Midterm Examination

CS 540: Introduction to Artificial Intelligence

March 8, 2006

LAST NAME: __________SOLUTIONS_____________________________

FIRST NAME: ____

Problem	**Score**	**Max Score**
1	————	14
2	————	14
3	————	18
4	————	18
5	————	16
6	————	10
Total	————	90

1. [14] **Search**
Consider a road map represented as a graph in which the cities are represented by nodes and the arcs connect some pairs of cities. An arc connecting cities *i* and *j* has a distance, *d*(*i*,*j*), indicating the length of the road connecting these two cities. If there is a road directly connecting city *i* and city *j*, then the predicate *Adjacent*(*i*,*j*) is true. Initially, two people are located in two different initial cities. At each "step" both people start at the same time and each moves to a city adjacent to where they started. The time required to travel to an adjacent city is equal to the distance between the cities. After *both* people have arrived at their new cities, the next step begins. The goal is to get the two people to meet in the same city as quickly as possible. Let a state be defined by a pair, (*i*,*j*), indicating that the first person is currently in city *i* and the second person is in city *j*.

(a) [2] Given a state, (*i*,*j*), define all its legal **successor** states.

The successors of state (*i*,*j*) are all pairs (*x*,*y*) such that *Adjacent*(*i*,*x*) is true and Adjacent(*j*,*y*) is true.

(b) [2] What is the condition for a state to be a **goal**?

Any state of the form (*i*,*i*) for some *i*.

(c) [2] What is the **cost** of a step (i.e., the time taken between the start of two successive steps)?

The cost of going from state (*i*,*j*) to state (*x*,*y*) is max(*d*(*i*,*x*), *d*(*j*,*y*)).

(d) [6] Let SLD(*i*,*j*) be the straight-line (i.e., Euclidean) distance between cities *i* and *j*, regardless of whether or not there is a road connecting them or what the distance is along a connecting road. Indicate for each of the following heuristic function definitions if it is **admissible** or not, and justify your answer.

(i) SLD(*i*,*j*)

At a step the two people may move straight towards each other the same distance, reducing the distance between them by 2 times the time cost at each step. So, SLD(i,j) could overestimate the true cost, meaning it is *not* admissible.

(ii) 2(SLD(*i*,*j*))

Not admissible for the reason given in (i).

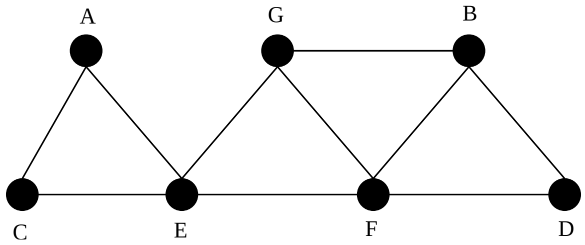
(iii) SLD(*i*,*j*)/2

Admissible for the reason given in (i).

(e) [2] If some roads are freeways, all with speed limit *sf*, and some roads are normal roads, all with speed limit *sr*, where *sr* < *sf*, and *h*(*i*,*j*) is an admissible heuristic when all roads have the same speed limit, define a heuristic function, *h2*(*i*,*j*), that is **admissible** given the two different types of roads.

h2(*i*,*j*) = *h*(*i*,*j*) / *sf* since this relaxes the constraint that some roads are slower than freeways.

2. [14] **Constraint Satisfaction**
Consider the problem of coloring the nodes, A, B, ..., G, in the following graph using the 3 colors, 1, 2 and 3, so that no two adjacent nodes (i.e., connected by an arc) have the same color.



(a) [6] Fill in the table below with the domain of each node after each of the following steps of selecting a node and assigning a color followed by **forward checking** (FC).

