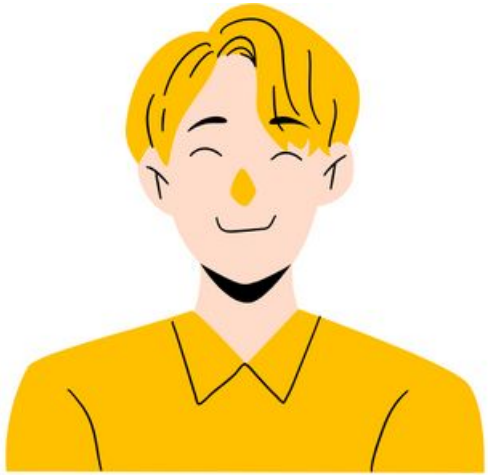
Algoritmul Minmax

Miruna-Andreea Zăvelcă
miruna-andreea.zavelca@unibuc.ro
Universitatea din București

Maria



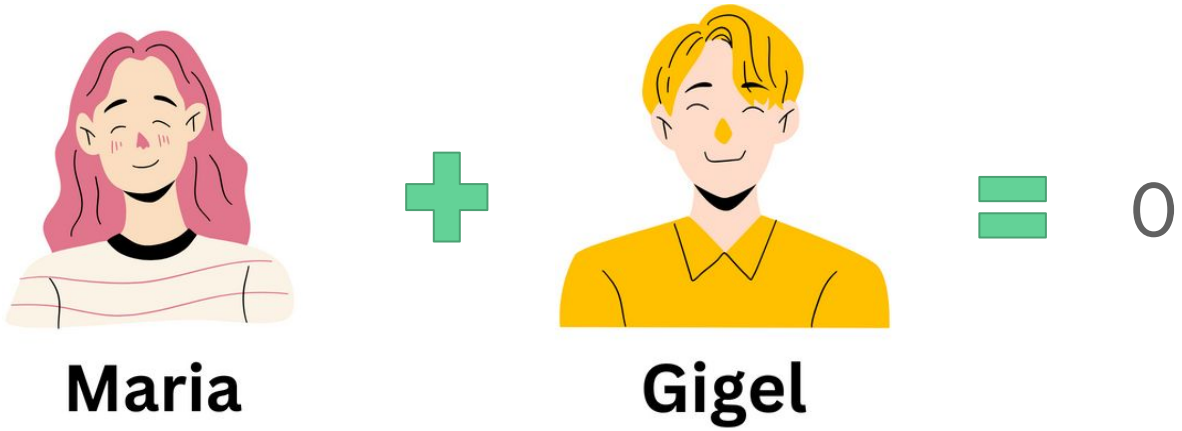
Gigel

Maria

Gigel

Joc de sumă zero (0-sum game)

Orice câștig al unui jucător reprezintă o pierdere cu aceeași valoare pentru celălalt jucător.

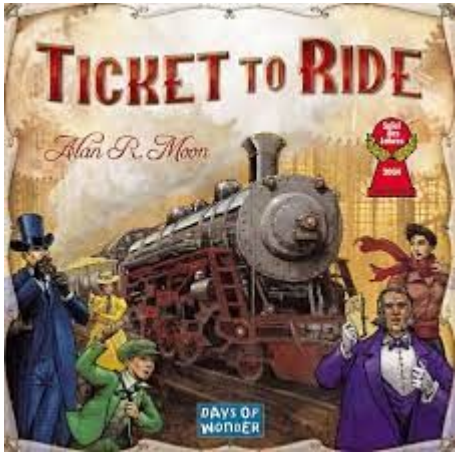


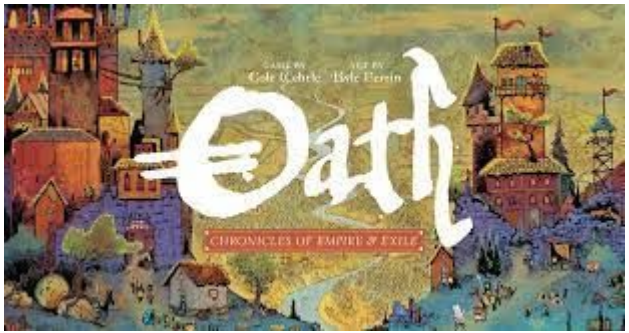
Joc cu informație perfectă

Există mai mulți jucători care efectuează mutări alternativ și au informație completă asupra stării curente a jocului.

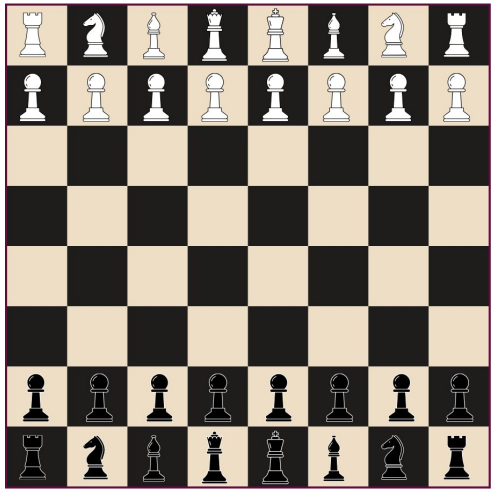
Joc cu informație completă

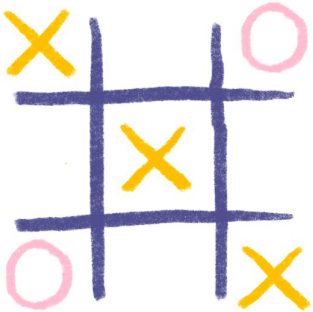
Există mai mulți jucători care efectuează mutări alternativ și cunosc obiectivele celuilalt jucător în fiecare moment.

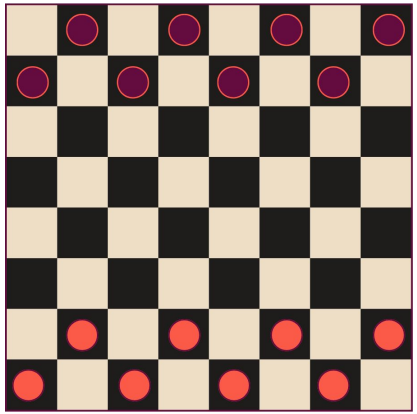




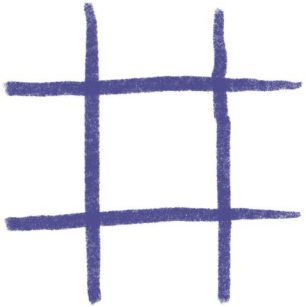
Jocuri cu 2 jucători, informație completă, perfectă și sumă 0



