INFO8006 Introduction to Artificial Intelligence

Exam of August 2020

I decided to not make a video for the mock exam correction. Instead, here is a detailed written solution. Please, feel free to reach me whenever something is not clear!
Antoine

Instructions

- *The exam starts at 8:30 AM.*
- *Question 1 must be answered on eCampus. The deadline is set to 9:30 AM.*
- *Questions 2 to 5 must be answered on paper. Your sheets must be scanned and submitted on eCampus before 12:30 AM.*
- *Answer the questions on separate sheets, labeled with the question number, your first name, last name and student id.*
- *Answer in English or in French.*

Question 1 [4 points]

Multiple choice questions. Choose one of the four choices. Correct answers are graded $+\frac{4}{10}$*, wrong answers are graded* $-\frac{2}{15}$ *and the absence of answers is graded* 0*. The total of your grade for Question 1 is bounded below at* 0/4*.*

1. Let us consider the task environment for the card game of Belotte. Which of the following is true?
   (a) Belotte is deterministic, partially observable, continuous and single agent.
   (b) Belotte is stochastic, partially observable, discrete and multiagent.
   (c) Belotte is stochastic, fully observable, discrete and multiagent.
   (d) Belotte is stochastic, partially observable, dynamic and multi-agents.
2. Let h_A and h_B be two admissible heuristics, which of the following propositions is correct:
   (a) $min(h_A, h_B)$ dominates h_A.
   (b) $h_A + h_B$ is admissible.
   (c) $min(max(h_A, h_B), min(h_A, h_B))$ is admissible.
   (d) $max(h_A + h_B, h_A - h_B)$ is admissible.
3. Let α and β denote two sentences from propositional logic. Which of the following propositions is not equivalent to the others:
   (a) $\alpha \models \beta$.
   (b) $M(\alpha) \subset M(\beta)$.
   (c) $(\alpha \wedge \neg\beta)$ is unsatisfiable.
   (d) $M(\alpha \wedge \neg\beta) = \{\}$.
4. Which of the following propositions is correct:
   (a) A transposition table can only be used for deterministic 2-player games.
   (b) A dictionary is an efficient way of implementing transposition tables.
   (c) Minimax is optimal if the adversary is not optimal.
   (d) In MCTS, the backpropagation phase optimizes the parameters of the algorithm.
5. Which of the following propositions is not equivalent to the others:
   (a) $\forall x, y, P(x, y) = P(x)P(y)$.
   (b) $\forall x, y, z, P(x)P(z|x, y)P(y) = P(x, y, z)$.
   (c) $X \perp Y$.
   (d) $X = f(Y)$ (read the random variable X is a function of the random variable Y).

6. You are tasked to implement a procedure for sampling from a 100-dimensional multivariate Normal distribution. Which of the following would be the most appropriate algorithm?

   (a) Inference by enumeration.

   (b) Inference by variable elimination.

   (c) The Kalman Filter.

   (d) Gibbs sampling.

7. Let us consider the RAGI robot wandering around at the Montefiore Institute. Which of the following is wrong?

   (a) Filtering can make use of its sensor data to maintain a belief state about its position.

   (b) A neural network can be used for detecting obstacles.

   (c) The H-Minimax algorithm can be helpful for decoding the speech of its visitors.

   (d) Policy iteration can be used to make rational decisions about its next move.

8. Let us consider the following version of the Bellman equation: $V(s) = R(s) + \gamma \max_a \sum_{s'} P(s'|s,a)V(s')$. Which of the following is false?

   (a) The equation holds even if the environment is continuous.

   (b) The equation holds even if the environment is deterministic.

   (c) The equation holds even if the environment is partially observable.

   (d) The equation holds even if the environment is fully observable.

9. Neural networks are typically trained using...

   (a) the Newton-Raphson method.

   (b) gradient descent (or a variant thereof).

   (c) the bisection method.

   (d) the simplex algorithm.

10. The search algorithm in AlphaGo is based on...

    (a) the Minimax algorithm.

    (b) Monte Carlo tree search.

