# Unit-3

## Adversarial Search

**Game Playing, Min-Max Search ,**
**Alpha-Beta Pruning**

# Topics

Games, Optimal Decision in Games, Alpha-Beta Pruning Minimax Search Procedure, Adding Alpha-beta Cut-offs, Iterative Deepening.

# Game Playing in Artificial Intelligence

Game Playing is an important domain of artificial intelligence.

Games don’t require much knowledge; the only knowledge we need to provide is the rules, legal moves and the conditions of winning or losing the game.

Both players try to win the game. So, both of them try to make the best move possible at each turn.

Searching techniques like BFS(Breadth First Search) are not accurate for this as the branching factor is very high, so searching will take a lot of time. So, we need another search procedures that improve –

- **Generate procedure** so that only good moves are generated.
- **Test procedure** so that the best move can be explored first.

# Minimax Search Procedure

- The most common search technique in game playing is Minimax search procedure.
- It is depth-first depth-limited search procedure.
- It is used for games like tic-tac-toe.

**Minimax algorithm uses two functions –**

- **MOVEGEN :** It generates all the possible moves that can be generated from the current position.
- **STATIC EVALUATION :** It returns a value depending upon the goodness from the viewpoint of two-player

# Minimax Search Procedure

- Backtracking Algorithm
- Best Move strategy is used
- Max will try to maximize its utility
- Min will try to minimize its utility

## Backtracking Technique: An Overview

**Backtracking** is a general algorithmic technique used for solving problems incrementally by building candidates for solutions and abandoning a candidate as soon as it is determined that it cannot be extended to a valid solution. It is particularly useful for constraint satisfaction problems (CSPs) and combinatorial problems.

# Key Concepts

- 1. State Space Tree:
- - Visual representation of the search space.
- - Nodes represent partial solutions.

- 2. Recursive Search:
- - Explores possible solutions through recursion.

- 3. Pruning:
- - Eliminates branches that cannot yield valid solutions.

# Steps in the Backtracking Algorithm

- 1. Choose a Candidate:
- - Start from an empty solution and select a candidate.

- 2. Check Constraints:
- - Validate the candidate against problem constraints.

- 3. Recursively Build the Solution:
- - Extend the current partial solution recursively.

- 4. Backtrack:
- - Undo the last choice if no valid solution is found.

- 5. Repeat:
- - Continue until all candidates are explored.

# Applications of Backtracking

- 1. Combinatorial Problems:
- - Generating permutations and combinations.

- 2. Constraint Satisfaction Problems (CSPs):
- - N-Queens problem, Graph coloring.

- 3. Puzzle Solving:
- - Logic puzzles, mazes, and games.

- 4. Pathfinding:
- - Finding paths in a grid or graph.

# Advantages of Backtracking

- 1. Flexibility:
- - Applicable to a wide range of problems.

- 2. Simplicity:
- - Easy to implement due to its recursive nature.

- 3. Efficient Search:
- - Prunes invalid solutions for performance gains.

# Disadvantages of Backtracking

- 1. Exponential Time Complexity:
- - May require exponential time in the worst case.

- 2. Inefficient for Large Problems:
- - Performance can degrade significantly as problem size increases.

# Example: N-Queens Problem

- Goal:
- - Place N queens on an N x N chessboard without threats.

- Backtracking Steps:
- 1. Choose a row and try placing a queen in each column.
- 2. Validate placement against existing queens.
- 3. Recursively attempt to place the next queen.
- 4. Backtrack if no valid placements are found.

# Minimax Search Procedure

- This algorithm is a two player game, so we call the first player as PLAYER1 and second player as PLAYER2.
- The value of each node is backed-up from its children.
- For PLAYER1 the backed-up value is the maximum value of its children and for PLAYER2 the backed-up value is the minimum value of its children.
- It provides most promising move to PLAYER1, assuming that the PLAYER2 has make the best move.
- It is a recursive algorithm, as same procedure occurs at each level.

# Minimax Search Procedure

- Minimax is a classic depth-first search technique for a sequential two-player game.
- The two players are called MAX and MIN.
- The minimax algorithm is designed for finding the optimal move for MAX, the player at the root node.
- The search tree is created by recursively expanding all nodes from the root in a depth-first manner until either the end of the game or the maximum search depth is reached.

