GAME THEORY

PAGE NO.
DATE: / 202

Prisoner's Dilemma

Police has custody of 2 suspects of a robbery, & they have no evidence. their only chance lies in a confession. they interrogate them in separate chambers. Each suspect can either confess or deny
- ~~Both confess: 5 years in jail to both.~~
- One confesses, other denies.
- Each suspect can either admit their partner committed the crime, or stay silent (cooperate & defect respectively)

Payoffs:
- x & y cooperate: one year in jail each
- x cooperates, y defects: x: 3yrs, y: 0yrs
- x & y defect: 2 years in jail each

Jberray

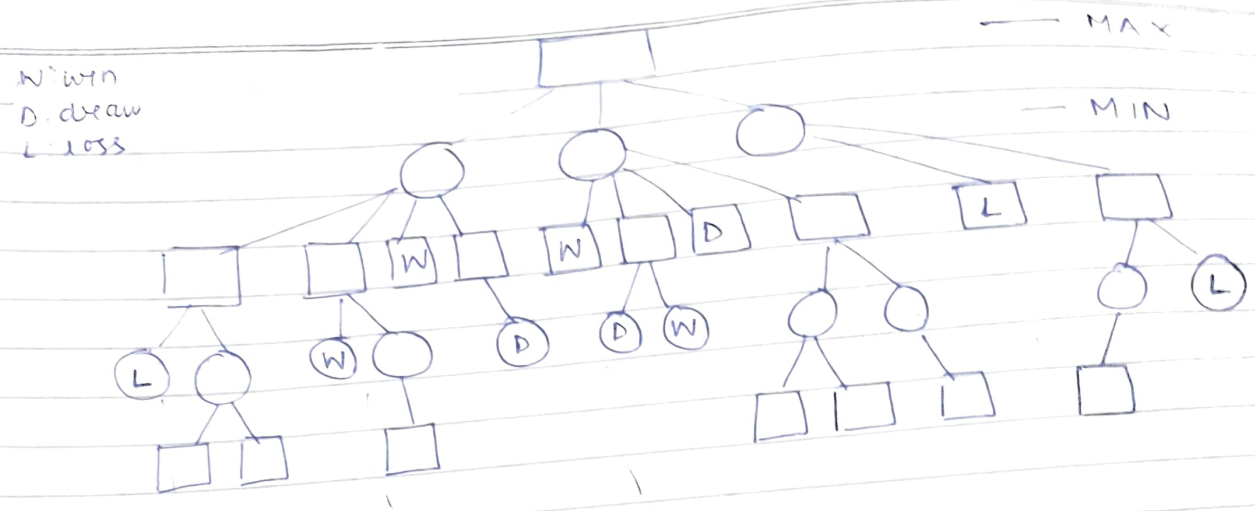
x/y	y coop	y def
x coop	1 / 1	3 / 0
x def	0 / 3	2 / 2

Two person games: exactly 2 players

Game Trees: Assumptions: 2 player game, zero sum game where two players are MAX & MIN. A game is represented by a game tree.
- Game tree is a layered tree where at each alternating level, one player or the other makes their choices.
- The layers are called MAX layers & MIN layers.

generally, MAX nodes : square
MIN : circle

PAGE NO.
DATE: / 202



- A game starts at the MAX node ~~& ends~~ playing first & ends at a leaf node.
- Leaves of the game tree are labelled with the outcome of the game & the game ends there.
- The task of each player is to choose the move when its turn comes
- MAX chooses at MAX levels & MIN at MIN levels.