	A	B	C	D	E	F	G
Initial domain	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3	1,2,3
After A=1 and FC	1	1,2,3	2,3	1,2,3	2,3	1,2,3	1,2,3
After B=1 and FC	1	1	2,3	2,3	2,3	2,3	2,3
After C=2 and FC	1	1	2	2,3	3	2,3	2,3

(b) [2] What general condition in a table such as the one above would indicate a "**deadend**" in a backtracking search with forward checking and constitute a backtrack point when searching for a solution?

```
The domain of at least one node is empty
```

(c) [2] What general condition in a table such as the one above would indicate that a **solution** has been found during a backtracking search with forward checking?

```
Every node has a single value associated with it
```

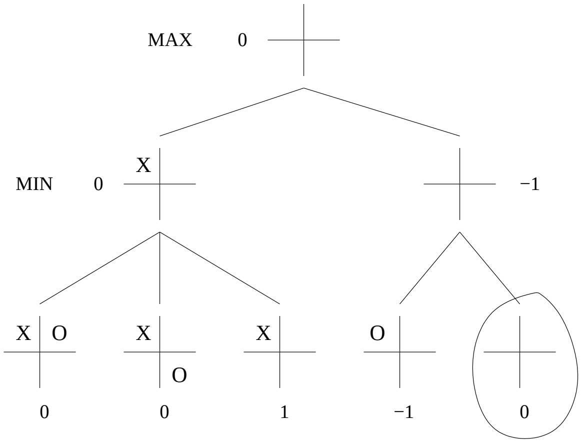
(d) [4] Apply **arc consistency** to the results of the third step above, when color 2 is selected for node C. Write below the resulting domains for each of the 7 nodes.

	A	B	C	D	E	F	G
After C=2, FC, & arc consistency	1	1	2	3	3	2 (or ∅)	∅ (or 2)

3. [18] **Adversarial Search**

(a) [14] Consider the game of 2 × 2 tic-tac-toe in which at each turn a player can either mark an empty square on the board or else "pass" meaning that they do not mark any square. Assume **X** goes first and represents the MAX player.

(i) [5] Draw the full game tree to depth 2 (the root is at depth 0). Do not show nodes that are rotations or reflections of siblings already included. Assume each "pass" move is generated last, making it the rightmost child of each node. Your tree should have 5 nodes at depth 2.



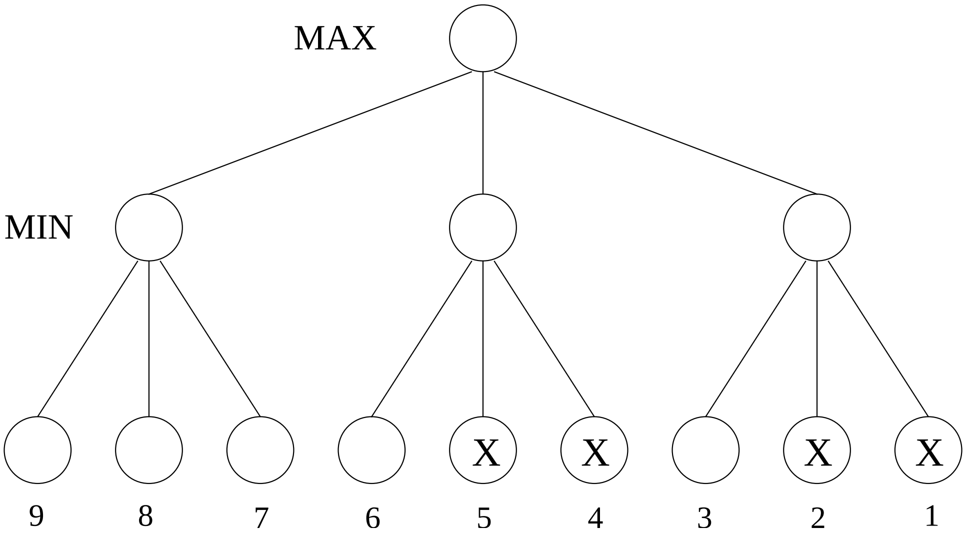
(ii) [3] Let the static evaluation function used be defined as the number of **X**s on the board minus the number of **O**s. Beside each leaf node mark its value, and then use minimax to compute the values at all non-leaf nodes.

(iii) [2] Assuming children are visited left-to-right in your tree in (a), circle any node that would not be evaluated by alpha-beta.

(iv) [4] Because this is such a simple game, we want to compute a complete playing strategy that will completely specify how player **X** should play (no matter what player **O** does) and guarantee that **X** wins. Explain how the alpha-beta pruning algorithm *can* be used to do this but the minimax algorithm *cannot*.

Because of the "pass" move, the game tree is of infinite size and minimax will loop forever following an infinite sequence of "pass" moves by both players. Assuming the last (rightmost) child of each node is the "pass" move, alpha-beta will prune the pass move as soon as it finds a sure win for **X**. Hence infinite looping is avoided.

