KOÇ UNIVERSITY
COLLEGE OF ENGINEERING

COMP 341: INTRODUCTION TO ARTIFICIAL INTELLIGENCE

Final Exam

FALL 2018, 05/01/2019
DURATION: 140 MINUTES

Name: Solutions

ID: 00110001 00110000 00110000

- This exam contains 8 pages including this cover page and 8 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	Total
Points:	9	10	16	12	8	15	18	15	103
Score:									

1. (9 points) True or False :

False — A rational agent will always achieve its goal.

True — Suppose that h_1 and h_2 are both admissible heuristics to a search problem. The heuristic, $max(h_1, h_2)$, is also admissible.

False — If X and Y are conditionally independent given Z, then $P(Z|X,Y) = P(Z|X)P(Z|Y)$.

True — Regularization helps with overfitting.

True — Markov Decision Processes are used to model problems involving uncertain action outcomes.

False — In Q-learning, the agent must follow its own policy.

2. (10 points) You are working at the credit department of a bank. You are trying to decide whether to give credit to new customers or not. You have data about your previous customers (e.g. their occupation, age, gender, money in their accounts etc.) and whether they were able to pay or not. Considering only the LR: linear regression, NB: Naïve Bayes and kNN: k-Nearest Neighbors methods, answer the following questions.

   (a) (1 point) If you formulate this as a machine learning problem, what is your input and what is your output? You are allowed to discretize all the data.

   Input: Customer Data. Output: Whether they paid or not

   (b) (3 points) You do not always know everything about a new customer, i.e., you may have missing data. Which method would you use and why?

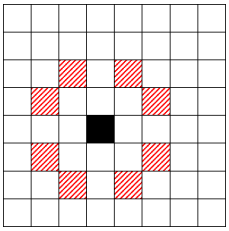
   NB. It is basically a Bayesian Network and the missing data can be marginalized out (treated as hidden). Method 1.5, justification 1.5. Also applies to the following parts

   (c) (3 points) You look at your training data and the problem turns out to be highly nonlinear. Assuming that you know everything about a new customer, which method would you use and why?
   kNN since it works well with nonlinear data. I will give points for LR with special features if described well enough

   (d) (3 points) This is not usually known to customers but banks are usually fine with their customers making their credit payments within a certain time period (e.g. 90 days) after their deadline. Your boss wants you to predict how late will an existing customer be with their payments. Assuming you have the correct data, which method would you use and why?

   LR since the days to payment can be regarded as a continuous variable to be predicted which regression approaches do.

3. (16 points) A chessboard is a grid structure made up of squares. Only one chess piece can occupy a square but a square can be empty. A knight can move in 2 steps in any canonical direction followed by a single step in any of the perpendicular direction (e.g. it can move 2 steps left and then 1 step up). In the figure, the black box is the location of the knight and the striped/light gray boxes are the potential locations that it can move.



Consider the problem of placing k knighs on an $n \times n$ chessboard such that no two knights are attacking each other, where k is given and $k \leq n^2$.

(a) (6 points) Formulate this as a CSP. In your formulation what are the variables and what is the domain of these variables?

Variables: Locations of knights $(X_1, \ldots X_k)$, Domains: $(x_i, y_i), i = 1 \ldots k, 0 < x_i, y_i \leq n)$ OR
Variables: The occupancy of each cell $(C_{11}, C_{12}, \ldots C_{nn})$, Domains: $(\{0, 1\})$

(b) (3 points) What are the constraints based on your formulation? You may describe them verbally.

First formulation: No two knights can attack each other $NotAttacking(X_i, X_j), i, j = 1 \ldots k$ and knights cannot occupy the same location $X_i \neq X_j, i \neq j$
Second Formulation: No two knights can attack each other $NotAttacking(C_{ij} == 1, C_{kl} == 1), i, j, k, l = 1 \ldots n$ and number of occupied cells must be k, $\sum_{i,j} C_{ij} = k$ (which is to force knights not occupying the same cells)

(c) (3 points) You want to use **local search** to solve this problem. Describe a reasonable objective funcion for the problem formulation (i.e. answer to part (a)) you have chosen.

First Formulation: Number of attacking knights + number of knights occupying the same square. May multiple the latter one to make it more severe.
Second Formulation: Number of attacking knights + $|k - \sum_{i,j} C_{ij}|$

(d) (4 points) What would a complete state look like for your chosen formulation? Based on this answer, decribe a reasonable successor function.

