National University of Computer and Emerging Sciences

CL 461 Artificial Intelligence

Adversarial Search (Minimax Algorithm- Alpha-Beta Pruning)

Adversarial search

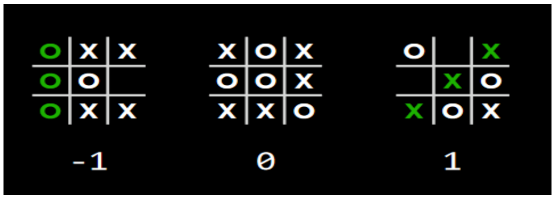
Adversarial search is a game-playing technique where the agents are surrounded by a competitive environment. A conflicting goal is given to the agents (multiagent). These agents compete with one another and try to defeat one another in order to win the game. Such conflicting goals give rise to the adversarial search. Here, game-playing means discussing those games where human intelligence and logic factor is used, excluding other factors such as luck factor. Tic-tac-toe, chess, checkers, etc., are such type of games where no luck factor works, only mind works.

We are opponents- I win, you loose.

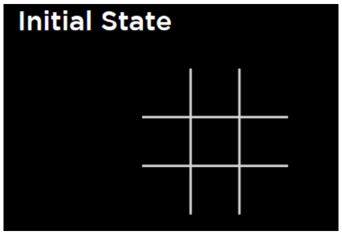
Minimax Algorithm

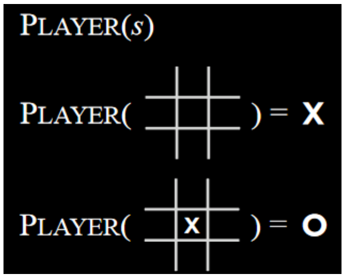
Minimax is a kind of backtracking algorithm that is used in decision making and game theory to find the optimal move for a player, assuming that your opponent also plays optimally. It is widely used in two player turn-based games such as Tic-Tac-Toe, Backgammon, Mancala, Chess, etc.

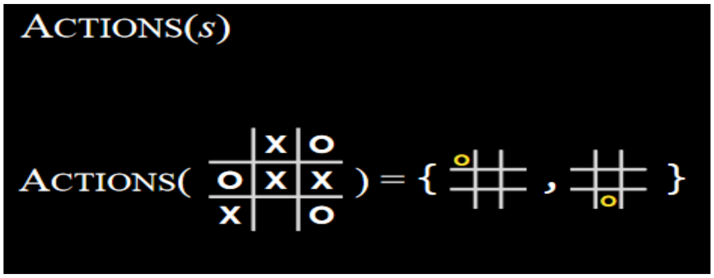
Max(x) aims to maximize score
Min(o) aims to minimize score

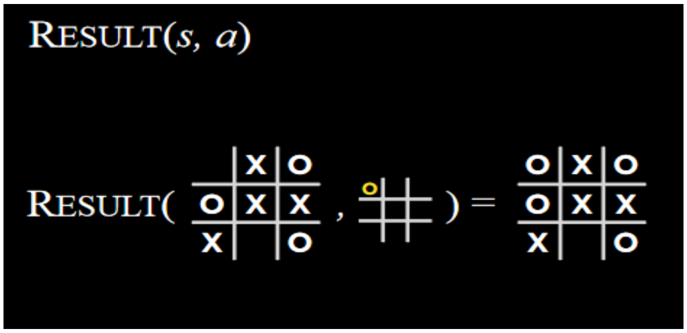


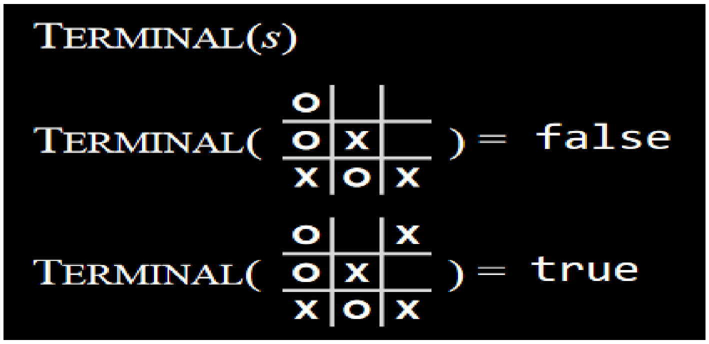
Elements:
S(0) : Initial state
Player(s): returns which player to move in state s
Action(s): returns legal moves in state s
Result(s, a): returns state after action a taken in state s
Terminal (s): checks if state s is a terminal state
Utility(s, p): final numerical value for terminal state s