⇒ Thus, a game is a path from a the root to a leaf node chosen at alternating levels by the two players.

$$value(leaf) = \begin{cases} 1, & \text{if MAX wins} \\ 0, & \text{if game is a draw} \\ -1, & \text{if MIN wins} \end{cases}$$

Given a game tree, it is possible to analyse the game & determine the outcome when both players play perfectly. We can do this by backing up values from the leaf nodes up to the root. The backup rule is as follows:

Minimax rule:

- if the node is a MAX node, back up the maximum of the values of its children

value(node) = max { value(c) | c is a child of node }

" MIN " minimum

"

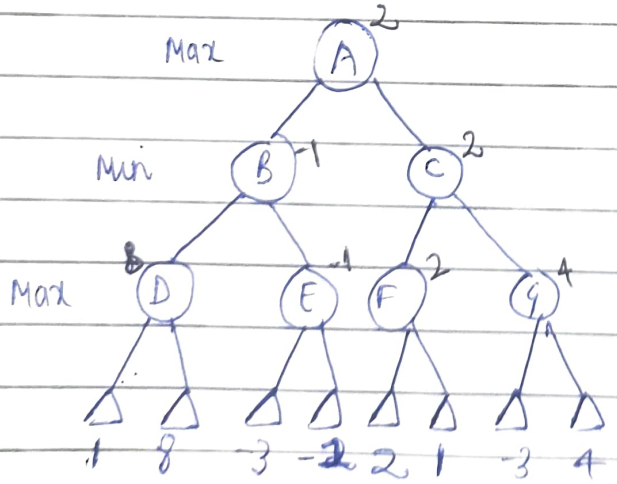
value(node) = min { value(c) | c is a child of node }

GAME PLAYING ALGORITHMS

- Explore the tree upto a finite ply depth
- Compute the evaluation function of the nodes on the frontier
- Use the minimax backup rule to determine the value of the partial game tree, & the best move.
- Make the chosen move, wait for opponent's move, then again search for your best move

1. MINIMAX ALGORITHM

- Simplest algo for computing minimax value
- Recursive, backtracking algo



PAGE NO.
DATE: / 202

- ~~M~~ Does a ~~DF~~ depth first traversal of the tree applying the minimax backup rule
- We choose a node N. 3 cases arises:

1. N is a terminal node: return eval(N)
2. N is a MAX node: (initialize MAX to -Large & look for higher value)

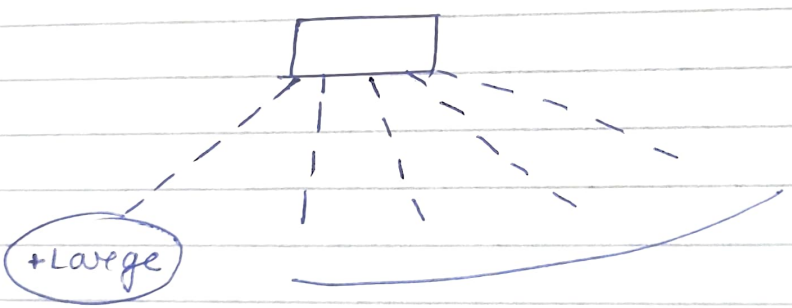
```
value ← -∞
for each child C of N
        value ← max(value, MINIMAX(C))
```

(Recursive)

3. N is a MIN node: (initialize Min to +large & look for lower values)

```
value ← ∞
for each child C of N
        value ← min(value, MINIMAX(C))
```

Pruning the search tree: when a winning move has been found, other options need not be explored.



In this case, MAX has found winning move

PAGE NO.
DATE: / 202

2. ALPHA-BETA ALGORITHM

- MAX nodes : Alpha nodes, which store alpha values. Alpha value is the value found so far for the alpha node, & is a lower bound on the value of the node. It can only be revised upwards. - rejects lower values subsequently
- MIN nodes : Beta nodes, " beta "
  Beta value " beta "
  " upper bound. "
  " downwards. - rejects higher values subsequently

For pruning, $\alpha \geq \beta$.

3. SSS* ALGORITHM

- Main problem w/ α-β is that is sensitive to the order in which moves are generated.
- It always searches the game tree from L to R, in an uninformed fashion.
- SSS* searches the game tree guided by heuristic ~~fashion~~ info
- Maintains priority queue.
- ~~Start by adding~~
- Each node is either LIVE - unknown minimax value
  SOLVED - known "
- Add the root node to the PQ & continue till root ~~occurs~~ appears SOLVED at the end
- ~~Fo~~

Initialization phase

PAGE NO.
DATE: / 202

Forward phase : add LIVE nodes till the horizon. Evaluate on the horizon.

Backward phase : when SOLVED, apply backup rule.