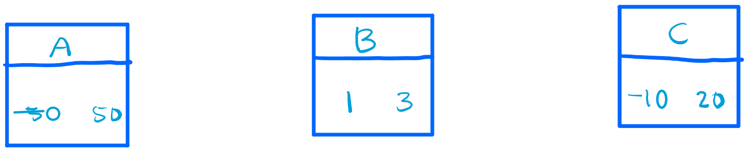
Lecture 19: Two player competitive games

27 March 2024 11:02

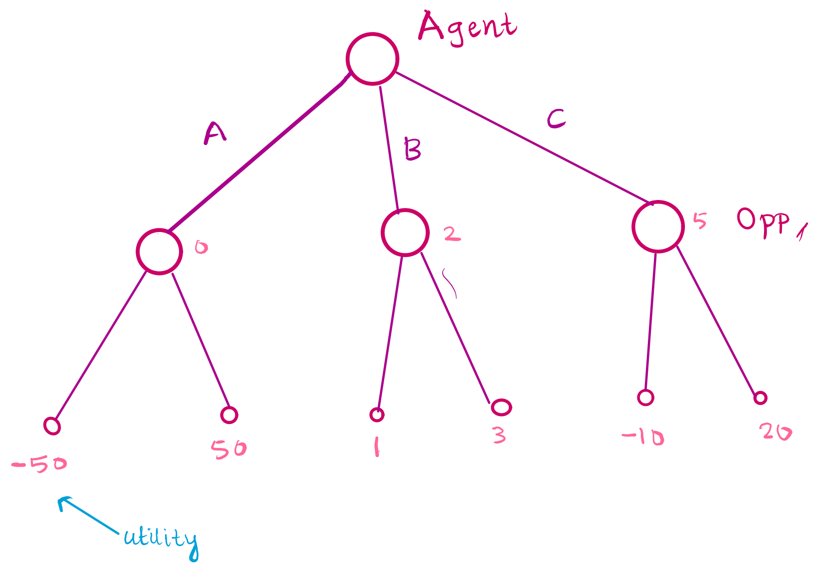
Recall the game we discussed last lecture. There are three buckets. Two players



Agent: Picks a bucket

Opponent: Pick a no. from selected bucket.

Game tree:



- If opponent is stochastic (prob = 1/2) maximize expected.
- If opponent is min player choose maximum among minimums.

Similarly strategy depends on opponent behaviour.

Two player Zero-Sum Game (Sequential mode)

Players = {agent, opponent}

S_0 = starting state

actions (s) = possible actions at a state s.

Player (s) = the player who makes the move at state s.

Succ(s,a) = resulting state if action 'a' is taken at state 's'

s is an inter-mediate state

isEnd (s) = is state s an end state

utility (s) = agent's utility at an end-state, s.

Example: (Chess)

Players = {white, black}

s = a board position

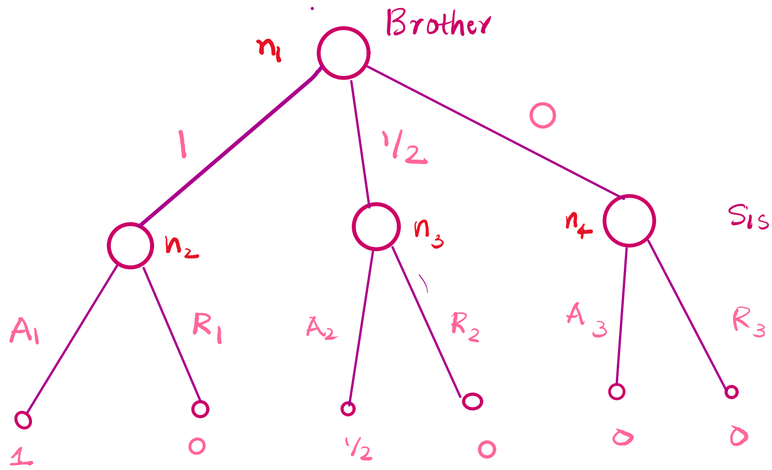
actions(s) = all legal moves by the player s.

isEnd(s) = whether s is checkmate or a draw

$$\text{Utility}(s) = \begin{cases} +M & \text{if white wins} \\ -M & \text{if white loses} \\ 0 & \text{if draw} \end{cases}$$

Constant sum game

Consider a Brother and Sister. They want to divide chocolate among themselves as shown below:



A_i : accept

R_i : reject.

