Use the learning rate = 1, and the initial weights all equal to 0. Recall that the threshold is considered to be a new input that always have value "1". Please give your answer by filling in the following table, where weight vector (w_1, w_2, w_3, w_4, t) means that w_i is the weight of input x_i, and t is the weight for the new input corresponding to the threshold. Stop when the weight vector converges. If it doesn't converge, explain why not.

ID	Weight vector	Weighted Sum	Actual	Desired
Initial	0, 0, 0, 0, 0			
1				
2				
3				
4				
5				
...	...	...	...	...

2.2 What is the Boolean function corresponding to your perceptron?

2.3 From the same training set, apply the GSCA algorithm to try to learn a set of rules. Give the set of rules if it succeeds. If it fails to learn a set of rules, explain why it failed.

Solution:

1.

ID	Extended input	Weight vector	Sum	Actual	Desired
Initial		0, 0, 0, 0, 0			
1	1, 0, 1, 0, 1	0, 0, 0, 0, 0	0	1	1
2	0, 0, 1, 0, 1	0, 0, 0, 0, 0	0	1	1
3	1, 0, 0, 0, 1	-1, 0, 0, 0, -1	0	1	0
4	1, 1, 0, 1, 1	-1, 0, 0, 0, -1	-2	0	0
5	0, 1, 1, 1, 1	-1, 0, 0, 0, -1	-1	0	0
1		0, 0, 1, 0, 0	-2	0	1
2		0, 0, 1, 0, 0	1	1	1
3		-1, 0, 1, 0, -1	0	1	0
4		-1, 0, 1, 0, -1	-2	0	0
5		-1, -1, 0, -1, -2	0	1	0
1		0, -1, 1, -1, -1	-3	0	1
2		0, -1, 1, -1, -1	0	1	1
3		0, -1, 1, -1, -1	-1	0	0
4		0, -1, 1, -1, -1	-3	0	0
5		0, -1, 1, -1, -1	-2	0	0
1		0, -1, 1, -1, -1	0	1	1

It converges to give the weight vector $(0, -1, 1, -1)$ with the threshold 1.

2. Boolean function: $\neg x_2 x_3 \neg x_4$, i.e. $\overline{x_2} x_3 \overline{x_4}$.

3.
We begin with: $true \supset OK$.
Choose the feature that yields the largest value of r_α:
$r_{x_1} = 1/3, r_{x_2} = 0, r_{x_3} = 2/3, r_{x_4} = 0$
. So we choose x_3, this will generate : $x_3 \supset OK$
This rule still covers the negative instance ID5, so we still need to narrow it. This time we have $r_{x_1} = 0.5, r_{x_2} = 0, r_{x_4} = 0$. We can add $\{x_1\}$: $x_1 \wedge x_3 \subset OK$. This rule covers only positive instances, so we have learned one rule. To learn the next one, we eliminate data with ID1 and continue like above.
We begin with: $true \supset OK$.
Choose the feature that yields the largest value of r_α:
$r_{x_1} = 0, r_{x_2} = 0, r_{x_3} = 0.5, r_{x_4} = 0$.
So we choose x_3, this will generate : $x_3 \supset OK$
This rule still covers the negative instance ID5, so we still need to narrow it. This time we have $r_{x_1} = 0$, and x_2=0 and x_4=0. So we add either of x_1, x_2, x_4, but there are no positive examples covered by the new generated rule. So we track back to $x_3 \supset OK$ and replace it by $x_1 \subset OK$ or $x_2 \subset OK$ or $x_4 \subset OK$. All of them cover no positive examples. So it backtrack to the beginning and abort (or loop for ever). So GSCA only learns one rule, for the reason that this training set cannot be learned without making use of negative literals like $\neg x_2$ or $\neg x_4$ in the rules.

Question 1, 2 points for each entry.

Question 2, your Boolean function should correspond to your perceptron, not necessarily the original data.

Question 3, you can get at most 4 points if you directly use $x_1 \wedge x_3 \supset OK$ without claiming GSCA fails.

Question 3 [12 points]

Imagine you are playing the following dice game. You can either quit, which will end the game and reward you \$10, or stay, which will reward you \$5 immediately and then a fair coin will be tossed. If it ends up with the head side up, then the game continues. If it ends up with the tail side up, then the coin will be tossed again. This time if it ends up with the head side up, then you pay \$9 (meaning your net loss is -\$4 in this case), and the game continues; but if it ends up with the tail side up, then the game ends. Answer the following questions by assuming the discount factor of 1.

