Midterm Examination

CS 540: Introduction to Artificial Intelligence

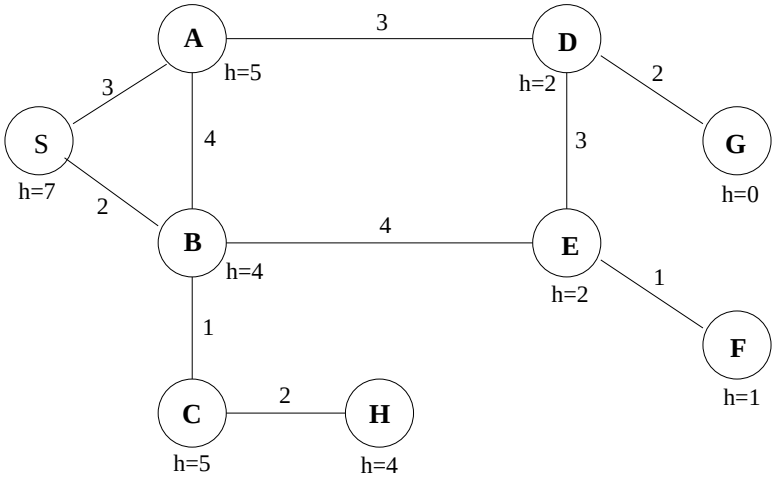
March 11, 2004

LAST (FAMILY) NAME: _____SOLUTIONS_____

FIRST NAME: _____

Problem	Score	Max Score
1	———	16
2	———	10
3	———	10
4	———	8
5	———	16
6	———	20
7	———	20
Total	———	100

1. [16] **Search Methods**
Consider the following *state space* where the arcs represent all of 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 each successor function 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) [4] Depth-First search (expand A before B)

SABCHEDG

(b) [4] A* heuristic search

SBACEDFG

(c) [4] Greedy Best-First search

SBEFDG

(d) [4] Hill-Climbing

SBEF

2. [10] **Search: Short Answer Questions**

(a) [2] True or False: The main difference between Simulated Annealing and conventional Hill-Climbing is that Simulated Annealing allows restarts from multiple, random initial states.

False

(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.

False

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

True

(d) [2] Search using evaluation function f(n) = –depth(n) corresponds to which search method?

Depth-First search

(e) [2] True or False: If it is known that all arc costs are greater than or equal to -1 (i.e., both positive and negative real-valued costs are allowed but the cost of an operator cannot be less than the fixed constant -1) and the state space does not contain cycles, uniform-cost search is still guaranteed to find an optimal solution.

False. Consider, for example, the state space where s --(3)--> g, s --(4)--> n1, n1 --(-1)--> n2, n2 --(-1)--> n3, n3 --(-1)--> n4, n4 --(-1)--> n5, n5 --(-1)--> g. UCS finds the non-optimal solution path s--g, which has cost 3, while the optimal path s--n1--n2--n3--n4--n5--g has cost -1.

3. [10] **Constraint Satisfaction**
The 8-Queens problem is to place 8 queens (Chess pieces) onto an 8×8 board so that no queen can "capture" (in the Chess sense) any other queen. In other words, no two queens are in the same row, same column, or same diagonal (note: there are two diagonal directions).

Setting this up as a constraint satisfaction problem, the variables (nodes) are the 8 queens. Assuming the *i*th queen must be located in the *i*th column means there are 8 possible values for each variable specifying the row number where that queen can be located. There are arcs in the constraint graph between every pair of nodes because the given constraints apply between every pair of queens. Say we have a state in the search for a solution in which 4 queens' positions have been uniquely determined as shown in the following figure.

			Q	F	F	F	F
				F	F		A
		Q		F	F	F	F
				A	A	F	
	Q			F	F	F	F
					F	A	A
Q				F	F	F	F
				F		A	F

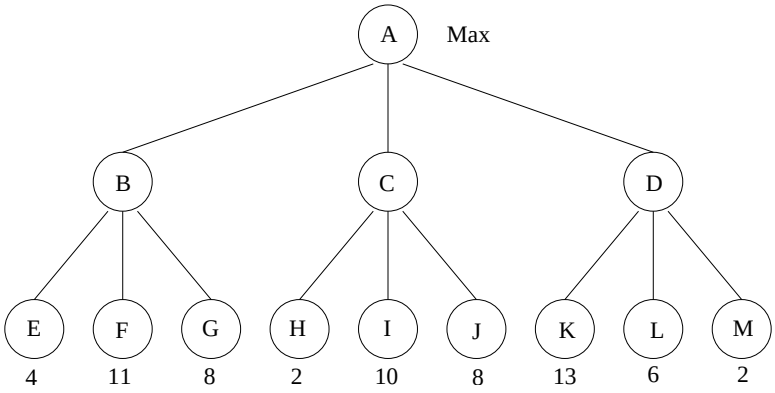
(a) [6] Using an "F," mark the positions in the last four columns that will be *eliminated* as possible positions for a queen by **forward checking**.