Players = {B, S}

$$\text{actions}(s) = \begin{cases} \{1, 1/2, 0\}, & s = n_1 \\ \{A_1, R_1\}, & s = n_2 \\ \{A_2, R_2\}, & s = n_3 \\ \{A_3, R_3\}, & s = n_4 \end{cases}$$

Stratergy of a player

Deterministic stratergy: $\pi_i(s) \in \text{actions}(s)$, if player($s$) = i

Randomized stratergy: $\pi_i(s) \in \Delta\,\text{actions}(s)$ eg: $\pi_B(n_1) = \left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)$

prob distribution

ΔA = set of all probability distributions over A.

set

Games with partial information:

$$U_{agent}(s) = \begin{cases} utility(s) & \text{if } isEnd(s) \\ \sum_{a \in actions(s)} \pi_{agent}(s)[a] \cdot U_{agent}(succ(s,a)) & \text{if } player(s) = agent \\ \sum_{a \in actions(s)} \pi_{opponent}(s)[a] \cdot U_{agent}(succ(s,a)) & \text{if } player(s) = opponent \end{cases}$$

$\pi_{agent}(s)[a]$ ↪ propability that agent picks action a

Opponent is stochastic, but player is utility maximizer (max-player)

In max player case instead of summation,

we can use

$$\max_{a \in actions(a)} \pi_{agent}(s)[a] \cdot U_{agent}(succ(s,a)) \quad \text{if } player(s) = agent$$

↪ propability that agent picks action a

opponent is utility minimizer for agent (min player)

$$\min_{a \in actions(a)} \pi_{opponent}(s)[a] \cdot U_{agent}(succ(s,a)) \quad \text{if } player(s) = opponent$$

So we've the following:

$$U_{agent}(s) = \begin{cases} utility(s) & \text{if } isEnd(s) \\ \max_{a \in actions(s)} \pi_{agent}(s)[a] \cdot U_{agent}(succ(s,a)) & \text{if } player(s) = agent \\ \min_{a \in actions(s)} \pi_{opponent}(s)[a] \cdot U_{agent}(succ(s,a)) & \text{if } player(s) = opponent \end{cases}$$

$\pi_{agent}^{maximum}(n_1) = B$

$\pi_{opponent}^{minimum}(n_2)$ minimizing player's utility

Q: Is $\pi_{agent}^{maximum}$ optimal if $\pi_{opponent}$ is stochastic?

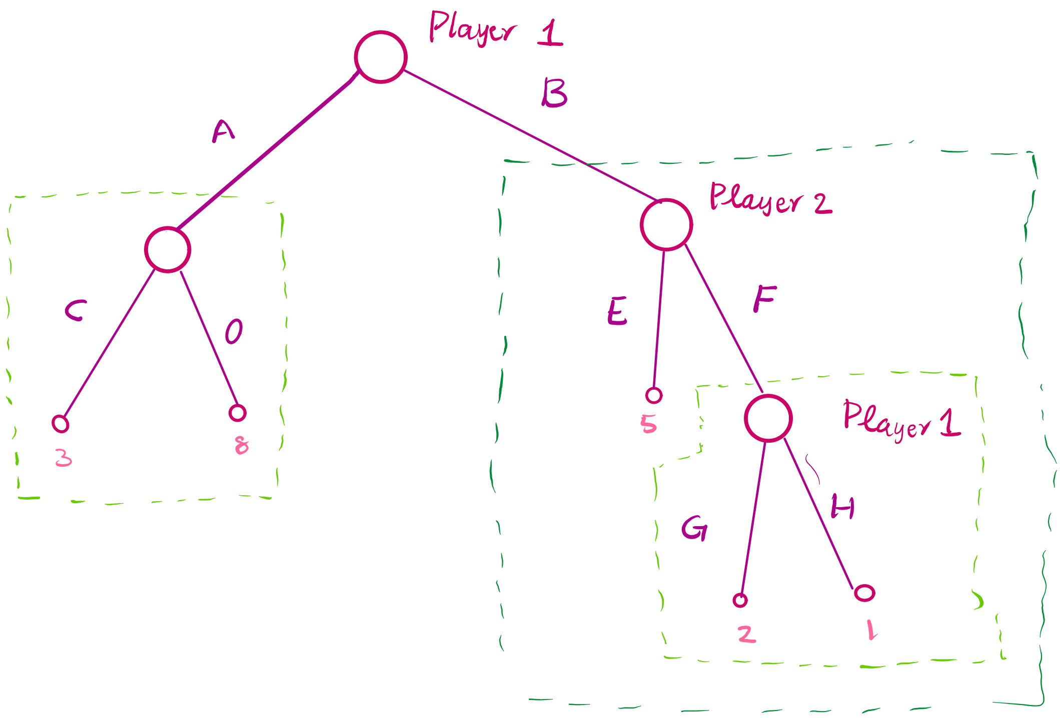
Ans: NO! Depends on opponent.

