Your Name__________________________ and ID _______________________

Solutions

ECS 170 Introduction to Artificial Intelligence

Final Examination, Open Text Book and Open Class Notes.
Answer All questions on the question paper in the spaces provided
Add additional sheets if necessary

Time: 2 hours

(1) Quickies (12 points)
Decide if each of the following is True or False. Please provide a brief justification to your answer.

(a) **(3 points)** Breadth first Search is complete even if zero step-costs are allowed.

Ans: True. What matters in DFS is the depth of the goal, not cost.

(b) **(3 points)** Depth-first iterative deepening always returns the same solution as breadth-first search if b is finite and the successor ordering is fixed.

Ans. True. Both return the "leftmost" among the shallowest solutions.

(c) **(3 points)** Any decision tree with Boolean attributes can be converted into an equivalent feedforward neural network.

Ans. True. A neural network with enough hidden units can represent any Boolean function.

(d) **(3 points)** Assume that a rook can move on a chessboard any number of squares ina straight line, vertically or horizontally, but cannot jump over other pieces. Manhattan distance is an admissible heuristic for the problem of moving the rook from square A to Square B in the smallest number ofmoves.

Ans. False. A rook can cross the board in one move, so Manhattan is pessimistic and hence not an admissible heuristic.

2. **Logic. (8 points)**

(a) (**3 points**) For the following sentence in English, decide if the associated first-order logic sentence is a good translation. If not, explain why not and correct it. (There may be more than one error or none at all)

John's social security number is the same as Mary's

$\exists n\_HasSS\#(John, n) \wedge HasSS\#(Mary, n)$

Ans. This is ok.

(b) **(5 points)** Prove using Resolution that the sentence below entails G

Definition: A propositional 2-CNF expression is a conjunction of clauses, each containing *exactly* two literals, e.g.,

$$(A \vee B) \wedge (\neg A \vee C) \wedge (\neg B \vee D) \wedge (\neg C \vee G) \wedge (\neg D \vee G)$$

Ans. Here is the standard resolution.
Step1. Combine
$(A \vee B)$ with $(\neg A \vee C)$ to get $(B \vee C)$
Step 2. Now combine
$(B \vee C)$ with $(\neg B \vee D)$ to get $(C \vee D)$
and keep doing it until you get G.

3. Bayes' Theorem (10 points)
Suppose that we have two bags each containing black and white balls. One bag contains three times as many white balls as blacks. The other bag contains three times as many black balls as white. Suppose we choose one of these bags at random. For this bag we select five balls at random, replacing each ball after it has been selected. The result is that we find 4 white balls and one black. What is the probability that we were using the bag with mainly white balls?
[Hint: Start by defining A as the random variable "bag chosen" " then A={a1,a2} where a1 represents "bag with mostly white balls" and a2 represents "bag with mostly black balls"]

Solution: Let A be the random variable "bag chosen" then A={a1,a2} where a1 represents "bag with mostly white balls" and a2 represents "bag with mostly black balls" . We know that P(a1)=P(a2)=1/2 since we choose the bag at random.
Let B be the event "4 white balls and one black ball chosen from 5 selections".
Then we have to calculate P(a1|B). From Bayes' rule this is:

$$P(a_1|B) = \frac{P(B|a_1)\cdot P(a_1)}{P(B|a_1)\cdot P(a_1) + P(B|a_2)\cdot P(a2)}$$

Now, for the bag with mostly white balls the probability of a ball being white is ¾ and the probability of a ball being black is ¼. Thus, we can use the Binomial Theorem, to compute P(B|a1) as:

$$P(B|a_1) = \binom{5}{1}\left(\frac{3}{4}\right)^4\left(\frac{1}{4}\right)^1 = \frac{405}{1024}$$

