Artificial Intelligence Qual Exam

Fall 2016

ID:

Note: This is exam is closed book. You will not need a calculator and should not use a calculator or any other electronic devices during the exam.
1 Search (20 points)

For this problem, we will consider three search algorithms, BFS, DFS, and IDDFS (Iterative Deepening Depth First Search), with nodes that are pushed onto the queue in left to right order. We will use $b \geq 1$ for the branching factor of the search tree and $d$ for the depth of the goal. We will assume that the branching factor is uniform, i.e., it is never less than $b$ anywhere. The location of the goal at depth $d$ will be specified by the $1 \leq g \leq b^d$. For example, in the tree below, we have $d = 2$, $b = 2^d - 1 = 2$, and $g = 3$, and the goal would be found before the node labeled never is popped off the queue by any of the three algorithms we're considering.

For each question below, you must provide some justification for your answer to receive full credit.

1.1 Worst Case Time Complexity (5 points)

Provide values of $b$ and $g$ for which all three algorithms have the same asymptotic time complexity in $d$. $g$ should be expressed as a function of $b$ and $d$. 
1.2 Best Case Time Complexity (5 points)
Provide values of $b$ and $g$ for which DFS has polynomial time complexity, but BFS and IDDFS have exponential time complexity in $d$. $g$ should be expressed as a function of $b$ and $d$.

1.3 Tricky Time Complexity (5 points)
Provide values of $b$ and $g$ for which IDDFS has quadratic time complexity (with respect to $d$) while BFS and DFS have linear complexity. $g$ should be expressed as a function of $b$ and $d$.

1.4 Space Complexity (5 points)
Provide values of $b$ and $g$ for which DFS and IDDFS have polynomial space complexity in $d$, but BFS has exponential space complexity in $d$. $g$ should be expressed as a function of $b$ and $d$. 


2 Games (20 points)

For this question, we will consider a perfect information, zero-sum, 2-player, alternating move game with a max node at the root. Assume that nodes are expanded depth-first, in left-to-right order.

2.1 Minimax Values (6 points)

In the search tree below, suppose the root node takes value $D$. Label the other nodes with their minimax value, taking into account that the root has value $D$. In some cases, you may need to use a mathematical expression, e.g., $\max(G, H)$, instead of a specific value because there isn’t enough information to determine a specific value for the node. Your answer should be as specific as possible given your knowledge that the root takes on value $D$. (Note that in a slight abuse of notation, we are using the letters $A \ldots G$ to represent both the leaf nodes in the tree and the values of those nodes.)
2.2 Pruning I (7 points)

Specify a condition on nodes E and F that will suffice for nodes G and H to be pruned by alpha-beta pruning.

2.3 Pruning II (7 points)

Assume that nodes E, F, and G have been visited under alpha-beta pruning. Specify a condition that would suffice for node H to be pruned.
3 Bayesian Networks (20 points)

For this problem, we will follow the standard convention of using capital letters for random variables and lower case letters for the values these variables take on. For example, the binary, random variable $A$ can take on values $a$ and $\bar{a}$. We will also use the standard convention of using $P(a)$ as shorthand for $P(A = a)$.

The Bayesian network shown below is defined over binary random variables:

![Bayesian Network Diagram]

3.1 CPTs (5 points)

What conditional probabilities are needed to specify the parameters of this network? You should write down one expression for each conditional probability table that must be stored, e.g., $P(X|Y Z)$. 
3.2 CPTs (5 points)

How many numbers must be stored to represent the conditional probabilities? Your answer should exclude conditional probabilities that are uniquely determined by others.

3.3 Marginalization (5 points)

Write an expression for $P(D)$ in terms of the CPTs.

3.4 Marginalization (5 points)

If you have not already done so in your previous answer, use the distributive law to write your expression for $P(D)$ in way that minimizes the amount of computation required to compute $P(D)$ by summing out some variables before others. You should also eliminate any unnecessary computations, if possible.
4 Markov Chains (20 points)

For this problem, we will use $P(S^t)$ for the probability distribution over the state at time $t$, and $s_i$ to indicate state $i$. For example, $P(S^2 = s_5)$ is the probability that we are in state 5 at time step 2.

4.1 Markov Property (6 points)

Write down a simplified expression for $P(S^t|S^{t-1} \ldots S^0)$ that must hold if the Markov property holds.

4.2 Matrix Representation (7 points)

Suppose we represent $P(S^t)$ as a row vector and we have a matrix $A$ such that $P(S^t) = P(S^{t-1})A$ for all $t$. Write a non-trivial, generic expression for the $A$ matrix by providing an expression for $a_{ij}$, the entry in row $i$ and column $j$ of $A$. 
4.3 Stationary Distribution (7 points)

Some Markov chains have the nice property that no matter what the initial distribution over states is, the state distribution will converge (as $t \to \infty$) to the same distribution over states. This is called a stationary distribution. Write down an expression that the stationary distribution must satisfy using the notation from question 4.2.
5 MDPs (20 points)

Consider an MDP for a simplified problem of holding or selling a stock. Think of this problem as answering the question of whether it’s worth holding on to a low-valued stock in hopes that it will eventually get a higher value.

The MDP has two states that we care about, corresponding to whether the stock is valued **high** or **low**. Each of these states has two actions, **sell** or **hold**. If you hold when the stock is high, there is a 50% chance that it will stay high, and a 50% chance that it will transition to low. If you hold when the stock is low, there is an 80% chance that it will stay low, and a 20% chance that it will transition to high. If you sell in the high state, you get an immediate payoff of $100 and then transition to an absorbing state. By convention, an absorbing state has its value fixed at 0. Selling in the low state has an immediate payoff of $50, then makes a transition to an absorbing state. Note that as in all MDPs, we do not apply the discount to the immediate reward; it applies to the value of future states only.

5.1 Optimal Policy and Value function (10 points)

Assume a discount factor of 0.8, then compute the optimal policy and value for the high and low states under this policy. Show your work. Hint: This problem was set up to be easy if you use fractions.
5.2 Discounting (10 points)

Provide a value of the discount factor that would cause the optimal policy to change, and justify why you have picked that choice.