In fact, we don't have an optimal policy but an equilibrium policy.

$$\text{Equilibrium}\left(\pi^{max}_{agent}, \pi^{min}_{opp.}\right)$$

Notice that $\left(\pi^{stoc.}_{agent}, \pi^{stoc.}_{opp.}\right)$ is not equilibrium.

Subgame and Subgame Perfection

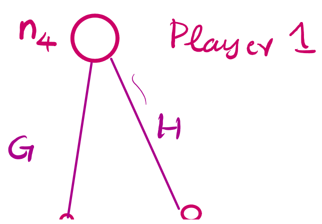


A subgame at s is the restriction of the game at the subtree rooted at s where is End(s) is false.

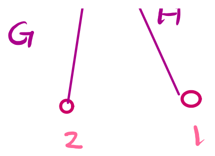
Player 1 is max $\rightarrow \underset{a \in actions(s)}{\arg\max} \; U_{agent}(succ(s,a))$

Player 2 is min $\rightarrow \underset{a \in action(s)}{\arg\min} \; U_{agent}(succ(s,a))$

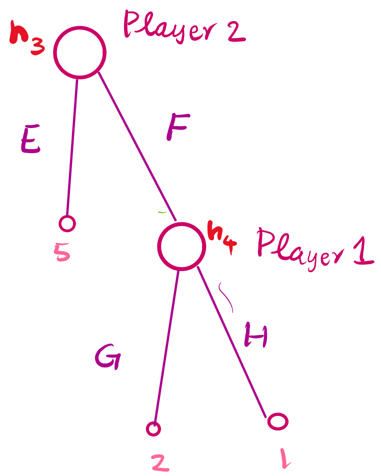
Consider the following subgame



At n_4, Player 1 should pick G



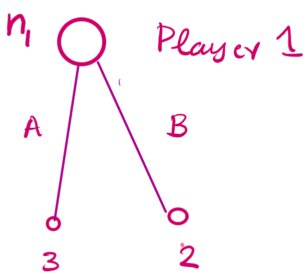
Consider this subgame:



At n_3, player 2 should pick F

At n_2, player 2 should pick C (in the other subgame)

In the entire game, after these choices we've



At n_1, player 1 should pick A

Subgame Perfect Equillibriam is an equillibrium at every subgame

Backward Induction:

function BackInd(s):

- if isEnd(s):

  return $n_{agent}(s)$, ϕ

- if player(s) = agent

  - bestUtil = $-\infty$
  - for all $a \in$ actions(s), do:

    # utilAtChild, bestAvect $\leftarrow$ BackInd(succ(s,a))

    # if utilAtChild > bestUtil:

* if utilAtChild > bestUtil:
    * bestUtil = UtilAtChild
    * bestAvect = append(a, bestAvect)

— if player(s) = opponent   do reverse, ie min stratergy.

— return bestUtil, bestAvect.

Can we use this to play Chess, Go, Checkers ?

The game tree of Checkers has 10^{20} nodes

The game tree of Chess has 10^{40} nodes

" Go has 10^{170} nodes

For checkers, it was solved in 2007 after 18 years of computation (to get a draw ☺!)