Midterm Examination

CS 540: Introduction to Artificial Intelligence

July 15, 2004

LAST (FAMILY) NAME: ________________________________

FIRST NAME: ________________________________

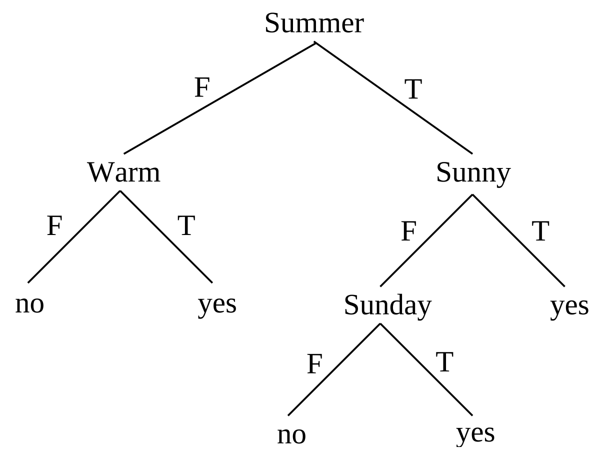
Problem	**Score**	**Max Score**
1	————	15
2	————	16
3	————	10
4	————	12
5	————	18
6	————	21
7	————	8
Total	————	100

1. [15] **Decision Tree Learning**

(a) [6] Describe briefly two (2) different causes of overfitting when learning a decision tree using the MaxGain criterion.

(b) [3] Suppose we generate a training set from a given "correct" decision tree and then use the decision tree learning algorithm with MaxGain to that training set. Is it the case that the algorithm is guaranteed to construct the same "correct" decision tree as the training set size goes to infinity? Briefly explain why or why not.

(c) [3] Given the following decision tree for making a binary decision about whether or not to go on a bike ride, write a single sentence in Propositional Logic that expresses the same information, i.e., when to go on a bike ride.



(d) [3] In a problem where each example has *n* binary attributes, to select the best attribute for a decision tree node at depth *k*, where the root is at depth 0, how many attributes must be compared?

2. [16] **Search Methods**
Consider the 3-puzzle problem, which is a simpler version of the 8-puzzle where the board is 2 x 2 and there are three tiles, numbered 1, 2, and 3. There are four operators, which move the blank **up**, **down**, **left**, and **right**, and these operators are applied *in this order* for all uninformed searches and in case of sibling ties for other searches. Break other ties by increasing time on OPEN. The cost of each operator is 1. The start and goal states are

Start

2	3
1	

Goal

1	2
	3

(a) [4] Draw the entire state space for this problem, labeling nodes and arcs clearly.

(b) [12] Assuming there is no checking for repeated states of any kind, draw the search trees produced by each of the following search methods. For each node in a tree, label it with a number indicating when it was removed from the OPEN list (and expanded or detected as a goal node). If a method does not find a solution, show the part of the search tree and then explain why no solution is found.

(i) Breadth-First search

(ii) Depth-First search

(iii) A* search with the heuristic equal to the number of misplaced tiles

3. [10] **Evaluation Functions for Heuristic Search**
Say we define the evaluation function for use in a heuristic search problem as

$$f(n) = (1-w)g(n) + wh(n)$$

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 $0.0 \le w \le 1.0$.

(a) [9] What search algorithm do you get when
(i) $w = 0.0$

(ii) $w = 0.5$

(iii) $w = 1.0$

(b) [1] Based on your answer to (a), for what range of values of w would you expect your algorithm is admissible?

4. [12] **Search and Propositional Logic**

The "SAT" problem is to determine if a given sentence in Conjunctive Normal Form (CNF) in Propositional Logic (PL) is true given some assignment of {True, False} to each of the symbols in the sentence. (CNF means the form of the sentence is a conjunction of disjunctions.) Say we want to solve this problem using Greedy Hill-Climbing Search. Each state corresponds to a complete assignment of True or False to each symbol. The successors of a state are all states that have exactly one symbol with a different truth value. For example, if a state were (A=True, B=False), the successor states would be (A=True, B=True) and (A=False, B=False). The evaluation function used is the number of clauses in the (CNF) sentence that are true.

(a) [3] If there are *n* distinct symbols in a given PL sentence (in CNF), how many neighboring states does each state have in the state space?

