procura adversarial ou, simplesmente, jogos (determinísticos)

Ciências ULisboa

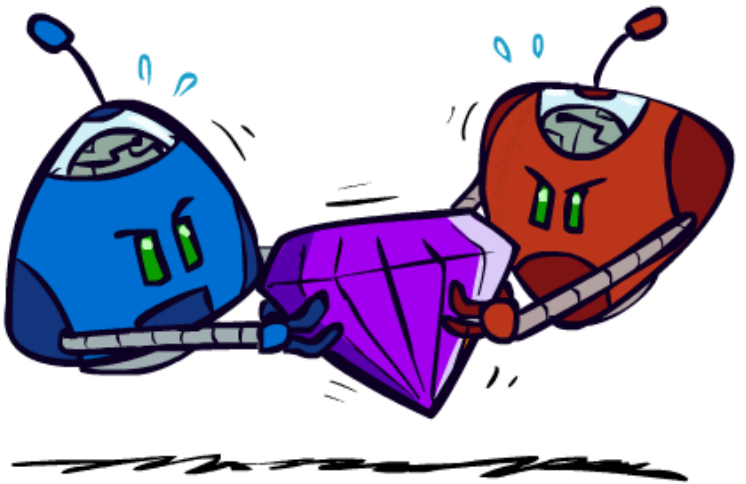
jogos

… há muitos

os que vamos estudar agora:

- determinísticos
- informação perfeita
- dois jogadores, à vez

e soma zero



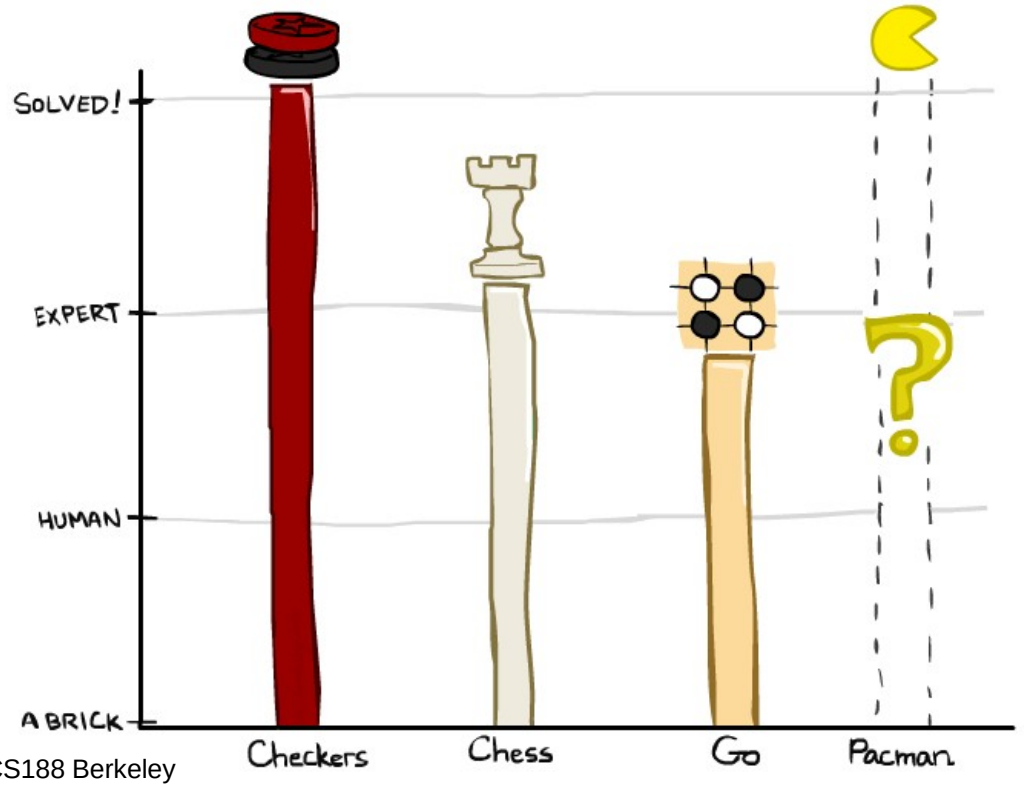
fonte: CS188 Berkeley

o que um ganha o outro perde

Ciências ULisboa

jogos soma zero 2 jogadores

- **Damas**. 1950: 1º programa. 1994: Chinook bateu campeão humano (há 40 anos). 2007: resolvido
- **Xadrez**. 1997: Deep Blue vence campeão humano. Programas atuais melhores ainda
- **Go**. 2016: AlphaGo vence campeão humano
- **Pacman**. 2019: MuZero com muito melhor desempenho que humanos e outros progrs. (12h de aprendizagem)



fonte: CS188 Berkeley

jogos são complexos

xadrez

factor de ramificação, $b \sim 35$

árvore de procura com $\sim 35^{100} \sim 10^{154}$

tempo é determinante!

