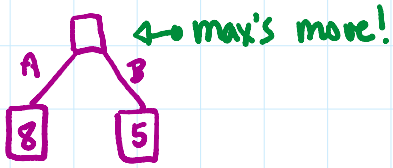
CMSC351-2023-11-29

Wednesday, November 29, 2023 8:28 AM

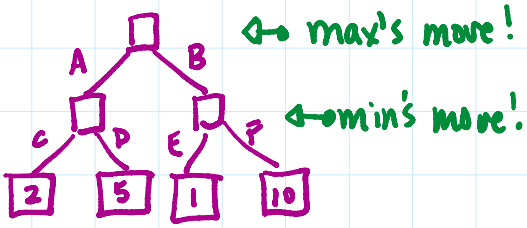
Minimax Algorithm

① Intro: The minimax algorithm is an algorithm which can help make a best choice in a situation involving worst-case scenarios.

Sps Max and Min are playing a game.
Sps the state of the game can be rep. by a number ↗ Max wants max #. Min wants min #.
Sps it's max's turn and max has 2 choices:



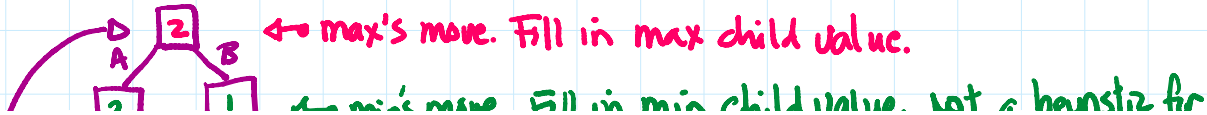
Best choice for Max is move A to get 8.
Now let's complicate it:

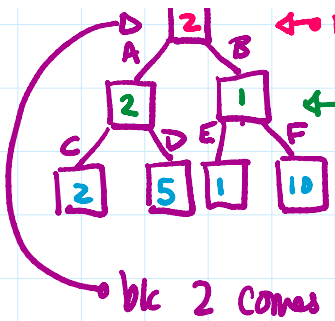


If Max sees 10 and picks B then Min will choose E, yielding 1. :(
So Max should choose A then min will choose C, yielding 2.
That's the best poss. for max!

② Algorithm: The situation is:

- Max's move
- Max can look some # moves ahead, rep. by a tree.
- There is a heuristic fctn which assigns a number to each leaf situation (game position).
- If we fill in those leaves, how do we then get back to the root to determine Max's best move?!





← max's move. Fill in max child value.

← min's move. Fill in min child value. not a heuristic for the game state!

← values of the heuristic fctn!

blc 2 comes from move A, Max should do A!

③ Pseudocode: For filling in values:

```
function minimax(node nd,depth):
  if nd.value is not undefined:   ← if a leaf
    return(nd.value)
  else:
    if depth is even:   ← max's turn
      return( max(minimax(ndc,depth+1) for all children ndc of nd) )
    else:   ← min's turn.
      return( min(minimax(ndc,depth+1) for all children ndc of nd) )
    end if
  end if
end function
```

Have a tree!
Heuristic has been applied to the leaves!

Call: minimax(root, 0)

Time Complexity: Function executes once for each node, hence if n nodes its $\theta(n)$.

Another way: if tree depth = d
if max branch factor (# children) = b
then

$$n \leq \underbrace{1 + b + b^2 + \ldots + b^d}$$

$$n \leq \frac{b^{d+1} - 1}{b - 1}$$

about the best we can do!

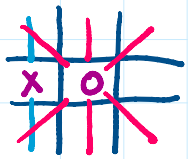
④ Tic-Tac-Toe

Max plays X, Min plays O. Max goes first.

Say: for a board, an open winning line (OWL) is a row/col/diag which contains at least one of the player's marks and none of the opponent's marks.