(b) [4] Consider the following game tree. Assuming children nodes are visited left to right, assign the static evaluation function values of 1, 2, ..., 9 to the nine leaf nodes in such a way that alpha-beta pruning eliminates as *many* nodes as possible. Indicate which nodes would not be visited in this case.



4. [18] **Decision Trees**

Consider the following set of 10 training examples, each containing three attributes, A1, A2 and A3, and a desired binary classification, + or –. For your information, $\log 0.1 = -3.3$, $\log 0.2 = -2.3$, $\log 0.3 = -1.7$, $\log 0.4 = -1.3$, $\log 0.45 = -1.15$, $\log 0.5 = -1.0$, $\log 0.55 = -0.85$, $\log 0.6 = -0.7$, $\log 0.7 = -0.5$, and $\log 0.8 = -0.3$, $\log 0.9 = -0.15$, and $\log 1 = 0$, where all logs are to base 2.

A1	**A2**	**A3**	**Class**
0	0	1	+
0	0	2	+
0	0	3	+
0	0	4	+
0	1	1	–
0	1	2	–
0	1	3	–
1	0	4	–
1	1	1	+
1	1	2	+

(a) [2] What is the **Information** content associated with the entire set of examples?

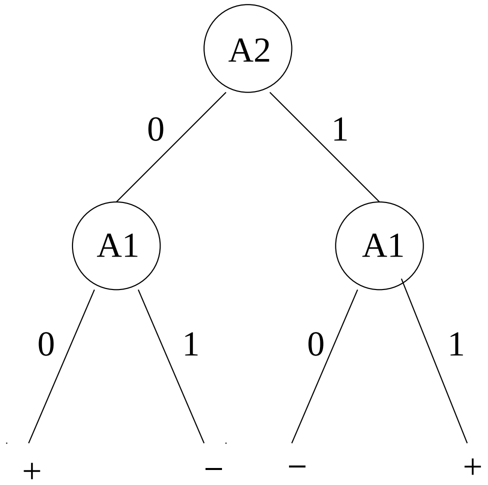
$$I(6/10, 4/10) = -0.6 \log 0.6 - 0.4 \log 0.4 = (-0.6)(-0.7) + (-0.4)(-1.3) = 0.94$$

(b) [5] Compute the **Remainder** and **Information Gain** associated with choosing attribute A1 for the root of the decision tree. Show your work. Round values in order to use the log table given above.

$$\begin{aligned}
Remainder(A1) &= (7/10)\, I(4/7, 3/7) + (3/10)\, I(2/3, 1/3) \\
&= (0.7)\, I(0.57, 0.43) + (0.3)\, I(0.67, 0.33) \\
&= 0.7[(-.57)\log .57 + (-.43)\log .43] + 0.3[(-.67)\log .67 + (-.33)\log .33] \\
&= 0.7[(-.57)(-.85) + (-.43)(-1.15)] + 0.3[(-.67)(-.5) + (-.33)(-1.7)] \\
&= 0.7(0.4845 + 0.4945) + 0.3(0.335 + 0.561) \\
&= 0.979 + 0.269 \\
&= 1.248
\end{aligned}$$

$$Gain(A1) = 0.94 - 1.248 = -0.3 = 0.0(rounded)$$

(c) [3] Using only attributes A1 and A2, draw a complete decision tree for this set of examples, if possible. (Any tree consistent with the training examples is okay.)



(d) [2] True or False: Given an arbitrary decision tree containing only binary attributes, there is a equivalent sentence in Propositional Logic that represents the same information described by the classification tree.

```
True
```

(e) [2] True or False: Two different decision trees that both correctly classify a set of training examples, will also classify any other testing example in the same way (i.e., both trees will output the same class for any other example).

```
False
```

(f) [4] If there is “noise” in the training set (meaning the values of attributes and desired class may be wrong), the decision tree learning algorithm learns a poor tree because of what general problem? To alleviate this problem, how might the decision tree construction procedure be modified?

```
The problem is that the learning algorithm overfits the data.  To
lessen the problem, prune the tree by, for example, creating a
leaf node if the information content of the examples at a node is
close to 0 (i.e., within epsilon).
```

5. [16] **Logic**

(a) [6] Which of the following sentences in PL are **entailed** by the sentence $(A \vee B) \wedge (\neg C \vee \neg D \vee E)$? (Note: You should be able to do this without using a truth table.)

