MSBD5015 Artificial Intelligence
Fall 2022 Final
1500 on 10/12/2022
Time Limit: 3 hours

Name: ____________________

Stu ID: ____________________

Instructions:

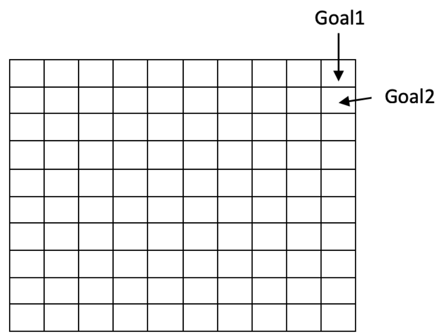
1. This exam contains 12 pages (including this cover page) and 12 questions.
2. This is a closed book exam.
3. Observe the honor code. Write the exam on your own.

Grade Table (for teacher use only)

Question	Points	Score
Reactive Agents and State Machines	10	
Genetic Programming	5	
Search Problem Formulation	5	
MDP	5	
Heuristic Search	10	
Alpha-Beta Pruning	5	
FOL Representation	10	
Propositional Logic	10	
Game theory and auction	10	
Uncertainty	10	
Perceptron and GSCA Learning	18	
Reinforcement Learning by ChatGPT	2	
Total:	100	

Question 1: Reactive Agents and State Machines .... 10 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 without any obstacles as shown below:



For each of the two goals indicated in the figure, can it be achieved by a reactive agent? By achieving a goal we mean that whatever the robot's initial position, the robot will get to the goal position, and then stop (nil action). For each of the two goals:

- If your answer is yes, give a production system for it. Do not call another production system. Write your own rules.
- If your answer is no, give your reason for it, and state how many steps the robot needs to remember in order to achieve the goal. No need to give a formal proof. An informal short explanation will be suffice. By k steps memory, we mean the robot remembers her past k actions and sensor readings. So 1 step memory means remembering last action and what the last sensor readings were.

Solution: Goal1: Yes.
Rules: $x_1x_2 \rightarrow nil$, $x_4\bar{x}_1 \rightarrow north$, $x_3\bar{x}_4 \rightarrow west$,
$x_2\bar{x}_3 \rightarrow south$, $x_1\bar{x}_2 \rightarrow east$, $1 \rightarrow north$
Goal2: No, the agent cannot distinguish goal2 from the one right below it. The agent needs one step memory. For a counter clock boundary following agent, it will first arrive at goal 1 then get goal2.

Question 2: Genetic Programming .... 5 points

Consider using genetic programming to evolve a program to control the robot to follow the boundary in the environment given in the last question. To do that, we need a fitness function to compute the fitness of any given program. Which of the following ones is most suitable, and why?

1. Ramdomly choose 10 starting positions, and for each of them, execute the given program until it has carried out **20** steps. The fitness value of the program is then the total number of next-to-wall cells visited during these executions.
2. Ramdomly choose 10 starting positions, and for each of them, execute the given program until it has carried out **200** steps. The fitness value of the program is then the total number of next-to-wall cells visited during these executions.

3. Ramdomly choose 10 starting positions, and for each of them, execute the given program until it has carried out **2,000,000** steps. The fitness value of the program is then the total number of next-to-wall cells visited during these executions.

Solution: The second one is most suitable as it is accurate and efficient. The first one is not accurate. The third one is costly.

Question 3: Search Problem Formulation .... 5 points

Consider the missionaries and cannibals problem:

> In the missionaries and cannibals problem, three missionaries and three cannibals must cross a river using a boat which can carry at most two people, under the constraint that, for both banks, if there are missionaries present on the bank, they cannot be outnumbered by cannibals (if they were, the cannibals would eat the missionaries). The boat cannot cross the river by itself with no people on board.

Assume that initially the missionaries and cannibals are at the left bank, and the goal is to get them all to the right bank safely. Also assume that the boat is initially at the left bank, and we do not care where it is in the goal state. Formulate this problem as a search problem by defining: states, initial state, goal condition (for checking if a state is a goal), operators and their costs.

Solution:

- States: (m, c, b) - m and c are the number of missionaries and cannibals, respectively, on the left bank, and b is either 0 (the boat is at the left bank) and 1 (the boat is at the right bank).
- Initial state: $(3, 3, 0)$.
- Goal condition: $(0, 0, b)$ for any b.
- Operators: $move(m, c)$ - move m missionaries and c cannibals from the bank where the boat is at to the other side, provided $m+c \leq 2$ and it does not cause the number of cannibals to exceed the number of missionaries on both side as a result of the action.
- Operator cost: all the same, say one unit.