3.1 Formulate this problem as a MDP.

3.2 Compute the optimal plan of your MDP.

Solution: MDP:

- States: $\{In, Out\}$.
- Starting state: In.
- End states: Out.
- Actions: $\{stay, quit\}$.
- $T(In, quit, Out) = 1$. $T(In, stay, Out) = .25$. $T(In, stay, In) = .75$.
- $R(In, quit, Out) = 10$. $R(In, stay, Out) = 5$, $R(In, stay, In) = [5 * 0.5 + (5 - 9) * 0.25]/.75 = 2$. *(Give partial mark for* $R(In, stay, In) = [5*0.5+(5-9)*0.25]$ *which will make quit better)*
- $\gamma = 1$.

There are only two policies: stay or quit in In.

$\pi_1 = stay, \pi_2 = quit$
$V(out) = 0$

$$\begin{aligned} V_{\pi_1}(in) &= T(in, stay, in) * [Reward(in, stay, in) + V(in)] \\ &\quad + T(in, stay, out) * [Reward(in, stay, out) + V(out)] \\ &= 0.75 * [2 + V_{\pi_1}(in)] + 0.25 * (5 + 0) \end{aligned}$$

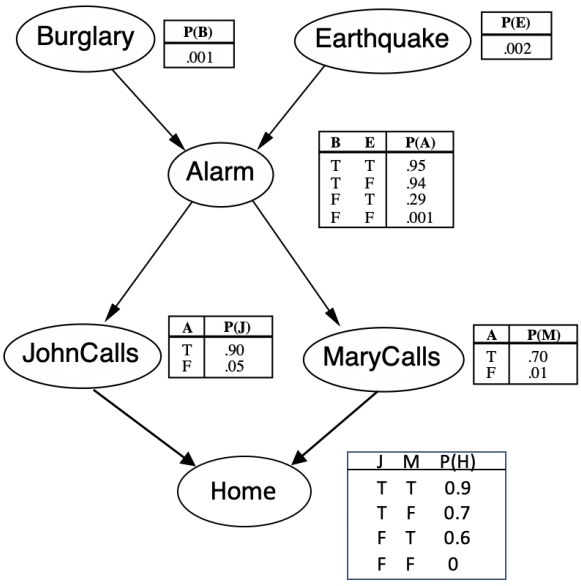
so, $V_{\pi_1}(in) = 11$

$$\begin{aligned} V_{\pi_2}(in) &= T(in, quit, out) * [Reward(in, quit, out) + V(out)] \\ &= 1 * (10 + 0) = 10 \end{aligned}$$

so, $V_{\pi_2}(in) = 10$
The optimal policy is to stay at every turn util being forced to end game *(Give full mark for policy if they compute the values correctly using the wrong rewards and/or transition probabilities)*

Question 4 [10 points]

Consider the following Bayesian network which adds one more node, *Home* (whether to go back home), to Pearl's example:



4.1 Are J (JohnCalls) and M (MaryCalls) independent given A (Alarm)? Explain your answer using D-separation.

4.2 Compute the probability of H (Home) being true given that A is true: $P(H|A)$. There is no need to perform numerical calculations. As long as your formula is right, you will get the full mark.

Solution: Yes, J and M are independent given A.
Two path from J to M, (1)J-A-M, (2)J-H-M
for (1) A is in E and is not the case that both path arrows lead in to A
for (2) H is not in E, both path arrows lead in to H, and neither H nor any descendants of H are in E

$$
\begin{aligned}
P(H=1|A=1) &= \Sigma_{J,M} P(H=1,J,M|A=1) \\
&= \Sigma_{J,M} P(H=1|J,M,A=1)P(J|A=1)P(M|A=1) \\
&= P(H=1|J=1,M=1,A=1)P(J=1|A=1)P(M=1|A=1) \\
&+ P(H=1|J=1,M=0,A=1)P(J=1|A=1)P(M=0|A=1) \\
&+ P(H=1|J=0,M=1,A=1)P(J=0|A=1)P(M=1|A=1) \\
&+ P(H=1|J=0,M=0,A=1)P(J=0|A=1)P(M=0|A=1)
\end{aligned}
$$

Question 5 [10 points]

A_1, A_2 and A_3 are three friends. We know the following facts about them:

- At least one of them is in the car.
- A_3 cannot drive.
- A_1 is in the car only if A_2 is in the car.

Use propositional logic to show that A_2 is in the car:

5.1 Encode the given facts as well as any necessary common sense knowledge as a KB using the following symbols:

- P_i: A_i is in the car, $i = 1, 2, 3$.
- D_i: A_i is the driver, $i = 1, 2, 3$.

5.2 Convert your KB to a set of clauses.

5.3 Use resolution (and proof by refutation) to show that your KB entails P_2.

Solution:

$$
\begin{aligned}
&P_1 \vee P_2 \vee P_3, (\text{not necessary})\\
&D_1 \vee D_2 \vee D_3,\\
&D_1 \supset P_1,\\
&D_2 \supset P_2,\\
&D_3 \supset P_3,\\
&\neg D_3,\\
&P_1 \supset P_2
\end{aligned}
$$

Clauses:

$$
\begin{aligned}
&P_1 \vee P_2 \vee P_3,\\
&D_1 \vee D_2 \vee D_3,\\
&\neg D_1 \vee P_1,\\
&\neg D_2 \vee P_2,\\
&\neg D_3 \vee P_3,\\
&\neg D_3,\\
&\neg P_1 \vee P_2
\end{aligned}
$$

Add negation of query $\neg P_2$.

