Artificial Intelligence

[see lectures & make notes from youtube
not slide]

Gopakumar Sir

Part Two

Local Search algorithms

Algorithms that perform purely local search in state space, evaluating & modifying one or more current states rather than systematically exploring paths from an initial state.

→ Hill - climbing
→ Simulated annealing
→ Genetic algorithms
→ local search in continuous spaces

* These algorithms are suitable for problems in which all that matters is solution state, not the path cost to reach it.

Artificial Intelligence

Gate Smashers

· Gopakumar Sir

Adversarial Search

Game playing

Types of games

	deterministic	Chance
perfect information	chess, checkers go	backgammon monopoly
imperfect information	battleships, blind tictactoe	poker, nuclear war

→ here we consider games which are deterministic with perfect information like chess, tictactoe etc.

Since no luck factor is involved here and proper logic is there behind the game.

2 methods/algo's are used

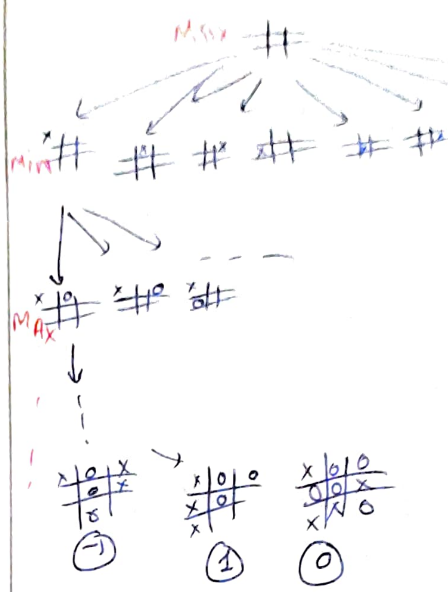
→ Min-max algorithm

→ Alpha-Beta (α-β) pruning

Adversarial Search

Adversarial search is a search, where we examine problem which arises when we try to plan ahead of the world and other agents are planning against us.

*) tictactoe



game tree / search tree / space graph /

here MAX = my chance, MAX refers to increasing prob. of winning

MIN = opponent chance, MIN specify the decreasing prob. of my winning

→ here utility / playoff is what winner will get.

here this is zero-sum game means win = +1, loss = -1 & draw = 0.

- ply → ply is one turn taken by a player.

- here traversal is DFS since each level shows one turn & since games run as alternate turns. So if we use BFS it goes in a level which shows single turns of a player & not an alternate one.

MiniMAX Algorithm

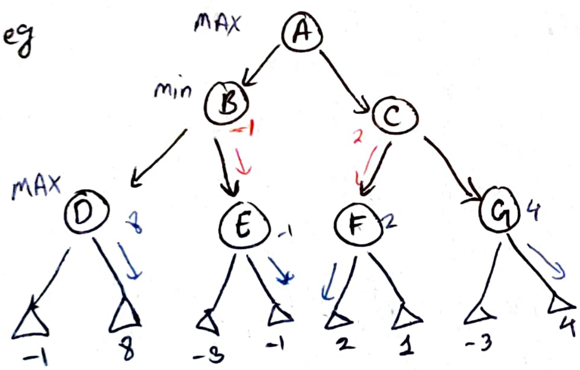
→ is Backtracking algorithm
first travel from root to leaf to see all cases. then again leaf to root to get Best ans.

→ Best move strategy used.

→ Max will try to maximize its utility (Best move)

→ Min will try to minimize utility (worst move for MAX player)

eg



1) first traveled till leaf
2) put scores

→ Min tries to decrease/minimize utility
So select least one.

→ Now A choses max one
= 2

So utility = A
& path = A → C → F → Δ2

Properties of Mini-Max

→ Complete
Yes if tree is finite

→ Optimal
Yes against an optimal opponent

→ Time Complexity
$O(b^m)$

→ Space complexity
$O(bm)$ (depth first exploration)

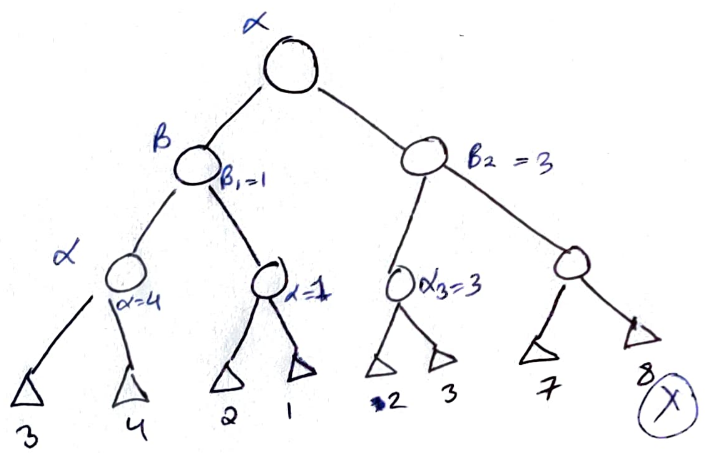
for chess $b \approx 35$, $m \approx 100$
for "reasonable" games
⇒ exact solution completely infeasible

