Midterm Examination
CS540: Introduction to Artificial Intelligence
July 14, 2016

LAST NAME: ____________________________________________

FIRST NAME: ____________________________________________

<table>
<thead>
<tr>
<th>Problem</th>
<th>Score</th>
<th>Max Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>_______</td>
<td>18</td>
</tr>
<tr>
<td>2</td>
<td>_______</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>_______</td>
<td>13</td>
</tr>
<tr>
<td>4</td>
<td>_______</td>
<td>9</td>
</tr>
<tr>
<td>5</td>
<td>_______</td>
<td>15</td>
</tr>
<tr>
<td>6</td>
<td>_______</td>
<td>8</td>
</tr>
<tr>
<td>7</td>
<td>_______</td>
<td>13</td>
</tr>
<tr>
<td>8</td>
<td>_______</td>
<td>15</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>_______</td>
<td><strong>100</strong></td>
</tr>
</tbody>
</table>
Question 1. [18] Search

(a) [3] Which uninformed search algorithm should you use if you want to use the smallest amount of space and also guarantee finding an optimal solution, assuming all actions have cost 1 and the goal is at an unknown depth?

(b) [2] True or False: If we use an admissible but not consistent heuristic with A* graph search, then when a node is expanded and put on Explored, we can guarantee that we have already reached that node’s state via the minimum-cost path from the start state.

(c) [2] True or False: If \( h_1 \) and \( h_2 \) are both admissible heuristics, then their average, \( h_3(n) = \frac{1}{2} h_1(n) + \frac{1}{2} h_2(n) \), must also be admissible.

(d) [2] True or False: If we know there exists a non-optimal solution with cost \( C \), then any node generated by the A* algorithm that has \( f(n) > C \) does not need to be put on Frontier (i.e., it can be thrown away) and A* will still find an optimal solution.

(e) [9] Say we define an evaluation function for a heuristic search problem as

\[
f(n) = (w \cdot g(n)) + ((1 - w) \cdot h(n))
\]

where \( g(n) \) is the cost of the best path found from the start state to state \( n \), \( h(n) \) is an admissible heuristic function that estimates the cost of a path from \( n \) to a goal state, and \( 0.0 \leq w \leq 1.0 \). What search algorithm do you get when:

(i) [3] \( w = 0.0 \)

(ii) [3] \( w = 0.5 \)

(iii) [3] \( w = 1.0 \)
Question 2. [9] Simulated Annealing

(a) [2] True or False: Simulated Annealing can escape local optima.

(b) [2] True or False: Simulated Annealing with a constant, positive temperature at all times is the same as Hill-Climbing.

(c) [3] What kind of search does Simulated Annealing do (approximately) if the temperature is very large (i.e., close to $\infty$) at every iteration? Select one (1) of the following.

   (i) It will halt immediately and do no search
   (ii) Breadth-First search
   (iii) A random walk
   (iv) Hill-Climbing

(d) [2] True or False: Simulated Annealing with a linearly decreasing temperature is guaranteed to converge to a *global optimal solution* after a finite number of iterations.

Consider a 2-person, non-zero-sum game in which players A and B alternate turns. Instead of using a single static board evaluation (SBE) function, there are two distinct SBEs, one for player A, \( f_A \), and a different one for player B, \( f_B \). Both SBEs are known to both players. Each function indicates the estimated value of a board position with respect to that player, with \textit{the larger the value meaning the better that position looks for the given player}. For example, \( f_B(p) = -5 \) means that position \( p \) is not good for player B, whereas \( f_B(p) = 100 \) means that position \( p \) looks very good for player B. We want to modify the Minimax algorithm to work for this game. At each leaf node, \( n \), at the cutoff depth we compute a \textit{pair of values} \((f_A(n), f_B(n))\), giving the value of that state from each player’s viewpoint. HINT: The backed-up value from a child to its parent corresponds to a single best move from the parent to one child.

(a) [5] What are the \textit{backed-up pairs of values} computed for the following game tree, where the root (at depth 0) corresponds to player A’s turn, nodes at depth 1 are positions where it’s player B’s turn, etc. Show your answers by adding them to the tree below.

(b) [1] What \textit{move} should player A make based on your result from (a)? Show your answer by darkening the selected arc.

The following three questions do not pertain to the above game tree.

(c) [2] True or False: The Minimax algorithm using SBE function \( f_1 \) is guaranteed to always choose the same next move as the Minimax algorithm using SBE function \( f_2 \) when \( f_2(n) = f_1(n) + c \), \( n \) is an arbitrary state in a game tree, and \( c \) is a real-valued constant.
(d) [2] Which one (1) of the following best describes when/why the **horizon effect** occurs with the Minimax algorithm:
   (i) large branching factors at nodes
   (ii) having a depth bound on the search
   (iii) inadequate utility function
   (iv) inadequate state representation

(e) [3] For games that have a very large search space with very large branching factor and use a relatively weak evaluation function, what search method can best attain human expert-level win performance?

**Question 4. [9] Hierarchical Agglomerative Clustering**

Consider a dataset containing 6 one-dimensional feature points with values: {2, 4, 7, 8, 12, 14}. After the first three iterations of Hierarchical Agglomerative Clustering using Euclidean distance between points we get the three clusters: {2, 4}, {7, 8}, and {12, 14}.

(a) [6] What is the distance between each pair of these three clusters using
   (i) [3] Single Linkage
   (ii) [3] Complete Linkage

(b) [3] What are the clusters formed after the next iteration using *Single Linkage*?
Question 5. [15] Constraint Satisfaction

Consider the problem of coloring the six regions (numbered 1…6) in the following map using three colors: R, G, and B, so that no adjacent regions have the same color. Two regions are adjacent if they share part of an edge (note: they are NOT adjacent if they only share a corner).