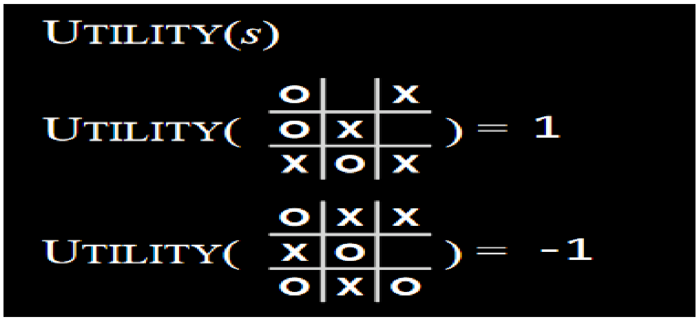








For Chess, the outcomes are a win, loss, or draw and its payoff values are +1, 0, ½.
For tic-tac-toe, utility values are +1, -1 , and 0.



Given a state s:

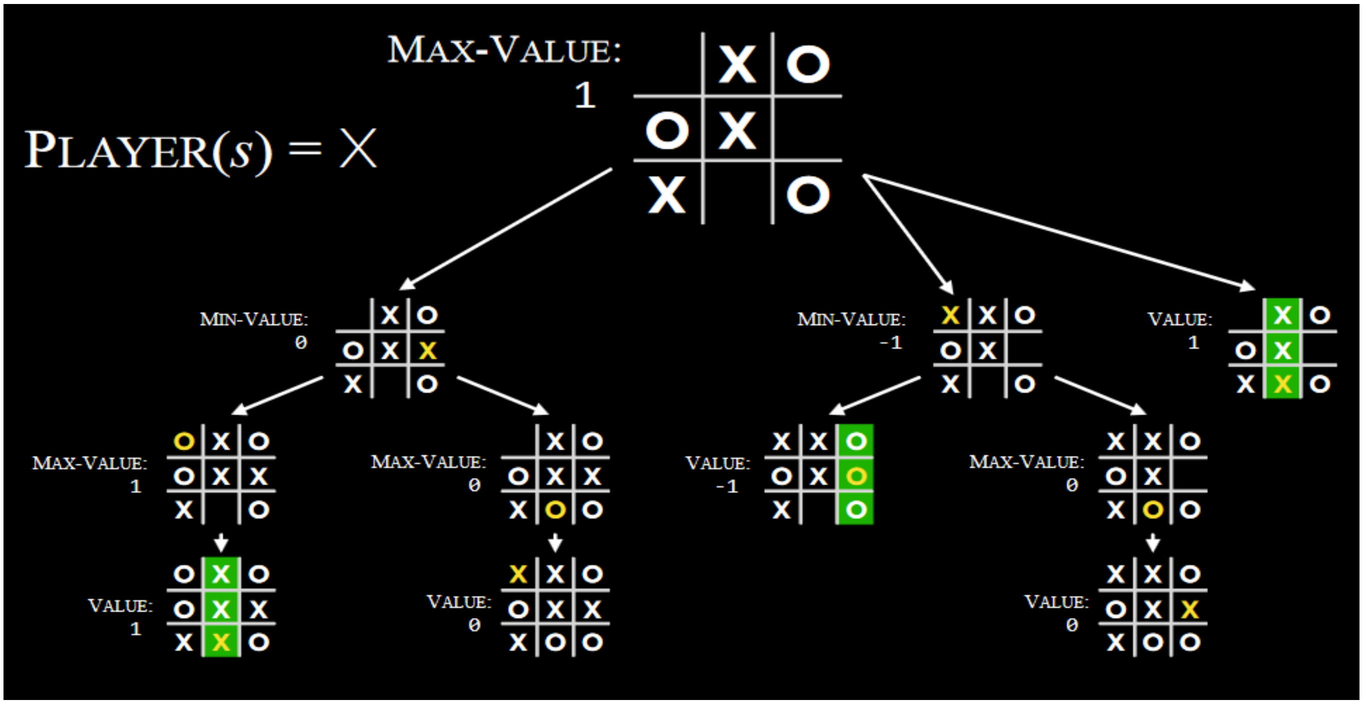
- MAX picks action a in Action(s) that produces highest value of MIN-value (Result(s,a))
- MIN picks action a in Action(s) that produces smallest value of MAX-value (Result(s,a))

