CS6601 Challenge Question 3 - Game Playing

Raymond Jia

Fall 2024 - Solutions

Note*: Please review the course textbook AIMA 4th Edition Chapter 5, Canvas Game Playing Modules, and Assignment 2 before attempting this Challenge Question.

Introduction

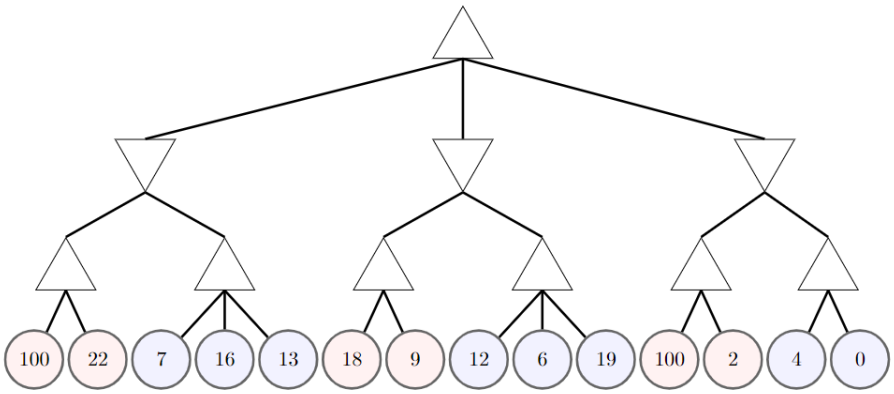

Figure 1: Game Tree A

Consider a two-player game where players take turns to make moves and the utility of each state is a whole number. We are playing as player one, and it is currently our turn to make a move (represented by a max node at depth 0). The utility for our game states have a minimum value of -100 (game lost) and a maximum value of 100 (game won). The game tree of depth 3 at this current turn can be see in Figure 1 above, with the leaf nodes showing the utility values of the game states at depth 3.

Q1 [1pt]

Using **Minimax**, what is the value of the utility value that will be propagated to the top in Game Tree A seen in Figure 1? (*AIMA 4th Ed. Section 5.2*)

Answer: 18

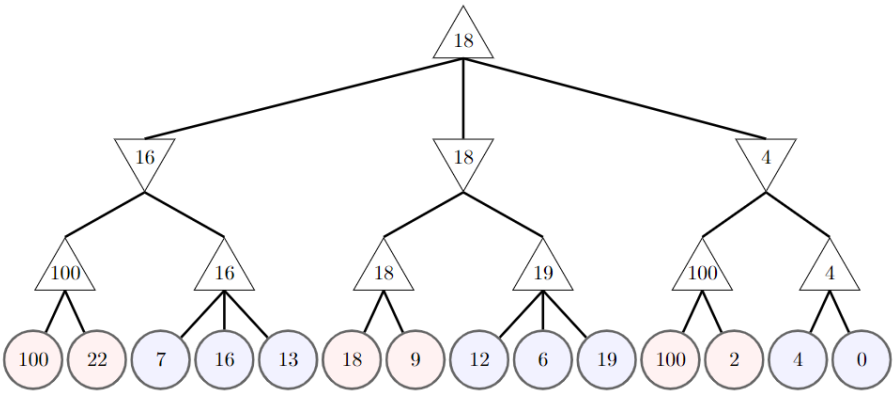

Figure 2: Game Tree A - Minimax Solution

Q2 [2pt]

If we perform **Minimax** with **iterative deepening** on Game Tree A seen in Figure 1 starting at a depth of 0 and incrementing the depth limit by 1 until a depth limit of 3 is reached, how many nodes will be explored in total during the process?

Answer: 39

Solution: At depth limit 0, 1 node is explored. At depth limit 1, 4 nodes are explored. At depth limit 2, 10 nodes are explored. At depth limit 3, 24 nodes are explored. Altogether $1 + 4 + 10 + 24 = 39$ nodes are explored.

Q3 [2pt]

If we use **Alpha-Beta Pruning** on Game Tree A seen in Figure 1, how many leaf nodes will be pruned? (*AIMA 4th Ed. Section 5.2.3*)

Answer: 2

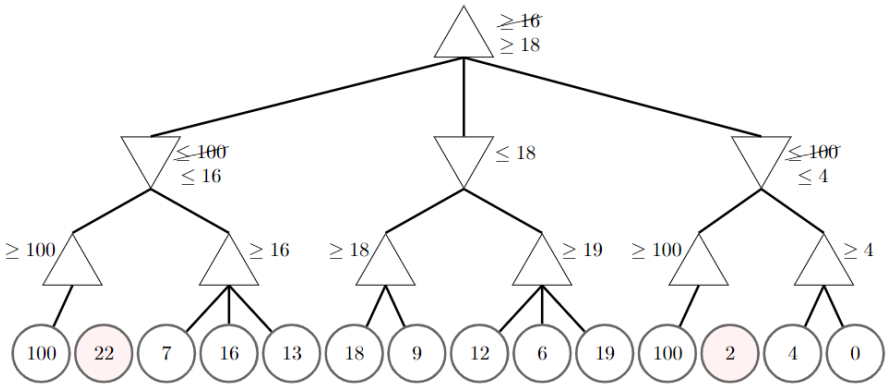

Figure 3: Game Tree A - Alpha-Beta Pruning Solution

Solution: Indexing leaf nodes from left to right starting with 0. Leaf node 1 with value 22 is pruned because the parent max node takes on limit ≥ 100 before exploration and 100 is the maximum value a utility score can have, therefore no further exploration of this area of the tree is necessary and the leaf can be pruned. Leaf node 11 with value 2 is pruned because of the same reason.