Similarly

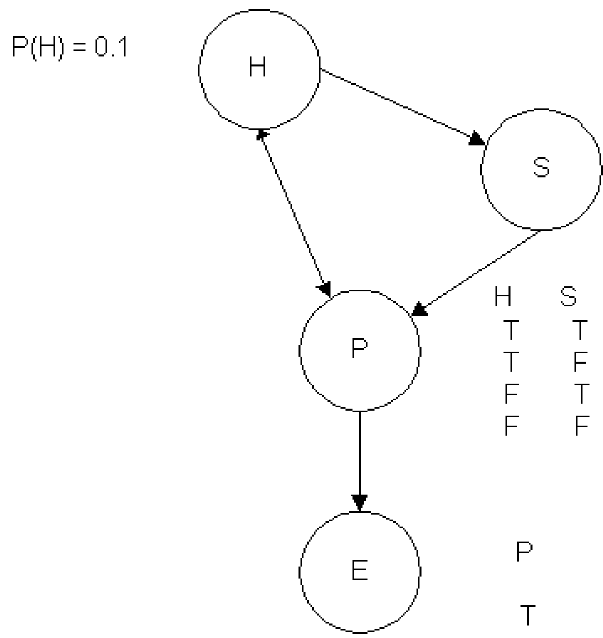
$$P(B|a_2) = \binom{5}{1}\left(\frac{1}{4}\right)^4\left(\frac{3}{4}\right)^1 = \frac{15}{1024}$$

hence

$$P(a_1|B) = \frac{405/1024}{405/1024 + 15/1024} = \frac{405}{420} = 0.964$$

4. Belief Networks (20 points)

Consider a simple belief network with the Boolean variables H = Honest, S = Slick, P = Popular and E = Elected as shown in the figure below



H	P(S)
T	0.3
F	0.9

H	S	P(P)
T	T	0.9
T	F	0.5
F	T	0.8
F	F	0.2

P	P(E)
T	0.6
F	0.1

Figure A Simple Bayes Network with Boolean variables H = Honest, S = Slick, P = Popular, E= Elected

a. **(3 points)** For now, do not look at the CPT tables, but only at the structure of the network. Which, if any, of the following are asserted by the netwrok structure.
   - P(H, S) = P(H) P(S)
   - P(E|P, H) = P(E|P)
   - $P(E) \neq P(E \mid H)$

(b) **(4 points)** Calculate $P(h, s, \neg p, \neg e)$

(c) **(5 points)** Calculate the probability that someone is elected given that they are honest.

(d) **(8 points)** Suppose we want to add the new variable L = LosOfCampaignFunds to the belief network. Describe with justifications all the changes you would make to the network.

Solution.(a) (ii) is asserted, by the local semantics of the BN. A node is conditionally independent of its non-descendents, given its parents. (i) is NOT asserted since H and S are linked by an arc. (iii) is NOT asserted by the structure alone, because arcs do not deny independence. (The CPT can deny independence).

(b) $P(h, s, \neg p, \neg e) = P(h)P(s \mid h)P(\neg p \mid h, s)P(\neg e \mid \neg p)$ = 0.1 x 0.3 x 0.1 x 0.9 = 0.00027

(c) One way to do this is to construct the full joint distribution for H = True (8 rows) and add them up.
$P(E \mid H) = \alpha P(h) \sum sP(s \mid h) \sum pP(p \mid h, s)P(E \mid p)$
= α 0.1[0.3x(0.9(0.6, 0.4)+0.1(0.1, 0.9)+0.7x(0.5(0.6, 0.4) + 0.5 (0.1, 0.9))]
= (0.41, 0.59)

(d) Let us assume that honesty does not influence fund-raising ability, but slickness does. Funds support advertising which increases popularity but not necessarily mprove electability. So L should be a child of S and parent of P. We would need CPT for P(L|S) and an augmented CPT for P(P|H,S,L). Any CPT's reflecting the above mentioned influences will do.

5. Learning. (9 points)

(a) **(3 points)** Give a simple argument to show that a Perceptron cannot represent some data sets generated by decision trees.

Ans: A decision tree can represent any Boolean function, including XOR. A Perceptron cannot represent XOR.

(a) **(7 points)** Now consider the representation of Boolean functions by Perceptrons, where "True" is +1 and "False" is –1. Assume a fixed input $a_0 = -1$ and three other inputs a_1, a_2, a_3 and assume that the activation function is a step function $g(x)$ if x > 0, -1 otherwise. Draw a Perceptron that represents the disjunction of the three attributes a_1, a_2, a_3.