First Formulation: All the k knights are at some location. Local Move: Changing the position of a single knight.
Second Formulation: Assignment of 0s and 1s to cells. Local Move: Flipping a cell.

4. (12 points) Blocks World: Imagine a problem where there are n numbered blocks on top of a table. Blocks can be stacked on top of each other. Further assume that there are k discreet locations on the table. The goal is to stack the blocks such that they are ordered from the smallest to the largest. The starting block positions can be random. An example for $n = 4$ and $k = 3$ and the desired end-state is given below. You can only move one block at a time. You cannot move a block if there is another block on top of it. If you were to formulate this as a search problem:

   (a) (4 points) What would be your state representation? What would be your goal test for this representation?

   A 2D $k \times nk$ integer array, where the first dimension represents the table locations and the second dimension represents the blocks for each location (3 points). Note that this handles stacks as well. Goal test would be to check whether the order is achieved by looking at each location (1 point). Other answers, such as using linked lists for each location, are possible.

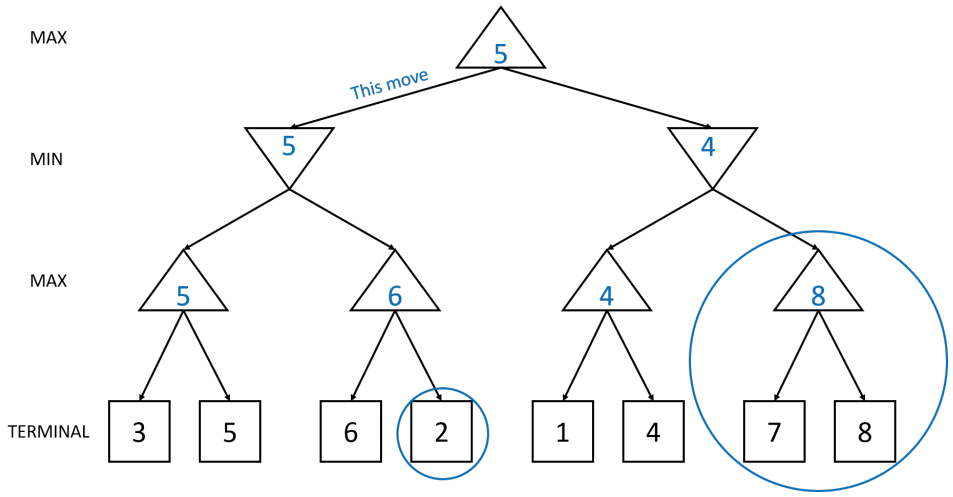
   (b) (8 points) What algorithm would you chose to solve this problem? If any, what heuristic would you use and why? Would this heuristic favor solution efficiency or computational efficiency?

   A* search if given a heuristic would get 3 points. A* without a heuristic, Uniform Cost Search, Breadth First Search, Iterative Deepeining Depth First Search would get 2 points. Depth First Search with a justification for computational efficiency get 2 points. Just DFS would get 1 point. Some A* Heuristics:

   - Number of blocks out of place: 1 point by itself. Admissible justification 2 points. Easy to compute additional 1 point.
   - In addition to the above, +2 for each block that is not directly above the one it is supposed to. Note that to move a block to its appropriate place in this situation, you must at least make 2 moves. Such a heuristic, along with justification gets at least 2 points.
   - Sum of Manhattan Distances to Place: 1 point by itself. Inadmissible but potentially faster to find a (non-optimal) solution justification 1 point. Easy to compute additional 1 point.

   There are other heuristics. Grade will depend on the justification and admissibility. If your heuristic looks to be very good, you may get bonus points!