Q4 [2pt]

If we re-order the nodes in the game tree, we can make **Alpha-Beta Pruning** more effective by pruning more nodes. How many leaf nodes will be pruned if we use Game Tree B seen in Figure 4 instead of Game Tree A?

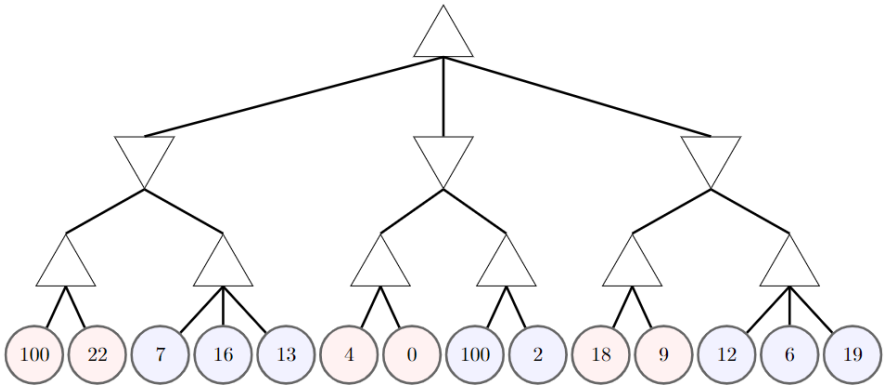

Figure 4: Game Tree B

Answer: 3

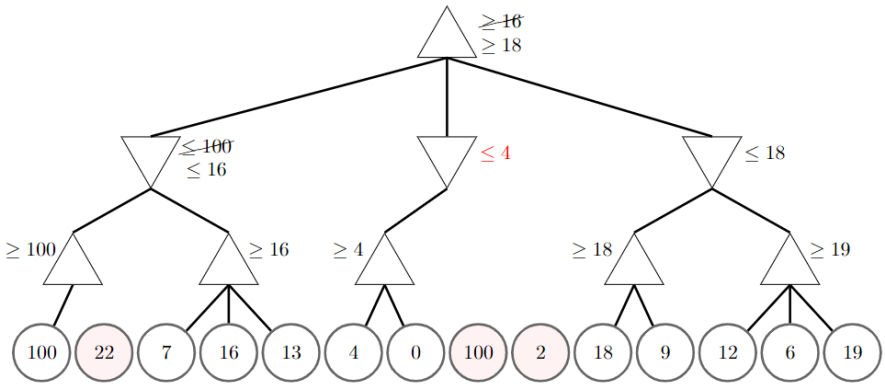

Figure 5: Game Tree B - Alpha-Beta Pruning Solution

Solution: Indexing leaf nodes from left to right starting with 0. Leaf node 1 with value 22 is pruned because the parent max node takes on limit ≥ 100 before exploration and 100 is the maximum value a utility score can have, therefore no further exploration of this area of the tree is necessary and the leaf can be pruned. Leaf nodes 7 and 8 with values 100 and 2 are pruned because the parent of their parent discovers requirement ≤ 4 which directly contradicts with the lower bound of ≥ 16, hence further exploration of that section of the tree is no longer necessary and nodes 7 and 8 are not explored.

Q5 [3pts]

Re-order the game tree such that the number of nodes pruned using **Alpha-Beta Pruning** is maximized. How many nodes will be pruned when using the re-ordered tree?

Answer: 6

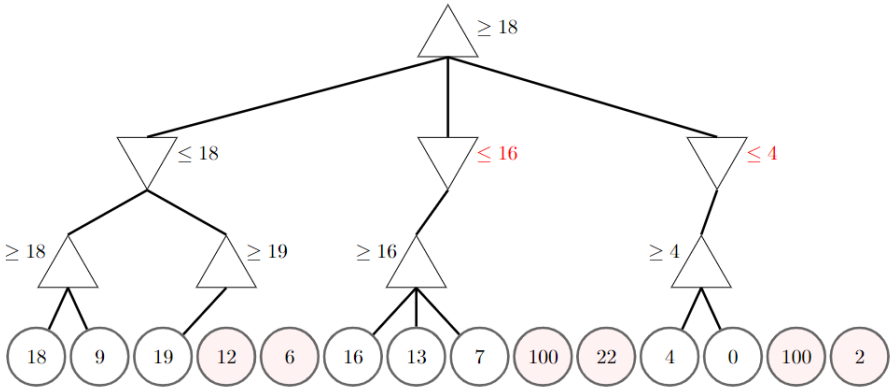

Figure 6: Best Alpha-Beta Pruning Tree Solution

Solution: There are multiple trees that can yield the maximum leaf nodes pruned, but the most simply way to obtain such a tree is to sort nodes at each level based on their desired max or min operation. For example at depth 3 in this problem, we can group the nodes by their parents and sort them in descending order because the parents are max nodes. This ensures that if the first node seen will be the greatest node and increases the likelihood that either a utility limit (e.g. 100) is seen immediately or a Alpha-Beta bound is violated immediately. The tree created through this method can be seen in Figure 6

Indexing leaf nodes from left to right starting with 0. Leaf nodes 3 and 4 with values 12 and 6 are pruned because their parent discovers a violation of the lower bound ≥ 18 when it obtains the bound of ≥ 19. Leaf nodes 8 and 9 with values 100 and 22 are pruned because the parent of their parent discovers a violation of the lower bound of ≥ 18 when it obtains the bound of ≤ 16. Leaf nodes 12 and 13 with values 100 and 2 are pruned because the parent of their parent discovers a violation of the lower bound of ≥ 18 when it obtains the bound of ≤ 4.