Ans. We need a Perceptron with three external inputs, the fourth one being –1. So we need to specify 4 weights. Because this is a disjunction of a_1, a_2, a_3, there is symmetry and all three associated weights, namely w_1, w_2, w_3 are equal, say = 2. The "lowest" input requiring a +1 at the output is (say) +1, -1, -1. This input gives a weighted sum of 2*1 + 2* (-1) + 2 * (-1) = -2. The input requiring an output of –1 is –1, -1, -1. This gives a weighted sum of 2*(-1) + 2* (-1) + 2 * (-1) = -6.

One way we can separate the two classes is to choose the weight w_0 to get the desired effect. Suppose we choose $w_0 = 4$, we get the desired effect.

(6) Bayesian Updating. (20 points)
A doctor knows that the disease meningitis causes the patient to have stiffneck 50% of the time. The doctor also knows some unconditional facts: Prior probability of a person having meningitis is 1/50,000. Prior probability of any patient having a stiff neck is 1/20. During these calculations, use the following additional information – as necessary.
$P(F \mid \neg M) =$ same as the background frequency of fever in a population = 0.02.
$P(F \mid M) =$ 0.8

(a**) (6 points)** Starting with a patient about whom we know nothing, show how the probability of having meningitis, P(M), is updated after we find the patient has stiff neck.

(b(**(14 points)** Next show how P(M) is updated, again, when we find the patient has fever.

Show all steps and calculations very clearly so I can follow what you are doing.

Ans. Part (a)
P(S) = 1/20= 0.05; P(S|M) = 50% = 0.5
Initially P(M) = 1/50,000 = 0.00002;
Update for Stiffneck

$$P(M \mid S) = P(M)\frac{P(S \mid M)}{P(S)} = 0.00002\frac{0.5}{0.05} = 0.0002$$

Update again for fever, assuming that the symptoms (fever and stiffneck) are conmditionally independent given meningitis)

$$P(M \mid S, F) = P(M \mid S)\frac{P(F \mid M, S)}{P(F \mid S)} = P(M \mid S)\frac{P(F \mid M)}{P(F \mid S)}$$

Here we know everything except P(F|S). We cannot reasonably make a guess of this because there is no causal relationship between fever and stiffneck.
One way to avoid the calculation of P(F|S) is to do normalization. Toward this goal, write

$$P(\neg M \mid S, F) = P(\neg M \mid S)\frac{P(F \mid \neg M)}{P(F \mid S)}$$

Now we follow these steps:

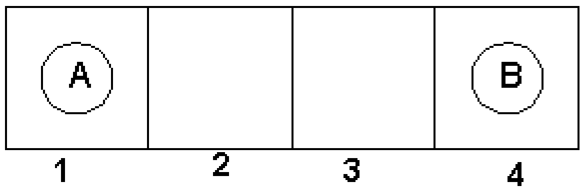
$$P(M \mid S, F) \alpha P(M \mid S)\frac{P(F \mid M)}{} = 0.0002 \times 0.8 = 0.00016$$

$$P(\neg M \mid S, F) \alpha P(\neg M \mid S)\frac{P(F \mid \neg M)}{} = 0.9998 \times 0.02 = 0.019996$$

Using normalization

$$P(M \mid S, F) = \frac{0.00016}{0.00016 + 0.019996} = 0.008$$

7. Games (20 points)
Consider a two-player game featuring aboard with four locations, numbered 1 through 4 and arranged in a line. Each player has a single token. Player A starts with his token in space 1 and Player B starts with his token in space 4. Player A moves first.



The two players take turns moving, and each player must move ghis token to an open adjacent space *in either dierction*. If an opponent occupies an adjacent space, then the player may jump over the opponent to the next open space, if any. (For example, if A is on 3 and B is on 2, then A may move back to 1.) The game ends when one player reaches the opposite end of he board. If Player A reaches space 4 first, the value of the game is +1; if Player B recahes space 1 first, the value fo the game is –1.