(b) [4] Using an "A," mark the positions in the last four columns that are *not* already marked with an "F" and that will be *eliminated* by **arc consistency**. (Use this process repeatedly until no inconsistencies remain.)

First, the A at (row=4,column=6) is added first because it is not consistent with any of the possible remaining positions for the queen in the 8th column. Consequently, the only remaining position for the queen in column 6 is (8,6). This implies A's must be added at (8,7) and (6,8) because they are not consistent with a queen at (8,6). Now (4,5) must be an A because it is not consistent with the two possible remaining positions in column 7. This implies a queen at (6,5), which in turn implies an A at (6,7), which implies a queen at (2,7), which implies an A at (2,8), which implies a queen at (4,8).

4. [8] **Adversarial Search**

(a) [4] Circle all the nodes, or state that none exist, that are *not* visited because **alpha-beta pruning** occurs in the following game tree. Assume the root is a maximizing node and children are visited left to right.



Nodes I and J are not visited.

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

(b) [2] True or False: The Minimax algorithm using static evaluation function (aka utility function) *f1* is guaranteed to always choose the same next move as the Minimax algorithm using static evaluation function *f2* when $f2(n) = f1(n) + c$, n is an arbitrary state in a game tree, and c is a real-valued constant.

True, any order-preserving transformation of the values at the leaf nodes does not affect the choice of move.

(c) [2] Which one 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 static evaluation function

(iv) inadequate state representation

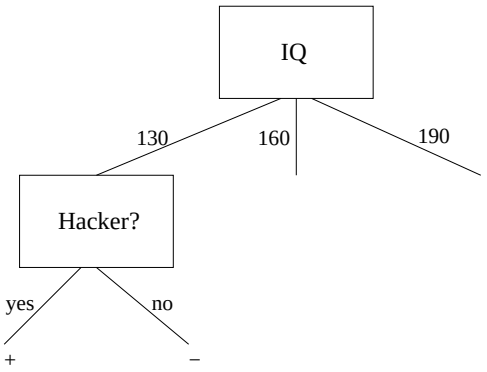
(ii)

5. [16] **Decision Trees**

In order to automate the department's graduate admissions decision process, we'd like to learn the concept "Will be a good CS graduate student." To this end we have obtained the following set of 14 training examples:

IQ	Hacker?	Publications?	Hobby	Good CS graduate student?
190	yes	no	movies	+
130	yes	no	music	+
130	yes	yes	cooking	+
190	no	yes	cooking	+
160	yes	yes	cooking	+
130	yes	yes	music	+
160	no	yes	music	+
190	no	no	music	+
190	yes	yes	movies	+
160	yes	no	movies	–
160	no	no	movies	–
130	no	yes	cooking	–
160	yes	no	music	–
130	no	no	music	–

(a) [8] Two nodes of the decision tree computed from this data are shown below. For each of the remaining two children of the root, compute either (1) if it is a non-leaf, the **information gain** associated with choosing the attribute *Hacker?* (only), or (2) if it's a leaf, its **classification** (i.e., + or –). In the case of (1), show your answer as an expression containing fractions and terms of the form $I(*,*)$, where I is the information content. You do *not* need to simplify this or compute a real-valued answer.



For the IQ=160 child node,
Remainder(Hacker) = 3/5 I(1/3, 2/3) + 2/5 I(1/2, 1/2),
I(IQ=160 node) = I(2/5, 3/5), and
Gain(Hacker) = I(2/5, 3/5) – 3/5 I(1/3, 2/3) – 2/5 I(1/2, 1/2).

For the IQ=190 child node, all 4 examples are +, so make this a leaf with classification +.

5. (continued)

(b) [5]

(i) [3] What is **n-fold cross-validation**?

Cross-validation is a technique that divides the training set into n parts of approximately equal size, and using n-1 of those parts to build a decision tree and the last part to test the classification accuracy of the tree. Do this n times using a different 1/n part of the data for testing each time, and average the results.

(ii) [2] How can n-fold cross-validation be used for deciding which of several decision tree pruning algorithms is best in terms of reducing the **overfitting problem**?

Cross-validation gives a better estimate of the classification accuracy of decision trees for a given domain, especially when the number of training examples is relatively small. To pick a pruning strategy that is best for a given domain, use n-fold cross-validation to compute, using each pruning method, the classification accuracy. Select the pruning method that gives the highest accuracy.