necessidade de tomar decisões mesmo que não seja viável saber qual é o ótimo

Ciências ULisboa

terminologia

S_0: estado inicial

Player(s): jogador que tem vez no estado *s*

MAX – 1º a jogar
MIN – 2º a jogar

Actions(s): movimentos possíveis no estado *s*

Result(s, a): resultado após movimento *a* no estado *s*

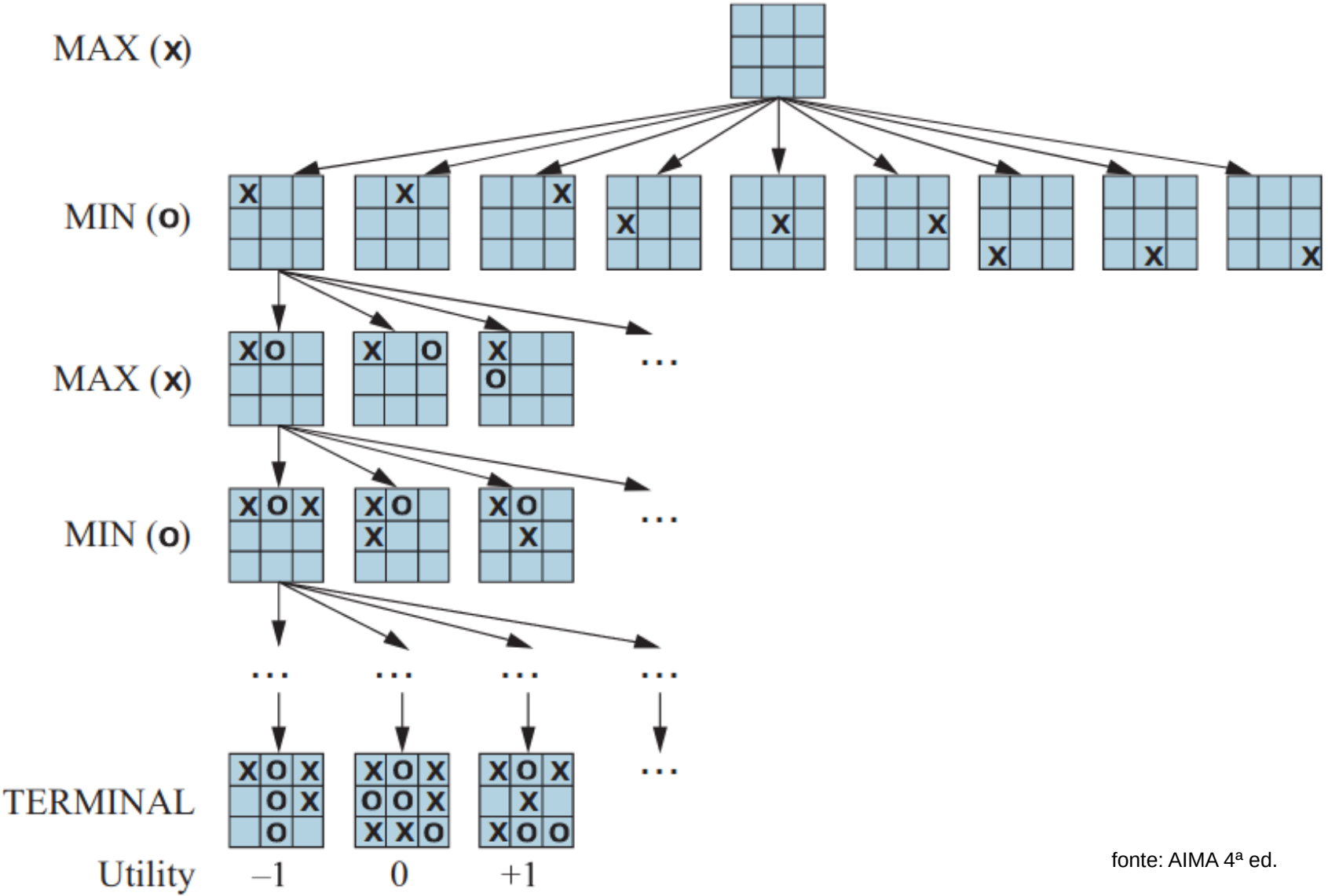
TerminalTest(s): teste de fim de jogo. V => estado terminal

Utility(s, p): ganho para o jogador *p* quanto o jogo termina no estado *s* (ex: no xadrez, 0, ½, +1)

utilidade ou ganho

Ciências ULisboa

jogo do galo



fonte: AIMA 4ª ed.

nota sobre espaços de procura

	estrutura	**ciclos**	**estados repetidos**
xadrez	grafo	sim	não
galo	árvore	não	sim
?	árvore	não	não

Ciências ULisboa

árvore de procura

nº de nós (estados) terminais

- jogo do galo: $< 9! = 362.880$
- xadrez: $> 10^{40}$

estratégia ótima produz resultado pelo menos tão bom como qualquer outra, se o adversário for infalível

árvore de procura usada é um subconjunto da árvore completa para permitir escolher jogada