# Example-

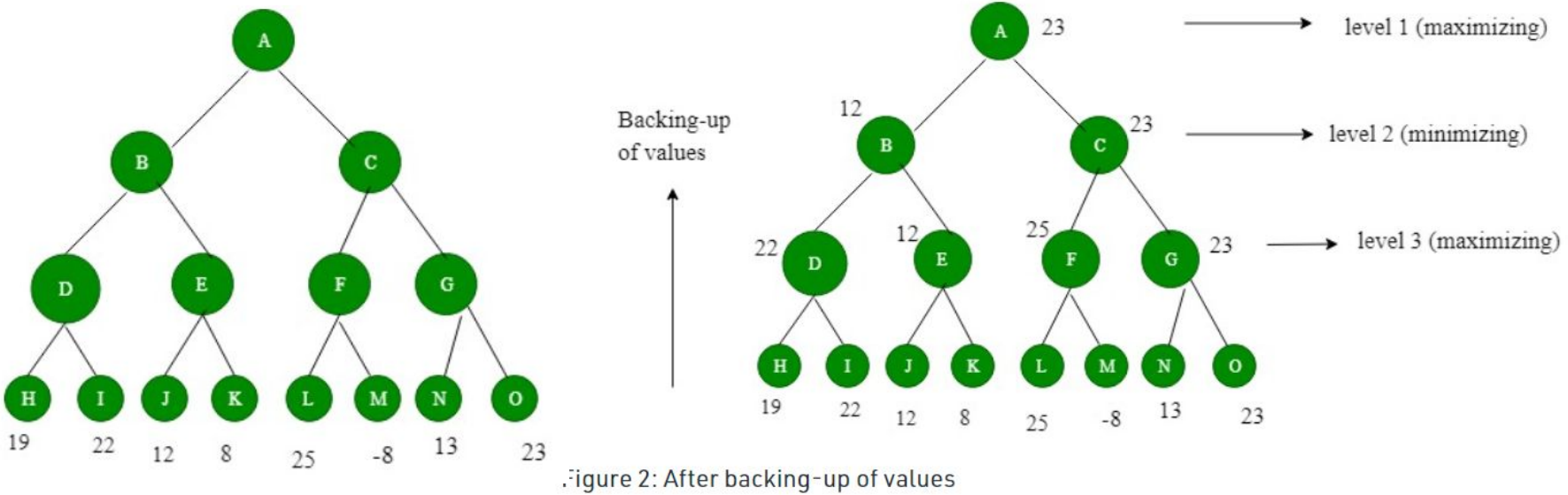



Figure 2: After backing-up of values

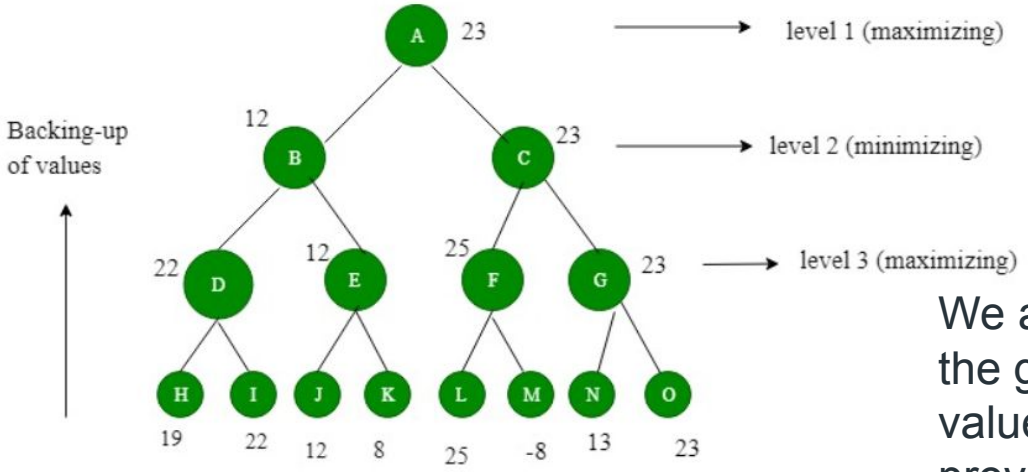
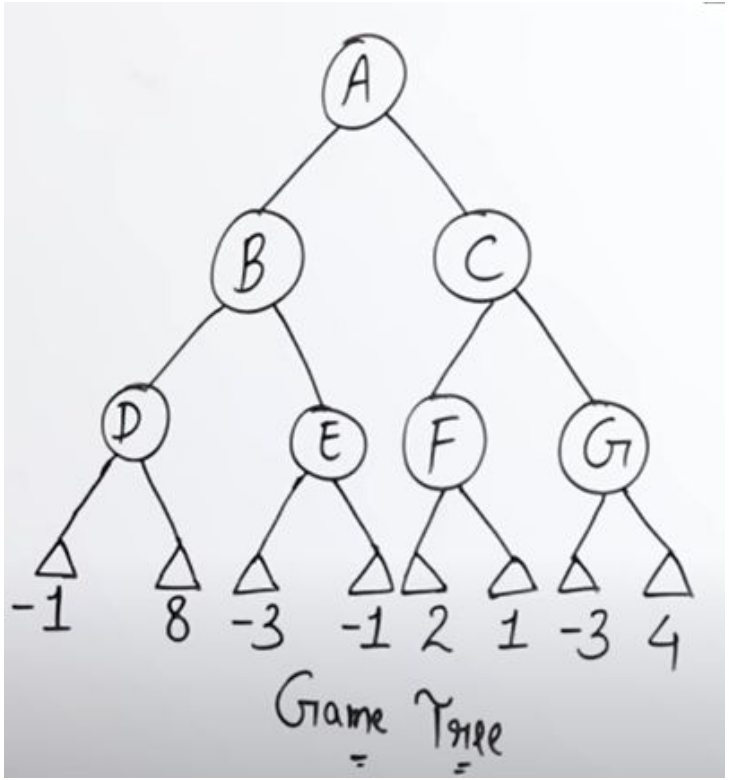
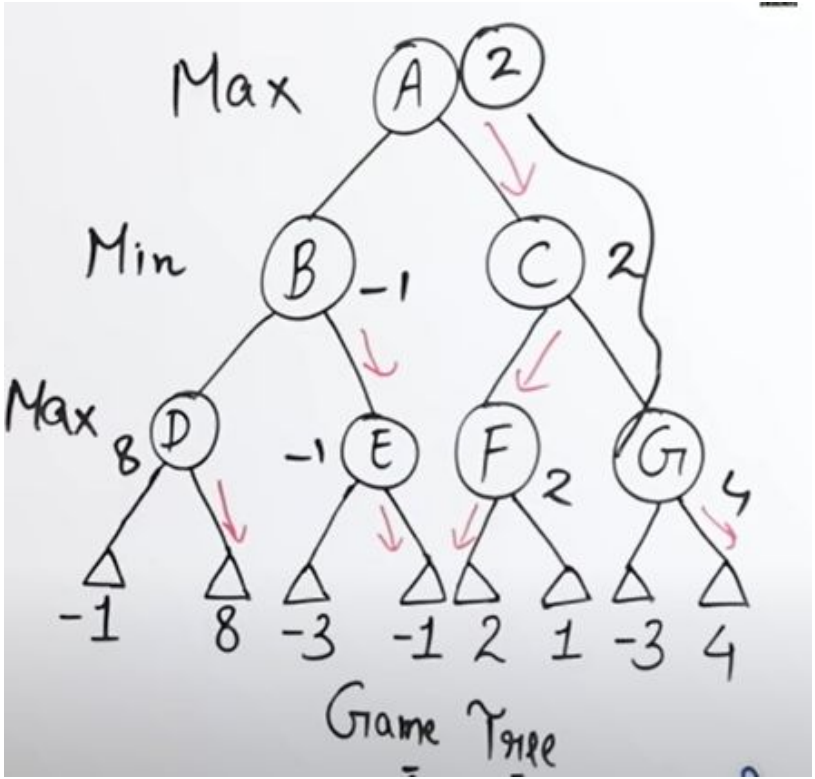



Figure 2: After backing-up of values

We assume that PLAYER1 will start the game. 4 levels are generated. The value to nodes H, I, J, K, L, M, N, O is provided by STATICEVALUATION function. Level 3 is maximizing level, so all nodes of level 3 will take maximum values of their children. Level 2 is minimizing level, so all its nodes will take minimum values of their children. This process continues. The value of A is 23. That means A should choose C move to win.