(b) [3] How many states are there in the state space?

(c) [6] Given the sentence $(\neg A \vee B \vee C) \wedge (A \vee \neg B \vee C) \wedge (A \vee B \vee \neg C) \wedge (A \vee B \vee C)$, is the state (A=False, B=False, C=False)

(i) a goal state? Briefly explain.

(ii) a local, but not a global, maximum of the state space? Briefly explain.

5. [18] **Constraint Satisfaction**

Say you work at a factory that makes flashlights, which requires four tasks to be completed: constructing the light bulb (L), charging the battery (B), doing the wiring (W), and assembling the housing (H). Task L takes 2 hours, B takes 1 hour, W takes 2 hours, and H takes 1 hour. We'd like the total time to make each flashlight to be no more than 4 hours. Tasks can be started at the beginning of each hour, designated 1, 2, 3, and 4. Tasks can done concurrently except L and B cannot be done simultaneously, L must be done sometime before H, and B must be completed sometime before W.

This problem can be formulated as a constraint satisfaction problem (CSP) by letting the variables be the start times for each of the tasks; we'll call these times SL, SB, SW, and SH. The possible values for each variable are the possible start times, 1, 2, 3, and 4. Based on the information above, the initial possible values for each variable are SL={1,2,3}, SB={1,2,3,4}, SW={1,2,3}, and SH={1,2,3,4}. The constraints are $SL + 1 < SH$, $SB < SW$, $SB \neq SL$, and $SB \neq SL + 1$.

(a) [3] Draw the constraint graph showing all of the above information.

(b) [3] Assuming we've already chosen (as the first step in a backtracking search) SL=2, what is the domain of each variable after applying **forward checking**?

(c) [12] Now assume we have all of the original domains for each variable and we apply **arc consistency**. Show the domain of each variable after each of the following steps of removing inconsistent values by filling in the following table. "Propagate X to Y" means that values in the domain of Y should be removed if they are inconsistent with the values in the domain of X. For example, given the initial domains specified above, after "propagating SL to SH" the domain of SH will change from {1,2,3,4} to {3,4}.

Fill in the following table resulting from the given sequence of propagation steps.

Action	**SL**	**SB**	**SW**	**SH**
Initial Domain	1,2,3	1,2,3,4	1,2,3	1,2,3,4
Propagate SL to SH	1,2,3	1,2,3,4	1,2,3	3,4
Propagate SH to SL				
Propagate SL to SB				
Propagate SW to SB				
Propagate SB to SW				

6. [21] **Propositional Logic**

(a) [2] One way of defining that two sentences in PL, α and β, are logically equivalent is to show that α <-> β is a tautology. Give an alternative definition of logical equivalence in terms of entailment.

(b) [8] Using the definition you gave as your answer to (a), prove that the clause ($\neg$P1 $\vee$ $\neg$P2 $\vee$ Q) is logically equivalent to the implication sentence (P1 $\wedge$ P2) $\rightarrow$ Q using the Resolution Refutation algorithm.

(c) [8] (i) Using the three propositional symbols, J means "I get the job," H means "I work hard," and P means "I get promoted," convert the following English sentences into three sentences in Propositional Logic.

If I get the job and work hard, I will be promoted. I was not promoted. Thus, either I did not get the job or I did not work hard.

(ii) Give an inference rule based on your sentences in (i) and then prove whether or not it is a sound rule of inference.

(d) [3] Is the following FOL sentence a tautology (aka valid), a contradiction (aka unsatisfiable), satisfiable, or none of these? Explain your answer using part or all of a truth table.

$(A \rightarrow \neg B) \rightarrow (C \rightarrow B)$

7. [8] **First-Order Logic**
For each of the following sentences in English, is the accompanying FOL sentence a good translation? If your answer is no, explain why not and correct it.

(a) [4] "Any course in Computer Science is harder than some courses in Psychology."

$$\forall x\ (Course(x) \wedge Dept(x, CS)) \rightarrow \exists y\ ((Course(y) \wedge Dept(y, Psychology)) \rightarrow Harder(x, y))$$

(b) [4] "If a course is harder than all courses in Math, it must be in Computer Science."

$$\forall x\ Course(x) \wedge (\forall y\ Course(y) \wedge Dept(y, Math) \wedge Harder(x, y)) \rightarrow Dept(x, CS)$$