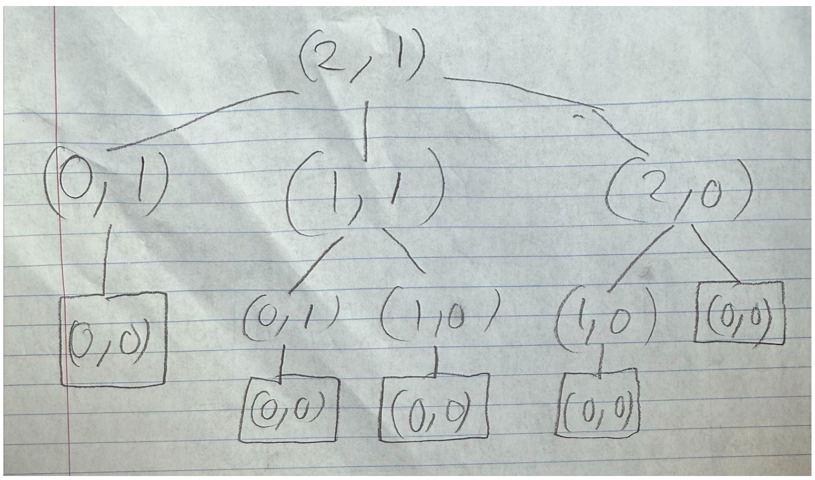
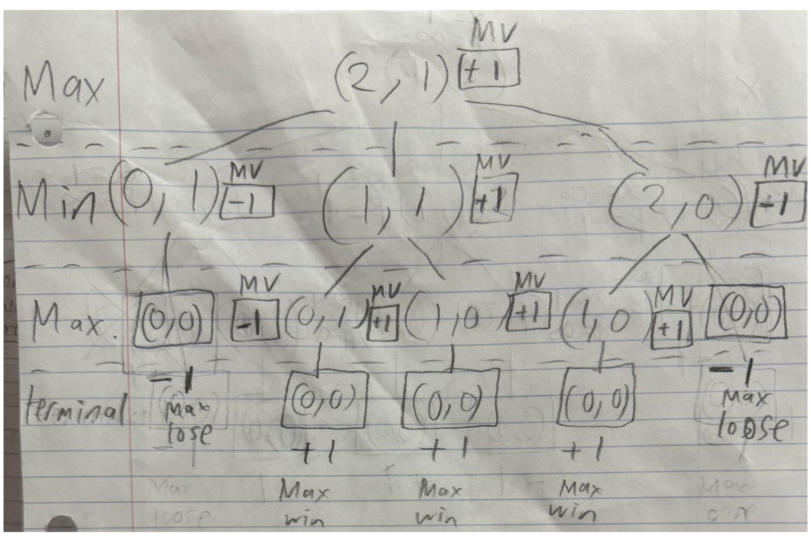
Problem 1

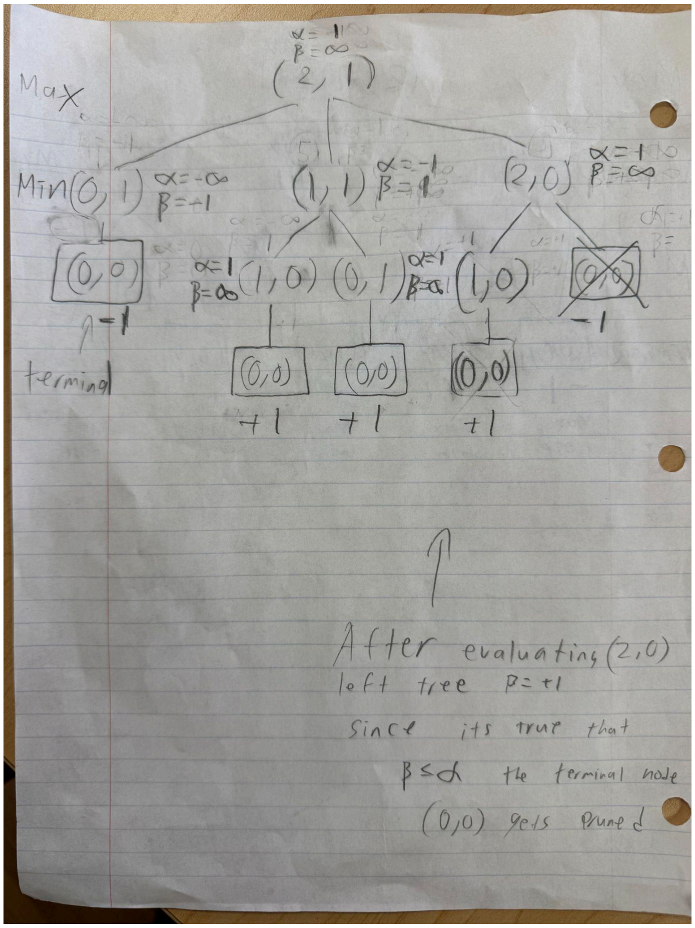
Nim’s game is a two-player strategy game played on a finite collection of piles containing discrete tokens, coins, or stones. The players take turns alternately, and on each turn a player must choose one pile and remove any positive number of objects from it. A player may not remove objects from more than one pile during a single turn. The game continues until all piles are empty, and the player who removes the last object from the final pile is declared the winner.