(a) [5] If initially every variable has all three possible values except region 1 has known value R and region 2 has known value G, what is the result of the Forward Checking algorithm? Do not generate a search tree; just do Forward Checking deletions.

1 = {R}, 2 = {G}, 3 = {}, 4 = {}, 5 = {}, 6 = {}

(b) [3] In general (i.e., not just the map problem in (a)), when using Forward Checking combined with backtracking search, if we ever reach a node in the search tree where there is a variable with no remaining values, what does this mean?

(c) [5] Assume the initial domains of the regions in the map above are given as 1={R,G,B}, 2={R,G}, 3={R,G,B}, 4={R}, 5={R,G,B}, and 6={R}. What is the result of applying the Arc Consistency algorithm, AC-3? Do not generate a search tree; just do AC-3 deletions.

1 = {}, 2 = {}, 3 = {}, 4 = {}, 5 = {}, 6 = {}

(d) [2] True or False: In general (i.e., not just the map problem in (c)), the AC-3 algorithm combined with backtracking search guarantees finding a solution, assuming at least one solution exists.
Question 6. [8] k-Means Clustering

(a) [2] True or False: k-Means Clustering is guaranteed to converge to a clustering that is a local minimum of the distortion measure of cluster quality (i.e., for a given value of $k$, the sum of squared distances of each data point to its cluster center is a local minimum).

(b) [2] True or False: Given a linearly-separable dataset with one group containing 5 points and a second group containing 20 points, $k$-Means Clustering with $k = 2$ is guaranteed to find these two groups as two clusters.

(c) [2] True or False: In $k$-Means Clustering it’s possible that a centroid may have no points assigned to it.

(d) [2] True or False: To find the best number of clusters, $k$, to use with $k$-Means Clustering for a given dataset, you should pick the value of $k$ that minimizes the distortion measure of cluster quality.

(a) [2] True or False: If $A$ is one of $B$'s $k$-nearest-neighbors for a given value of $k$, then $B$ must be one of $A$'s $k$-nearest-neighbors.

(b) [2] True or False: k-NN classification results may change if we change the value of each feature by multiplying it by 0.1 for every example in both the training set and the testing set, assuming we use Euclidean distance to compute nearest neighbors.

(c) [9] Consider the following training set consisting of 6 points in class 1 (the squares) and 5 points in class 2 (the triangles). Each point has two features $(x, y)$ corresponding to its 2D coordinates. A new point (the circle) is to be classified using $k$-NN and Euclidean distance.

(i) [3] When $k=3$, how should the new point be classified?

(ii) [3] When $k=5$, how should the new point be classified?

(iii) [3] When $k=1$, draw on the figure below the approximate decision boundary that separates the two classes.

(a) [8] Starbucks researchers have collected the following data about how some customers like their coffee. There are two binary flavor attributes, *Hazelnut* and *Vanilla*, indicating if that flavor was added (Y) to the coffee or not (N). And there is one 3-valued *Roast* attribute, indicating whether the roast type is Light (L), Medium (M) or Dark (D). The class variable is *Likes*. Use \( \log 0.1 = -3.32, \log 0.2 = -2.32, \log 0.3 = -1.74, \log 0.33 = -1.59, \log 0.4 = -1.32, \log 0.45 = -1.15, \log 0.5 = -1.0, \log 0.55 = -0.86, \log 0.6 = -0.74, \log 0.67 = -0.58, \log 0.7 = -0.51, \text{and} \log 0.8 = -0.32, \log 0.9 = -0.15, \text{and} \log 1 = 0, \) where all logs are to base 2. There are 10 training examples:

<table>
<thead>
<tr>
<th>Hazelnut</th>
<th>Vanilla</th>
<th>Roast</th>
<th>Likes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>Y</td>
<td>L</td>
<td>Y</td>
</tr>
<tr>
<td>Y</td>
<td>Y</td>
<td>M</td>
<td>Y</td>
</tr>
<tr>
<td>Y</td>
<td>Y</td>
<td>D</td>
<td>Y</td>
</tr>
<tr>
<td>Y</td>
<td>Y</td>
<td>L</td>
<td>Y</td>
</tr>
<tr>
<td>Y</td>
<td>N</td>
<td>M</td>
<td>N</td>
</tr>
<tr>
<td>Y</td>
<td>N</td>
<td>D</td>
<td>N</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>L</td>
<td>N</td>
</tr>
<tr>
<td>N</td>
<td>Y</td>
<td>M</td>
<td>N</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>D</td>
<td>Y</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>L</td>
<td>Y</td>
</tr>
</tbody>
</table>

(i) [4] What is the entropy, \( H(\text{Likes}) \), of the class variable, *Likes*, for this set of examples? 

(ii) [4] What is the conditional entropy \( H(\text{Likes} \mid \text{Hazelnut}) \)?
(b) [3] In general, what search strategy is used by the decision tree pruning algorithm presented in class that uses a tuning set, where the search is defined by the succession of trees produced by the algorithm?

(i) A*
(ii) Breadth-First search
(iii) Depth-First search
(iv) Hill-Climbing

(c) [2] True or False: In general, $I(A; B) = I(B; A)$ for any two variables (attribute or class) and any training set.

(d) [2] True or False: Given a binary attribute, $A$, that splits a set of examples, $E$, into two nonempty subsets, $E_1$ and $E_2$ such that the $i^{th}$ child, $i = 1, 2$, has $p_i$ positive examples and $n_i$ negative examples associated with a binary class variable $C$, it is possible for $I(C; A) = 0$. 