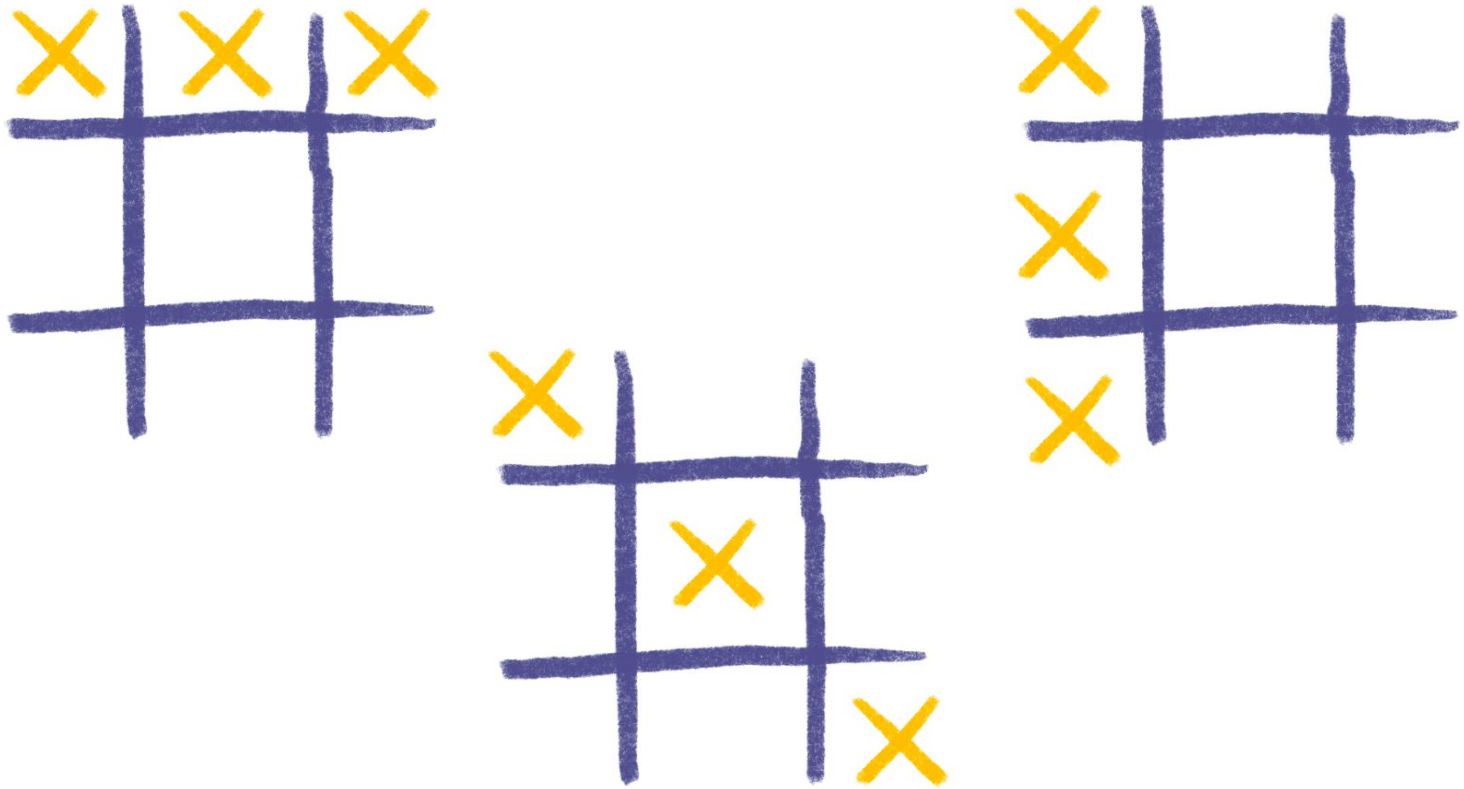


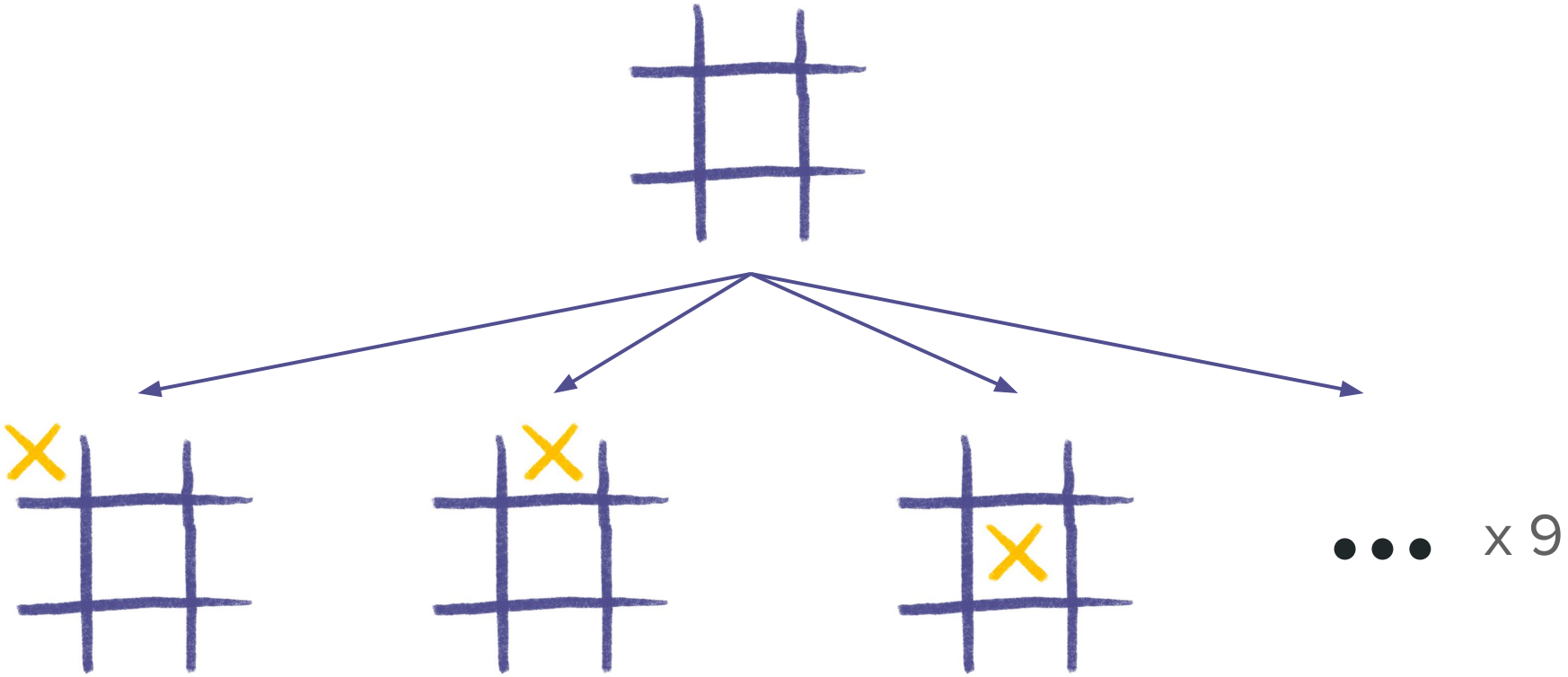


Maria

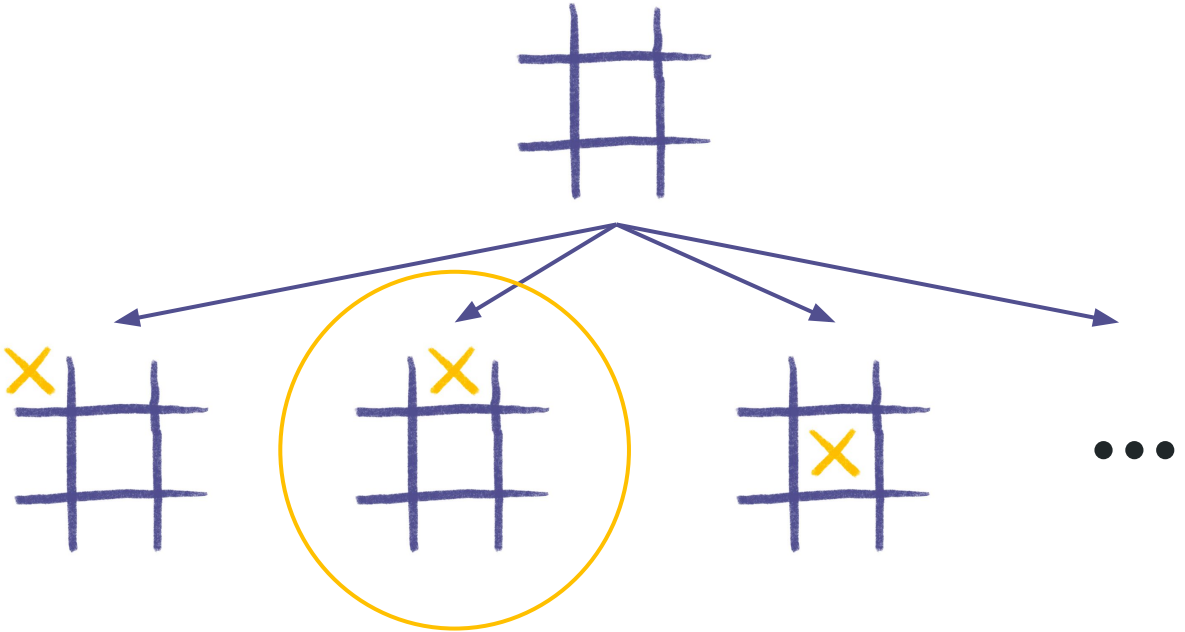


Gigel



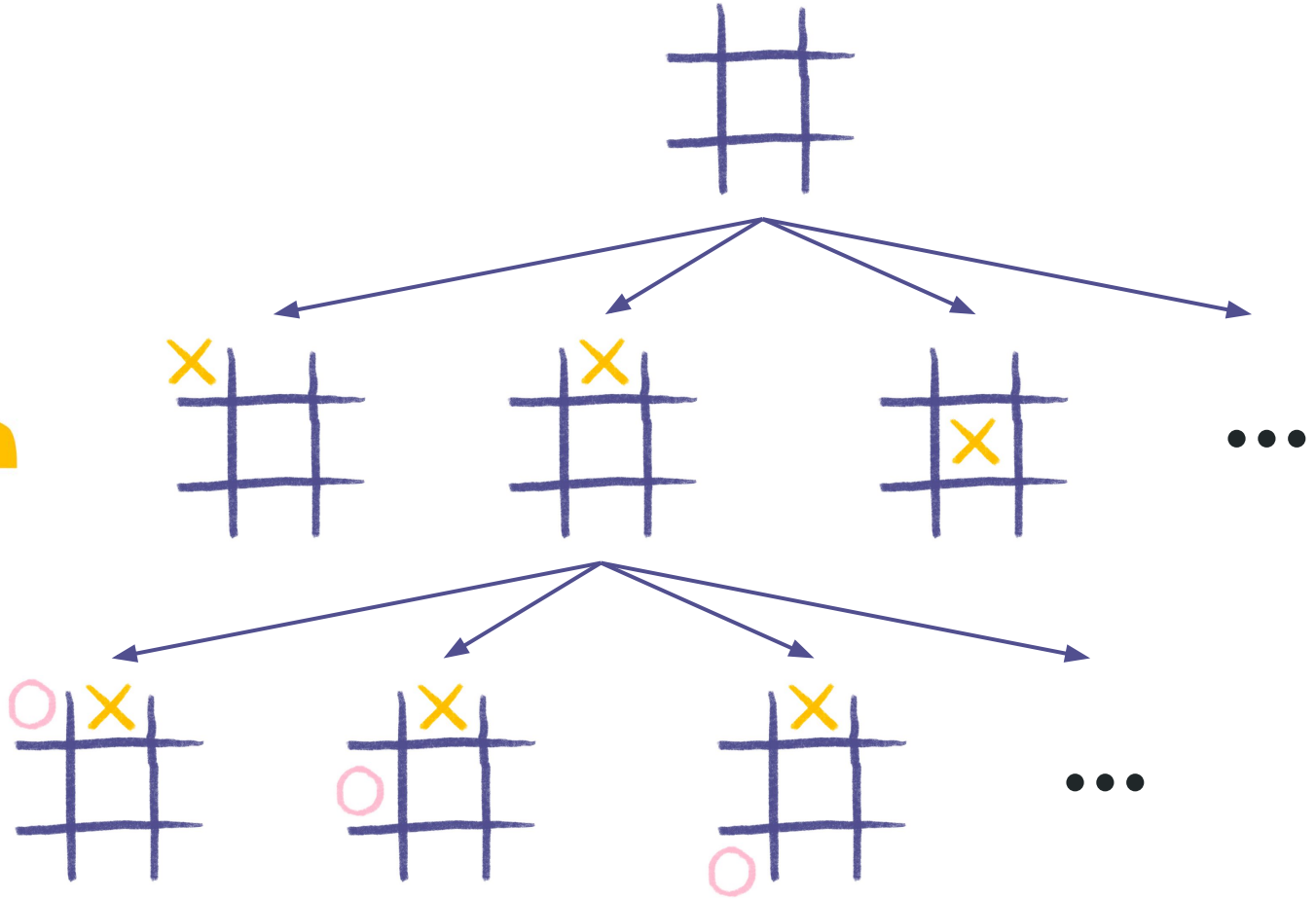

x 9

Gigel



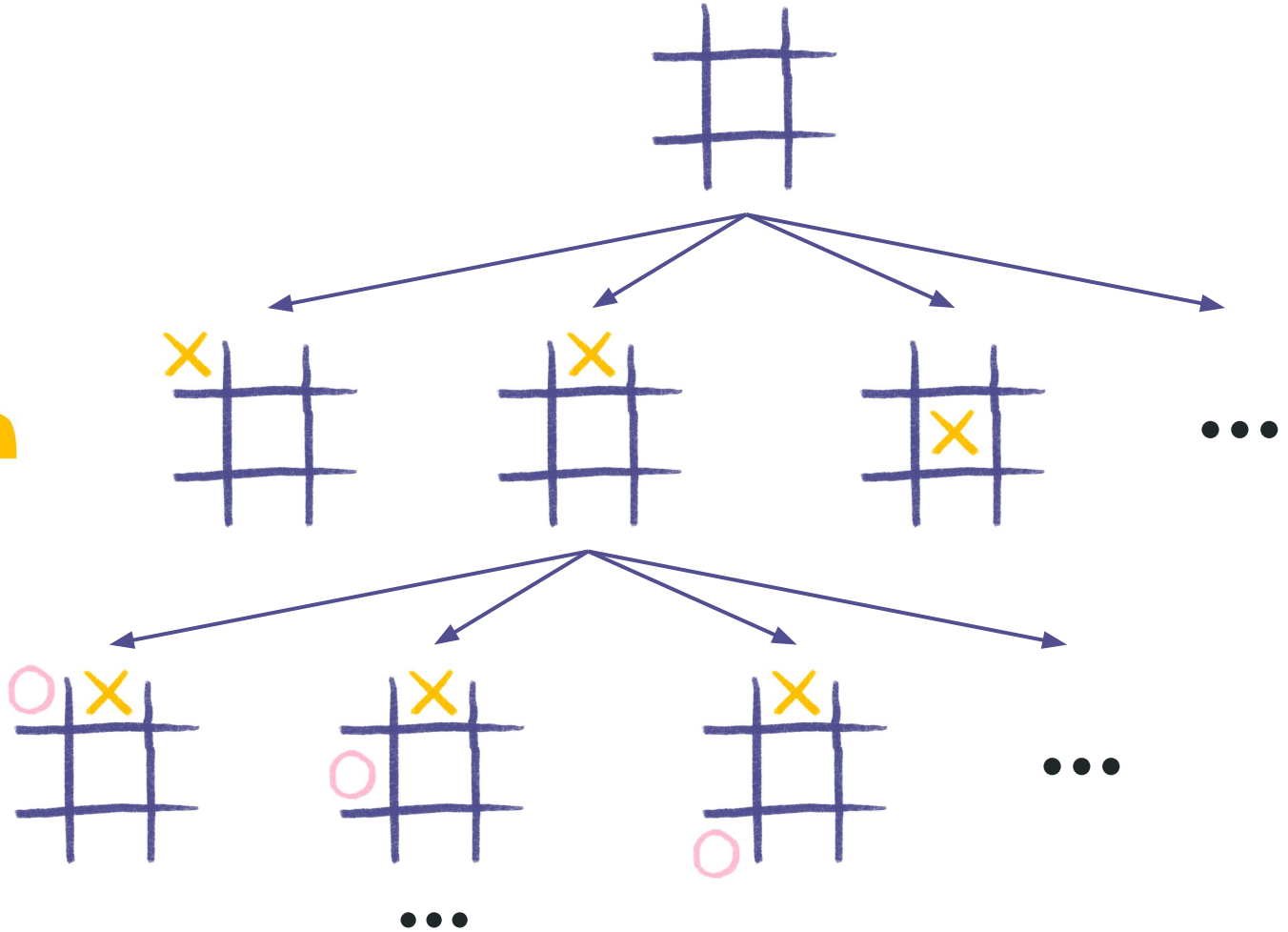
Gigel

Maria



Gigel

Maria



Cum măsurăm utilitatea?