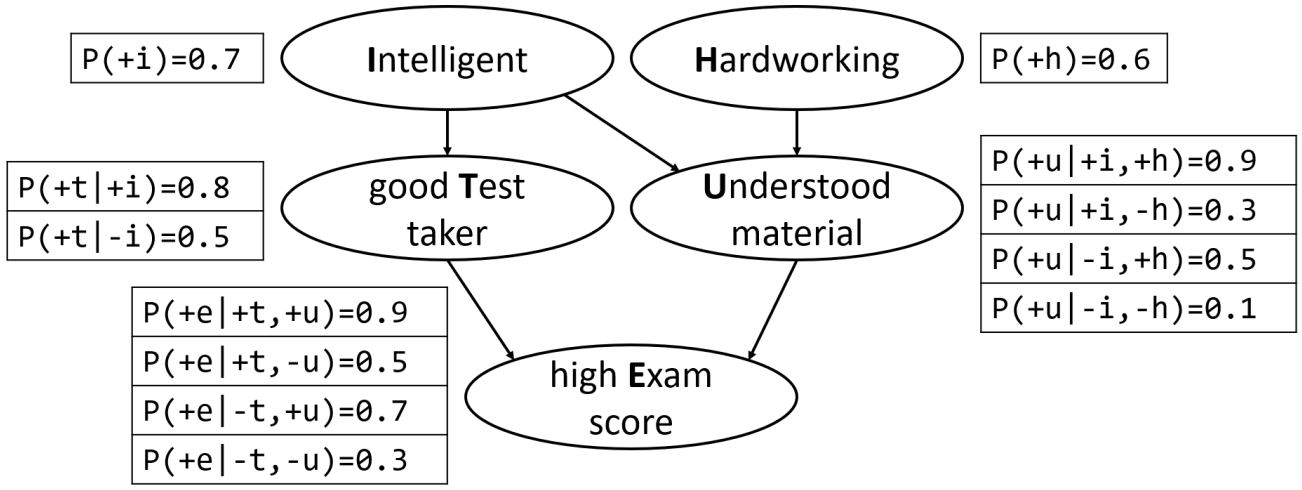
5. (8 points) Consider the game tree below.



   (a) (4 points) Fill in the values of each state on the tree. Mark the optimal first move of the player MAX. Each correct value 0.5, including the selected move

   (b) (4 points) Circle the pruned nodes and/or subtrees during alpha-beta pruning if the moves are explored from left to right. Two pruned parts, 2 points each

6. (15 points) You are given the following Bayesian Network (BN) that models the exam scores of students. All the nodes represent Boolean random variables with the bold capital letters being used in the mathematical notation. Note that only the positive instance probabilities are given but you can get the negative instance values by subracting them from 1.



(a) (1 point) Write down the joint probability distribution, $P(I, H, T, U, E)$, defined by this BN.
$P(I, H, T, U, E) = P(I)P(H)P(T|I)P(U|I, H)P(E|T, U)$

(b) (1 point) What is the most likely sample from this BN, i.e., if we were to run prior sampling, which sample would be repeated the most?
$\{+i, +h, +t, +u, +e\}$

(c) (3 points) Calculate the weights of the following samples if you were to do likelihood weighting given the evidence variables $\{-t, +u\}$:

- $\{+i, +h, -t, +u, +e\} \Rightarrow 0.2 \times 0.9 = 0.18$
- $\{+i, +h, -t, +u, -e\} \Rightarrow 0.2 \times 0.9 = 0.18$
- $\{+i, -h, -t, +u, +e\} \Rightarrow 0.2 \times 0.3 = 0.06$
- $\{-i, +h, -t, +u, -e\} \Rightarrow 0.5 \times 0.5 = 0.25$
- $\{-i, -h, -t, +u, +e\} \Rightarrow 0.5 \times 0.1 = 0.05$

(d) (10 points) Compute the probability that a student who did well on the test actually understood the material, i.e. $P(+u|+e)$? You may leave mathematical expressions as is, you do not need to calculate the multiplications and summations.

This can systematically be solved by variable elimination and factors. However, we are going to follow this slightly fine tuned plan:

1. Join $P(I)$ and $P(T|I)$ to get $P(T,I)$, and sum out I to get $P(T)$ (do it only for $+t$, easy to get $-t$ then)
2. Join $P(H)$ and $P(U|I,H)$ to get $P(U,H|I)$, and sum out H to get $P(U|I)$. This is possible since H and I are independent. (do it only for $+u$).
3. Join $P(I)$ and $P(U|I)$, and sum out I to get $P(U)$ (do it only for $+u$).
4. Join $P(T)$, $P(U)$ and $P(+e|T,U)$, and sum out T to get $P(+e,U)$.
5. Then it is trivial to find $P(+u|+e)$

Step 1:
$P(+t,+i) = P(+i)P(+t|+i) = 0.7 \cdot 0.8 = 0.56, P(+t,-i) = P(-i)P(+t|-i) = 0.3 \cdot 0.5 = 0.15$
$P(+t) = P(+t,+i) + P(+t,-i) = 0.56 + 0.15 = 0.71, P(-t) = 1 - P(+t) = 0.29$

Step 2:

I	H	$P(+u,H\|I)$
$+i$	$+h$	$P(+u\|+h,+i)P(+h) = 0.9 \cdot 0.6 = 0.54$
$+i$	$-h$	$P(+u\|+h,+i)P(-h) = 0.3 \cdot 0.4 = 0.12$
$-i$	$+h$	$P(+u\|+h,-i)P(+h) = 0.5 \cdot 0.6 = 0.30$
$-i$	$-h$	$P(+u\|+h,-i)P(-h) = 0.1 \cdot 0.4 = 0.04$