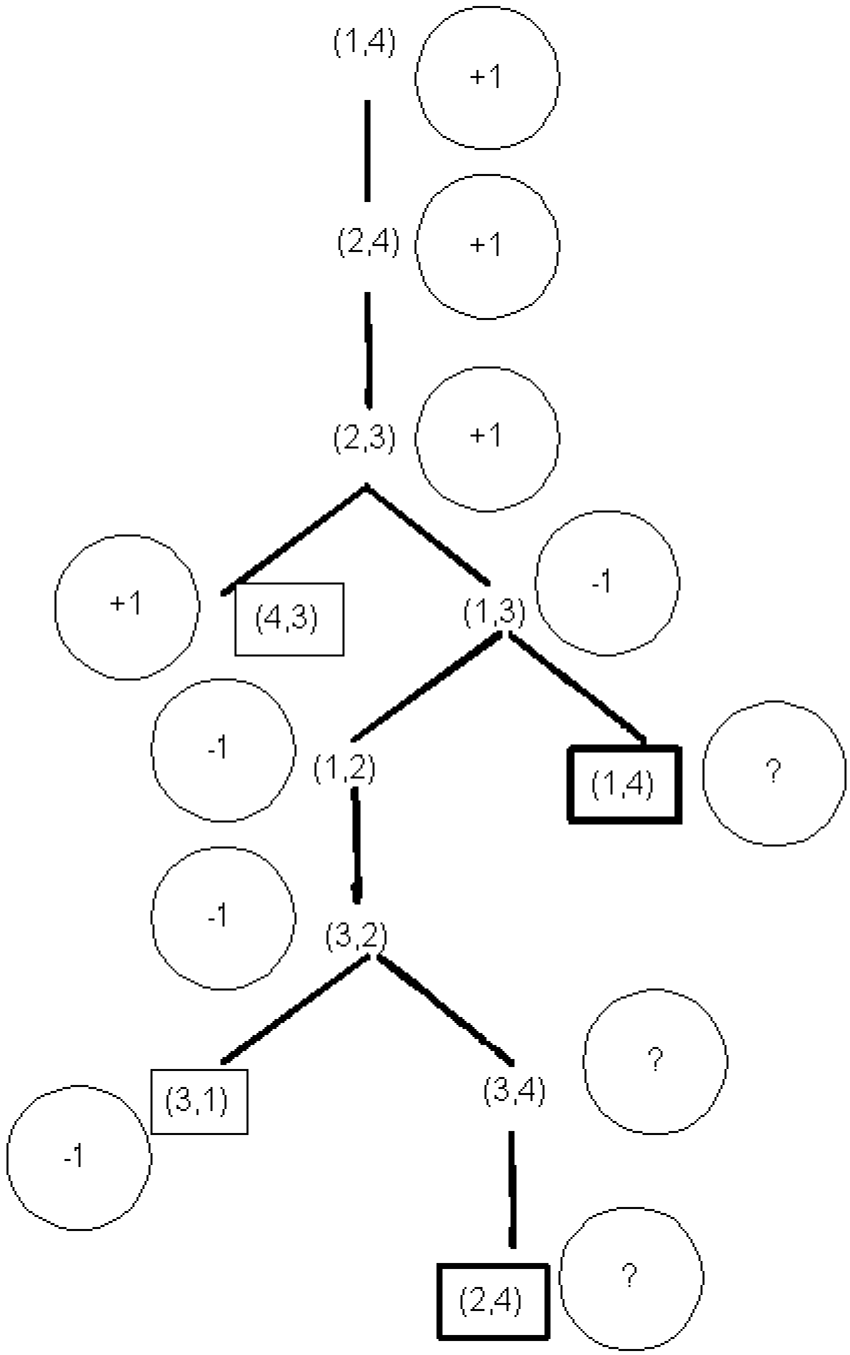
(a) **(10 points)** Draw the complete game tree, usoing the following conventions.
- Write each state as (s_A, s_B) where s_A and S_B denote the token locations.
- Put the terminal states in square boxes and annotate each with its game value in a circle.
- Put *loop states* (that is states that already appear on the path to the root) in double square boxes. Sincer it is not clear how to assign svalues to these states, put a ? in a circle.

(b) **(5 points)** Now mark each node with its backedup minimax value (also in a circle) . Expalin in words how you handles the ? values and why.

(c) **(5 points)** Explain why the standard minimax algorithm would fail on this game tree and briefly sketch how you might fix, drawing on your answer to part (b). Does your modified algorithm give optimal decisions on all gams with loops? Why?

Ans. (a) The game tree looks as below, complete with annotations and backed up values.