# Example-


A
B
C
D
E
F
G
-1
8
-3
-1
2
1
-3
4
Game Tree


Max
A
2
Min
B
-1
C
2
Max
8
D
-1
E
F
2
G
4
-1
8
-3
-1
2
1
-3
4
Game Tree

Time complexity -

$O(b^d)$

**‘b’** is the branching factor (choices/children) and ‘**d**’ is the depth(ply)

# Alpha Beta Pruning - Example

Alpha beta pruning is an optimisation technique for the minimax algorithm.

The word ‘pruning’ means cutting down branches and leaves.

Alpha-beta pruning is nothing but the pruning of useless branches in decision trees.

The need for pruning came from the fact that in some cases decision trees become very complex. In that tree, some useless branches increase the complexity of the model.

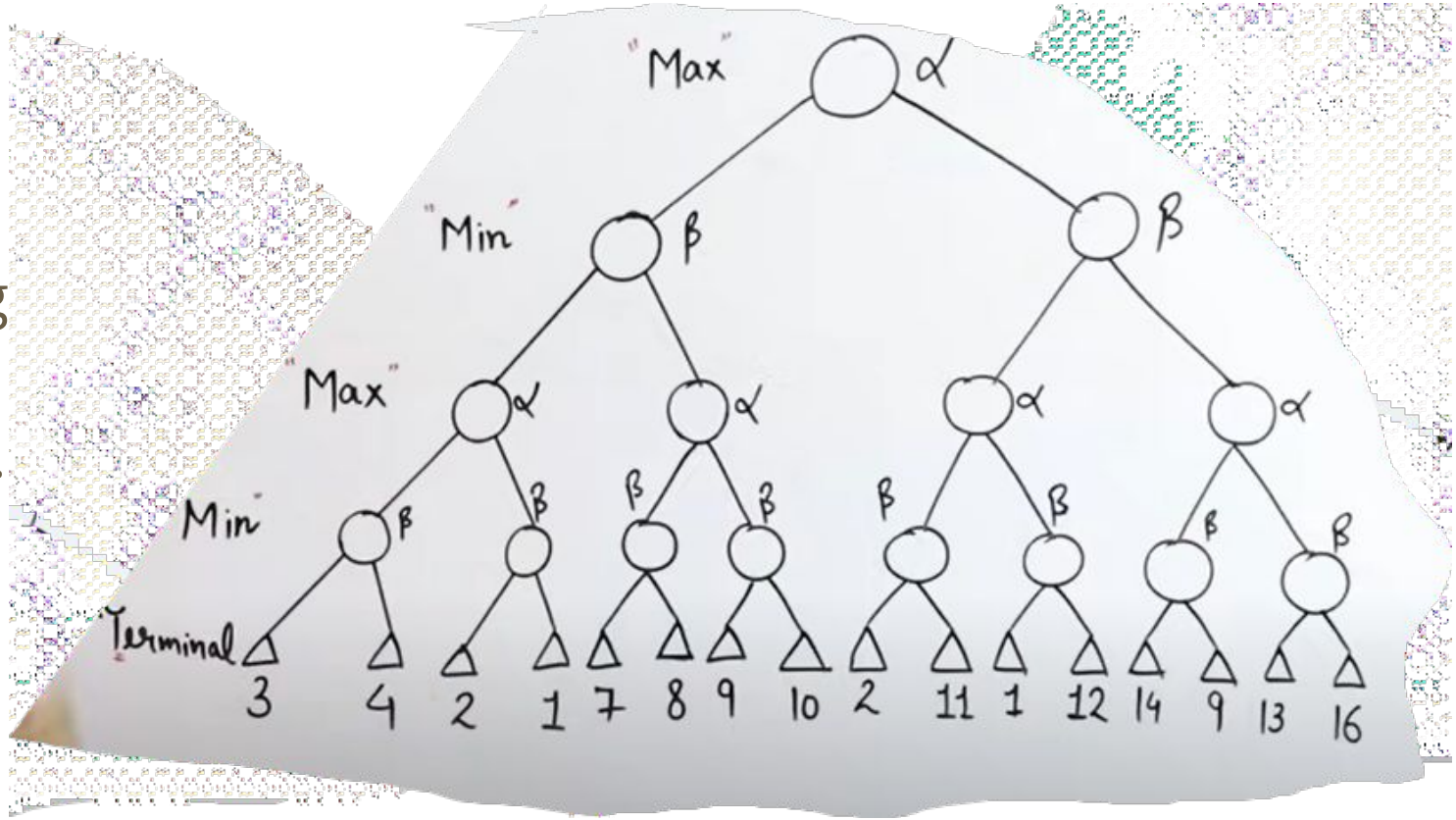
So, to avoid this, Alpha-Beta pruning comes to play so that the computer does not have to look at the entire tree.