Question 4: MDP .... 5 points

Consider again the missionaries and cannibals problem in the last question. Now there are two uncertainties:

- Even if the number of cannibals (C) is greater than the number of missionaries (M), it may still be safe: if $C > M$, then with probability M/C it is safe and probability $(C-M)/C$ it is unsafe (i.e. the missionaries got eaten).
- The boat is 100% safe for one person but has a 10% probability of sinking if two people are in it.

Formulate this setting as an MDP by giving the following:

- states;
- starting state;
- terminating condition $End(s)$;
- reward function $R(s, a, s')$: -1 if the boat sinks or the state is unsafe, 1 if all of them have crossed the river (i.e. all of them are on the right bank), 0 otherwise (meaning we don't care how long it takes for them to cross the river).
- discount fact: Let it be 1.
- state transition relation $T(s, a, s')$: for the sake of time, define this relation only for the starting state s.

Solution:

- States and actions: same as Q3
- Starting state: (3, 3, 0)
- Terminating condition: $End(s)$: (1) if s denote the case when either the boat sinks or the state is unsafe, denote $s = END$ for this case. (2) if $s = (0, 0, b)$ for any b.
- Reward: $R(S, A, END) = -1$, $R(S, A, (0, 0, b)) = 1$, and 0 otherwise.
- Transition relation $T(S, A, S')$ for $S = (3, 3, 0)$: applicable actions are $A = move(m', c')$, where $1 \le m' + c' \le 2$. Note that $3 - m' > 3 - c' \iff m' < c'$, so we only need to care about one side of banks, say the right side.
  - if $m' + c' = 2 \wedge m' < c' \Rightarrow m' = 0, c' = 2$, actually no M to eat on the right bank
    1. $T(S, A, END) = 0.1$
    2. $T(S, A, (3 - m', 3 - c', 1)) = 0.9$
  - if $m' + c' = 2 \wedge m' = c' \Rightarrow m' = 1, c' = 1$, balanced on both sides
    1. $T(S, A, END) = 0.1$
    2. $T(S, A, (3 - m', 3 - c', 1)) = 0.9$

- if $m' + c' = 2 \wedge m' > c' \Rightarrow m' = 2, c' = 0$, all M will be safe on the right bank
  1. $T(S, A, END) = 0.1 + 0.9 * \frac{3-1}{3} = 0.7$
  2. $T(S, A, (3 - m', 3 - c', 1)) = 0.9 * \frac{1}{3} = 0.3$
- if $m' + c' = 1 \wedge m' < c' \Rightarrow m' = 0, c' = 1$, the boat will not sink and the C will have no M to eat on the right bank
  1. $T(S, A, END) = 0$
  2. $T(S, A, (3 - m', 3 - c', 1)) = 1$
- if $m' + c' = 2 \wedge m' > c' \Rightarrow m' = 1, c' = 0$, the boat will not sink but the M on the left bank will possibly be unsafe
  1. $T(S, A, END) = \frac{1}{3}$
  2. $T(S, A, (3 - m', 3 - c', 1)) = \frac{2}{3}$

Question 5: Heuristic Search .... 10 points

Consider the following state space. The number next to a state is the value of the heuristic function on the state, and the number next to an arc is the cost of the corresponding operator. If the number next to a state is 0, that means that the state is a goal state.

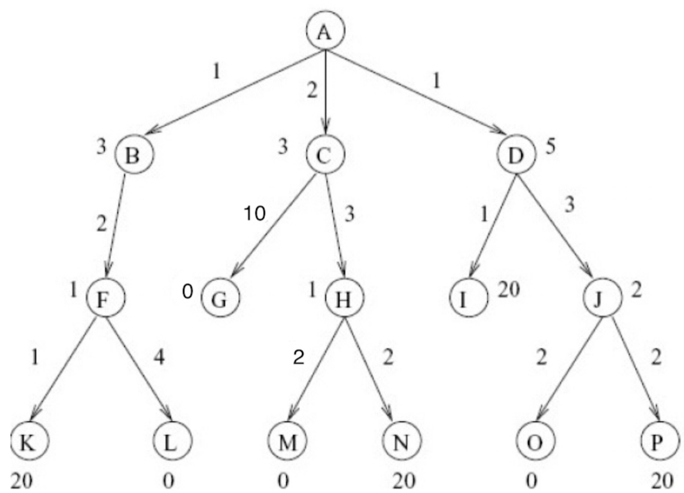


Figure 1: A search problem

(5.1) Give the sequence of the states expanded by A^* algorithm, starting from the root A and terminating at a goal state. Notice that whenever there is a tie, we prefer nodes at deeper levels, and on the same level, left to right.