Ciências ULisboa

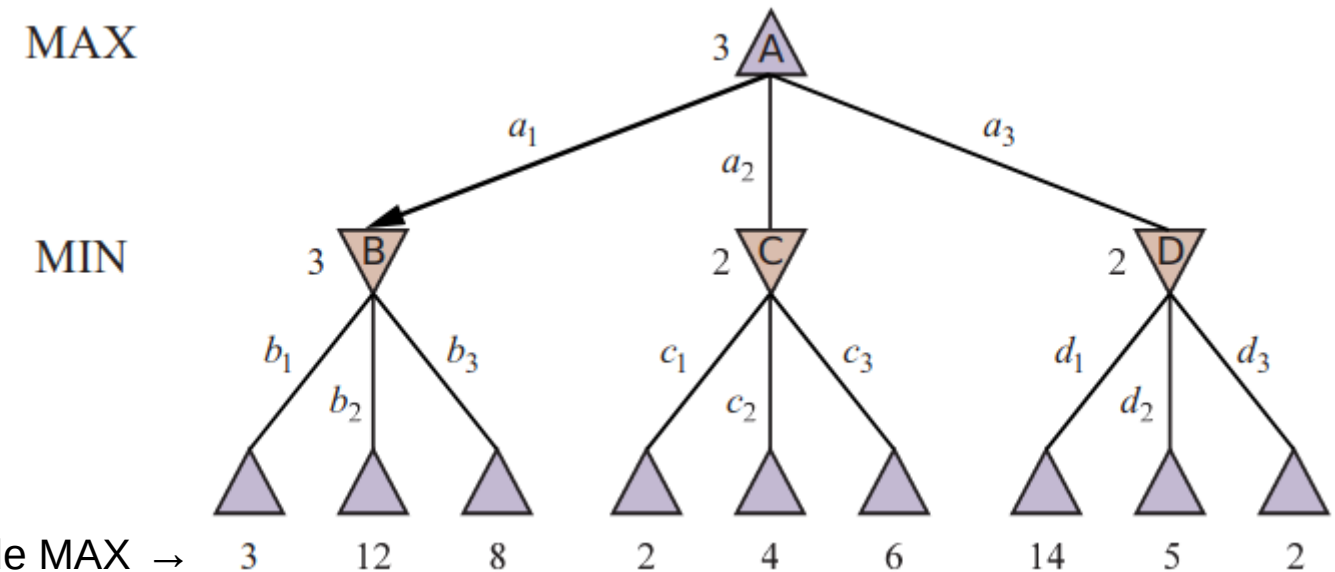
estratégia ótima

exemplo com um jogo mais simples

cada jogador só joga uma vez (jogo de 1 movimento)

△: ações de MAX

▽: ações de MIN



fonte: AIMA 4ª ed.

Ciências ULisboa

estratégia ótima

minimax (de um nó)

ganho de MAX num nó quando, daí em diante, ambos os jogadores escolhem jogadas ótimas

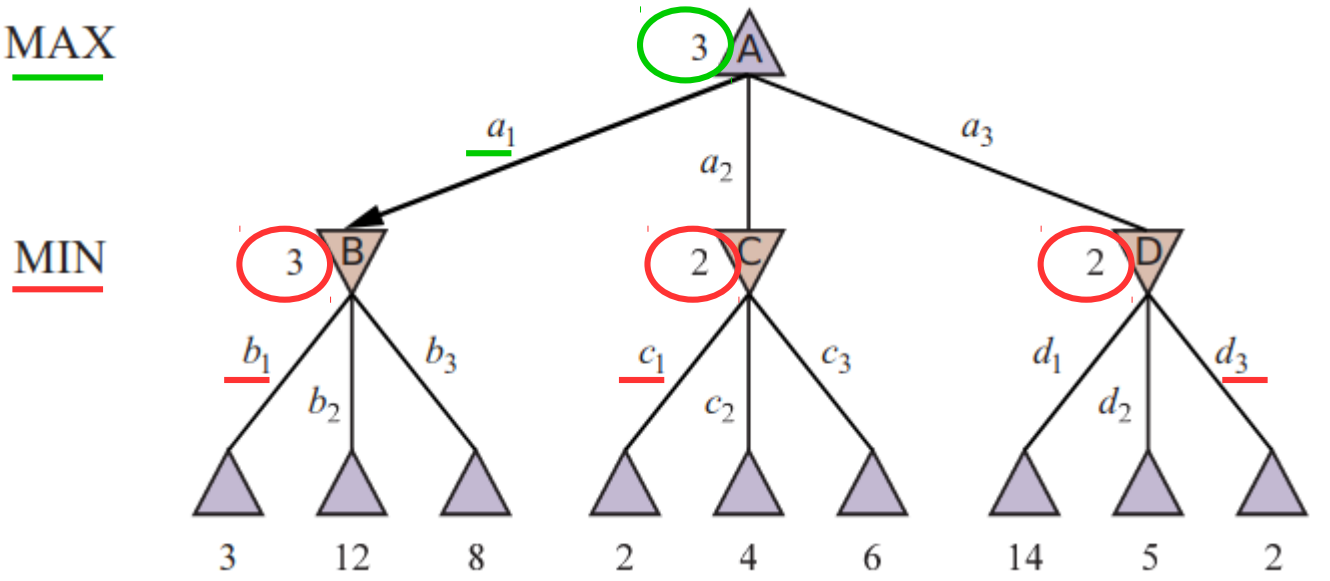
$$\textsc{Minimax}(s) = \begin{cases} \textsc{Utility}(s) & \text{se } \textsc{TerminalTest}(s) \\ \max_{a \in \textsc{Actions}(s)} \textsc{Minimax}(\textsc{Result}(s, a)) & \text{se } \textsc{Jogador}(s) = \text{MAX} \\ \min_{a \in \textsc{Actions}(s)} \textsc{Minimax}(\textsc{Result}(s, a)) & \text{se } \textsc{Jogador}(s) = \text{MIN} \end{cases}$$

MAX prefere ganho máximo; MIN prefere ganho mínimo

ganhos de MAX

Ciências ULisboa

no jogo de profundidade 1



adaptado de AIMA 4ª ed.