Question 6 [12 points]

We know that Carol has no sisters, i.e. she is the only daughter of her parent. One

day, someone saw Carol talking to a man and asked Carol whom she was talking to. Carol said that she was talking to a man whose mother is a daughter of my father. Use first-order logic to figure out the relationship between Carol and the man whom she was talking to.

6.1 Encode the known facts, including what Carol said, as a KB using the following constants and predicates:

- C - constant denoting Carol.
- M - constant denoting the man whom Carol was talking to.
- $daughter(x, y)$ - x is y's daughter.
- $mother(x, y)$ - x is y's mother.

Important You are not allowed to introduce any other constants, functions or predicates.

6.2 What are the possible queries to your KB to find out the answer?

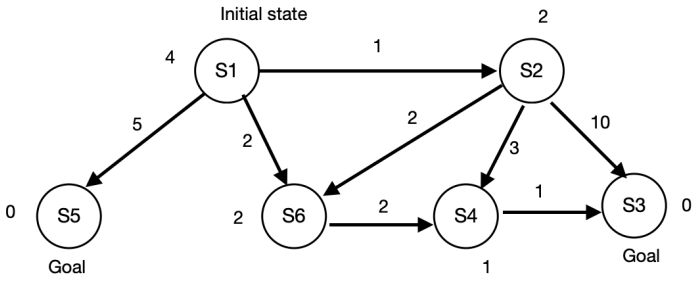
Solution:

$$\forall x, y.daughter(C, y) \wedge daughter(x, y) \supset x = C,$$
$$\exists x, y.mother(x, M) \wedge daughter(x, y) \wedge daughter(C, y)$$

Possible queries $mother(C, M)$ or $daughter(C, M)$.

Question 7 [10 points]

Consider again the search problem in the midterm:



Recall that the numbers on the edges are costs of the corresponding actions, and the numbers next to the states are their heuristic values. Answer the following questions:

7.1 Can you come up with a new *admissible* heuristic function so that A* algorithm will return the solution $S1 \rightarrow S6 \rightarrow S4 \rightarrow S3$ without using any tie-breaking rule? If you can, provide such a heuristic function. If not, explain why not.

7.2 Can you come up with a new *admissible* heuristic function so that A* will expand only nodes that are on a path of an optimal solution regardless of any tie-breaking rule? More precisely, this means that using your heuristic function, if a node is selected from the OPEN list for expansion, then there must be an optimal plan that goes through this node. If you can, provide such a heuristic function. If not, explain why not.

Solution:

1. No. There are multiple optimal solutions. Any of them can be returned depending on the tie-breaking rule.
2. Yes. Just let the heuristic function h to be the true cost: $h(S) = g^*(S)$ where $g^*(S)$ is the cost of an optimal solution starting in S. So $h(S1) = 5, h(S2) = 4, h(S3) = 0, h(S4) = 1, h(S5) = 0, h(S6) = 3$.

Question 8 [12 points]

Consider the Tic-Tac-Toe game. Describe how you can train player X to play the game using reinforcement learning:

- Describe the states;
- Describe the actions that X can play in the states;
- Describe the starting state;
- Describe the end states;
- Describe what the transition probability function means, and whether it needs to be learned;
- Describe what the reward function means, and whether it needs to be learned

There are articles on the internet about using reinforcement learning to play the game. You can read as many of them as you want. In the end, if your answer uses some ideas from them, you have to cite them.

Solution: One possible way:

- States: all positions with equal number of X's and O's on the board.
- Actions: X marks an un-occupied cell.
- Starting state: the empty board position.
- End states: end of the game.

- T-function: $T(s, a, s')$ means that after X makes the move a in s, the probability that O's response will lead to the new state s'. This cannot be known ahead of time so needs to be learned. What will be learned will depend on the opponent you used to train your strategy for X.
- Rewards: One way to define the reward would be to make it 0 if the transition does not lead to an end state, and X's utility if it does. So $R(s, a, s') = 0$ if s' is not an end state. In this case, this does not need to be learned.

Question 9 [14 points]

In your own words, answer the following questions:

9.1 What is supervised machine learning?

9.2 What is unsupervised machine learning? Is it the same as clustering?

9.3 What is a linear predictor?

9.4 What is reinforcement learning?

9.5 What is the gradient descent algorithm?

9.6 What is genetic programming?

9.7 Is AI and same as machine learning? If yes, why? If not, list some differences.