U3
Adversarial Search & Games.

Page No.
Date

Adversarial Search Problem :- having competitive activity involving 'n' players & follows protocols.
adversarial Games :- agents with conflicting goals.
- dealing with a competitive multi agent environment.
- many unpredictible moves / actions.
- Analyze & strategize.

Environment types.
① competative - every agent makes effort to win
② Cooperative - agents jointy perform activity

Zero Sum Game
player 1 win = +1 } sum = 0
player 2 loss = -1 } (+1, -1, 0 are called payoffs)

Non-Zero Sum Game
player 1 wins does not necessarily mean player 2 lost.
2 types.
① Positive Sum game : Cooperative (all have same goal)
② Negative Sum game : Competitive (no one wins eg WAR)

Relevant aspects of Game
① Accessible env :- Game with accessable env have all info at hand
② Search :- games requiring search possible game positions to play game
③ Unpredictable opponent :- introduces uncertainity in the game

Game Theory :-
Branch of Maths & AI that studies interaction among rational agents. where the outcome ~~of~~ is dependent ~~on~~ not only on their own actions but also on action of

Page No.
Date

others.
- used in AI for decision making, strategy formula & modeling competative & cooperative behaviours among agents.

Stochastic Game	Partial Game.
① Outcome influenced by random events.	① A game where players h incomplete or limited inf
② includes probabilistic state transitions.	② Not necessarily random.
③ fully observvaable env	③ partially observalabele en
④ Eg: Ludo	④ Eg. poker, card game
⑤ Key technique used. MDP, Q Learning	⑤ Key techniques used PO MDP

Types of Game strategies
① Equalizing strategy – produces same avg winning
② Optimal strategy - min max strategy.
③ pure & mixed strategies - pure → easy stats. mixed → complex. se one

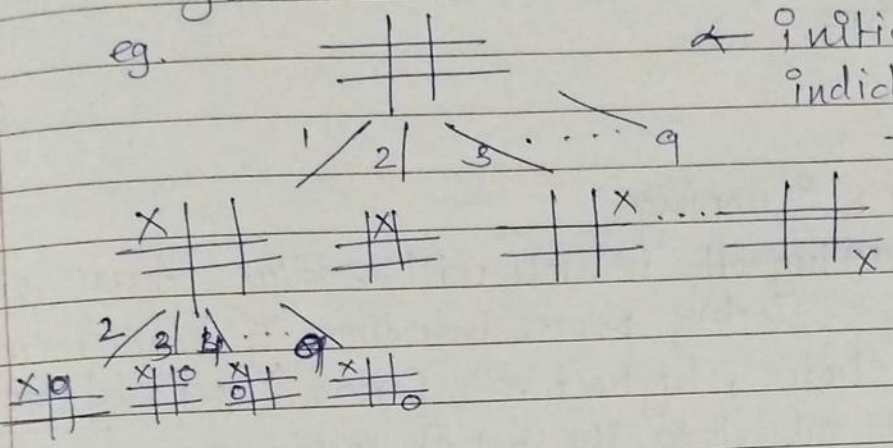
Types of Games.
① Deterministic → fully observalable env
② Non Deterministic
③ perfect info.
④ Non perfect info
⑤ Zero sum game
⑥ non zero sum game
⑦ constant sum game
⑧ n person game

Page No.
Date / /

Game tree

- directed graph with nodes & edges.
- nodes → position in game.
- edges → next action.

eg.



← initial move is always indicated with max to win (max chance

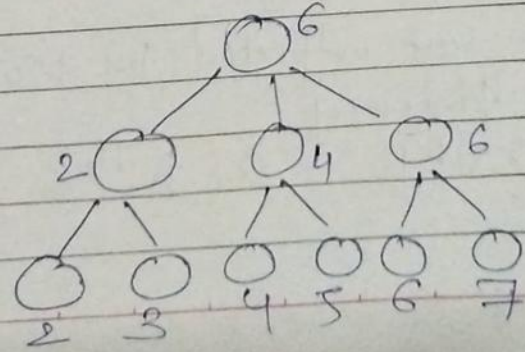
MinMax Algo.

- evaluates decision based on present state of game
- needs deterministic env with perfect info.
- based on the sucessor state every time the min max value is calculated. (by recursive computi
- the selected value, with highest minMax valu should be equal to the best posible outcome agains best play.

steps.

1. create a game tree
2. Assign val. to terminals / leaf nodes.
3. apply Min & Max opp. alternately & propoga upwards
4. find max for root. end.

can be done like leaf node 0/1 for ~~to~~ loss/win



~~max~~ Termina

properties of MinMax Algo.

1. Complete if tree is finite.
2. Optimal if limited oponents.
3. Time complexity $O(b^d)$ ← Space same.
4. uses DFS.

Alpha Beta Pruning.

pruning:- cutting off. which will reduce tree size & help in reducing processing time & save memory