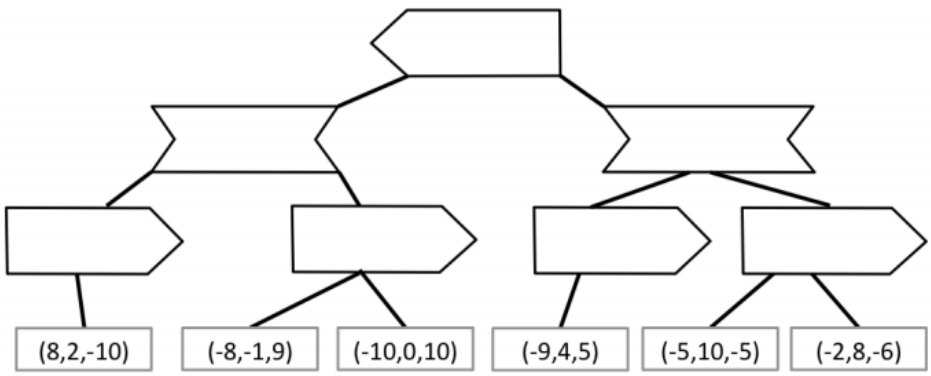
    (c) A*.

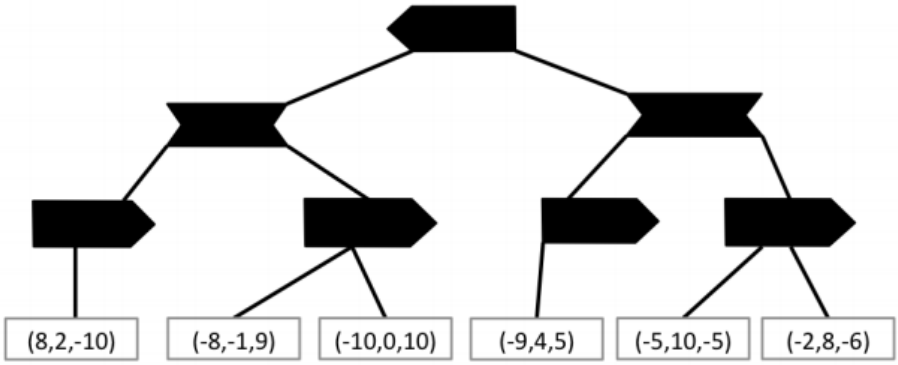
    (d) a Markov Chain Monte Carlo method.

Question 2 [4 points]

A solution for this exercise is provided on the following website `http://ai.berkeley.edu/exams/fa12_final_solutions.pdf`. Note howeber that the "min" conditions in the solution of the 4th sub-question is useless.

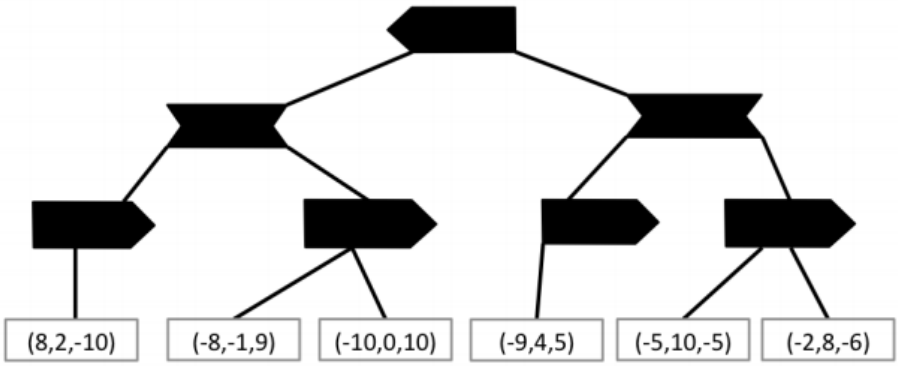
1. Yann, Yoshua and Geoffrey are playing poker together. The objective of the game is to become the wealthiest player and so to avoid being liable. Fill in the following multiplayer minimax tree, in which the players utilities is given in dollars and as (Yann's utility, Yoshua's utility, Geoffrey's utility). Yann is the first player, Yoshua is second and Geoffrey the third.