Algorithm

```
function MAX-VALUE(state):
        if Terminal(state):
                return Utility(state)
        v=-inf
        for action in Actions(state):
                v=MAX(v , MIN-Value(Result(state, action)))
        return v

function MIN-VALUE(state):
        if Terminal(state):
                return Utility(state)
        v=-inf
        for action in Actions(state):
                v=MIN(v , MAX-Value(Result(state, action)))
        return v
```

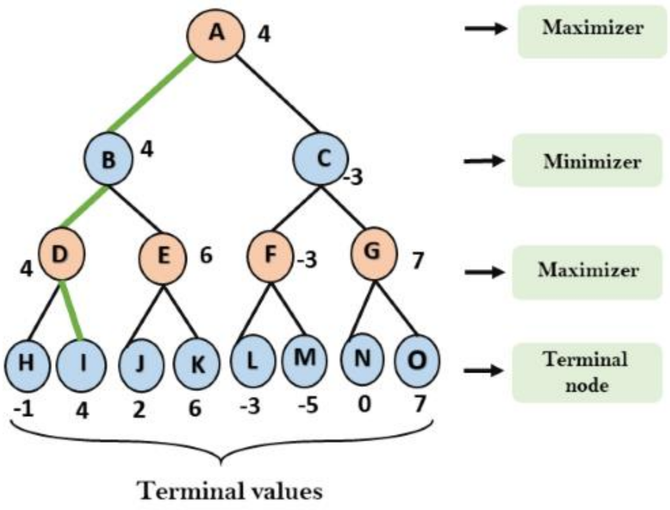
Game Tree:

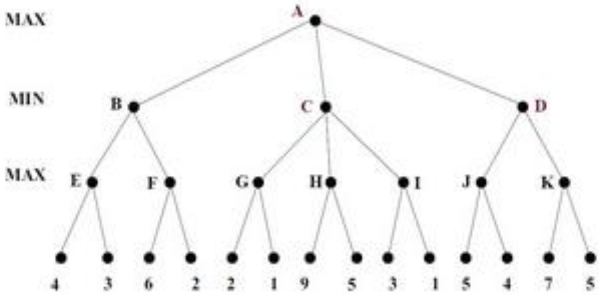
A game tree is a tree where nodes of the tree are the game states and Edges of the tree are the moves by players. Game tree involves initial state, actions function, and result Function.



Exercise 1 :

Implement the following tree using minimax algorithm.





Alpha Beta Pruning

Alpha-beta pruning is a modified version of the minimax algorithm. It is an optimization technique for the minimax algorithm. Alpha-beta pruning can be applied at any depth of a tree, and sometimes it not only prune the tree leaves but also entire sub-tree.

The two-parameter can be defined as:

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

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 +∞.

Condition:

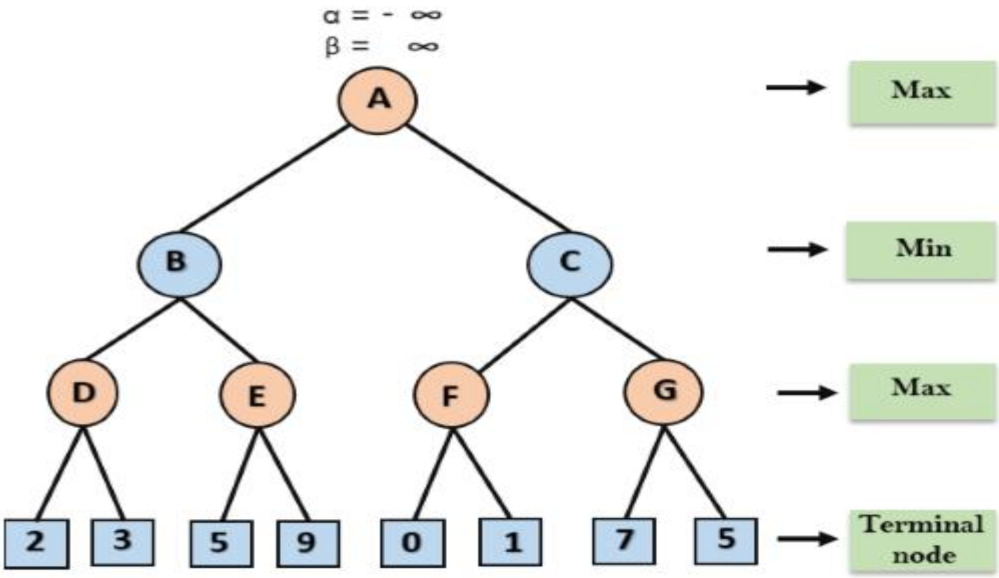
$\alpha>=\beta$

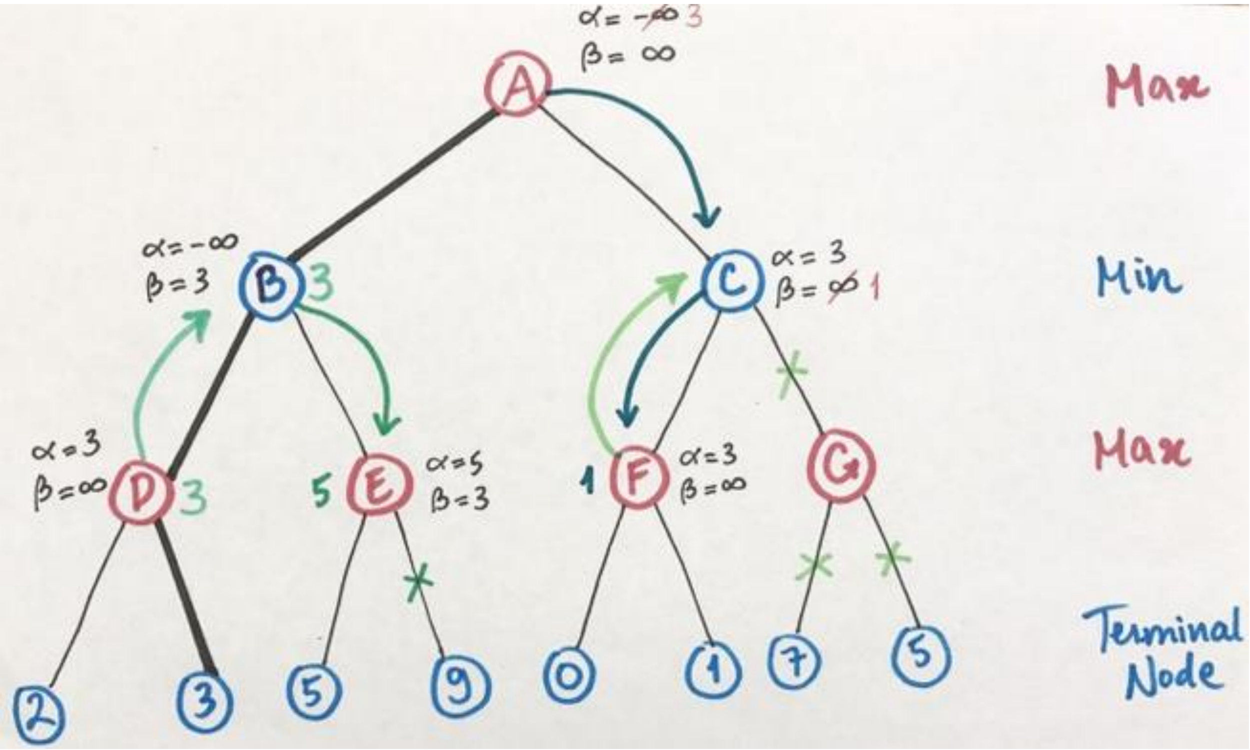
Algorithm:

```
function Alpha-Beta-Search(state):
    v=MAX(state, -inf, +inf)
    return action in Actions(state) with value v

function MAX-VALUE(state,alpha,beta):
    if Terminal(state):
        return Utility(state)
    v=-inf
    for action in Actions(state):
        v=MAX(v , MIN-Value(Result(state, action, alpha,beta)))
        if v >= beta:
            return v
        if  v>=alpha:
            alpha=v
    return v

function MIN-VALUE(state, alpha, beta):
    if Terminal(state):
        return Utility(state)
    v=+inf
    for action in Actions(state):
        v=MIN(v , MAX-Value(Result(state, action), alpha, beta))
        if v <= alpha:
            return v
        if  v<=beta:
            beta=v
    return v
```

Key- Points:

