Midterm Examination

CS 540: Introduction to Artificial Intelligence

March 14, 2007

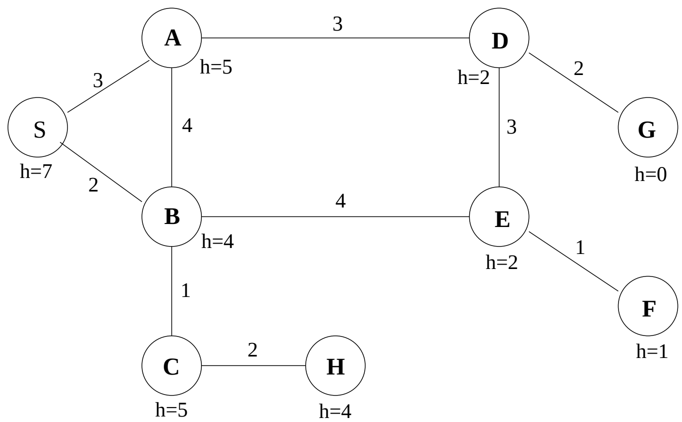
LAST (FAMILY) NAME: ______________________________

FIRST NAME: ______________________________

Problem	Score	Max Score
1	______	15
2	______	12
3	______	10
4	______	10
5	______	16
6	______	12
7	______	15
8	______	10
Total	______	100

1. [15] **Search Methods**

Consider the following *state space graph* where the arcs represent the legal successors of a node, and all arcs are bi-directional, meaning the successor function can be applied from either node. The cost of moving to a successor node is given by the number on the arc. The value of a heuristic evaluation function, *h*, if computed at a state, is shown along side each node. The start state is **S** and the goal is **G**.



When a node is expanded, assume its children are put in the NODES list in alphabetical order so that the child closest to the front of the alphabet is removed before its other siblings (for all uninformed searches and for ties between siblings in informed searches). Do *not* generate a child node if that same node is an *ancestor* of the current node *in the search tree*. When selecting a node from NODES, in case of ties between non-siblings, use FIFO order to select the node that has been in NODES longest.

Give the sequence of nodes as they are **removed** from the NODES list (for expansion or before halting at the goal) for each of the following search methods.

(a) [5] Uniform-Cost search

(b) [5] A* heuristic search

(c) [5] Greedy Best-First search

2. [12] **Search: Short Answer Questions**

(a) [3] Under what general conditions will A* search expand the same nodes as Breadth-First search? (Do not consider issues related to ties.)

(b) [2] True or False: Greedy Best-First search with $h(n) = 0$ for all nodes n is guaranteed to find an optimal solution if all arc costs are 1.

(c) [2] True or False: In a finite search space containing no goal state, Algorithm A* will always explore all states.

(d) [3] Say we have a search space that is very large with a very large branching factor at most nodes, there may be infinite paths in the search space, and we have no heuristic function. What search method would be good to use in this situation and why?

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

3. [10] **Constraint Satisfaction**

Consider the problem of assigning colors to the five squares on board below such that horizontally adjacent and vertically adjacent squares do not have the same color. Assume there are possible two colors, red (R) and black (B). Formulated as a constraint satisfaction problem, there are five variables (the squares) and two possible values (R, B) for each variable.

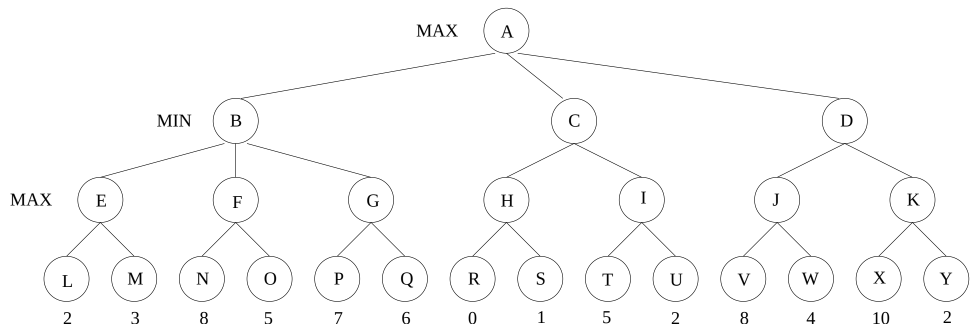
1	2	3
4	5	

(a) [3] If initially every variable has both possible values and we then assign variable 1 to have value R, what is the result of the Forward Checking algorithm?