2. As you may have noticed, poker is a zero-sum game. Below, apply an algorithm similar to $\alpha - \beta$ pruning. Cross off any branches that can be safely pruned. If no branches can be removed, explain why not. Make the assumption that the nodes are explored from left to right.

3. In addition to the zero-sum game assumption, we now further assume that all utilities are bounded between $[-10, 10]$. Below, cross off any branches that can be safely unexplored. If no branches can be removed, explain why not. Make the assumption that the nodes are explored from left to right.



4. Again, consider the zero-sum, bounded-utility, 3-player game described above. Here, we assume the utilities are bounded by $[-C, C]$ for some constant C (the previous problem considered the case $C = 10$). Assume there are only 3 actions in the game, and the order in which players play is Yann, Yoshua, then Geoffrey. Below is pseudo-code that computes the utilities at each node, but the pruning conditions are left out. Fill in the conditions below that prune wherever safe.

Algorithm 1 Compute palyers' values.

```
1: procedure COMPUTE-GEOFFREY-VALUE(state, C, β)
2:     v_G = −∞
3:     γ = −∞
4:     for each successor of state do
5:         (w_Ya, w_Yo, w_G) = EVALUATE-UTILITY(successor)
6:         if w_G > v_G then
7:             v_Ya = W_Ya
8:             v_Yo = W_Yo
9:             v_G = W_G
10:        end if
11:        if ???? then
12:            return (v_Ya, v_Yo, v_G)
13:        end if
14:        γ = max(γ, v_G)
15:    end for
16:    return (v_Ya, v_Yo, v_G)
17: end procedure
```

```
1: procedure COMPUTE-YOSHUA-VALUE(state, C, α)
2:     v_Yo = −∞
3:     β = −∞
4:     for each successor of state do
5:         (w_Ya, w_Yo, w_G) = COMPUTE-GEOFFREY-VALUE(successor, C, β)
6:         if w_Yo > v_Yo then
7:             v_Ya = W_Ya
8:             v_Yo = W_Yo
9:             v_G = W_G
10:        end if
11:        if ???? then
12:            return (v_Ya, v_Yo, v_G)
13:        end if
14:        β = max(β, v_Yo)
15:    end for
16:    return (v_Ya, v_Yo, v_G)
17: end procedure
```

```
1: procedure COMPUTE-YANN-VALUE(state, C)
2:     v_Ya = −∞
3:     α = −∞
4:     for each successor of state do
5:         (w_Ya, w_Yo, w_G) = COMPUTE-YOSHUA-VALUE(successor, C, α)
6:         if w_Ya > v_Ya then
7:             v_Ya = W_Ya
8:             v_Yo = W_Yo
9:             v_G = W_G
10:        end if
11:        if ???? then
12:            return (v_Ya, v_Yo, v_G)
13:        end if
14:        α = max(α, v_Ya)
15:    end for
16:    return (v_Ya, v_Yo, v_G)
17: end procedure
```

Question 3 [4 points]

A small robot going under the name of WALL-E wanders around the University to find and collect waste on the campus. When it detects an interesting object on the ground, WALL-E computes and executes a sequence of instructions specific to the type of the object it has recognized. Unfortunately, random bugs introduced by the students who wrote its program may appear, which results in the sequence of instructions to be improperly executed and therefore in more frequent failures when trying to grasp an object. Consider the Boolean variables S (the grasp succeeds) and B (it has randomly reached a state in which its program causes the gripper to be buggy), as well as the multivalued variables P (the sequence of pick up instructions), I (the image recorded by its camera) and O (the actual object to pick up).

(a) Given the following factorization of the joint distribution,

$$P(B, I, O, P, S) = P(B)P(I|O)P(O)P(P|I)P(S|B, P, O),$$

draw the most sparse Bayesian network that correctly captures the independence assumptions.

