Name: Niraj Milind Amrutkar

Roll No. 332002

PRN: 22010910

Batch B1

Assignment No. 4

- Implementation of Min-Max Search Procedure with alpha beta for finding the solutions of games

Aim:

Implementation of Min-Max Search Procedure with alpha-beta pruning for finding the solutions of games.

Objective:

To Implement Min-Max Search Procedure with alpha-beta pruning for finding the solutions of games.

Theory:

Mini-max algorithm is a recursive or backtracking algorithm which is used in decision-making and game theory. It provides an optimal move for the player assuming that opponent is also playing optimally.

Mini-Max algorithm uses recursion to search through the game-tree. Min-Max algorithm is mostly used for game playing in AI. Such as Chess, Checkers, tic-tac-toe, go, and various tow-players game. This Algorithm computes the minimax decision for the current state. In this algorithm two players play the game; one is called MAX and other is called MI. Both the players fight it as the opponent player gets the minimum benefit while they get the maximum benefit. Both Players of the game are opponent of each other, where MAX will select the maximized value and MIN will select the minimized value.

The minimax algorithm performs a depth-first search algorithm for the exploration of the complete game tree.The minimax algorithm proceeds all the way down to the terminal node of the tree, then backtrack the tree as the recursion.

ALPHA BETA PRUNING

Alpha-beta pruning is a modified version of the minimax algorithm. It is an optimization technique for the minimax algorithm. As we have seen in the

minimax search algorithm that the number of game states it has to examine are exponential in depth of the tree. Since we cannot eliminate the exponent, but we can cut it to half. Hence there is a technique by which without checking each node of the game tree we can compute the correct minimax decision, and this technique is called pruning. This involves two threshold parameter Alpha and beta for future expansion, so it is called alpha-beta pruning. It is also called as Alpha-Beta Algorithm.

Alpha-beta pruning can be applied at any depth of a tree, and sometimes it not only prunes the tree leaves but also entire sub-tree.

The two-parameter can be defined as:

a. Alpha: The best (highest-value) choice we have found so far at any point along the path of Maximiser. The initial value of alpha is -∞.

b. Beta: The best (lowest-value) choice we have found so far at any point along the path of Minimizer. The initial value of beta is +∞.

Code:

```
MAX, MIN = 1000, -1000
def minimax(depth, nodeIndex, maximizingPlayer,values, alpha, beta):
    if depth == 3:
        return values[nodeIndex]

    if maximizingPlayer:
        best = MIN
        for i in range(0, 2):
            val = minimax(depth + 1, nodeIndex * 2 + i,False, values, alpha,
beta)
            best = max(best, val)
            alpha = max(alpha, best)
            # Alpha Beta Pruning
            if beta <= alpha:
                break
        return best

    else:
        best = MAX
        for i in range(0, 2):
            val = minimax(depth + 1, nodeIndex * 2 + i,True, values, alpha,
beta)
            best = min(best, val)
            beta = min(beta, best)
            # Alpha Beta Pruning
```

```
            if beta <= alpha:
                break

        return best

if __name__ == "__main__":
    values = [3, 5, 6, 9, 1, 2, 0, -1]
    print("The optimal value \n->", minimax(0, 0, True, values, MIN, MAX))
```

Output:

```
ligence\Assignment 4\MinMax.py"
The optimal value
-> 5
```