Utilitatea = valoarea euristică a unui nod

= cât de mult îl apropie de câștig pe jucătorul MAX

Exemplu pentru X&0:

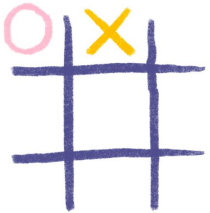
Diferența între scorul jucătorului curent și scorul jucătorului opus, unde scorul unui jucător este:

+ 100 pentru 3 în linie

+10 pentru fiecare 2 în linie (cu o căsuță liberă)

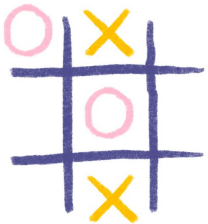
+1 pentru fiecare 1 în linie (cu 2 căsuțe libere)

0 altfel



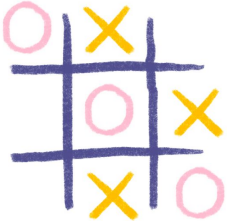
O 2 - 1 = 1

X 1 - 2 = -1



O 12 - 1 = 11

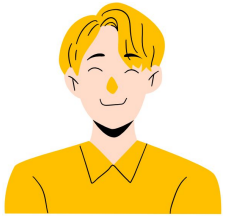
X 1 - 12 = -11



O 101

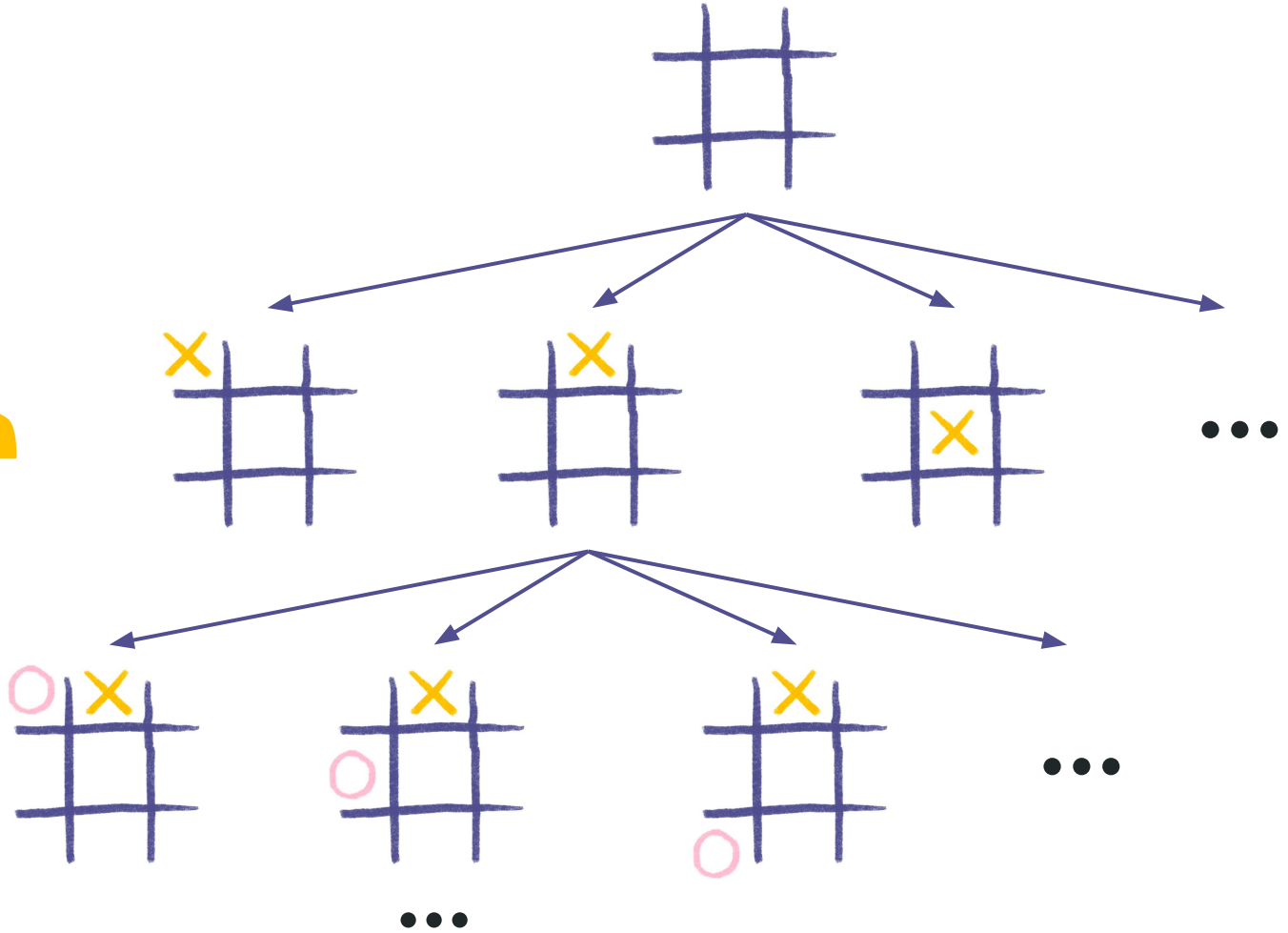
X -101

MINMAX

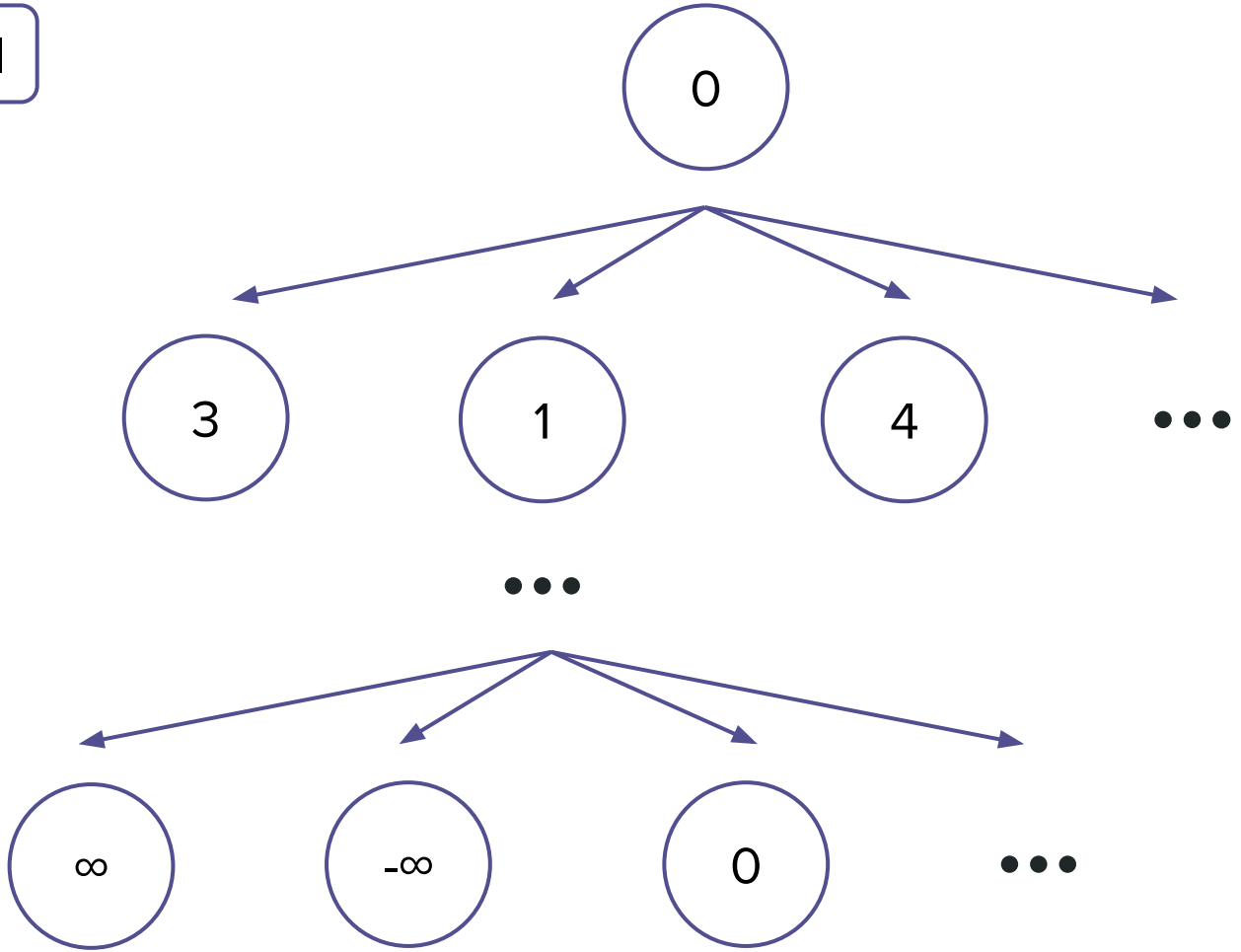


Max

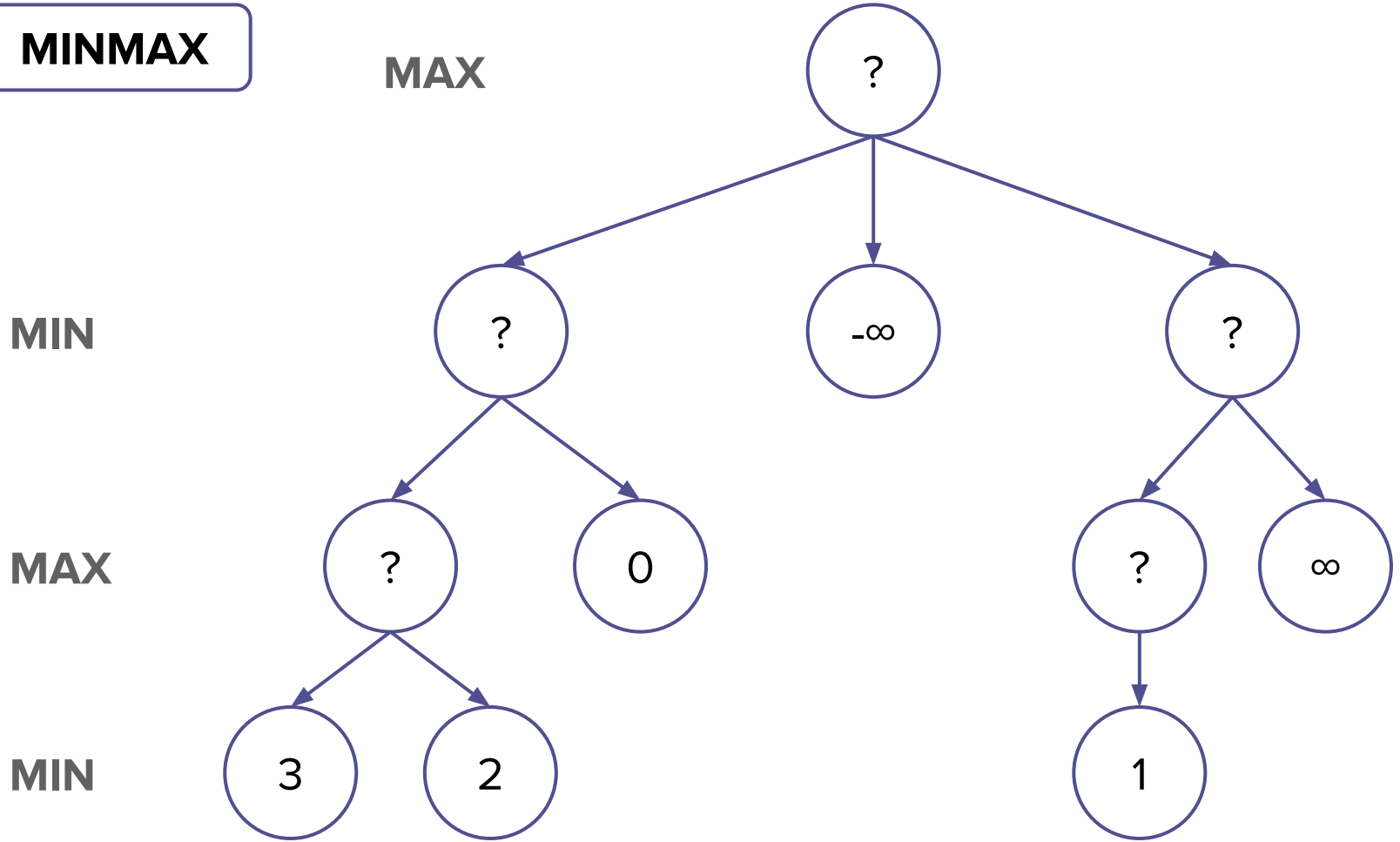
Min



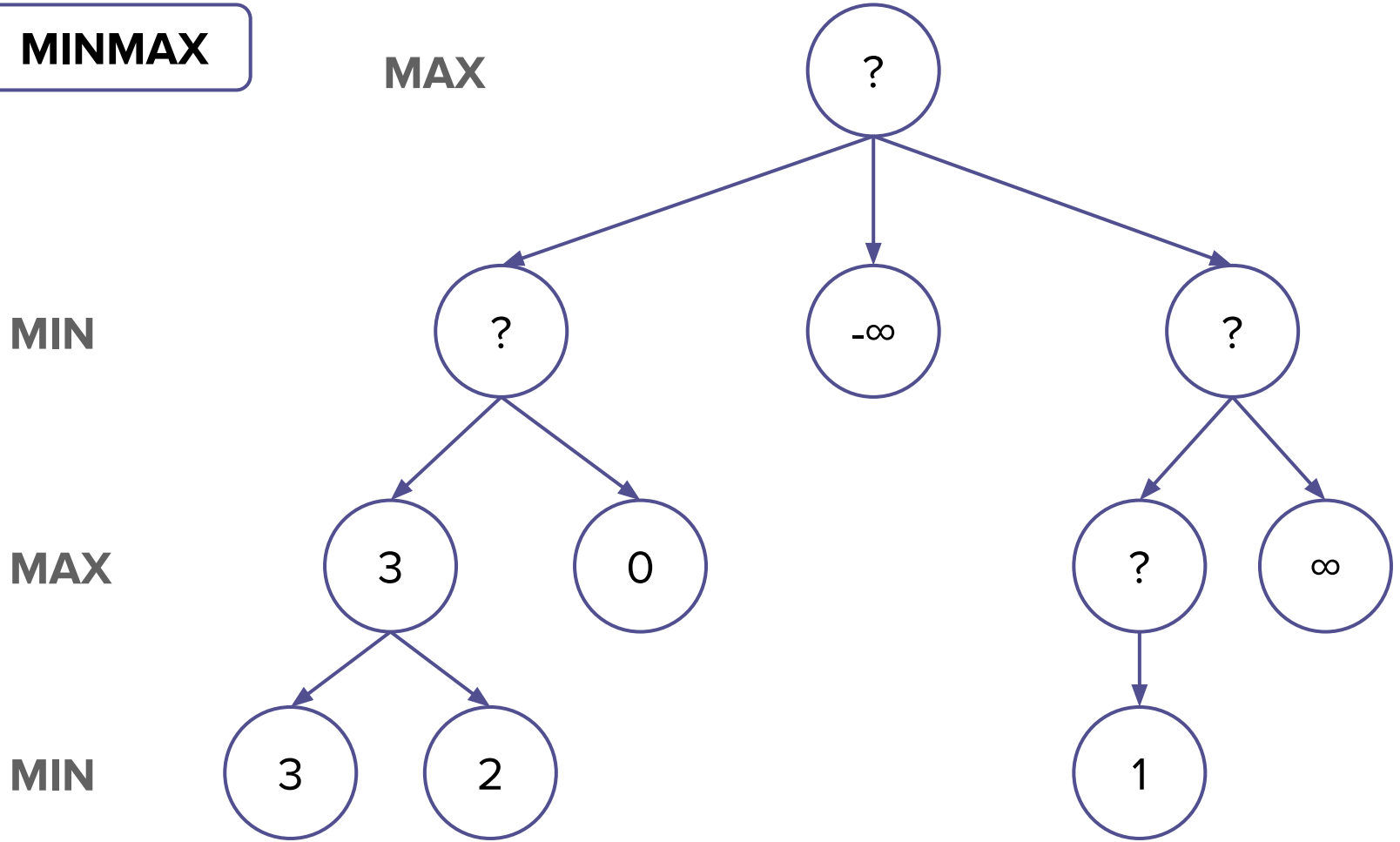
EURISTICI



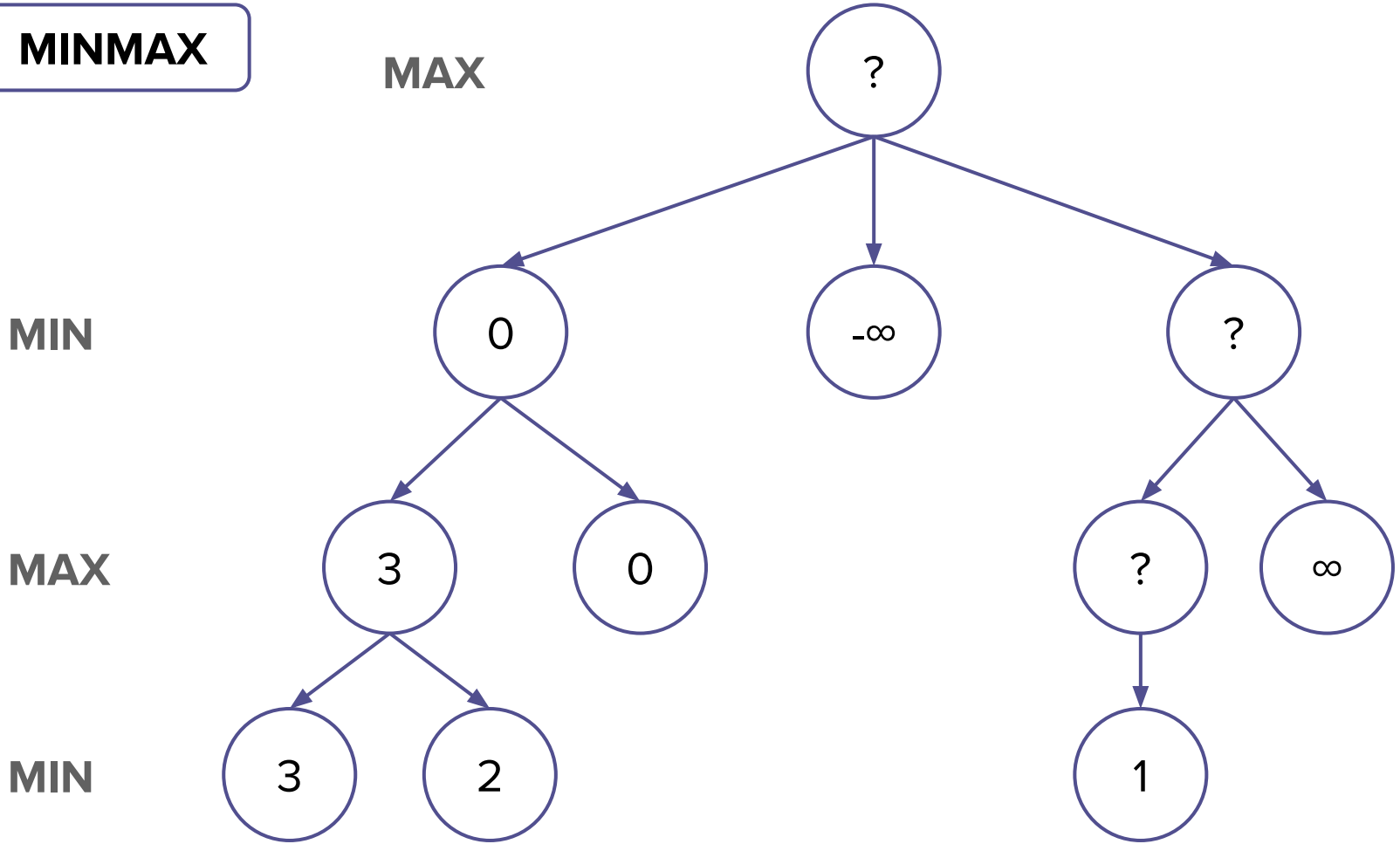
MINMAX



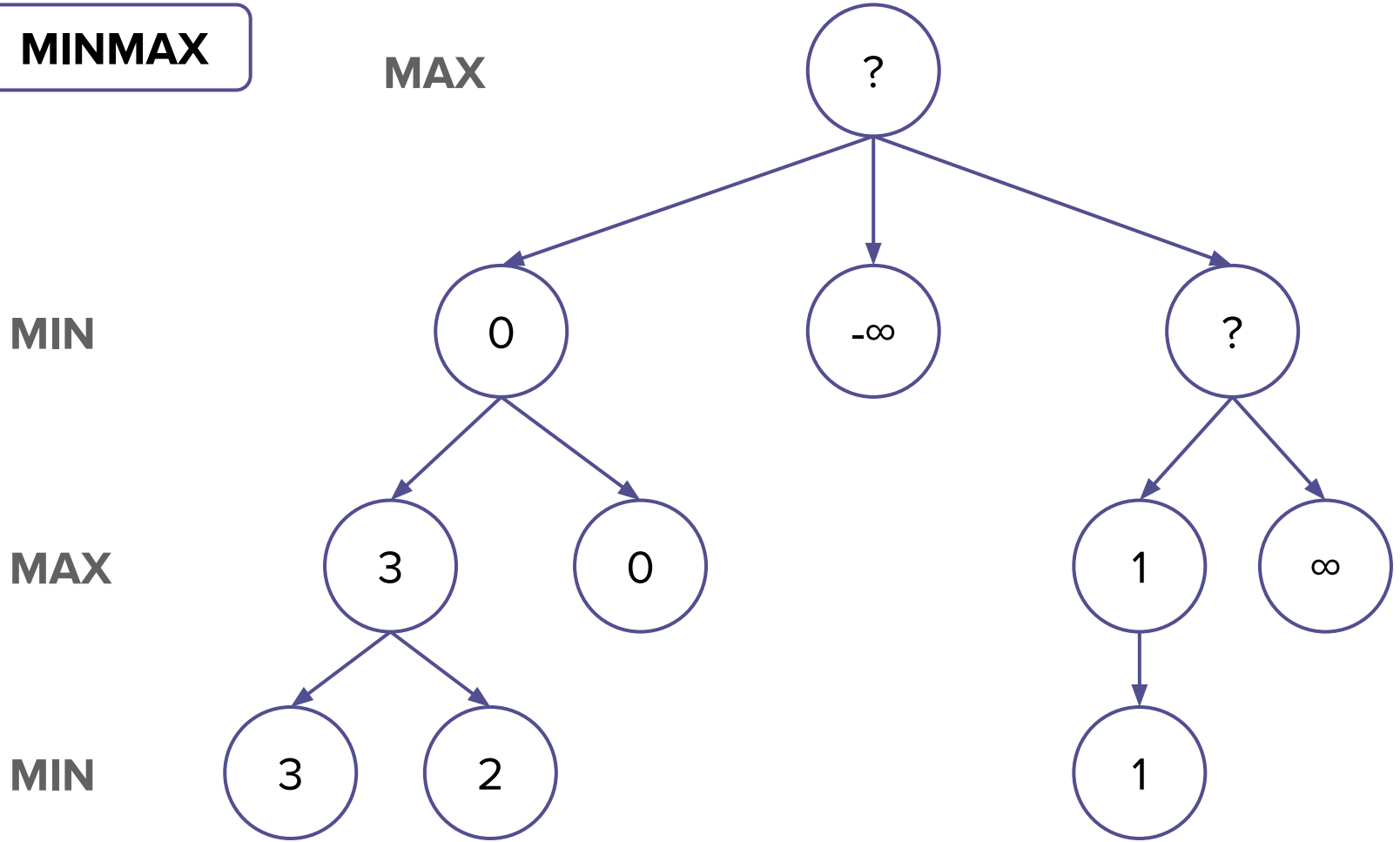
MINMAX



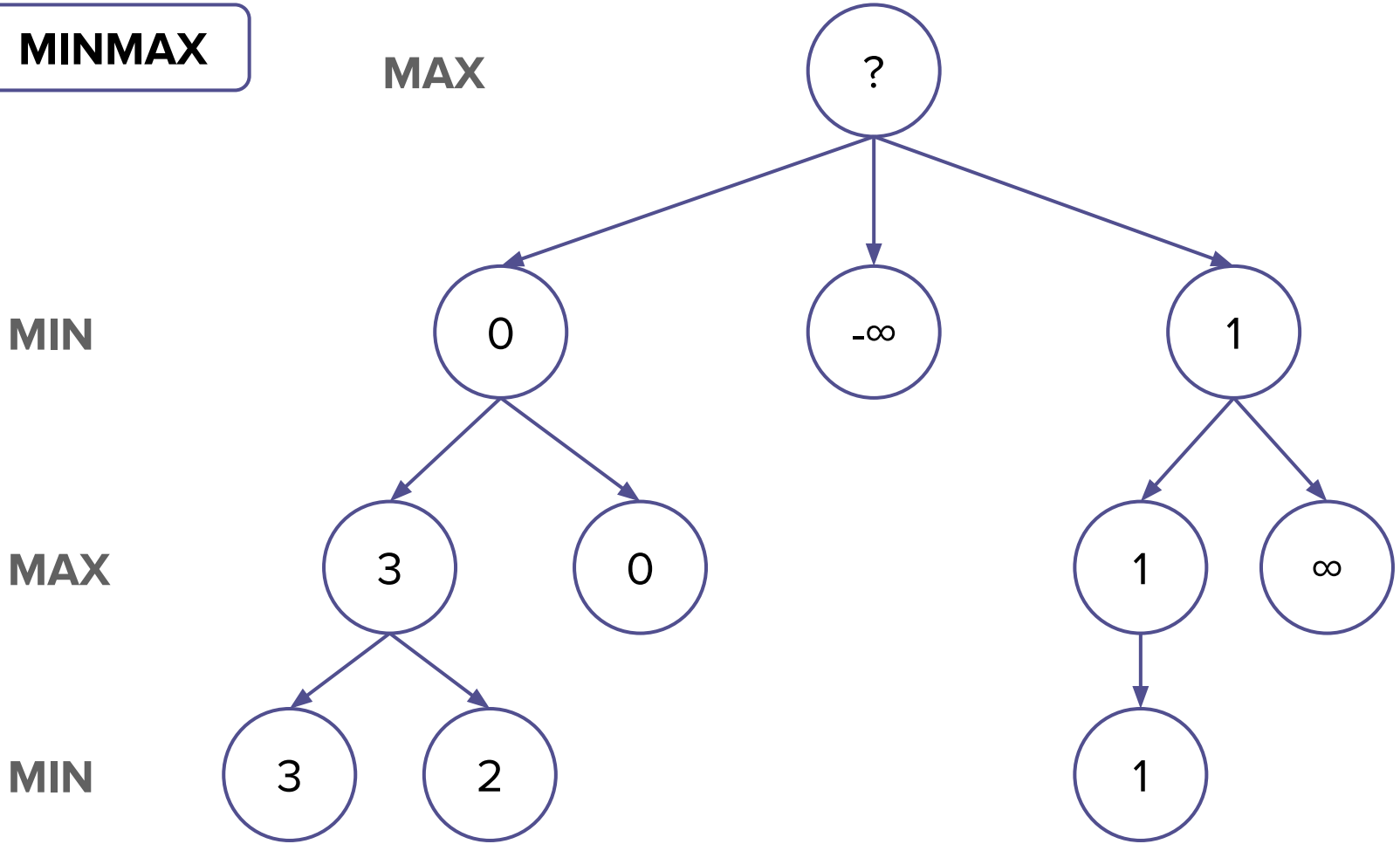
MINMAX



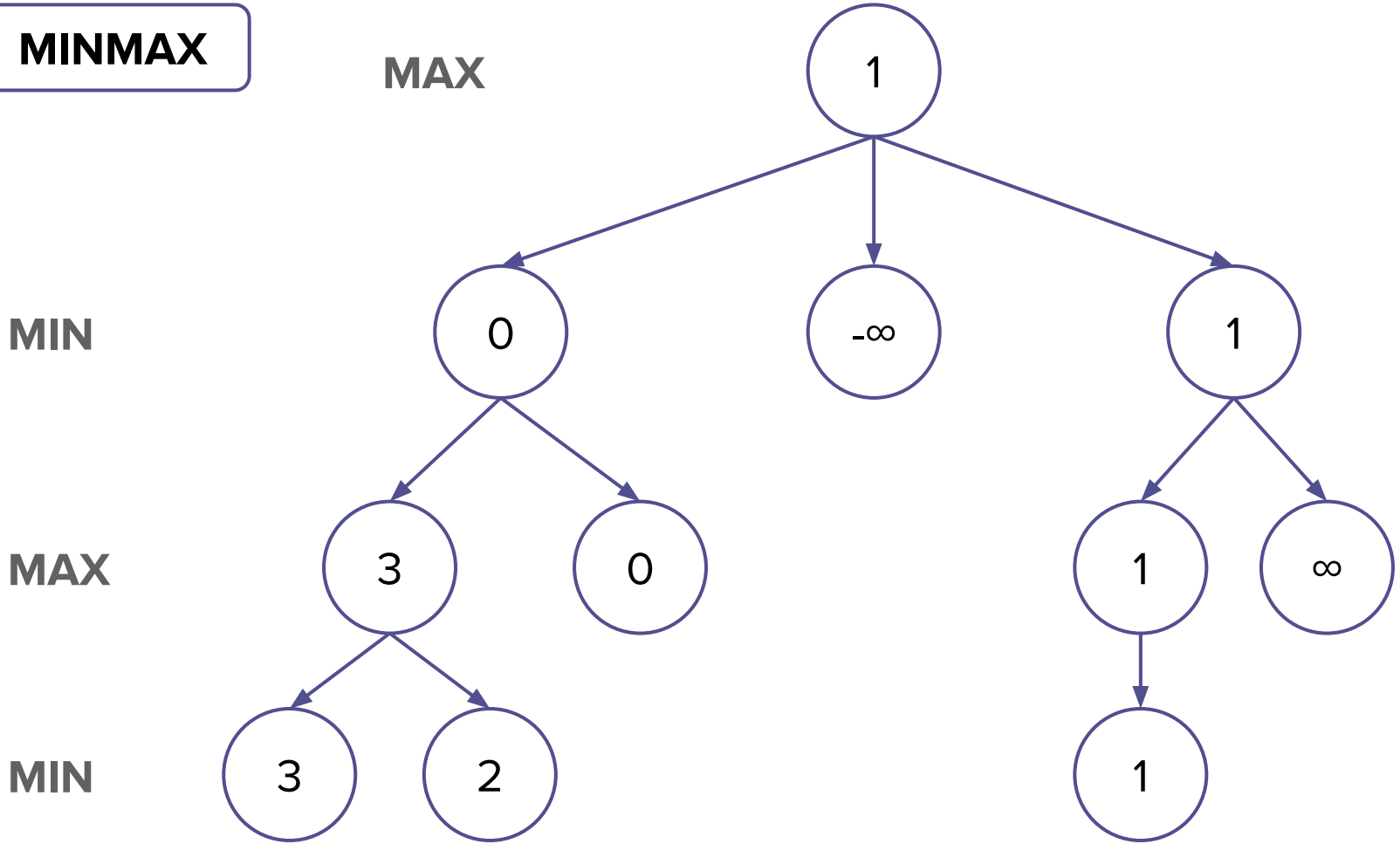
MINMAX



MINMAX



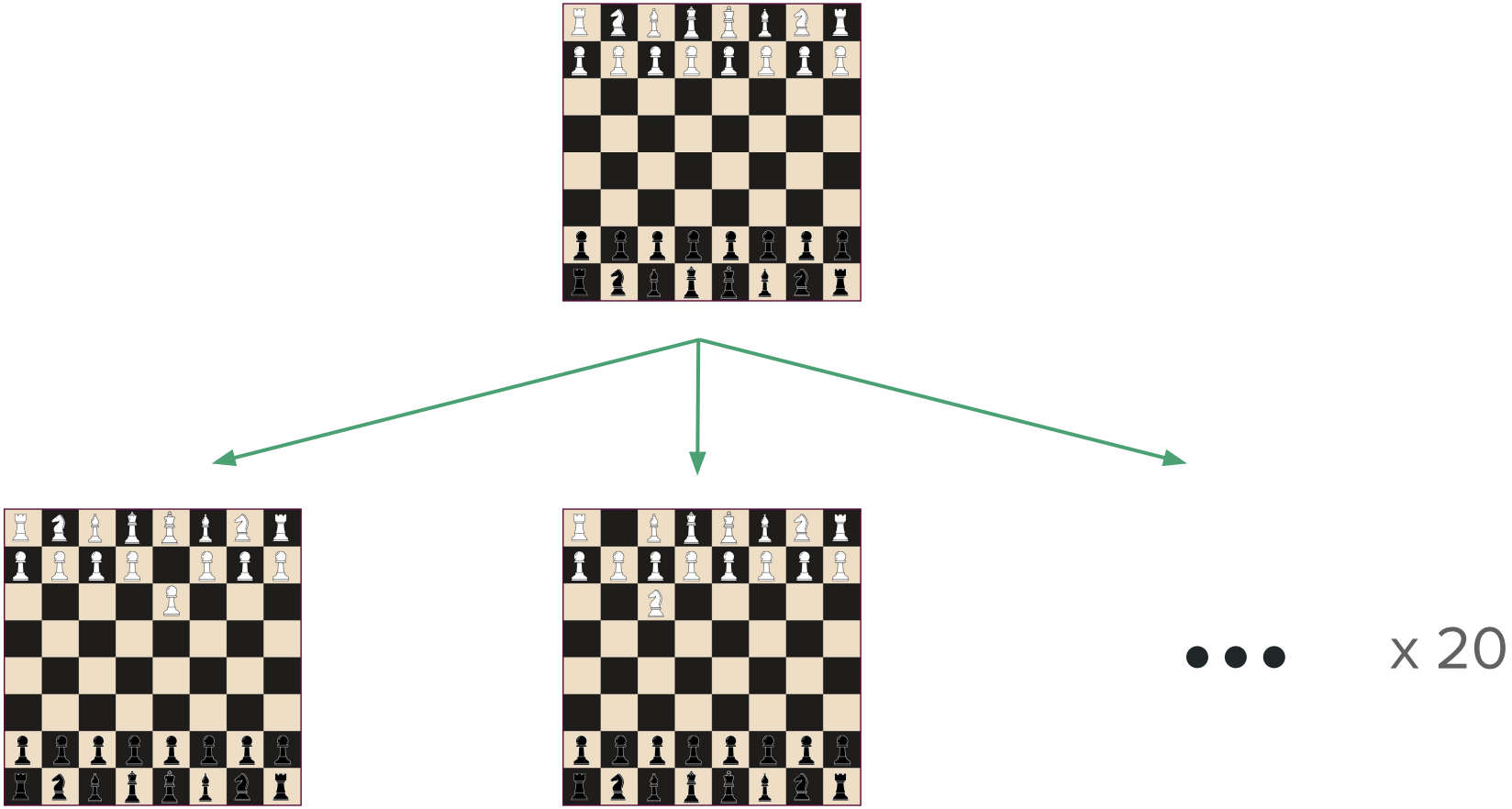
MINMAX



Pseudocod Minmax

```
function minimax(node, depth, maximizingPlayer) is
    if depth = 0 or node is a terminal node then
        return the heuristic value of node
    if maximizingPlayer then
        value := −∞
        for each child of node do
            value := max(value, minimax(child, depth − 1, FALSE))
        return value
    else (* minimizing player *)
        value := +∞
        for each child of node do
            value := min(value, minimax(child, depth − 1, TRUE))
        return value
```