(5.2) Can you adjust the heuristic function so that the goal state G is returned? Notice that your heuristic function still needs to be admissible, and you don't need to do this question if G is already the terminating goal state in your solution to (5.1).

Solution:

(5.1) state sequence: A, B, F, C, H, D, J, O.

(5.2) Impossible as the path to G is not an optimal solution.

Question 6: Alpha-Beta Pruning .... **5 points**

Perform a left-to-right alpha-beta prune on the following minimax game tree. Which nodes are pruned? What's the value of the root? Left-to-right means that whenever a node is expanded, it's children are considered in the order from left to right. This means the leaf nodes are generated from left right. So the first leaf node considered is D, followed by E, followed by M and so on.

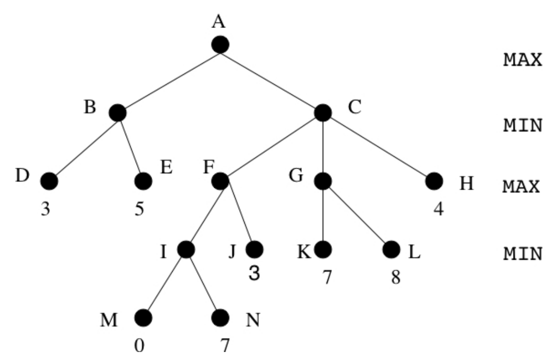


Figure 2: A minimax search tree

Solution: The value of A is 3. The nodes N, G (and its childern), and H are pruned.

Question 7: FOL Representation .... **10 points**

Let $Fools(x, y, t)$ stands for "x can fool y at time t". Represent the following statements in first-order logic:

1. Everyone can fool every other person sometimes. Notice that this means that for example, Jane can fool both Jack and John but not necessarily at the same time.
2. There is exactly one time when someone can fool everyone else.

3. At no time can someone fool all other people.
4. A person cannot fool another person all the time.
5. For any two people x and y, if x can fool y some time, then y can also fool x some time.

Solution:

1. $\forall x, y. x \neq y \supset \exists t. Fools(x, y, t)$,
   Not ideal but ok if condition $x \neq y$ is dropped:
   $\forall x, y \exists t. Fools(x, y, t)$
   Similarly for answers below.
2. $\exists t[\exists x \forall y(x \neq y \supset Fools(x, y, t)) \wedge \forall u(\exists x \forall y(x \neq y \supset Fools(x, y, u)) \supset u = t)]$,
3. $\forall t, x \exists y. y \neq x \wedge \neg Fools(x, y, t)$ Or
   $\neg \exists t \exists x \forall y(y \neq x \supset Fools(x, y, t))$,
4. $\forall x, y(x \neq y \supset \exists t. \neg Fools(x, y, t))$,
5. $\forall x, y. (\exists t. Fools(x, y, t)) \supset (\exists u. Fools(y, x, u))$.

Question 8: Propositional Logic .... 10 points

A, B, and C are three friends, and we know:

1. A drinks tea.
2. B drinks wine and dates C.
3. C drinks either tea or wine but not both, and dates A.

Assuming the following propositions:

- T_x: x drinks tea. So this yields three propositions: T_A, T_B, and T_C.
- W_x: x drinks wine. This will yield three propositions.
- $D_{x,y}$: x dates y. This will yield 9 propositions.

Answer the following questions:

(8.1) Represent the above facts as propositional formula using the given propositions.

(8.2) Convert your formulas to clauses.

(8.3) Represent the following query as a formula: A wine drinker dates a tea drinker. To make your formula shorter, you can assume that by a wine drinker, we mean either B or C, and by a tea drinker either A or C. Also no one dates themselves.

(8.4) Convert the negation of your query formula to clauses.

(8.5) Use resolution to derive the empty clause from all the clauses (those from your facts and from the negation of your query).

