CSCI 3202 PS 2; due Wednesday 10/25

Problem 2.1 (35 points) SIM game

Write a program to play the game of SIM with eight points (labeled A-H). Again, the two players are RED and BLUE; you can assume that RED moves first by convention; each player alternately chooses an edge to "claim" for the given color; and the loser is the player who is forced to create a complete triangle of edges in their own color. Your program should play interactively against the user. It needn't involve any fancy graphics or interface, but it should play a decent and competitive game of SIM. The interaction could be entirely textual, and could look something like this:

Are you playing Red? (Y/N): Y
RED move: A, B
BLUE move: E, G
etc.

Some notes: you might want to use alpha-beta pruning (it'll come in handy for the next problem anyway). In any event, whatever algorithmic strategy you use for your program, you should explain and document your approach. Since the computer program will probably not display the current state of the game (though of course it could), you may want to play against the program with a paper-and-colored-pencil representation of the game board handy. You should submit the important source code for your program (i.e., the major procedures for deciding on a next move) and a sample game or two to show how it plays the game

Problem 2.2 (15 points) Alpha-Beta Pruning

Construct a tree with a branching factor of 3 at each level, and 81 leaves. This tree will represent two complete MAX-MIN pairs of moves. The top level of the tree is MAX; the next level (3 nodes) is MIN; the third level (9 nodes) is MAX; the fourth (27) is MIN; and the fifth (81 nodes) are the leaves of the tree, labeled with the following numbers, grouped in sets of 27, and reading from left to right:

8 3 7 2 5 9 5 3 8 2 5 7 9 4 8 6 3 2 4 9 5 1 7 2 2 4 6

1 4 7 8 4 2 5 7 8 2 4 6 9 5 6 1 0 2 4 5 3 9 2 7 3 1 7

9 5 6 4 0 1 3 4 5 1 2 3 7 3 4 4 3 2 8 2 7 1 9 3 2 7 3

As usual, MAX is trying to achieve the highest number; MIN the lowest.

(a) What is the overall value of this tree to MAX?

(b) How many of these leaves 81 leaves actually have to be evaluated by an alpha-beta game search algorithm assuming that possible moves are examined from left to right?

(c) How many leaves have to be evaluated by alpha-beta assuming that moves are examined from right to left?

(d) Suppose the root player is MIN. (That is, the levels of the tree are now MIN, MAX, MIN, and MAX.) Now how many of the 81 leaves have to be evaluated by an alpha-beta searcher if moves are examined from left to right?

Problem 3. (20 points) 4x4 Sudoku

Suppose, instead of the regular (9x9) Sudoku game, we make a much simpler 4x4 version. In this version, each square contains the number 1, 2, 3, or 4; a square only contains one number; each of the four rows contains the values 1, 2, 3, 4; each of the columns contains 1, 2, 3, 4; and each two-by-two block (upper left, upper right, lower left, lower right) likewise contains 1, 2, 3, 4. The rows are counted from the top (i.e., the top row is row number 1), and the columns are counted from the left.

3a. (2 points) Here's a 4x4 Sudoku puzzle for you to solve "by hand":

1			4
			2
2			
	3		1

3b. (18 points) Let's approach this puzzle using propositional logic. Let the sentence variable "Sxyz" stand for the sentence "The Square in row x, column y, has contents z." Thus, there are 64 sentences whose truth value we are interested in:

S111 ("Row 1 Column 1 contains 1")
S112 ("Row 1 Column 1 contains 2")
S113 ("Row 1 Column 1 contains 3")
S114 ("Row 1 Column 1 contains 4")
S121 ("Row 1 Column 2 contains 1")

...
S444 ("Row 4 Column 4 contains 4")

In the sample puzzle above, then, as it happens S144 is true, S111 is true, and S113 is false.

Representing this puzzle in propositional logic means that we have to state many of the constraints of the puzzle in the appropriate form. For example, to represent the idea that the square in row 1 column 1 contains a number, we could write:

S111 OR S112 OR S113 OR S114

We need more than this, however. Since the square can only contain a single number, we need statements like:

NOT(S111 AND S112)
NOT (S111 AND S113)

and so forth.

Your job is as follows:

(a) Write the necessary statements of propositional logic to ensure that there is exactly one "3" in the fourth column of the puzzle.

(b) Write the necessary statements of propositional logic to derive the placement of a "3" in row 3, column 4; these statements should include starting constraints of the puzzle (e.g., "S144", "S242", and so on) and rules like the one you wrote in part (i) above. Finally, show how you can use resolution, along with those statements, to derive the truth of S343.

Problem 4 (30 points) Mastermind

In this problem, you will write a program to play as the “code-guesser” in the game of Masetermind. For simplicity, this game of Mastermind has ***four*** colors (not six, as usual) and ***three*** positions (not four, as usual). The colors are Red, Blue, Orange, and White. This means that there are only 64 possible codes.

Your program should allow you (as user) to think of a “secret” code. The program will then make successive guesses at the code, and you answer with X and O responses as in the standard MasterMind game. The program should simply keep track of the current set of “open” guesses consistent with the answers so far; you can then decide how the next guess can be selected. When the game starts, the program has 64 “open” possibilities; each answered guess should narrow the possibilities.

Here’s an example. Suppose you think of the code: R R W. The program might begin by asking:

B B O Your answer: nothing (no X's or O's)

The program now has limited the set of open codes to a total of eight (the ones that involve only R and W). The program might next ask:

R W W Your answer: XX

The program now has three remaining possibilities (WWW, RRW, and RWR). The program might now guess:

WWW Your answer: X

The program now has only one remaining possibility (RRW), so it guesses that, and it's done.

Show your program at work for an example code.

This strategy is a small example of what in machine learning is referred to as "candidate elimination": as we get more information, we can eliminate hypotheses that are incompatible with the information that we've received so far. Would this strategy work for a large-scale Mastermind game with (say) 30 colors and (say) 25 positions?