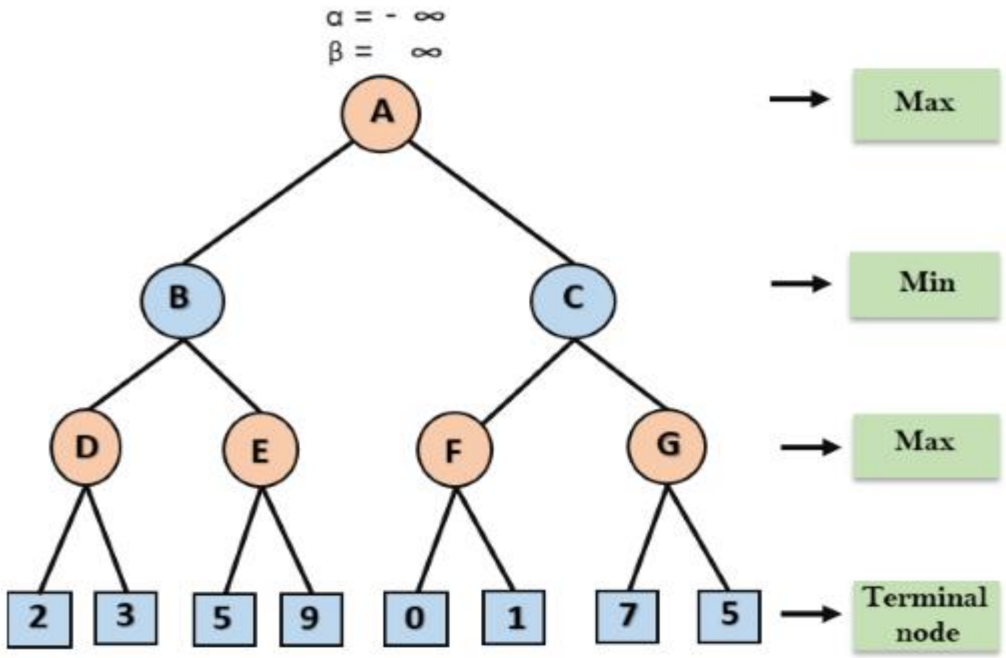
- The Max player will only update the value of alpha.
- The Min player will only update the value of beta.
- While backtracking the tree, the node values will be passed to upper nodes instead of values of alpha and beta.
- We will only pass the alpha, beta values to the child nodes.

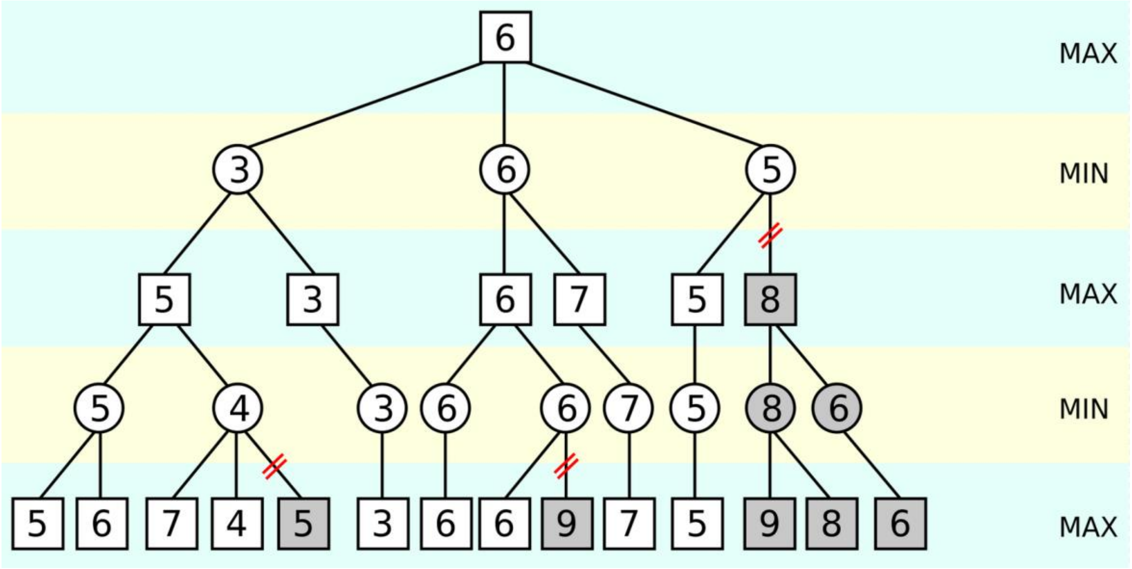




Exercise 2 :

Implement the following tree using alpha-beta pruning algorithm.


α = - ∞
β = ∞
A
B
C
D
E
F
G
2
3
5
9
0
1
7
5
Max
Min
Max
Terminal node


6
MAX
3
6
5
MIN
5
3
6
7
5
8
MAX
5
4
3
6
6
7
5
8
6
MIN
5
6
7
4
5
3
6
6
9
7
5
9
8
6
MAX