Solution: 1. $T_A \wedge W_B \wedge D_{B,C} \wedge D_{C,A} \wedge (T_C \vee W_C) \wedge (\neg T_c \vee \neg W_C)$
2. $\{T_A, W_B, D_{B,C}, D_{C,A}, T_C \vee W_C, \neg T_C \vee \neg W_C\}$
3. $(W_B \wedge D_{B,A} \wedge T_A) \vee (W_B \wedge D_{B,C} \wedge T_C) \vee (W_C \wedge D_{C,A} \wedge T_A)$
4.$\{\neg W_B \vee \neg D_{B,A} \vee \neg T_A, \neg W_B \vee \neg D_{B,C} \vee \neg T_C, \neg W_C \vee \neg D_{C,A} \vee \neg T_A\}$

5. $D_{B,C}$ and $\neg W_B \vee \neg D_{B,C} \vee \neg T_C$ yield $\neg W_B \vee \neg T_C$.
T_A and $\neg W_C \vee \neg D_{C,A} \vee \neg T_A$ yield $\neg W_C \vee \neg D_{C,A}$.
$\neg W_C \vee \neg D_{C,A}$ and $D_{C,A}$ yield $\neg W_C$.
$\neg W_C$ and $T_C \vee W_C$ yield T_C.
W_B and $\neg W_B \vee \neg T_C$ yield $\neg T_C$.
T_C and $\neg T_C$ yield the empty clause.

Question 9: Game theory and auction .... 10 points

Consider the single item first-price auction with two bidders B_1 and B_2, and assume that when there is a tie, B_1 will win. Suppose B_i values the item x_i, $i = 1, 2$, and the information is common knowledge (thus B_1 knows that B_2's value is x_2, and vice versa and so on). Suppose further that each can bid with any integer in the interval $[1, 100]$.

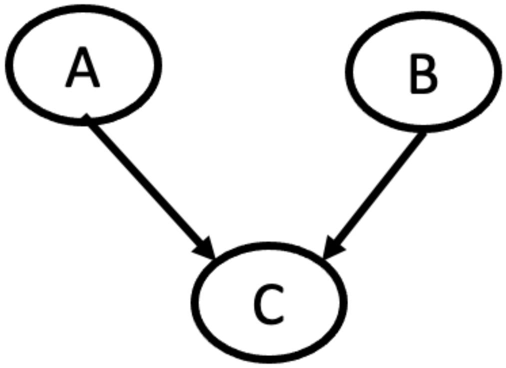
- Make this auction into a game in normal form $(\{B_1, B_2\}, R_1, R_2, u_1, u_2)$ by defining R_i (the set of actions for player B_i) and u_i (player B_i's utility function). You can assume that both players are risk neutral.
- Suppose $x_1 = 10$ and $x_2 = 5$. What are the Nash equilibria of your game?

Solution: $R_1 = R_2 =$ set of integers in $[1, 100]$. $u_1(b_1, b_2) = x_1 - b_1$ if $b_1 \geq b_2$ and 0 otherwise. $u_2(b_1, b_2) = x_2 - b_2$ if $b_2 > b_1$ and 0 otherwise.

NEs: (b, b) for $10 \geq b \geq 4$. Consider (b_1, b_2). If $b_1 > b_2$. Then B_1 is the winner and he is better off lower the bid to b_1. If $b_2 > b_1$ then B_2 is the winner. If $b_2 > 5$, then B_2 overbid and will be better off lowr his bid so he'll lose. If $b_2 \leq 5$, then B_1 is better off by increasing his bid to b_2 thus winning the auction. So for (b_1, b_2) to be a NE, $b_1 = b_2$.

Question 10: Uncertainty .... 10 points

Consider the following Bayesian network:



Prove the two semantics for this network is equivalent. That is, prove that the following two are equivalent:

1. $P(A, B, C) = P(A)P(B)P(C|A, B)$.
2. A and B are independent.

Solution: $1 \to 2$: $P(A, B, C) = P(C|AB)P(A, B)$
since $P(A, B, C) = P(A)P(B)P(C|AB)$ we have $P(A, B) = P(A)P(B)$
$2 \to 1$: $P(A, B, C) = P(C|AB)P(A, B) = P(C|AB)P(A)P(B)$

Question 11: Perceptron and GSCA Learning .... 18 points

Consider the following data set:

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

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

(11.1) Run the perceptron error-correction procedure on this dataset using 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. Run it for 3 iterations.

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

(11.2) Does it converge? If so, give the Boolean function corresponding to the perceptron. If not, either give a perceptron that agrees with the training set thus showing that the procedure will converge but needs more iterations, or prove that there is no perceptron that can agree with the training set, i.e. the training set is not linearly separable, so the error-correction procedure will not converge. [HINT To show that the training set is not linearly separable, prove by refutation: suppose there is one, then the four examples in the training set will yield four contraints that are inconsistent, like how we proved the exclusive or is not linearly separable in class.]