MAX – escolhe melhor dos piores casos (MIN ótimo)

se MIN não for ótimo, MAX tem ganho maior ainda

computação naturalmente recursiva

Ciências ULisboa

complexidade

minimax faz uma pesquisa completa em profundidade

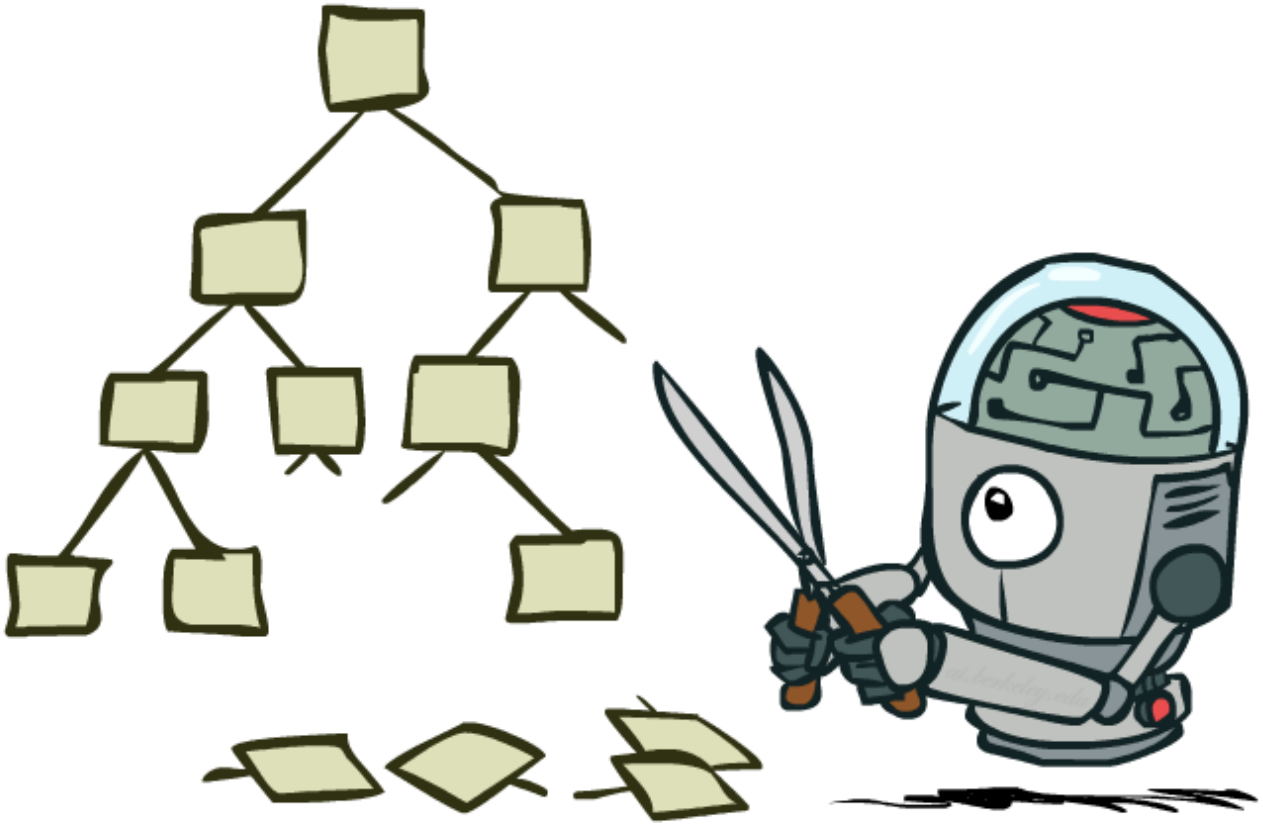
- complexidade temporal:
  - $O(b^m)$
- complexidade espacial:
  - $O(bm)$ - se expande de uma vez todas as ações de um nó
  - $O(m)$ – se expande à vez cada ação de um nó