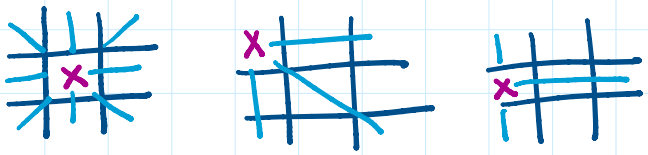
ex  max has 3 OWLs.

Ex  max has 3 owls.

Or  max has 1 owl
min has 3 owls.

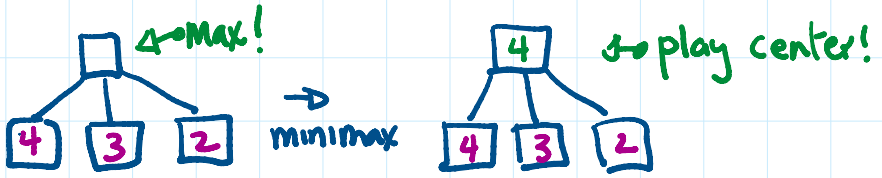
define heuristic: $h = (\#\text{owls X has}) - (\#\text{owls O has})$

- First, max can look 1 move ahead!
  max has 3 poss moves: center, corner, edge (up to symmetry)

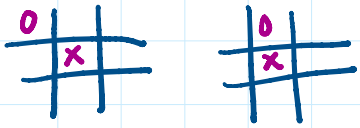


$h = 4-0$ $h = 3-0$ $h = 2-0$

In a tree:

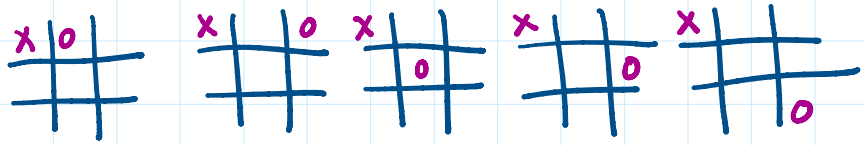


- Now two moves ahead!
  if max plays center, min has 2 moves:



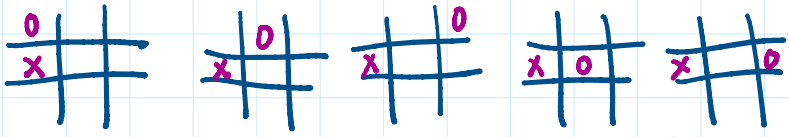
$h = 3-2 = 1$ $h = 3-1 = 2$

if max plays corner, min has 5 moves

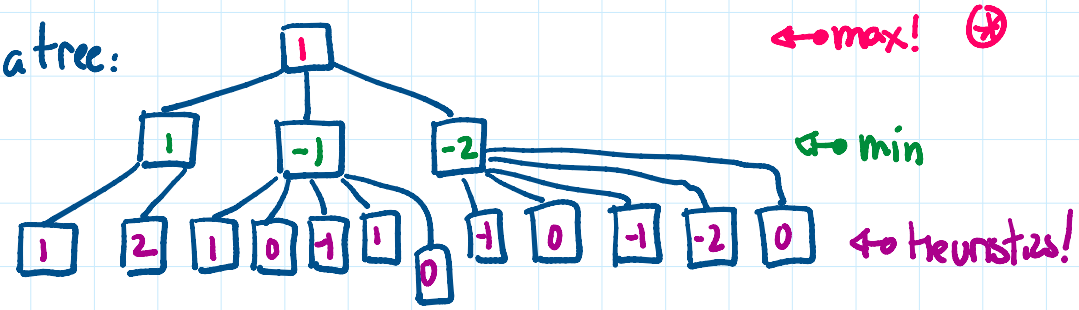


$h = 2-1 = 1$ $h = 2-2 = 0$ $h = 2-3 = -1$ $h = 3-2 = 1$ $h = 2-2 = 0$

if max plays edge, min has 5 moves



As a tree:



(*) Says max should play the center!

note: Not always one best move!