(11.3) 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	Weighted Sum	Actual	Desired
Initial		0,0,0,0,0			
1	1,0,0,1,1	0,0,0,0,0	0	1	1
2	0,1,1,0,1	0,0,0,0,0	0	1	1
3	0,0,1,0,1	0,0,-1,0,-1	0	1	0
4	1,1,0,1,1	0,0,-1,0,-1	-1	0	0
1	1,0,0,1,1	1,0,-1,1,0	-1	0	1
2	0,1,1,0,1	1,1,0,1,1	-1	0	1
3	0,0,1,0,1	1,1,-1,1,0	1	1	0
4	1,1,0,1,1	0,0,-1,0,-1	3	1	0
1	1,0,0,1,1	1,0,-1,1,0	-1	0	1
2	0,1,1,0,1	1,1,0,1,1	-1	0	1
3	0,0,1,0,1	1,1,-1,1,0	1	1	0
4	1,1,0,1,1	0,0,-1,0,-1	3	1	0

2. It won't converge. The training set corresponds to $x_1\overline{x_3} + \overline{x_1}x_3$, where x_2 is $\overline{x_1}$, and x_4 is $\overline{x_3}$. Despite the rewriting, it is still not linearly separable. Suppose there is a perceptron (w_1, w_2, w_3, w_4) with threshold θ that agrees with the dataset, then

$$\begin{aligned}
&w_1 + w_4 \geq \theta, \\
&w_2 + w_3 \geq \theta, \\
&w_3 < \theta, \\
&w1 + w_2 + w_4 < \theta.
\end{aligned}$$

Thus $w_1 + w_2 + w_3 + w_4 \geq 2\theta$ (from the first two) and $w_1 + w_2 + w_3 + w_4 < 2\theta$ (from the last two), a contradition. Thus the dataset is not linearly separable.

3. We begin with: $true \supset OK$. Choose the feature that yields the largest value of r_α : $r_{x_1} = r_{x_2} = r_{x_3} = r_{x_4} = \frac{1}{2}$. We choose x_1, this will generate : $x_1 \supset OK$. This rule still covers the negative instance ID4, so we still need to narrow it. This time we have $r_{x_2} = 0$, $r_{x_3} = 0$, $r_{x_4} = \frac{1}{2}$. We can add x_4 : $x_1 \wedge x_4 \supset OK$. This rule still covers the negative instance ID4. We cannot learn a rule from this.

Return to the fist step and choose x_2, there is a negative instance ID4, then $r_{x_1} = 0$, $r_{x_3} = 1$, $r_{x_4} = 0$. So we choose x_3, this will generate : $x_2 \wedge x_3 \supset OK$.

Delete ID2 instance from the training set, repeat the process, but this time cannot learn any rule because only instance ID1 is positive, while we have shown in the first step that $x_1 \wedge x_4 \supset OK$ still covers the negative instance ID4.

So GSCA learns only rule, $x_2 \wedge x_3 \supset OK$.

Question 12: Reinforcement Learning by ChatGPT .... 2 points

I asked ChatGPT, a recently released natural langauge conversational agent by OpenAI, to give me "a good exam question for reinforcement learning." Below is what ChatGPT returned:

> Consider a reinforcement learning agent interacting with an environment. The agent has a set of possible actions, A, and the environment has a set of possible states, S. At each time step t, the agent receives a reward R_t from the environment and observes the current state s_t. The goal of the agent is to learn a policy, π, that maps states to actions, such that the expected cumulative reward over time is maximized.
>
> Suppose the agent is currently in state s and is considering two possible actions: a_1 and a_2. The agent has observed the following rewards when taking these actions in state s:
>
> - a_1: 3, 2, 5, 4, 7
> - a_2: 4, 1, 1, 6, 1
>
> Which of the following statements is true?
>
> (a) The agent should choose a_1 because it has a higher average reward.
> (b) The agent should choose a_2 because it has a higher average reward.
> (c) The agent cannot determine which action to take because it does not have enough information about the environment.
> (d) The agent should choose a_1 because it has a higher maximum reward.

Solution: ChatGPT gave answer (a), which we will accept as a correct answer. You can also say (a) with ϵ probability, and a random action otherwise, assuming the RL agent is using an ϵ-greedy algorithm. An aside: ChatGPT is fun but don't bet your house on it. It gives wrong answers quite a few times. Also it does not understand RL. The question that it came up with above is most likely copied/paraphrased from a similar question on the internet.