Sum out H: $P(+u|+i) = 0.54 + 0.12 = 0.66, P(+u|-i) = 0.3 + 0.04 = 0.34$
Step 3:
$P(+u,+i) = P(+i)P(+u|+i) = 0.66 \cdot 0.7 = 0.462, P(+u,-i) = 0.34 \cdot 0.3 = 0.102$
$P(+u) = P(+u,+i) + P(+u,-i) = 0.462 + 0.102 = 0.564, P(-u) = 1 - P(+u) = 0.436$

Step 4:

T	U	$P(+e,T,U)$
$+t$	$+u$	$0.9 \cdot 0.71 \cdot 0.564$
$+t$	$-u$	$0.5 \cdot 0.71 \cdot 0.436$
$-t$	$+u$	$0.7 \cdot 0.29 \cdot 0.564$
$-t$	$-u$	$0.3 \cdot 0.29 \cdot 0.436$

Sum out T:
$P(+e,+u) = 0.639 \cdot 0.564 + 0.203 \cdot 0.564 = 0.564 \cdot 0.842$
$P(+e,-u) = 0.436(0.355 + 0.087) = 0.436 \cdot 0.442$
$P(+u|+e) = \frac{P(+e,+u)}{P(+e,+u)+P(+e,-u)} = \frac{0.564 \cdot 0.842}{0.564 \cdot 0.842 + 0.436 \cdot 0.442} \sim 0.7$

7. (18 points) Grandpac is hunting ghosts in a grid using his sonar. He cannot directly see the ghosts but he can make noisy observations about their locations. With years of practice, he has an idea of how ghosts might move. The world is such that the time progresses in discreet steps. Furthermore Grandpac and ghosts move in turns at each time step, Grandpac going first.

   (a) (4 points) Which representation approach among Static Bayesian Networks (BN), Hidden Markov Models (HMM) and Markov Decision Processes (MDP) should Grandpac chose to find the most likely location of a ghost at a given time step T? Formulate the problem based on your selection without going into details.

   Representation: HMMs with sonar as emissions and "idea of ghosts" as the transition function. Prior probs can be uniform.

   (b) (2 points) Hunting a ghost gives Grandpac 20 points. He must use his "Proton Pack" on the ghosts to hunt them but this special weapon costs 10 points to use. Suppose that he has estimated the position of a ghost at the grid location (x, y) with probability p. What is his expected utility as a function of p? For what value of p the Grandpac should use his weapon on the ghost? If he never or always should, state it so.

   $20p - 10$, use when $p > 0.50$.

   (c) (12 points) Ghastly is tired of Grandpac hunting his ghost friends and decides to intervene. When Ghastly is around, ghosts can no longer move and the grid becomes slippery such that Grandpac cannot always move where he wants to. Grandpac and Ghastly can see each other on the grid. Ghastly acts againts Grandpac. The turn taking nature of the world stays the same with Grandpac going first. How should Grandpac represent this problem? What algorithm(s) can he use to solve it? (You are free to chose anything from the topics that we have covered or come up with your own approach)

   This was an open-ended question with many possible answers. There are a few potential directions to select. You should have paid attention to:

   - Ghosts are immobile: This affects the state OR the reward function (if MDP route is selected). Assume known (1 point), low uncertainty due to stationary ghosts and many measurements Grandpac will take (2 points) or if reward, unknown to be discovered (2 points).
   - Ghastly and Grandpac can see each other: This affects the state. Implying no uncertainty for this part. (1 point)
   - Slippery Grid: Action uncertainty for Grandpac (2 points). May assume the same for Ghastly as well but do not have to.
   - Adversarial Nature: Need to somehow plan for multiple agents. (2 points)
   - Zero-Sumness: Can assume to be zero-sum with Grandpac's utility since Grandpac wants to hunt and Ghastly wants to protect ghosts. (1 point)

   Mentioning these somehow already gets you 8 points. The remaining is for the method:

   - Adversarial search: Expectiminimax for actions. Maybe roll the ghost uncertainty into the search or assume known. Up to 3 points based on formulation
   - MDPs: Define a zero-sum MDP with minimax value functions instead of just max. Up to 4 points. Need to assume known ghost places
   - Reinforcement Learning: Similar to above but making it online and having the ghosts be part of the unknown reward function. Up to 5 points.
   - Partially Observable MDPs (POMDP): MDPs where there is state and action uncertainty. In addition need to define a minimax formulation. Up to 8 points.