inviável em jogos normais

mas é a base para algoritmos práticos para jogos normais

Ciências ULisboa

poda



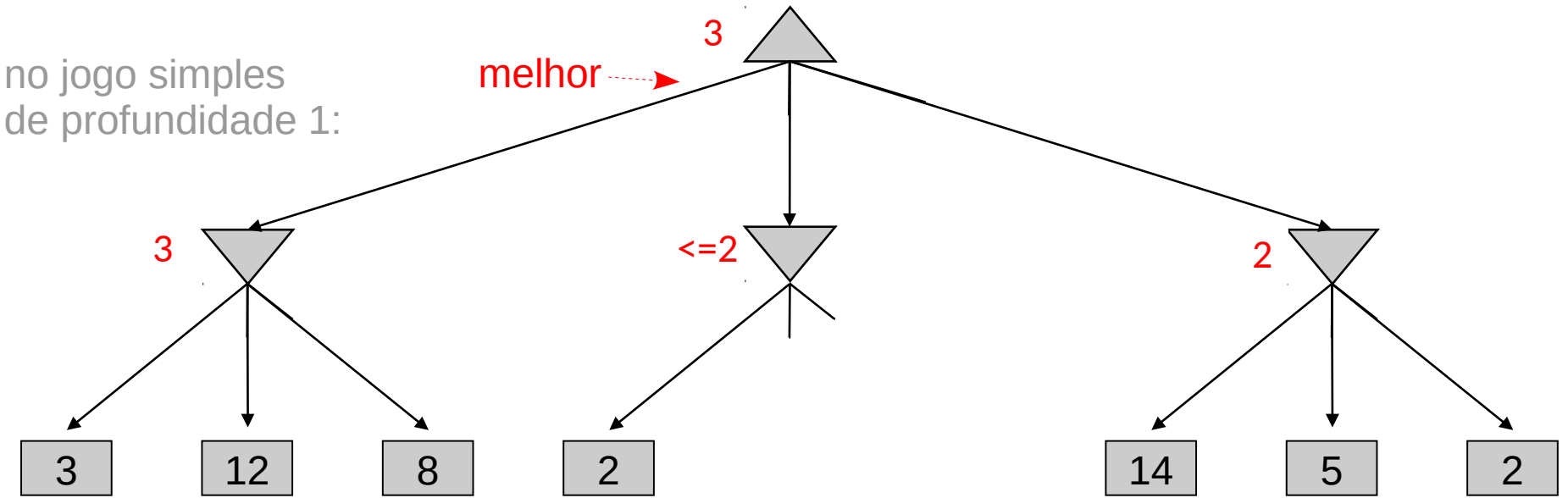
fonte: CS188 Berkeley

Ciências ULisboa

poda alfa-beta

retorna o mesmo resultado do minimax

mas não expande caminhos que não podem influir no resultado



fonte: CS188 Berkeley

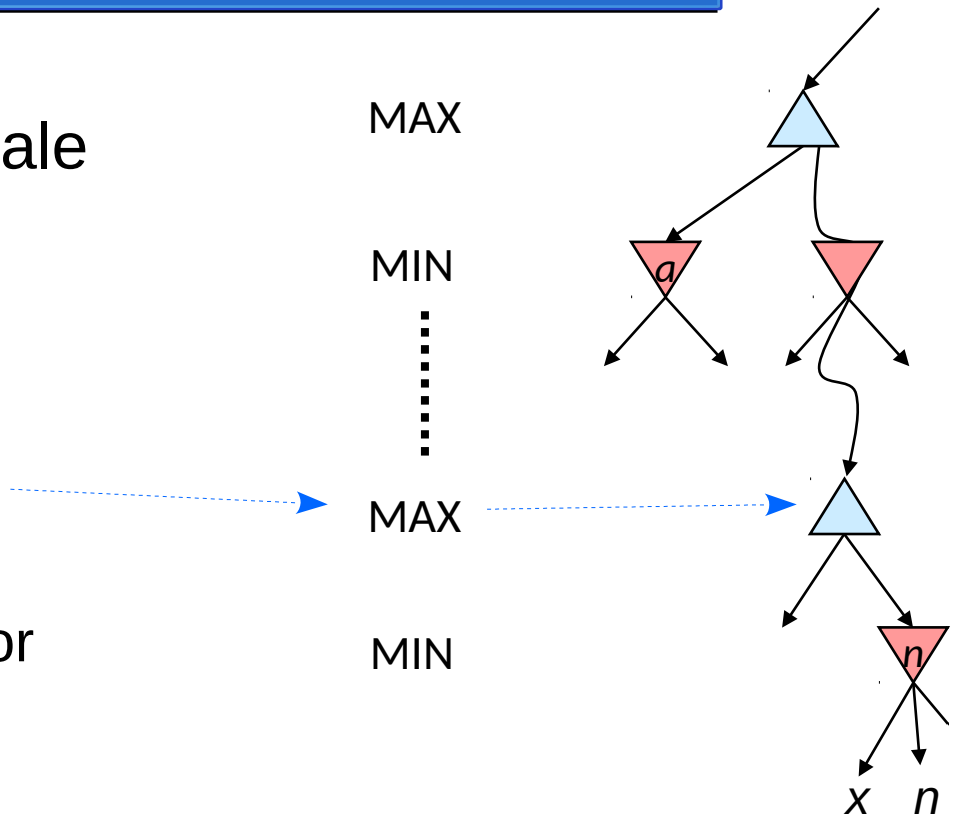
Ciências ULisboa

poda pelo menor valor – nó MIN

- se o valor de *n* for pior (menor) do que *a*, não vale a pena expandir mais irmãos de *x* e *n*

  porque MAX, na jogada acima, nunca há-de escolher esse ramo

  (comparação com melhor valor de MAX é feita no ciclo do MIN)



fonte: AIMA 4ª ed.