(b) [3] If initially every variable has both possible values and the arc consistency procedure is run, what are the resulting domains for each of the variables?

(c) [4] If initially every variable has both possible values and we then assign variable 5 to have value B, what is the result of the Arc Consistency algorithm?

4. [10] **Adversarial Search**
Consider the following game tree in which the root corresponds to a MAX node and the values of a static evaluation function, if applied, are given at the leaves.



(a) [2] What is the minimax value computed at the root node for this tree?

(b) [1] What move should MAX choose?

(c) [3] Which nodes are *not* examined when **Alpha-Beta Pruning** is performed? Assume children are visited left to right.

(d) [4] Is there a different ordering for the children of the root for which *more* pruning would result by Alpha-Beta? If so, state the order. If not, say why not.

5. [16] **Decision Trees**

(a) [5] Consider a domain in which there are two attributes, X and Y, each with three possible values, 1,2,3. The concept to be learned is $X \geq Y$. (E.g., if X=3 and Y=1, then the output is true.) Draw a decision tree that represents this concept (assume your training set contains all possible combinations of values).

(b) [3] How is the problem of overfitting the training data exhibited when constructing a decision tree?

(c) [8] Say we build a decision tree by adding nodes until there is one or a small number of examples at a node, which we then make a leaf. Then we prune the tree by deleting leaves until the "score" of the tree starts to get worse. The question is, how to score each possible pruning of the tree? For each of the following possible definitions of a scoring method, explain whether or not it would be a good idea to use it, and why or why not.

(i) The score is the percentage correct classification using the tree on the training set used to build the tree.

(ii) The score is the percentage correct classification using the tree on a separate set of examples from the training set used to build the tree.

6. [12] **Propositional Logic**

(a) [3] Define **soundness** in terms of Propositional Logic (PL) sentences α and β, and $\models$ and $\vdash$.

(b) [3] Is the PL sentence $((P \Rightarrow Q) \wedge Q) \Rightarrow P$ valid, unsatisfiable or satisfiable? Justify (i.e., prove) your answer.

(c) [6] Given two arbitrary sentences α and β in PL, $\alpha \models \beta$ if and only if

(i) [2] the sentence $\alpha \Rightarrow \beta$ is ________________________________

(ii) [2] the sentence $\alpha \wedge \neg\beta$ is ________________________________

(iii) [2] $\alpha \wedge \neg\beta \vdash$ ________________________________

7. [15] **Deductive Inference in Propositional Logic**
Consider the following 4 premise sentences in PL:

$P \Rightarrow (R \vee S)$, $\neg P \Rightarrow (R \vee S)$, $\neg S$, $(R \vee U) \Rightarrow T$

(a) [4] Convert the premise sentences into conjunctive normal form (CNF) and show the result as a set of **clauses**.

(b) [8] Prove the sentence T is true given the premises using the **Resolution Refutation algorithm**. Show your result as a proof tree.

(c) [3] Can the Resolution Refutation algorithm be used to show that a given sentence in PL is *not* true given a set of premise sentences in PL? If yes, explain how. If not, explain why not.

8. [10] **Representation in First-Order Logic**
For each of the following sentences in English, is the accompanying sentence in First-Order Logic a good translation? If yes, answer "yes." If no, explain why not and then give a correct answer.

(a) [5] No two people have the same social security number.

$\neg\, \exists x, y, n \;\; (IsPerson(x) \wedge IsPerson(y)) \Rightarrow (HasSS\#(x, n) \wedge HasSS\#(y, n))$

(b) [5] Everyone's social security number has nine digits.

$\forall x, n \;\; IsPerson(x) \Rightarrow (HasSS\#(x, n) \wedge NumDigits(n, 9))$