(c) [3] Say a given attribute, A, splits a nonempty set of examples, E, into two nonempty subsets, E_1 and E_2 (i.e., A is a binary attribute). Assume the ith child, i = 1 or 2, has p_i positive examples and n_i negative examples. Is it possible for the **information gain** for attribute A to be equal to 0? If yes, say how. If no, explain why not.

If p1 = n1 = p2 = n2, then Remainder(A) = 1/2 I(0.5, 0.5) + 1/2 I(0.5, 0.5) = 1. But at the given node half (= p1+p2) of the examples are positive and half (= n1+n2) are negative, so I(0.5, 0.5) = 1. Therefore Gain(A) = 1.0 - 1.0 = 0.

6. [20] **First-Order Logic**
Consider the following two sentences in first-order logic where the domain of both variables is the non-negative integers (aka natural numbers), and the "≥" predicate means "is greater than or equal to" (consider this predicate just like any other used with FOL).

$$\forall x \exists y \; x \geq y \tag{1}$$

$$\exists y \forall x \; x \geq y \tag{2}$$

(a) [4] Translate (1) and (2) into English.

(1) Every natural number has another one less than or equal to it.
(2) There is a least natural number.

(b) [3] Does (1) entail (2)?

No

(c) [3] Does (2) entail (1)?

Yes

(d) [5] Use the resolution refutation algorithm to try and prove that (1) follows from (2). If you succeed, show the refutation tree and the most general unifiers used. If you can't prove this, show the steps of the refutation that lead to a point where you cannot proceed and explain why the proof fails.

Convert (2) to clause form: $x \geq c_1$, where c_1 is a Skolem constant. Convert the negation of (1) to clause form: $\neg(c_2 \geq y)$ where c_2 is a Skolem constant. Unify the above 2 clauses to get the empty clause using mgu = $\{x/c_2, y/c_1\}$. So, the proof succeeds indicating that (1) is deducible from (2).

(e) [5] Use resolution refutation to try and prove that (2) follows from (1). Use the same instructions given in the previous question.

Convert (1) to clause form: $x \geq s_1(x)$ where $s_1(x)$ is a Skolem function. Convert the negation of (2) to clause form: $\neg(s_2(y) \geq y)$ where $s_2(y)$ is a (different) Skolem function. Attempting to unify these two clauses, first unify variable x with Skolem function $s_2(y)$, which succeeds, so we have so far that mgu =$\{x/s_2(y)\}$. Next, when we move to the next two terms, after substitution from the previous step, we need to unify $s_1(s_2(y))$ with y. But this fails because of the "occurs check", i.e., because y occurs in both terms. So, the proof fails here.

7. [20] **Soundness and Completeness**

(a) [8] Given proposition symbols P and Q, and binary connectives $\uparrow$ and $\downarrow$ defined by

P	**Q**	**P$\uparrow$Q**	**P$\downarrow$Q**
T	T	F	F
T	F	T	F
F	T	T	F
F	F	T	T

(i) [4] Is the inference rule $\dfrac{\neg\alpha,\ \alpha\downarrow\beta}{\neg\beta}$ **sound** for Propositional Logic (PL)? In addition to answering yes or no, justify your answer by showing an appropriate truth table.

Yes, because the only model for the premises is the last row where both premises are true, and in this case the conclusion is also true.

α	β	$\neg\alpha$	$\alpha\downarrow\beta$	$\neg\beta$
T	T	F	F	F
T	F	F	F	T
F	T	T	F	F
F	F	T	T	T

(ii) [4] Is the inference rule $\dfrac{\alpha\uparrow\beta}{\neg\beta}$ **sound** for PL? In addition to answering yes or no, justify your answer by showing an appropriate truth table.

No, because not all models for the premise are also models for the conclusion, as shown by the third row of the table:

α	β	$\alpha\uparrow\beta$	$\neg\beta$
T	T	F	F
T	F	T	T
F	T	T	F
F	F	T	T

(b) [8] Fill in the following table with Yes or No in each box assuming non-empty sentences α and β in the specified language, $\alpha \models \beta$, and using only the given inference rule and inference algorithm.

Language	Inference Rule	Inference Procedure	Sound?	Complete?
Horn Clauses in PL	Generalized Modus Ponens (GMP)	Natural Deduction (Forward Chaining)	Yes	Yes
PL	GMP	Natural Deduction	Yes	No
PL	Abduction	Natural Deduction	No	No
PL	Resolution	Resolution Refutation	Yes	Yes

(c) [4] A sentence α in PL entails sentence β if and only if

(i) [2] the sentence $\alpha \Rightarrow \beta$ is _____valid_____ (fill in one word)

(ii) [2] the sentence $\alpha \wedge \neg\beta$ is _____unsatisfiable_____ (fill in one word)