(i) $(A \vee B)$

Entailed by And-Elimination

(ii) $(A \vee B \vee C) \wedge ((B \wedge C \wedge D) => E)$

Entailed because $(B \wedge C \wedge D) => E$ is equivalent to $\neg B \vee \neg C \vee \neg D \vee E$ which is weaker than the second clause.

(iii) $(A \vee B) \wedge (\neg D \vee E)$

Not entailed because this removes the $\neg C$ literal which strengthens the second clause.

(b) [2] A set of sentences in PL, *s1*, *s2*, ..., *sn*, is unsatisfiable (aka a contradiction) if and only if which of the following is/are true? Your answer can be one or more of (i) - (v). (i) $\neg s1 \wedge \neg s2 \wedge \cdots \wedge \neg sn$ is satisfiable; (ii) $\neg s1 \wedge \neg s2 \wedge \cdots \wedge \neg sn$ is valid; (iii) $\neg s1 \vee \neg s2 \vee \cdots \vee \neg sn$ is satisfiable; (iv) $\neg s1 \vee \neg s2 \vee \cdots \vee \neg sn$ is valid; and (v) None of the above.

(iv) because a conjunction of sentences is unsatisfiable if and only if its negation is valid.

(c) [8] Which of the following are syntactically and semantically correct **translations** into First-Order Logic of "Everyone's zipcode within a state has the same first digit?" *Digit*(*n*,*z*) is a function that returns the *n*th digit of zipcode *z*. *Zipcode* is a function that returns a zipcode. *State* and *LivesIn* are predicates. Each answer should say either *correct* or *incorrect.*

(i) $\forall x \forall s \forall z\ [State(s) \wedge LivesIn(x,s) \wedge (Zipcode(x) = z)] =>$
$[\forall y \forall w\ (LivesIn(y,s) \wedge (Zipcode(y) = w)) => (Digit(1,z) = Digit(1,w))]$

Correct

(ii) $\forall x \forall s\ [State(s) \wedge LivesIn(x,s) \wedge \exists z\ (Zipcode(x) = z)] =>$
$[\forall y \forall w\ LivesIn(y,s) \wedge (Zipcode(y) = w) \wedge (Digit(1,z) = Digit(1,w))]$

Incorrect because the existentially quantified variable z has the wrong scope and the universally quantified variables y and w have $\wedge$ as their connective.

(iii) $\forall x \forall y \forall s\ [State(s) \wedge LivesIn(x,s) \wedge LivesIn(y,s)] => Digit(1,(Zipcode(x) = Zipcode(y)))$

Incorrect because it's syntactically wrong: $Zipcode(x) = Zipcode(y)$ is not a term, it's a predicate.

(iv) $\forall x \forall y \forall s\ [State(s) \wedge LivesIn(x,s) \wedge LivesIn(y,s)] =>$
$[Digit(1,Zipcode(x)) = Digit(1,Zipcode(y))]$

Correct

6. [10] **Inference in Propositional Logic**
You are given the following sentences in Propositional Logic defining a knowledge base.

$\neg P \vee \neg Q$

$(P \vee Q) \wedge (P \vee R)$

$(S \vee Q) \wedge (S \vee \neg R)$

$\neg R$

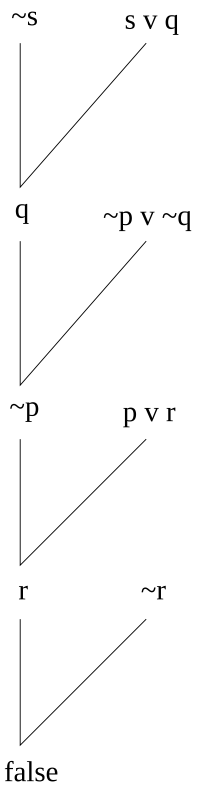
(a) [2] In order to prove *S* using the truth table method, how many **rows** would the table contain?

There are 4 Boolean variables so the table has $2^4 = 16$ rows.

(b) [2] Specify one **model** of the above sentences.

P=true, Q=false, R=false, and S=true

(c) [6] Prove *S* using the **Resolution Refutation algorithm**.


~s
s v q
q
~p v ~q
~p
p v r
r
~r
false