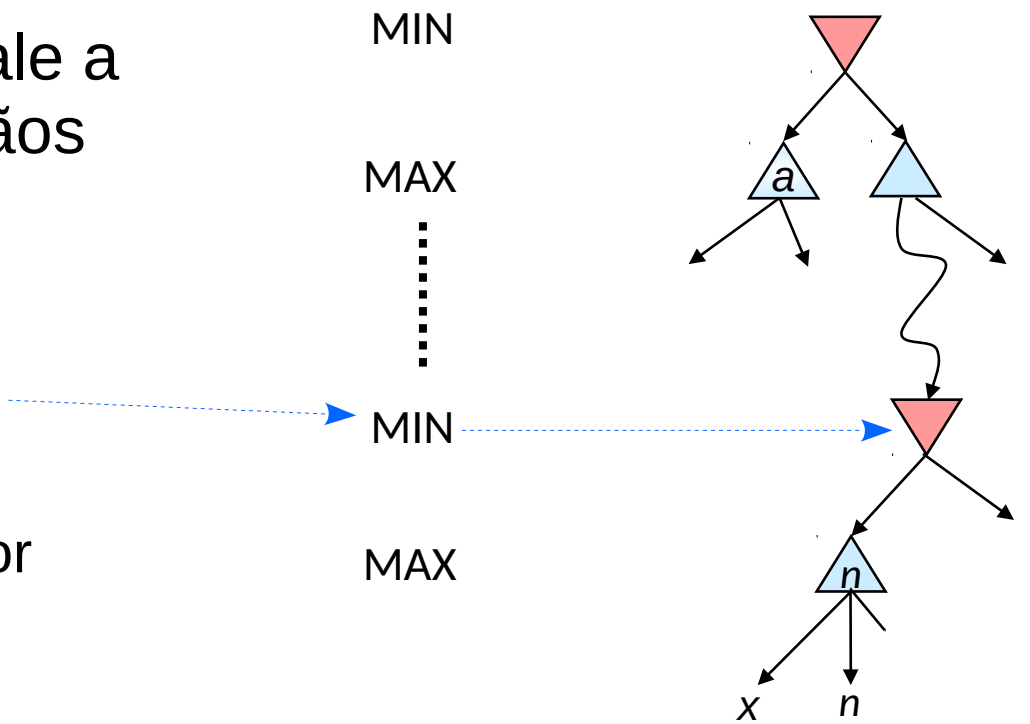
Ciências ULisboa

poda pelo maior valor – nó MAX

- se o valor de *n* for pior (maior) do que *a*, não vale a pena expandir mais irmãos de *x* e *n*

  porque MIN, na jogada acima, nunca há-de escolher esse ramo

  (comparação com melhor valor de MIN é feita no ciclo do MAX)



Ciências ULisboa

o alfa e o beta

- α = valor da melhor escolha (maior valor) do MAX até agora

- β = valor da melhor escolha (menor valor) do MIN até agora

pára a expansão do nó atual assim que o valor deste é pior do que α ou β, conforme jogada de MAX ou de MIN respetivamente

Ciências ULisboa

```
function ALPHA-BETA-SEARCH(state) returns an action
  v ← MAX-VALUE(state, −∞, +∞)
  return the action in ACTIONS(state) with value v

function MAX-VALUE(state, α, β) returns a utility value
  if TERMINAL-TEST(state) then return UTILITY(state)
  v ← −∞
  for each a in ACTIONS(state) do
    v ← MAX(v, MIN-VALUE(RESULT(s,a), α, β))
    if v ≥ β then return v        ◄ ponto de poda
    α ← MAX(α, v)
  return v

function MIN-VALUE(state, α, β) returns a utility value
  if TERMINAL-TEST(state) then return UTILITY(state)
  v ← +∞
  for each a in ACTIONS(state) do
    v ← MIN(v, MAX-VALUE(RESULT(s,a) , α, β))
    if v ≤ α then return v        ◄ ponto de poda
    β ← MIN(β, v)
  return v
```

complexidade temporal da poda alfa-beta

se sucessores estiverem por ordem aleatória

$$O(b^{3m/4})$$

mas...

