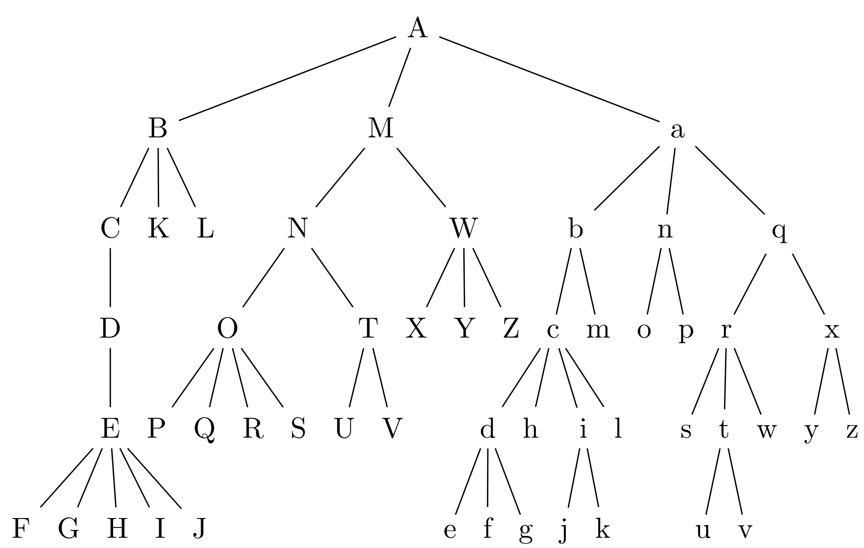
Work alone. Non-detailed discussion with a non-group member is allowed, but must be summarized and acknowledged by all involved. Viewing or exchanging written work, even in rough or preliminary form, is not allowed.

1. Acknowledge all resources,including discussions, texts, urls, etc. Acknowledge that you have read and understood the UAlberta document Understanding Plagiarism `http://www.science.ualberta.ca/en/FacultyAndStaff/~/media/science/Faculty%20And%20Staff/Documents/Understanding_Plagiarism.ashx`.

2. Theorem: Let $n \geq 2$. Let A be an n-Y board completely covered with stones. Each stone is black or white. Let A^- be the corresponding Y-reduced $(n-1)$-board. Then A has a winning configuration for black iff A^- does.

   Conjecture: Let $n \geq 2$. Let A be an n-Y position. Each cell is black, white, or empty. Let A^- be the corresponding Y-reduced $(n-1)$-position (use the majority rule with this tiebreaker: if a triangle has one black cell, one white cell, and one empty cell, then the corresponding cell on the reduced board is empty). Then A has a winning configuration for black iff A^- does.

   Prove or disprove: the theorem implies the conjecture.



3. For the above tree, assume that each leaf has proof and disproof value 1. (i) Find the proof and disproof values of all internal nodes, and all most proving nodes.

   (ii) Assume that a most proving node v is explored, with this result: v is a win for player to move (so, new negamax proof value of v is 0). For all tree nodes whose proof or disproof value is changed by this result, give the new value.

   (iii) Repeat (ii) for this result: loss for player to move.

   (iv) Repeat (ii) for this result: the node is expanded so that it has 5 children; their proof and disproof values are initialized to 1.

4. For a min-max tree, a *proof set* is a set L of leaves, such that if all elements of L are root-player wins, then root-player has a winning strategy. (i) Give a definition for

disproof set. (ii) A disproof set is *minimum* if it is a smallest proof set. Find a value of n and a min-max tree with n nodes, such that the number of minimum disproof sets is more than n.

5. The usual algorithm to find a most-proving node in a tree is to repeatedly descend to the child whose negamax disproof number is smallest. A min-max dag (directed acyclic graph) is the generalization of a min-max tree to games with transpositions but no cycles. For min-max dags, the following recursive negamax formula is used to approximate proof and disproof values: for each leaf t, define $p(t) = d(t) = 1$; for each internal node v, define $p(v) =$ min(over all children c) of $d(c)$, and define $d(v) =$ sum(over all children c) of $p(c)$. (i) Find a small min-max dag for which these formulas compute a wrong proof or disproof value for at least one node. (ii) Find a small min-max dag for which these formulas descend to a node that is *not* a most-proving node.