8. (15 points) An agent is in a 2×3 grid world. Each grid cell has a reward which is earned by entering that cell. The available actions are $\{N, S, E, W\}$ for north, south, east, and west. Actions may be stochastic; an action does not necessarily result in the expected next state.

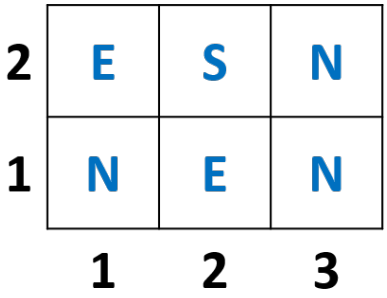
   (a) (4 points) Consider the following episodes performed in the grid world. These tuples are of the form $[s, a, s', r]$, where we start in state s, perform action a, end up in state s', and receive immediate reward r which is determined by the state entered. Let the discount factor $\gamma = 1.0$ for this MDP. Fill in the values computed by the Q-learning algorithm with a learning rate of $\alpha = 0.5$. Assume that initial Q values are 0. Fill out each row using values you have computed in previous rows.

Episode	$[s, a, s', r]$	$Q(s, a)$	Computed Value
1	$[(1,2), E, (2,2), 2]$	$Q((1,2), E)$	$0 + 0.5(2 + 1 \cdot 0) - 0) = 1$ or 1 or 1
2	$[(2,2), S, (2,1), 4]$	$Q((2,2), S)$	$0 + 0.5(4 + 1 \cdot 0) - 0) = 2$ or 2 or 2
3	$[(2,1), W, (1,1), 8]$	$Q((2,1), W)$	$0 + 0.5(8 + 1 \cdot 0) - 0) = 4$ or 4 or 4
4	$[(1,1), N, (2,1), 4]$	$Q((1,1), N)$	$0 + 0.5(4 + 1 \cdot 4) - 0) = 4$ or 2.5 or 4
5	$[(1,2), E, (1,1), 8]$	$Q((1,2), E)$	$1 + 0.5(8 + 1 \cdot 4) - 1) = 6.5$ or 5.75 or 6
6	$[(1,1), E, (2,1), 4]$	$Q((1,1), E)$	$0 + 0.5(4 + 1 \cdot 4) - 0) = 4$ or 4 or 5
7	$[(2,1), E, (3,1), 64]$	$Q((2,1), E)$	$0 + 0.5(64 + 1 \cdot 0) - 0) = 32$ or 32 or 35
8	$[(3,1), N, (3,2), 16]$	$Q((3,1), N)$	$0 + 0.5(16 + 1 \cdot 0) - 0) = 8$ or 8 or 8

   Keep applying $Q(s,a) \leftarrow Q(s,a) + \alpha(r + \gamma max_{a'}(Q(s',a')) - Q(s,a))$ with 0 initial q-values. The first OR corresponds to 4th episodes next state being (1,2) and the second OR corresponds to to the 5th episodes first state being (2,1). State (1,1)'s action changes to E in both of the OR cases!

   (b) (2 points) Which of the episodes show that the MDP is stochastic? 1 and 5 since the agent takes the same action in the same state but ends up in different states

   (c) (3 points) In the empty grid below, fill in the policy (i.e. write the corresponding actions in the grid) using the Q-values you have computed in the previous step. In case of equal values (including 0), pick using the given order: $\{N, S, E, W\}$



   (d) (6 points) (Unrelated to the above part) How can we modify the q-learning approach to calculate the q-values of a specific policy and not the optimal one? Write down the update rule if you can.

   The regular q-value update is as follows given the tuple (s, a, r, s'):
   $Q(s,a) \leftarrow Q(s,a) + \alpha(r + \gamma max_{a'}(Q(s',a')) - Q(s,a))$ or equivalently
   $Q(s,a) \leftarrow (1-\alpha)Q(s,a) + \alpha(r + \gamma max_{a'}(Q(s',a')))$
   Notice that we are taking the maximum over the next q-state given the action and not the action that a specific policy gives. For the question, the agent needs to take another action at its next state, i.e., need the tuple: (s, a, r, s', a'). This method is aptly named SARSA and it is the on-policy version of Q-learning with the update as follows:
   $Q(s,a) \leftarrow Q(s,a) + \alpha(r + \gamma Q(s',a') - Q(s,a))$