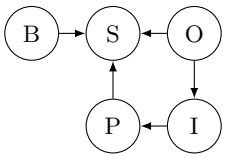


Figure 1: WALL-E Bayesian Network

(b) From the topology of your network, determine which of the following is true, false, or cannot be determined. From the topology of a Bayesian Network we can only determine independencies (not dependencies). Applying the d-separation algorithm we easily find:

i. $I \perp B$ - True

ii. $O \perp S$ - Cannot Be Determined

iii. $P \perp S|I$ - Cannot Be Determined

iv. $O \perp S|\{B, P\}$ - Cannot Be Determined

v. $I \perp B|S$ - Cannot Be Determined

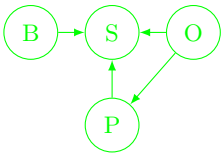
vi. $S \perp I|B$ - Cannot Be Determined

(c) Assume there are only two types of objects, bottles and boxes, each with their own and appropriate pickup sequence. Under normal conditions, if the appropriate pickup sequence is executed, then the probabilities of success are x_1 and x_2 respectively for bottles and boxes, and y_1 and y_2 when WALL-E attempts to execute the wrong pickup sequences. When WALL-E is in a state that causes bugs, all probabilities of success are halved. Give the conditional probability table associated with S.

O	Box		Bottle	
P	Box	Bottle	Box	Bottle
$P(S = True\|O, P, B = True)$	x_1	y_1	y_2	x_2
$P(S = True\|O, P, B = False)$	$\frac{x_1}{2}$	$\frac{y_1}{2}$	$\frac{y_2}{2}$	$\frac{x_2}{2}$

Table 1: CPT

(d) Assume the expected accuracy of the convolutional network used to recognize an object from images of the camera is 90% for bottles and 80% for boxes. Explain how and why you can make use of this assumption to simplify your Bayesian network. We can use the convolutional network to decide which sequence to execute and so remove the variable I from the Bayesian Network. This gives the following net:



We have the following CPT for $P|O$:

O	Box	Bottle
$P(P = Box\|O)$	0.8	0.1
$P(P = Bottle\|O)$	0.2	0.9

(e) Using this simplified Bayesian network, calculate an expression for the probability that the object is actually a box, given a successful grasp and a recognition of the object as a box.

$$P(O = Box|S = T, P = Box) = \frac{\sum_{b \in \{Box, Bottle\}} P(S = T, B = b, O = Box, P = Box)}{\sum_{o \in \{Box, Bottle\}} \sum_{b \in \{Box, Bottle\}} P(S = T, B = b, O = o, P = Box)}, \quad (1)$$

where $P(S = s, B = b, O = o, P = p) = P(B = b)P(O = o)P(P = p|O = o)P(S = s|B = b, P = p, O = o)$.

Question 4 [4 points]

You decide to start your own game company after graduating. Your first game starts with two ghosts, Greeny and Blinky. On each turn, the player clicks on a ghost. Clicking on any Greeny (action a_G) produces a new Greeny with probability $(2g + b)/(2(g + b))$ and a new Blinky otherwise, whereas clicking on any Blinky (action a_B) produces a new Blinky with probability $(g + 2b)/(2(g + b))$ and a new Greeny otherwise, where g and b are the numbers of Greenies and Blinkies in the current state. There is a reward +1 for producing an offspring of the same kind. The game ends after T steps.

(a) Sometimes MDPs are formulated with a reward function $R(s, a, s')$ defined as the reward obtained by performing the action a from state s and resulting in state s'. Write the Bellman equations for this formulation. $V^*(s) = \max_a \sum_{s'} P(s'|a, s)(R(s, a, s') + \gamma V^*(s'))$

(b) Assuming the formalism above, define the game as an MDP, with a minimal state space.

