CMSC 671 Artificial Intelligence - Fall 2013
Homework Assignment 4
Due at the start of class on October 16^{th}

Each of these problems is worth 25 points.

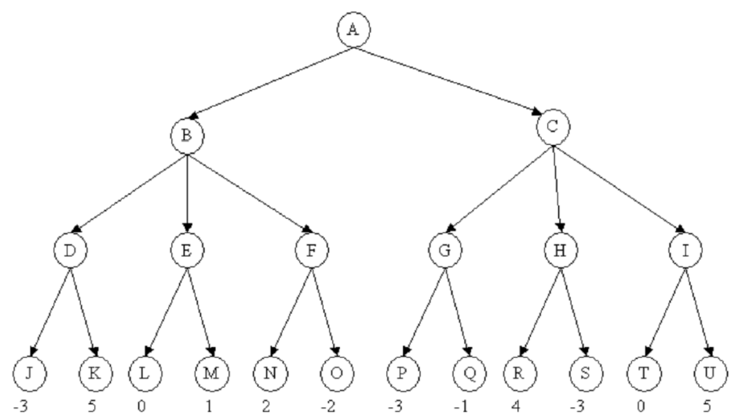
1. Consider the problem of constructing (not solving) crossword puzzles. You are given two things, a grid that specifies which squares are open (can be filled with letters by the puzzle solver) and which are shaded (cannot be filled with letters), and a dictionary of words. Your task is to fill in all of the blank squares creating a valid crossword puzzle using any subset of the words in the dictionary.

   Explain how you would formulate this as a constraint satisfaction problem. Should the variables be words or letters? Why? Write no more that 500 words total.

2. Binary constraints play an important role in constraint reasoning. We asserted that constraints involving more than two variables can always be rewritten as constraints involving only two. Some CSP software systems, in fact, can only constraints between a pair of variables.

   Show how a single ternary constraint such as "A + B = C" can be turned into three binary constraints by using an auxiliary variable. You may assume finite domains. (Hint: Consider a new variable that takes on values that are pairs of other values, and consider constraints such as "X is the first element of the pair Y.") Next, show how constraints with more than three variables can be treated similarly.

3. Consider the game tree below, in which the values assigned by the evaluation function are shown at the leaves. The root is a MAX player.



   – Suppose MAX uses alpha-beta pruning. Write down the final α and β values assigned to each node. What move will MAX choose?
   – What nodes will be pruned by the alpha-beta algorithm?

4. The games we discussed in class have only two players. Assume we have a perfect-information, deterministic game that has three players A, B and C who alternate taking turns – first A, then B, then C, then A, ... The game will result in one winner and two losers. A fellow student suggests modifying the minimax procedure to work for such a game.

   Based on information specific to each player, we have three independent static evaluation functions, Fa , Fb and Fc that are associated with players A, B and C, respectively. Each function indicates the estimated value of a board position with respect to that player. For example, $F_a(p) = -5$ means that position p is not good for player A, whereas $F_a(p) = 100$ means that position p looks very good for player A. At each leaf node n in the look-ahead minimax tree, we compute a triple of values ($F_a(n)$, $F_b(n)$, $F_c(n)$). At the level corresponding to A's choice, the value to be backed up is the triple which maximizes A's components. Similarly for B and C.

Using this scheme, what are the backed-up values that should be computed for the following game tree, where the root corresponds to player A's turn, nodes at depth one are positions where it's player B's turn, etc.

Identify and briefly discuss at least two problems with this approach, i.e., potential aspects of a three person game which this modification of minimax does not deal with.