se os sucessores estiverem ordenados (melhores 1º)

$$O(b^{m/2})$$

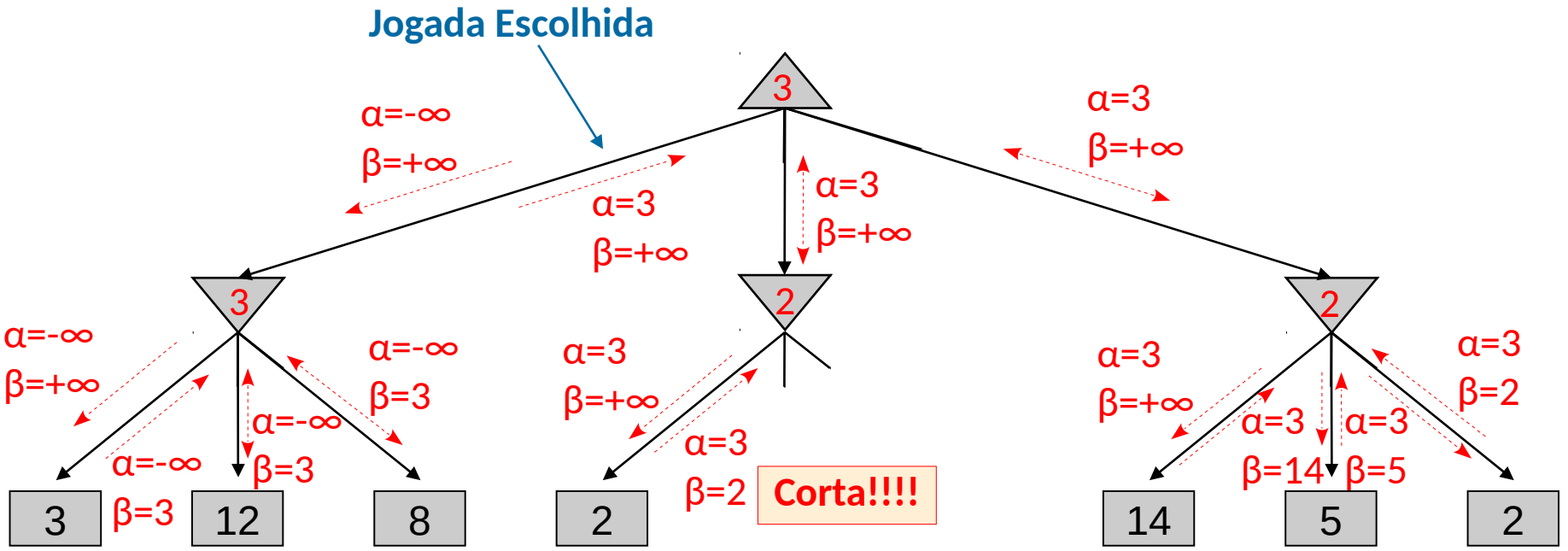
ex: no xadrez pode preferir-se captura-ameaça-avanço-recuo
o que aproxima a cerca 2x esse valor

Ciências ULisboa

extra

Ciências ULisboa

valores de alfa e beta no jogo simples



Ciências ULisboa

jogo dos peões (slides seguintes)