(b) The ? values are handled by assuming that an agent with a choice between winning the game and entering a ? state will choose to win. That is min(-1, ?) = -1 and max (+1,?) = +1. If all successors are ? then the backed up value is ?



(d) Standard Minimax is depth first and would go into an infinite loop.

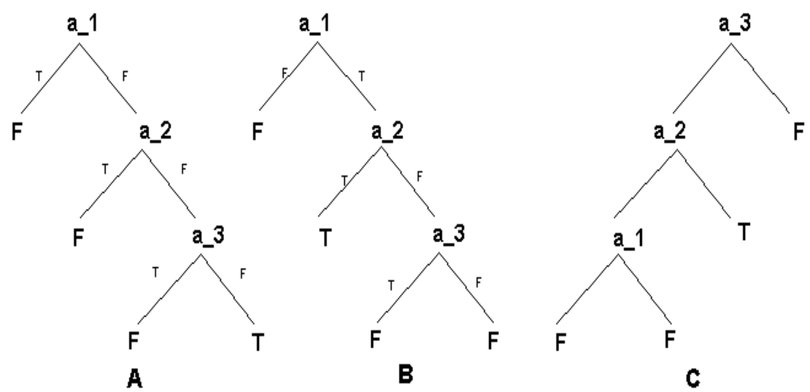
================End of Test================

A 1-decision list or 1-DL is a decision tree with Boolean inputs in which at least one branch from every attribute test leads immediately to a leaf. (Obviously, the final test leads to two leaves).

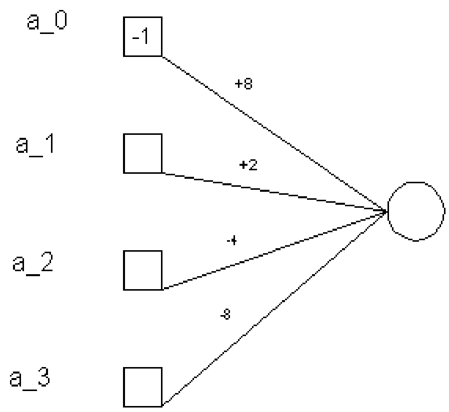
(b) **(3 points)** Draw a 1-DL that is equivalent to the *disjunction* of three attributes a_1, a_2, a_3.

Ans: A linear tree with three attributes; leaves are T, T, T, F

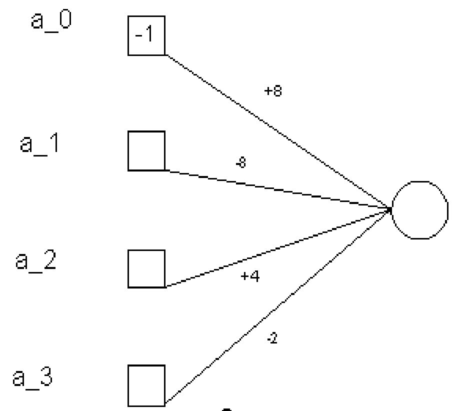
(c) **(8 points)** Consider the following three 1-DL's labelled A, B and C and the three Perceptrons labelled 1, 2, 3. (see the figure below). Each of the three figures from Group 1 is equivalent to one of the figures from Group 2. Show that equivalence with a brief explanation of why the answer is so. {Hint. I am asking you to show something like "A is equivalent to 1, B is to 2 and c is to 3. But, of course, this is NOT the correct answer. You have to figure that out.)



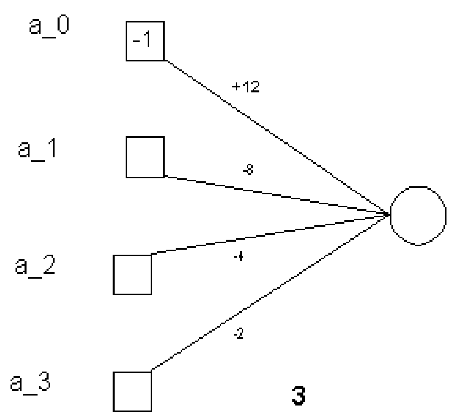
==



1



2



3

Ans: Intuitively, the root of a Decision list is the most important, so it has the highest weight. If a true attributes requires a false output, then its weight must be negative. So A=3, B=2, C=1.