# Alpha Beta Pruning - Example

- Alpha: At any point along the Maximizer path, Alpha is the best option or the highest value we've discovered.
- Beta: At any point along the Minimizer path, Beta is the best option or the lowest value we've discovered..
- The condition for Alpha-beta Pruning is that $\alpha >= \beta$.
- MAX will update only alpha values and the MIN player will update only beta values.
- The node values will be passed to upper nodes instead of alpha and beta values during going into the tree's reverse.
- Alpha and Beta values only are passed to child nodes.

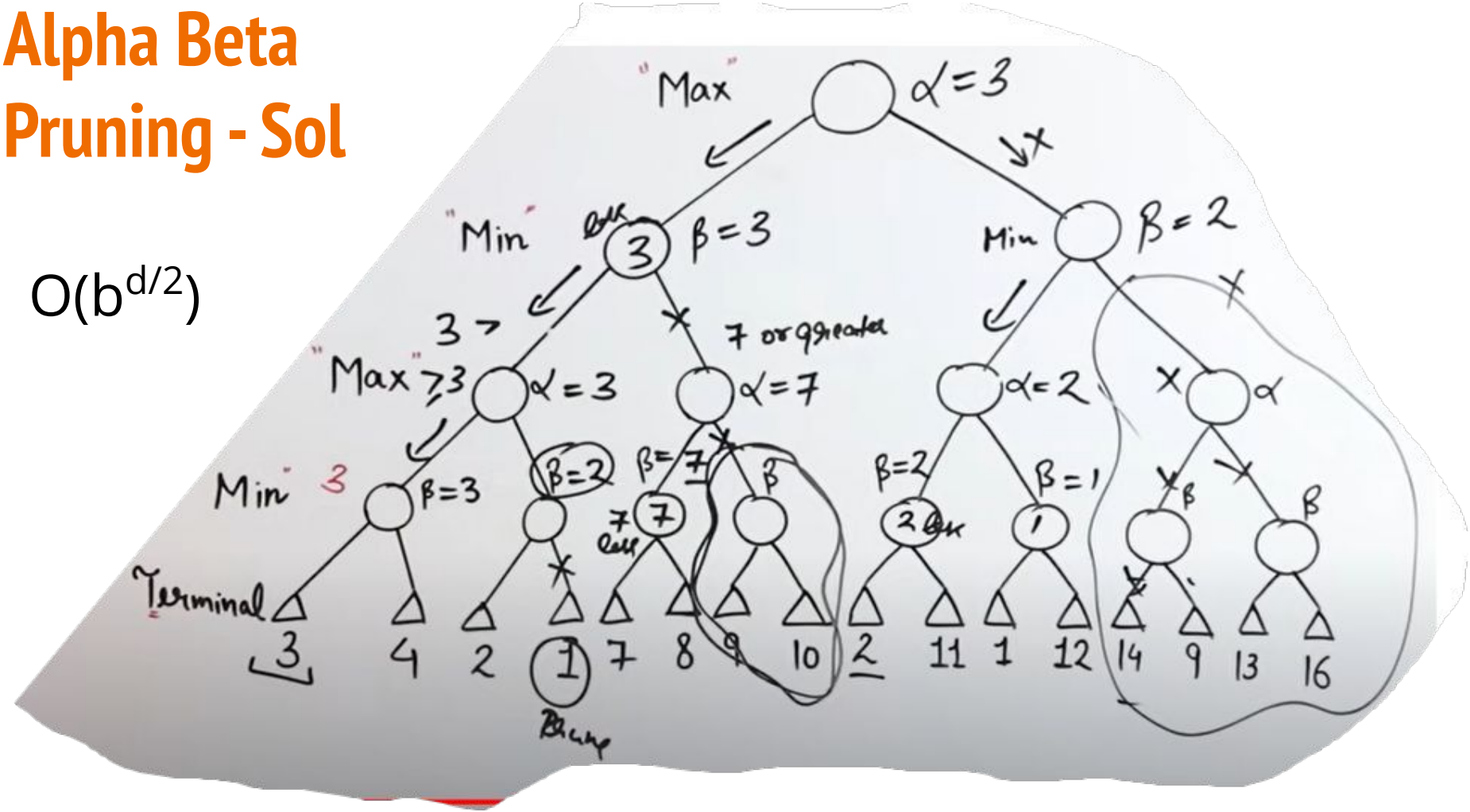
# Alpha Beta Pruning - Example

Cut-off search by exploring less no of nodes.