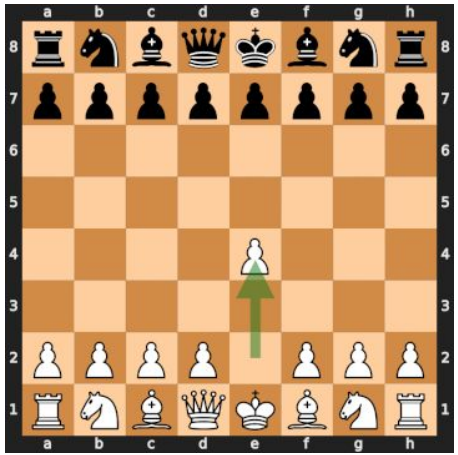
Alpha-Beta Prunning


• • •
x 20

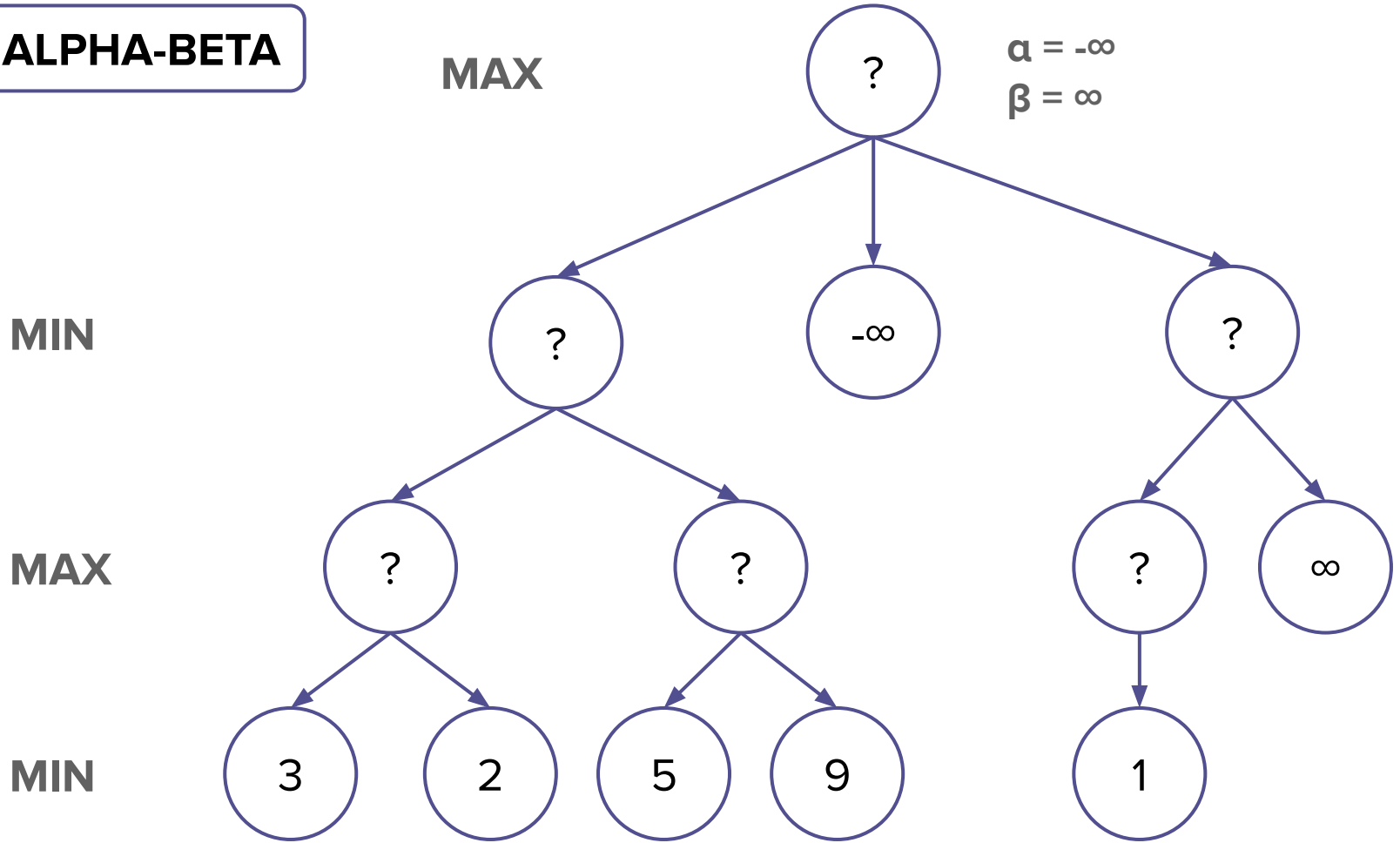
Min

Max

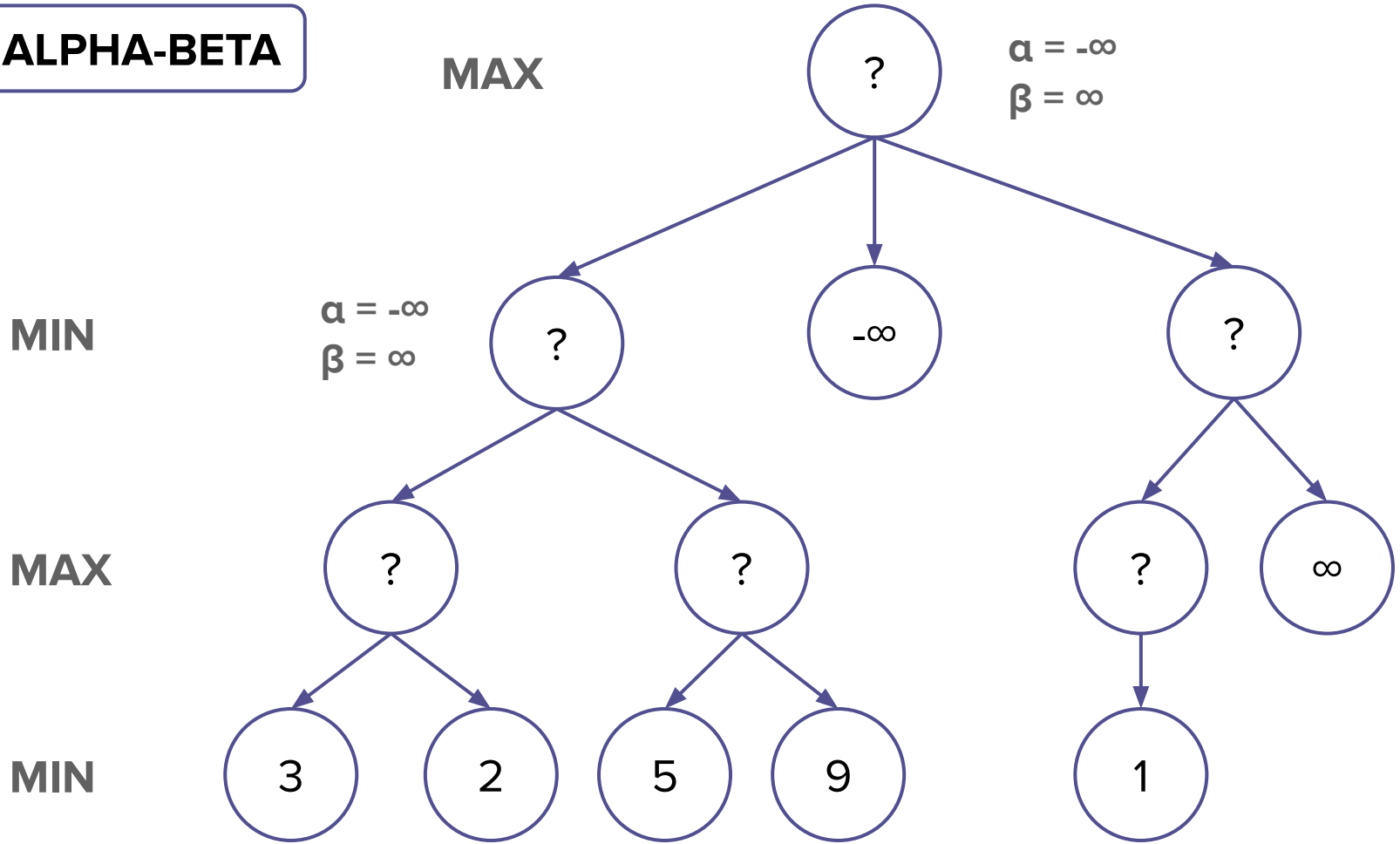
Deep Blue vs. Kasparov (1996)



