CS 188 Fall 2018

Introduction to Artificial Intelligence

Written HW 11 Sol.

Self-assessment due: Monday 5/6/2019 at 11:59pm (submit via Gradescope)

For the self assessment, **fill in the self assessment boxes in your original submission** (you can download a PDF copy of your submission from Gradescope – be sure to delete any extra title pages that Gradescope attaches). For each subpart where your original answer was correct, write “correct.” Otherwise, write and explain the correct answer. **Do not leave any boxes empty.** If you did not submit the homework (or skipped some questions) but wish to receive credit for the self-assessment, we ask that you first complete the homework without looking at the solutions, and then perform the self assessment afterwards.

Q1. The OMNIBUS

(a) Search

(i) [*true* or *false*] Uniform-cost search will never expand more nodes than A*-search.

(ii) [*true* or *false*] Depth-first search will always expand more nodes than breadth-first search.

(iii) [*true* or *false*] The heuristic $h(n) = 0$ is admissible for every search problem.

(iv) [*true* or *false*] The heuristic $h(n) = 1$ is admissible for every search problem.

(v) [*true* or *false*] The heuristic $h(n) = c(n)$, where $c(n)$ is the true cheapest cost to get from the node n to a goal state, is admissible for every search problem.

(b) CSPs

(i) [*true* or *false*] The most-constrained variable heuristic provides a way to select the next variable to assign in a backtracking search for solving a CSP.

(ii) [*true* or *false*] By using the most-constrained variable heuristic and the least-constraining value heuristic we can solve every CSP in time linear in the number of variables.

(c) Games

(i) [*true* or *false*] When using alpha-beta pruning, it is possible to get an incorrect value at the root node by choosing a bad ordering when expanding children.

(ii) [*true* or *false*] When using alpha-beta pruning, the computational savings are independent of the order in which children are expanded.

(iii) [*true* or *false*] When using expectimax to compute a policy, re-scaling the values of all the leaf nodes by multiplying them all with 10 can result in a different policy being optimal.

(d) MDPs For this question, assume that the MDP has a finite number of states.

(i) [*true* or *false*] For an MDP (S, A, T, γ, R) if we only change the reward function R the optimal policy is guaranteed to remain the same.

(ii) [*true* or *false*] Value iteration is guaranteed to converge if the discount factor (γ) satisfies $0 < \gamma < 1$.

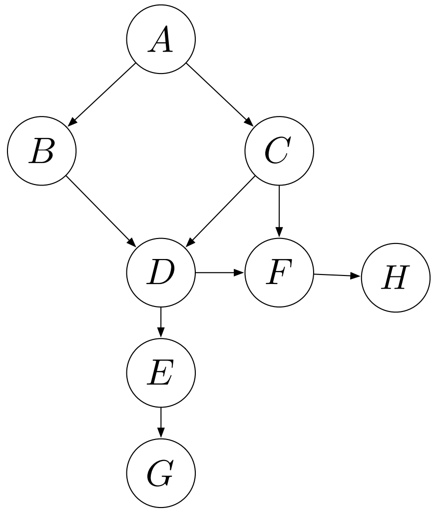
(iii) [*true* or *false*] Policies found by value iteration are superior to policies found by policy iteration.

(e) Reinforcement Learning

(i) [*true* or *false*] Q-learning can learn the optimal Q-function Q^* without ever executing the optimal policy.

(ii) [*true* or *false*] If an MDP has a transition model T that assigns non-zero probability for all triples $T(s, a, s')$ then Q-learning will fail.

(f) Bayes' Nets For each of the conditional independence assertions given below, circle whether they are guaranteed to be true, guaranteed to be false, or cannot be determined for the given Bayes' net.



$B \perp\!\!\!\perp C$	Guaranteed true	Guaranteed false	Cannot be determined
$B \perp\!\!\!\perp C \mid G$	Guaranteed true	Guaranteed false	Cannot be determined
$B \perp\!\!\!\perp C \mid H$	Guaranteed true	Guaranteed false	Cannot be determined
$A \perp\!\!\!\perp D \mid G$	Guaranteed true	Guaranteed false	Cannot be determined
$A \perp\!\!\!\perp D \mid H$	Guaranteed true	Guaranteed false	Cannot be determined
$B \perp\!\!\!\perp C \mid A, F$	Guaranteed true	Guaranteed false	Cannot be determined
$F \perp\!\!\!\perp B \mid D, A$	Guaranteed true	Guaranteed false	Cannot be determined
$F \perp\!\!\!\perp B \mid D, C$	Guaranteed true	Guaranteed false	Cannot be determined

Q2. Perceptron

(a) Suppose you have a binary perceptron in 2D with weight vector $\mathbf{w} = r\ [w_1, w_2]^T$. You are given w_1 and w_2, and are given that $r > 0$, but otherwise not told what r is. *Assume that ties are broken as positive.*
Can you determine the perceptron's classification of a new example x with known feature vector $f(x)$?
● Always ○ Sometimes ○ Never

(b) Now you are learning a multi-class perceptron between 4 classes. The weight vectors are currently $[1, 0]^T$, $[0, 1]^T$, $[-1, 0]^T$, $[0, -1]^T$ for the classes A, B, C, and D. The next training example x has a **label of A** and feature vector $f(x)$.
For the following questions, *do not make any assumptions about tie-breaking.* (Do not write down a solution that creates a tie.)

(i) Write down a feature vector in which no weight vectors will be updated.

$$f(x) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$ ○ Not possible

Any feature vector that points in the direction of w_A more than any other direction, such that $w_A \cdot f(x) > w_i \cdot f(x)$ for $i \neq A$.

(ii) Write down a feature vector in which **only** $\mathbf{w}_A$ will be updated by the perceptron.

$$f(x) = \begin{bmatrix} \ \end{bmatrix}$$ ● Not possible

(iii) Write down a feature vector in which **only** $\mathbf{w}_A$ and $\mathbf{w}_B$ will be updated by the perceptron.

$$f(x) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$ ○ Not possible

Any feature vector that points in the direction of w_B more than any other direction, such that $w_B \cdot f(x) > w_i \cdot f(x)$ for $i \neq B$.

(iv) Write down a feature vector in which **only** $\mathbf{w}_A$ and $\mathbf{w}_C$ will be updated by the perceptron.

$$f(x) = \begin{bmatrix} -1 \\ 0 \end{bmatrix}$$ ○ Not possible

Any feature vector that points in the direction of w_C more than any other direction, such that $w_C \cdot f(x) > w_i \cdot f(x)$ for $i \neq C$.

The weight vectors are the same as before, but now there is a bias feature with value of 1 for all x and the weight of this bias feature is $0, -2, 1, -1$ for classes A, B, C, and D respectively. As before, the next training example x has a **label of A** and a feature vector $f(x)$. The always "1" bias feature is the first entry in $f(x)$.

(v) Write down a feature vector in which **only** $\mathbf{w}_B$ and $\mathbf{w}_C$ will be updated by the perceptron.

$$f(x) = \begin{bmatrix} 1 \\ \\ \end{bmatrix}$$ ● Not possible

(vi) Write down a feature vector in which **only** $\mathbf{w}_A$ and $\mathbf{w}_C$ will be updated by the perceptron.

$$f(x) = \begin{bmatrix} 1 \\ 0 \\ \\ 0 \end{bmatrix}$$ ○ Not possible

Any feature vector that points in the direction of w_C more than any other direction, such that $w_C \cdot f(x) > w_i \cdot f(x)$ for $i \neq C$.