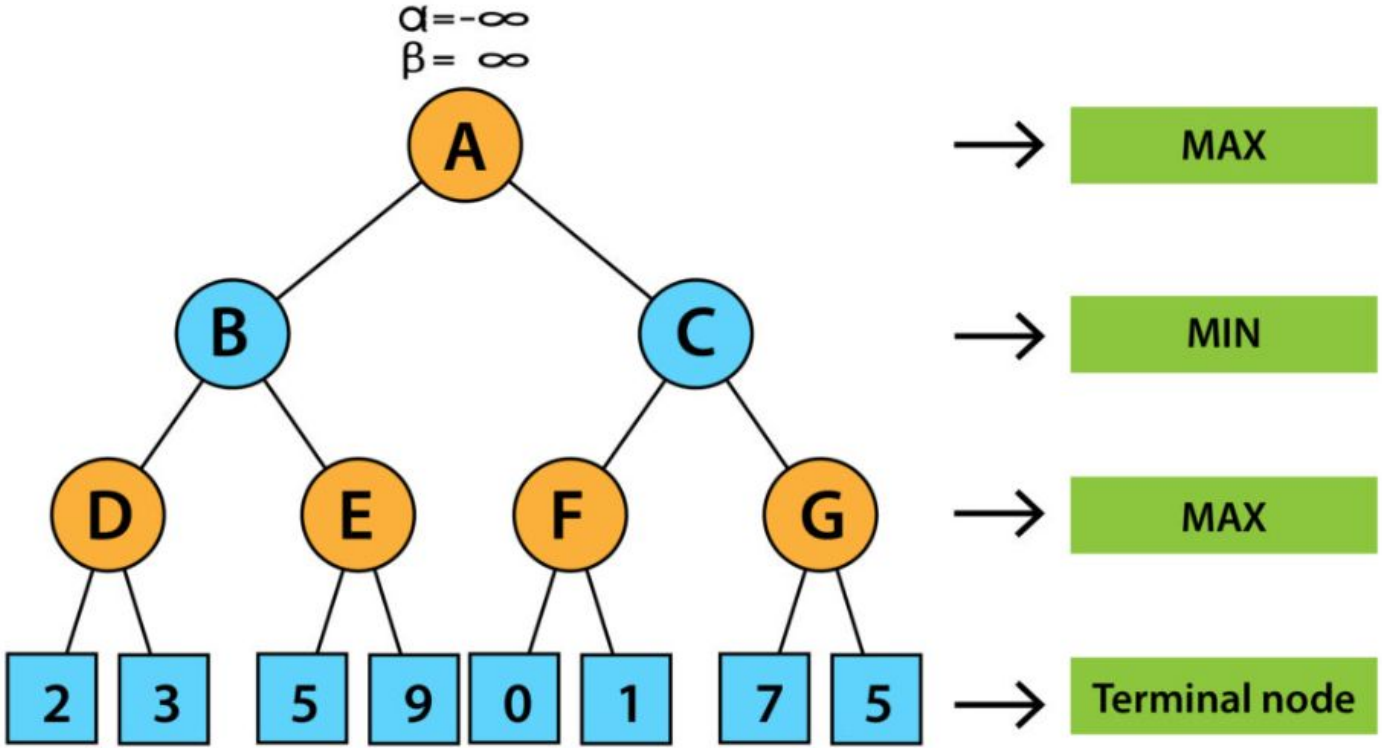


# Alpha Beta Pruning - Sol

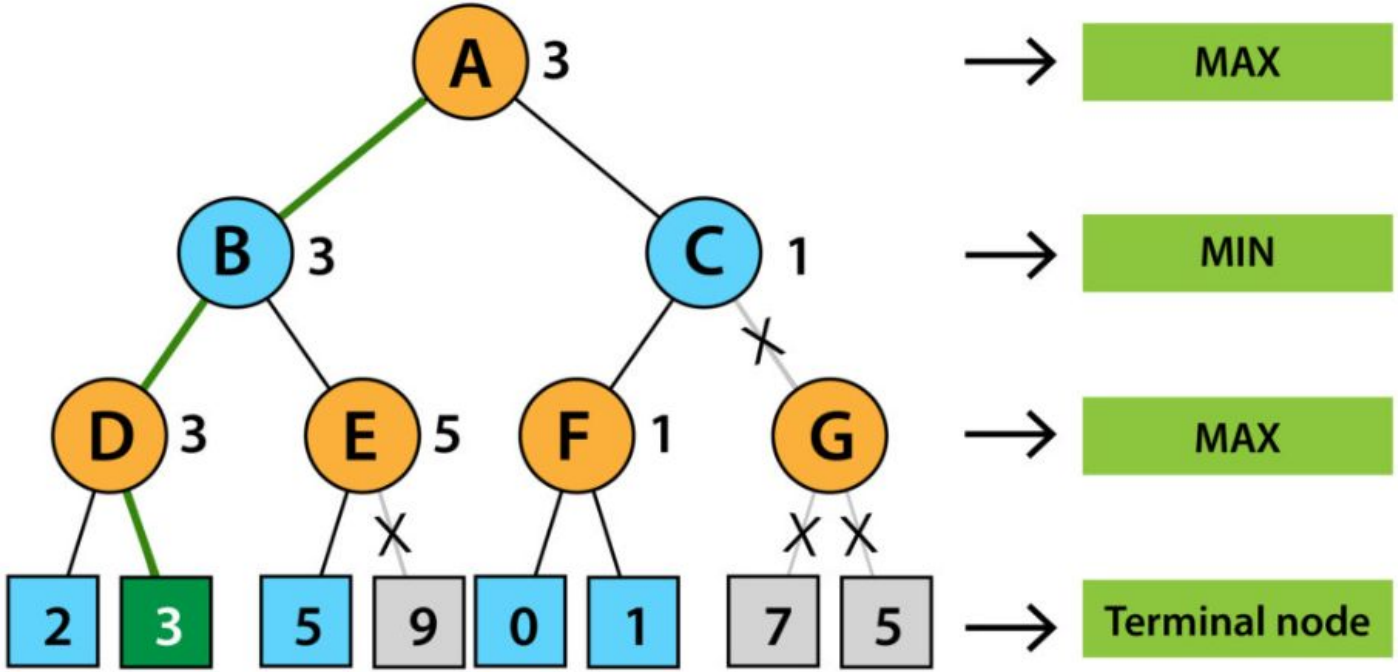
$O(b^{d/2})$



# Alpha - Beta Pruning example



# Alpha - Beta Pruning example solution



# Ref

https://www.mygreatlearning.com/blog/alpha-beta-pruning-in-ai/

https://www.youtube.com/watch?v=_i-lZcbWkps