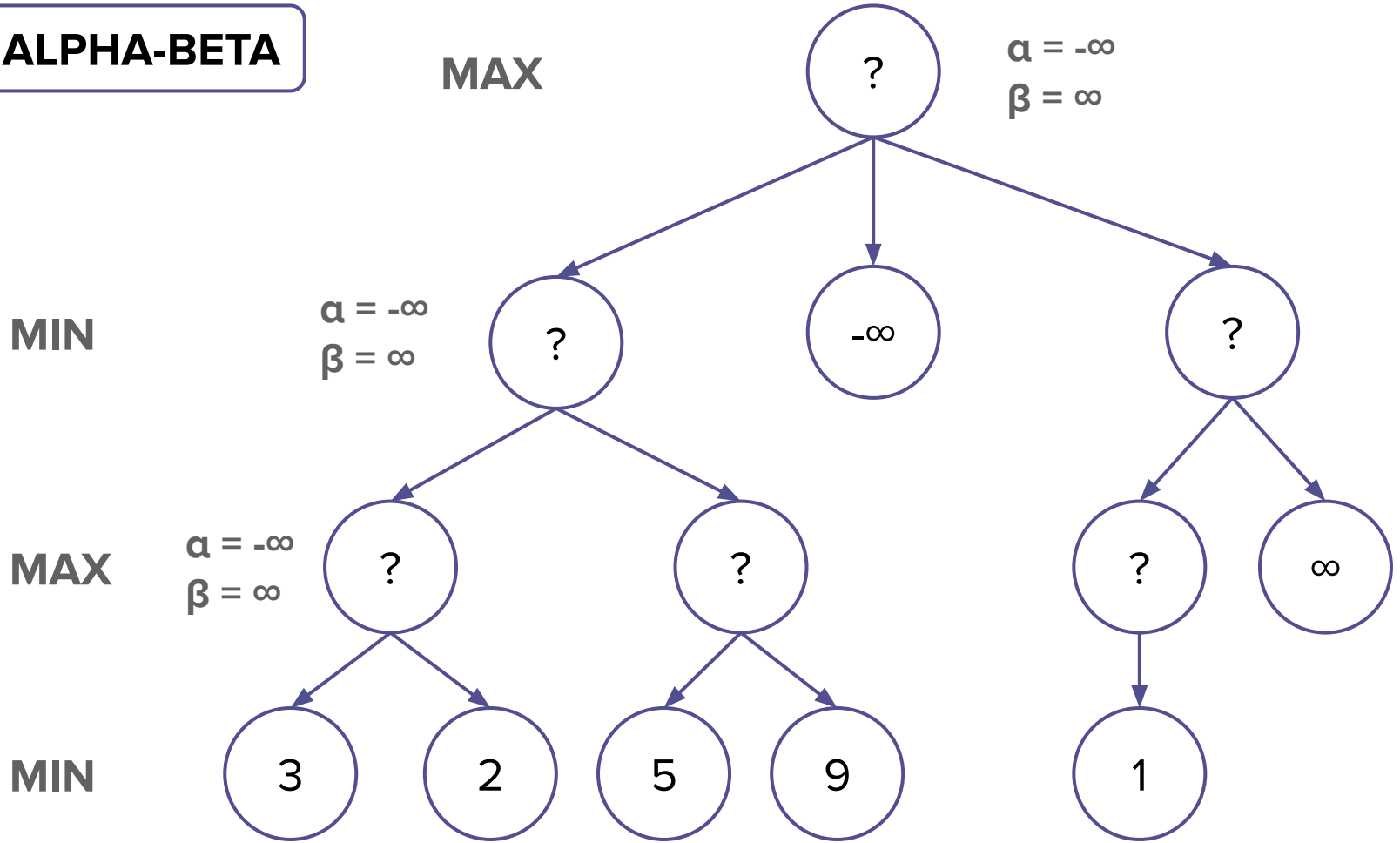
ALPHA-BETA



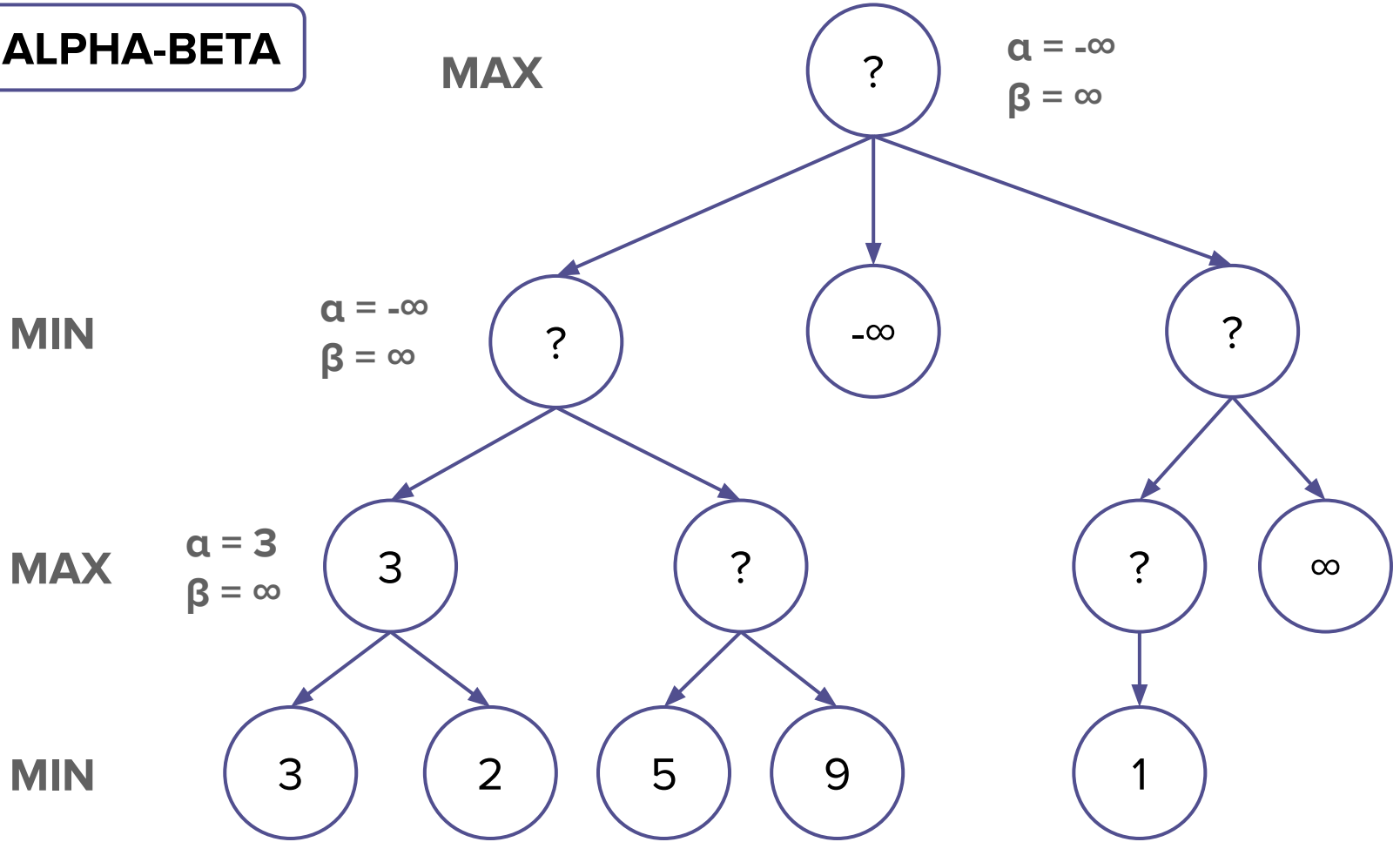
ALPHA-BETA



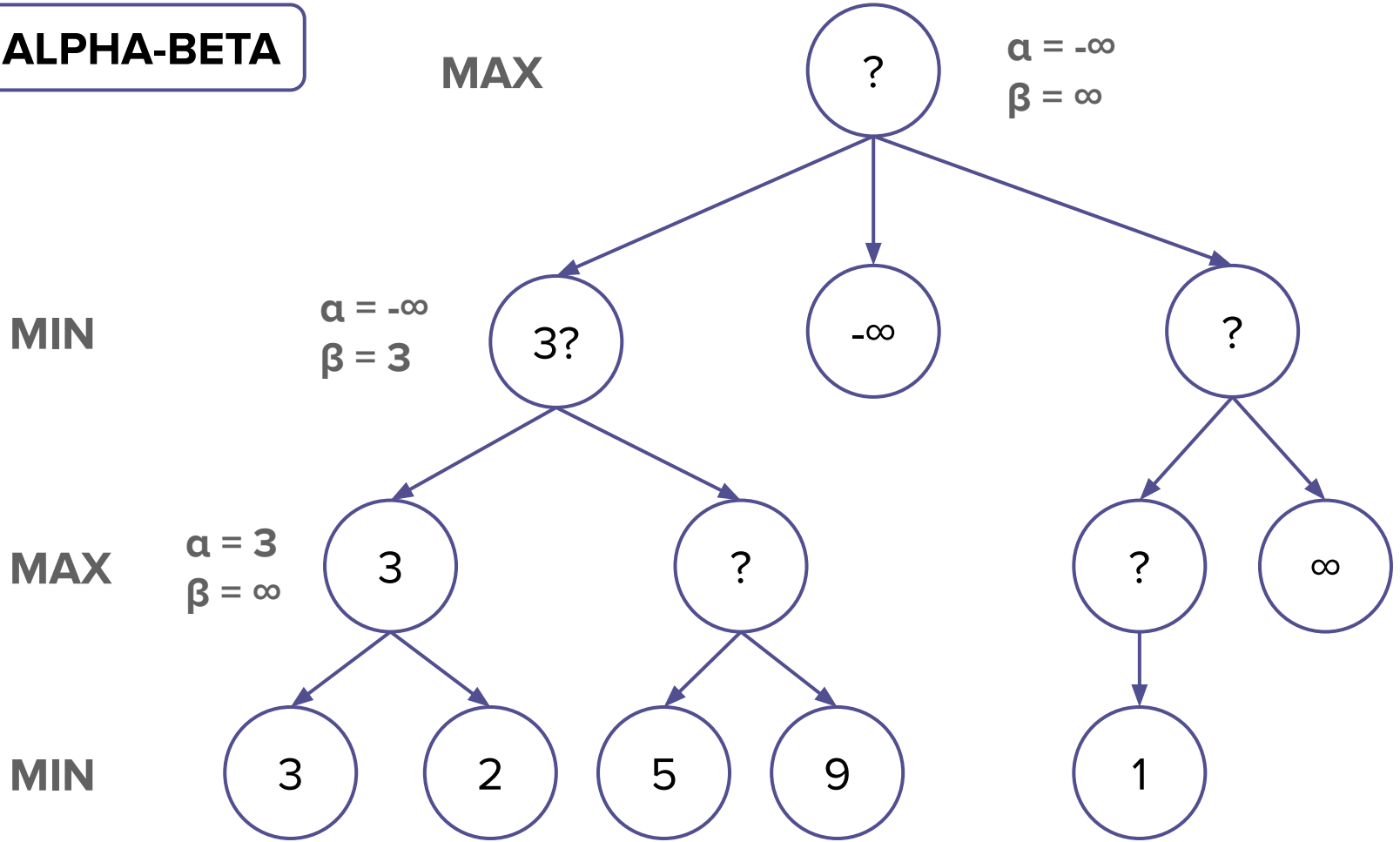
ALPHA-BETA



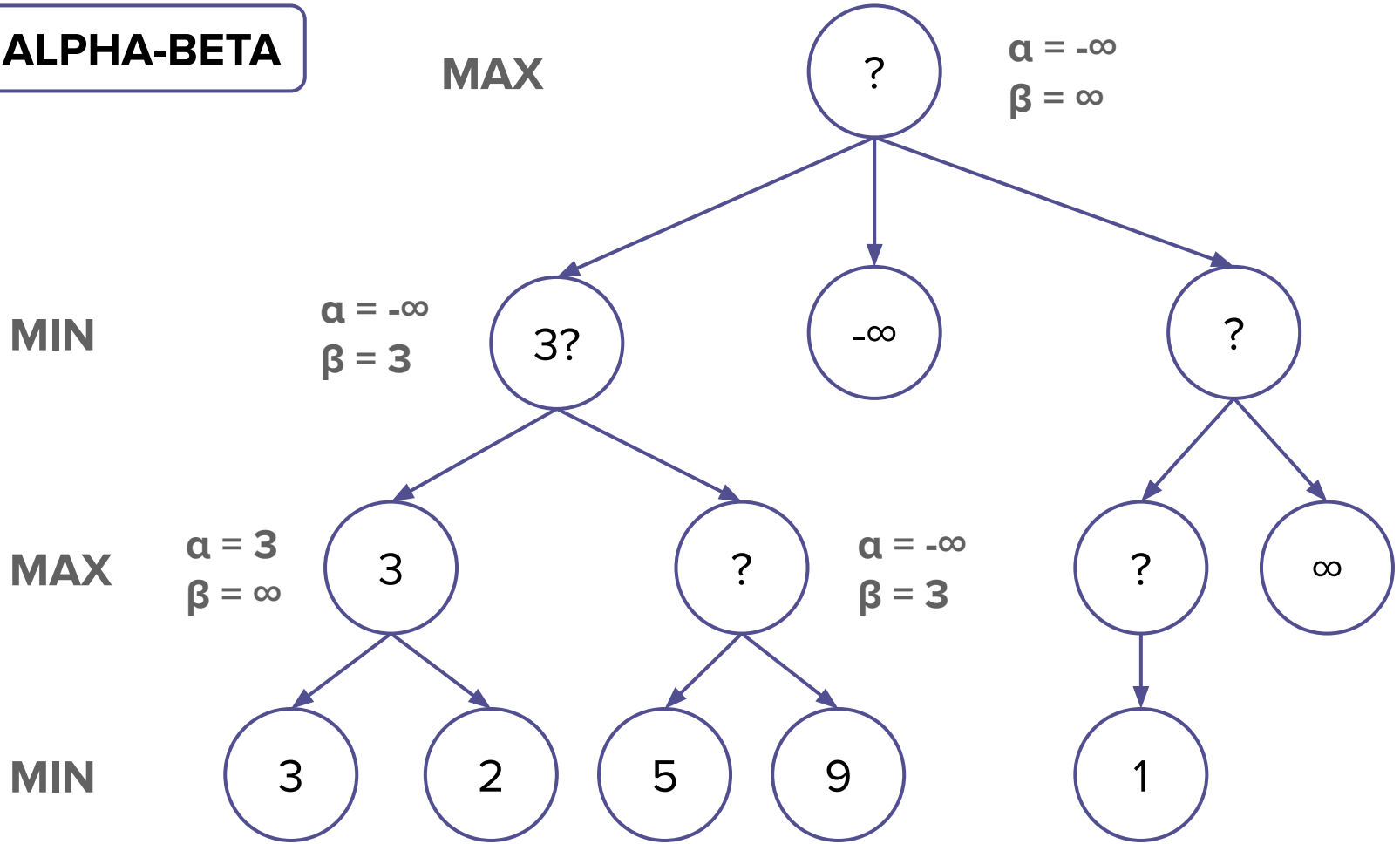
ALPHA-BETA



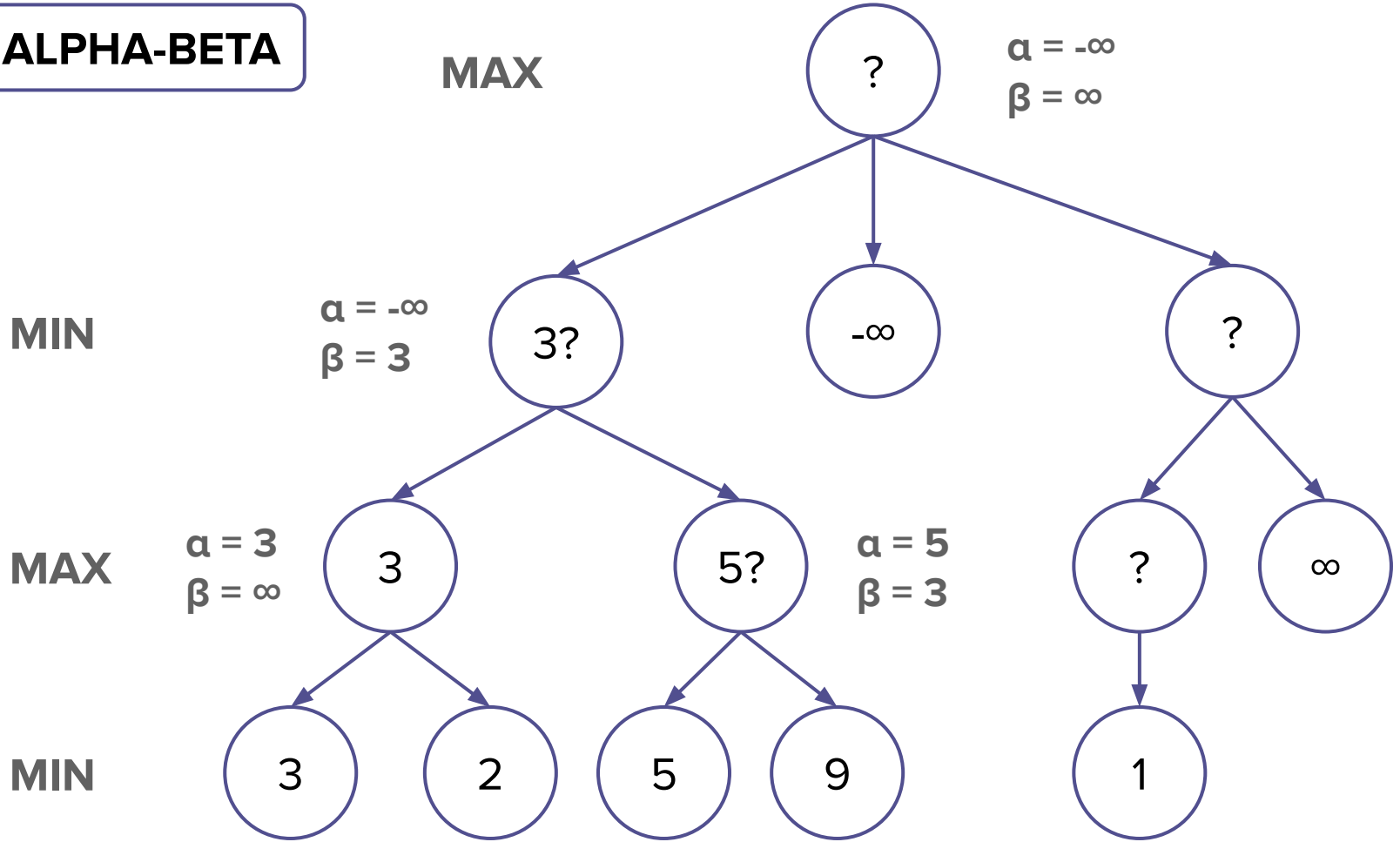
ALPHA-BETA



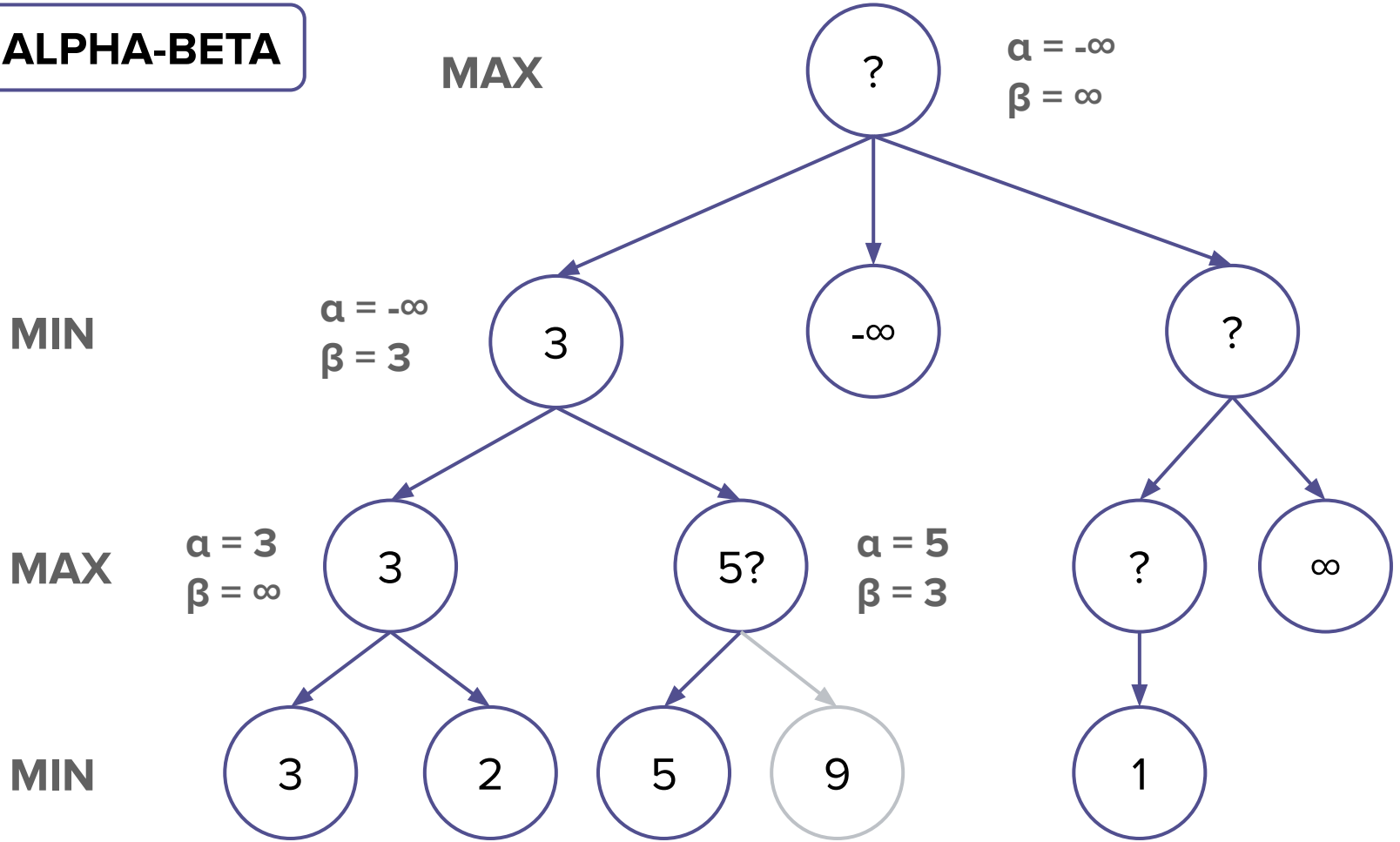
ALPHA-BETA



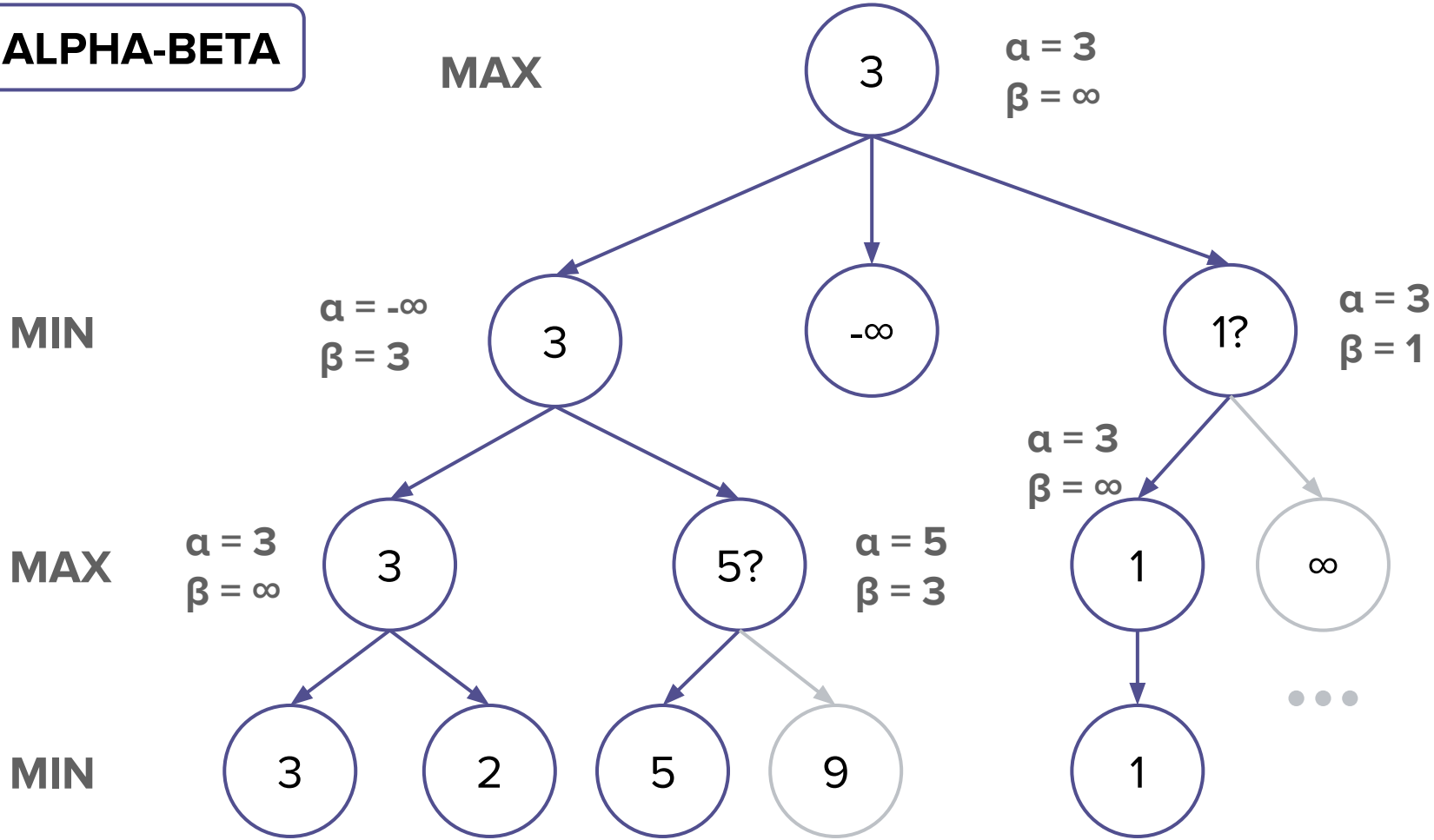
ALPHA-BETA



ALPHA-BETA



ALPHA-BETA

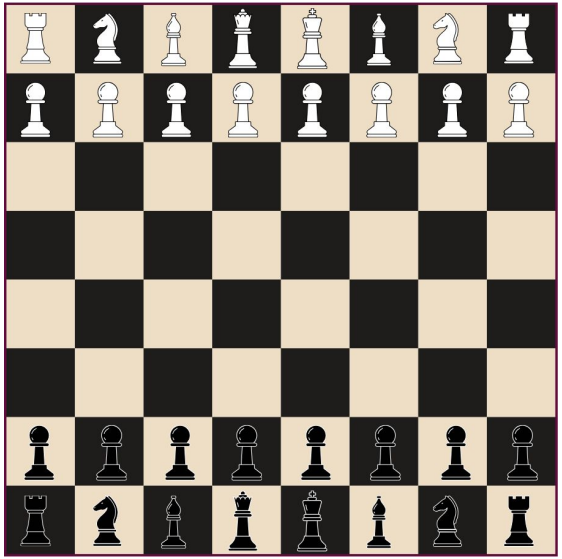


Pseudocod Alpha-Beta Prunning

```
function alphabeta(node, depth, α, β, maximizingPlayer) is
    if depth == 0 or node is terminal then
        return the heuristic value of node
    if maximizingPlayer then
        value := −∞
        for each child of node do
            value := max(value, alphabeta(child, depth − 1, α, β, FALSE))
            if value > β then
                break (* β cutoff *)
            α := max(α, value)
        return value
    else
        value := +∞
        for each child of node do
            value := min(value, alphabeta(child, depth − 1, α, β, TRUE))
            if value < α then
                break (* α cutoff *)
            β := min(β, value)
        return value
```

Q&A

Mulțumesc pentru atenție!




LEAGUE of LEGENDS


WORLD
WARCRAFT

Maria

Gigel

Maria

???

Gigel

Mai multe variații

- 3-player Minmax
- Max^n
- Multiplayer Alpha-Beta Prunning
- Monte Carlo Tree Search
- 0-sum 3-person game
- Expectiminimax:
  - Pentru jocuri cu informatie incompleta (ex: dat cu zarul, move by nature)
  - Environmentul nu are niciun scop / beneficiu în jocul nostru, așadar nu poate juca optim
  - Pentru mutarea lui algoritmul presupune media probabilităților peste toți fiii posibili