A tree with two heaps one being 2 and the other one 1.



Here shows the minimax algorithm applied to the game tree from above

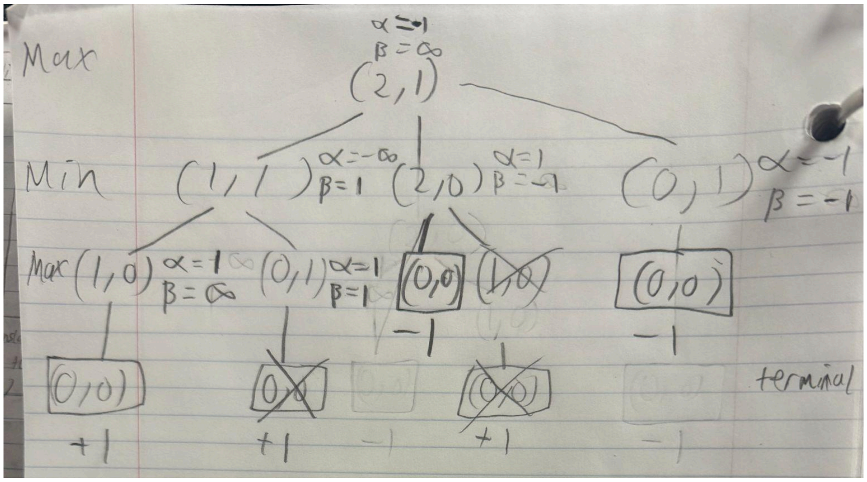


Here is the tree after the alpha beta pruning process removing nodes that don't need to be evaluated. The optimal path for max to win is going for the (1,1) state so he would have the perfect win.

The minimax algorithm is effective in solving the game of Nim because it evaluates every possible move and guarantees the best decision if both players play optimally. However, as the number of heaps and objects increases, the game tree grows exponentially, making a full search very complex. Alpha–beta pruning improves this by eliminating branches that cannot influence the final result, reducing the number of nodes that must be evaluated. The effectiveness of pruning depends on the move order—when good moves are explored first, many branches can be skipped. In smaller Nim games such as this one, minimax with alpha–beta pruning is both accurate and efficient, but in larger or more complex cases, even pruning cannot fully handle the exponential growth, making mathematical strategies more practical.

Problem 2

I looked into the optimization technique of move ordering, basically moving the most effective node to the left to be evaluated first. This way the alpha beta pruning works more effectively and improves the overall performance on bigger trees.



As for my example more branches got pruned so there are less nodes to be evaluated therefore increasing the performance of the algorithm.

The heuristic can improve the performance of the minimax algorithm by using the XOR of all heap sizes to estimate game states. In Nim, if the XOR (nim-sum) equals -1, the position is losing, but if it is +1, the position is winning. By applying this rule, the algorithm can quickly evaluate positions without fully exploring every possible move. This allows minimax to estimate the best move faster, reducing computation time while maintaining strong decision quality.