Alpha - Beta Pruning ($\alpha - \beta$)

⇒ See gate smashers video.

here α = for max

β = for min



1) Selects 4 ($\alpha_1 = 4$)
2) Selects 1 ($\alpha_2 = 1$)
3) β selects smallest so β_1
4) $\alpha_3 = 3$
5) $\alpha_4 = 7$ but it is greater than α_3
   So prune 8^{th}
6) $\alpha = \max(\beta_1, \beta_2)$
   $= 3$.

means don't go to path which don't counts in ans.

So in avg.

$$O(b^d) \xrightarrow{\text{Time}} O(b^{d/2})$$

minimax $\qquad$ $\alpha - \beta$ pruning

Pruning ⇒ is a data comparession technique in ml & search algo that reduce size of decision tree by removing sections of tree that are non-critical and redundant to classify instances.

⇒ Alpha-beta pruning is a modified version of minimax algorithm. It is optimization technique for minimax algorithm.

Initial values of $\alpha = -\infty$

$\beta = +\infty$

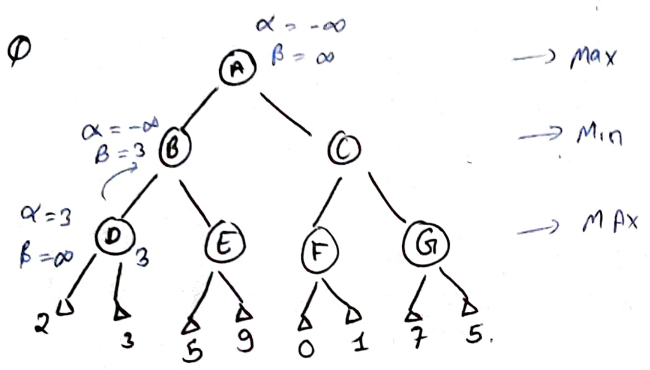
Main condn required for α-β pruning is

$\alpha >= \beta$.

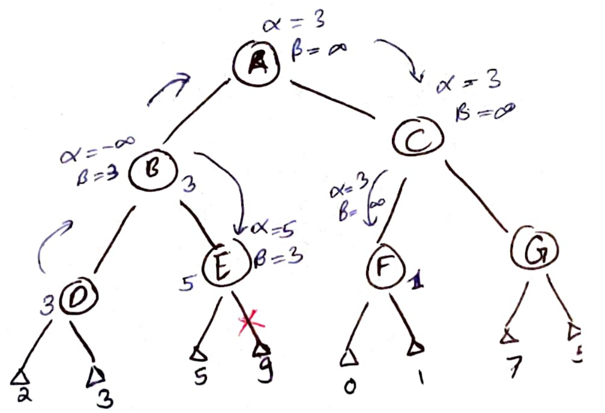
eg) $\alpha - \beta$ pruning

write in format α, β

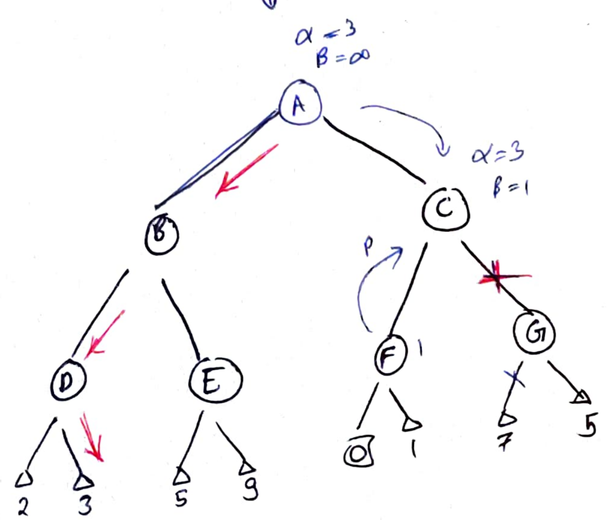
ϕ



↓



↓



Ans. = 3 (utility = 3)