- A state $s := (N_G, N_B) \in \mathbb{Z}^2$ where the components respectively counts the number of Greeny and the number of Blinky. Another possibility is to formalize the state as a pair that contains the number of Blinky and the total number of ghosts.
- The action $a \in$ G, B.
- The transition probabilities: $P(s' = (N'_G, N'_B)|s = (N_G, N_B), a)$
  - $= \frac{2N_G+N_B}{2N_G+2N_B}$ if $a = G$ and $N'_G = N_G + 1$ and $N'_B = N_B$
  - $= \frac{N_B}{2N_G+2N_B}$ if $a = G$ and $N'_G = N_G$ and $N'_B = N_B + 1$
  - $= \frac{N_G+2N_B}{2N_G+2N_B}$ if $a = B$ and $N'_G = N_G$ and $N'_B = N_B + 1$
  - $= \frac{N_G}{2N_G+2N_B}$ if $a = B$ and $N'_G = N_G + 1$ and $N'_B = N_B$
  - $= 0$ else.
- Reward $R(s', a, s) = 1$ if $[N'_G = N_G + 1 \quad \& \quad a = G] \quad || \quad [N'_B = N_B + 1 \quad \& \quad a = B]$, else 0

(c) Count the total number of states in the MDP. The total number of states is equal to $\sum_{i=1}^{T+1} = \frac{(T+1)(T+2)}{2}$

(d) For $\gamma = 0.9$, $T = 3$ and $V_0(s) = 0$ for all s, run the Value Iteration algorithm for 3 iterations. First let us notice that the optimal policy will be to play the most present ghost, that is G if $N_G > N_B$ and else B. Then we see that by symmetry the states $s = (a, b)$ is equivalent to the state $s' = (b, a)$. Then using the recursion $V^i(s) = \max_a \sum_{s'} P(s'|a, s)(R(s, a, s') + \gamma V^*(s'))$, we should eventually find the following sates' values:

s	$(1,1)$	$(1,2)$	$(2,2)$	$(1,3)$	$(2,3)$	$(1,4)$
$V^0(s)$	0	0	0	0	0	0
$V^1(s)$	$\frac{3}{4}$	$\frac{5}{6}$	$\frac{3}{4}$	$\frac{7}{8}$	$\frac{4}{5}$	$\frac{9}{10}$
$V^2(s)$	$\frac{3}{2}$	$\frac{769}{480}$	$\frac{147}{100}$	$\frac{1339}{800}$	$\frac{4}{5}$	$\frac{9}{10}$
$V^3(s)$	≈ 2.19	≈ 2.30	$\frac{147}{100}$	$\frac{1339}{800}$	$\frac{4}{5}$	$\frac{9}{10}$

(e) Explain why Value Iteration applied to this problem converges in $O(T)$ iterations. Because the time horizon is finite and each state is only reachable after a unique number of play. Thus, after i iterations the value functions of the states that requires to play $T - i$ times to be reached are exactly estimated.

Question 5 [4 points]

You observe a Grandmaster agent playing Pacman. How can you use the moves you observe to train your own agent?

(a) Describe formally the data you would collect, the inference problem you would consider, and how you would solve it. Ideally, I would collect the information about the action performed by the Grandmaster, the reward obtained as well as some features to define a state (e.g. the game frames). I would formalize the problem as finding the policy played by the Grandmaster. I would use temporal difference Q learning to learn the state-action value function of this policy and I would then extract the policy from it when required.

Alternatively I could formalize the problem as a supervised classification problem in which we aim at learning a model that predicts the action given the state.

(b) How would you design a neural network to control your agent? Define mathematically the neural network architecture, its inputs, its outputs, its parameters, as well as the loss you would use to train it. I would use a convolutional neural network to model the value function. Its inputs would the frame and its outputs would be a vector where each component is the corresponding state-action value. The parameters would be the weights of the network. To train it I would use a quadratic loss as when performing Q-value learning. Another possibility would be to use a simple multi layer perceptron if we have extracted meaningful features from the frame.

Alternatively the output of the network should correspond to the probability of each action, to do that I would use a softmax output layer and a cross-entropy loss.

(c) Discuss the expected performance of the resulting agent when (i) the Grandmaster agent is optimal, and (ii) the Grandmaster agent is suboptimal. Assuming a very large amount of data we can assume that we may be able to learn exactly the Grandmaster policy. If it is optimal we would expect our agent to be as well. Similarly it would be suboptimal if the Grandmaster is not so "grand".