Alpha-Beta Pruning algorithm

October 25, 2024

[2]:
```
import math

def minimax(depth, nodeIndex, maximizingPlayer, values, alpha, beta, maxDepth):
    if depth == maxDepth:
        return values[nodeIndex]

    if maximizingPlayer:
        maxEval = -math.inf
        for i in range(2):
            eval = minimax(depth + 1, nodeIndex * 2 + i, False, values, alpha,␣
 ↪beta, maxDepth)
            maxEval = max(maxEval, eval)
            alpha = max(alpha, eval)
            if beta <= alpha:
                break
        return maxEval
    else:
        minEval = math.inf
        for i in range(2):
            eval = minimax(depth + 1, nodeIndex * 2 + i, True, values, alpha,␣
 ↪beta, maxDepth)
            minEval = min(minEval, eval)
            beta = min(beta, eval)
            if beta <= alpha:
                break
        return minEval

# Get depth and values from user
maxDepth = int(input("Enter the depth of the game tree: "))
values = list(map(int, input("Enter the leaf node values separated by spaces:␣
 ↪").replace(',', ' ').split()))

alpha = -10000
beta = 10000
print("The optimal value is:", minimax(0, 0, True, values, alpha, beta,␣
 ↪maxDepth))
```

```
Enter the depth of the game tree:  3
Enter the leaf node values separated by spaces:  3 5 6 9 1 2 0 -1
The optimal value is: 5
```

```
[ ]:
```