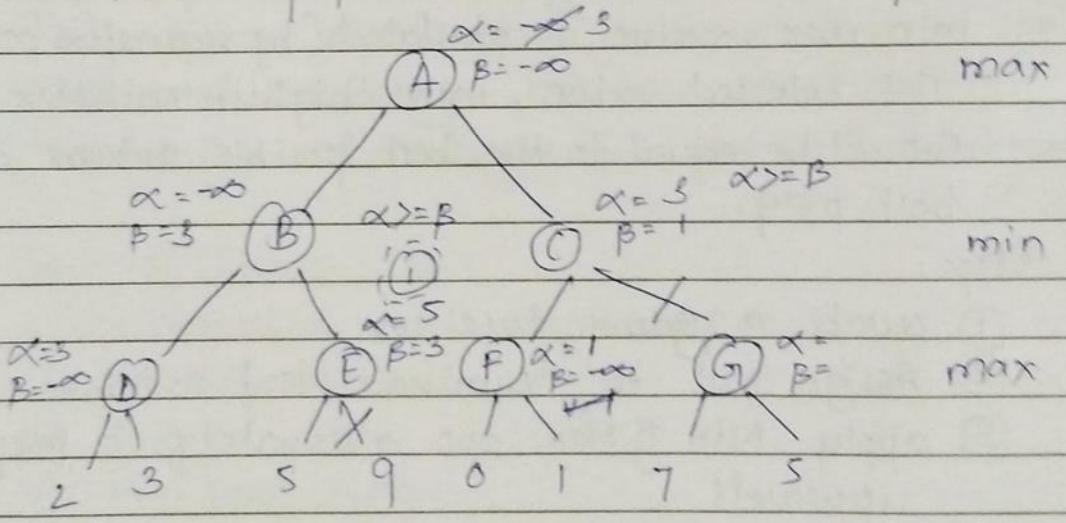
α ⇒ best choice, highest val (max) set to $-\infty$

β ⇒ lowest val along the pat of minimizer. $+\infty$

condition $\alpha >= \beta$

max update α val | min updates β.

- node val pased to upper nodes.
- α, β only pased to child node not upwards.



properties of α-β pruning

1. final result not affected by pruning
2. improved efficiency
3. time complexity $O(b^{d/2})$

↑ try to find best path

Monte Carlo tree Search

MCTS is a heuristic search algo used for decision making
- useful in large game trees (traditional game search not possible)
- builds search tree ~~acyc~~ asymmetrically
- focus on promissing moves by running simulations.

Steps.

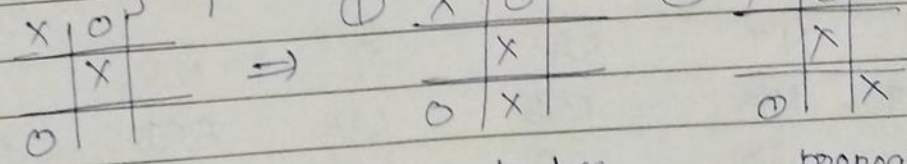
① Selection.
- Start at root, select child based on selection policy like UCT
- move down until u reach a node that is not fully expanded.

② Expansion.
- add child node to tree by exploring one or more untried moves from selected node

③ Simulation.
- simulate a game till the end using random or semi-random moves

④ Back propogation.
- propogate the result (win | loss | draw) & repeat.

eg.



~~propogate~~ propogate loss. propogate win

Adv. ① works well with large spaces
② No need of compleate evaluation func.
③. Search focused on promissing node

Application ① games like chess.
② AI agent in planning & Robotic.

(I) Node Consistancy:- A single variable is node consistent if all the values in the variable domain satisfy ~~all~~ the variables unary constraint. (for single var)

(II) Path consistancy (A, B has consistant assignment) C doesent disrupts that & have an assignment.

Constraint Satisfaction problem

- process of finding a sol to a set of constraints
- finding a group of variables that fulfill a set of restrictions or rules. ← AIM
- Components of CSP
  1. Variables → fields to cover. (work on this part is done)
  2. Domain → posible values.
  3. Constraints. - a pair of scope + relation

Scope ⇒ variables that participate in constraint
relation ⇒ list of valid value combinations.

- back tracking used if struck in a psoble due to constraints

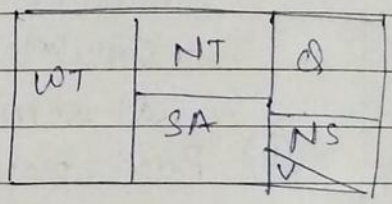
Eg → Variables → WT, NT, SA, Q, NS, V, T

Domains → R, G, B.

Constraints →

WT ≠ (NT, SA)

SA ≠ NT, Q, NS, V | NT ≠ Q | N ≠ V | T



Domain	WT	NT	SA	Q	NS	V	T
Initial Domain	RGB	RGB	RGB	RGB	RGB	RGB	RGB
R	R	GB	GB	RGB	RGB	RGB	RGB
G	R	G	B	RG	RG	RG	RGB
B	R	G	B	R	RG	RG	RGB
R	R	G	B	R	RG	RG	RGB
R	R	G	B	R	R	G	R
G	R	G	B	R	R	G	R

- if traped back track & change color.

Arc consistancy :- A var is in Arc consistent ~~if~~ with other var. if for every val in its domain, there is one value in the other variables domain that satisfies the binary constraints b/w two

binary constraints :- rule involving 2 variables

Page No.
Date

CRYPTO Arithmetic.

x_3 x_2 x_1

C T W O
C T W O
F O U R

0
1 → F, X3
2
3 ← U.
4 → Q
5 ← O.
6 ← T
7 ← W
8
9

2O = R
2T = O
2W = U

(1). consider leftmost as 1.
(2). take Rightmost vals small as po
ie.
(3) R to L

0 ← O
1 ← M
2 ← X
3
4
5 ← E
6 ← N
7 ← D
8
9 ← S

E+O = N
O = N - E + 1

S M E N D
M O R E
M O N E Y

Note :- 7 ≠ 6 = 13

(1)
N 7
+ R 8
E ← 5

E only = 5 not 15

Constraint propogation / legal values

- narrowing down possible domain values for a variable (step by step).
- communicating domain (reduced) for further variables.
- This process results in more domain reductions.
- domain reductions are propogated to appropriate constraints.

this ← map to data table

Adv.

(1). reduce search space by pruning dead end branches
(2). reveal hidden structures & symmetries in problem

in CSP only constraint propogation

local consistancy :-

- used to reduce no. of posible values by checking if they work with their immediate neighbors.