jogar peões de xadrez
ganha o 1º a atingir o lado oposto

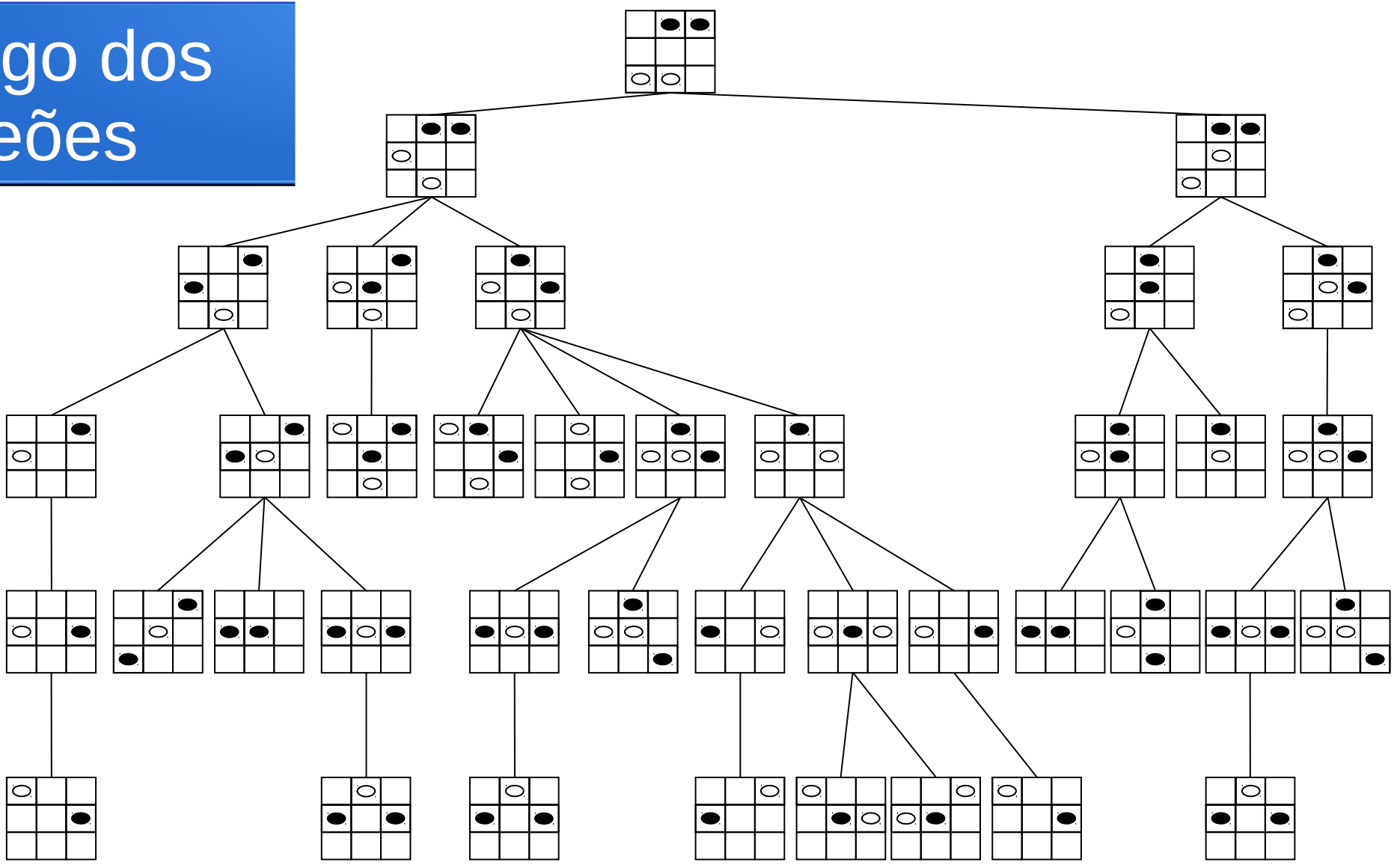
- fazer a análise da poda alfa-beta

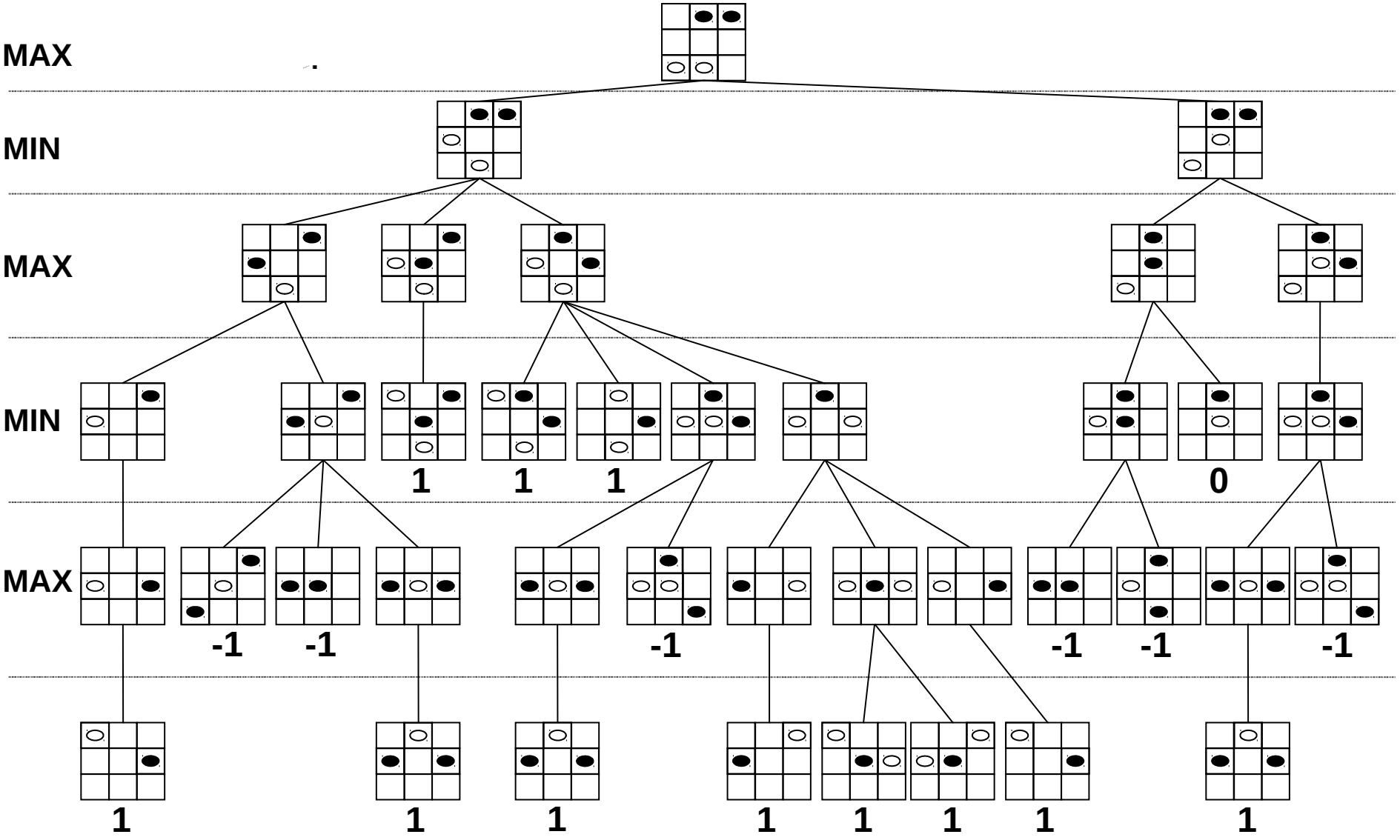
  marcar os valores de alfa e beta na descida (chamada) e no retorno

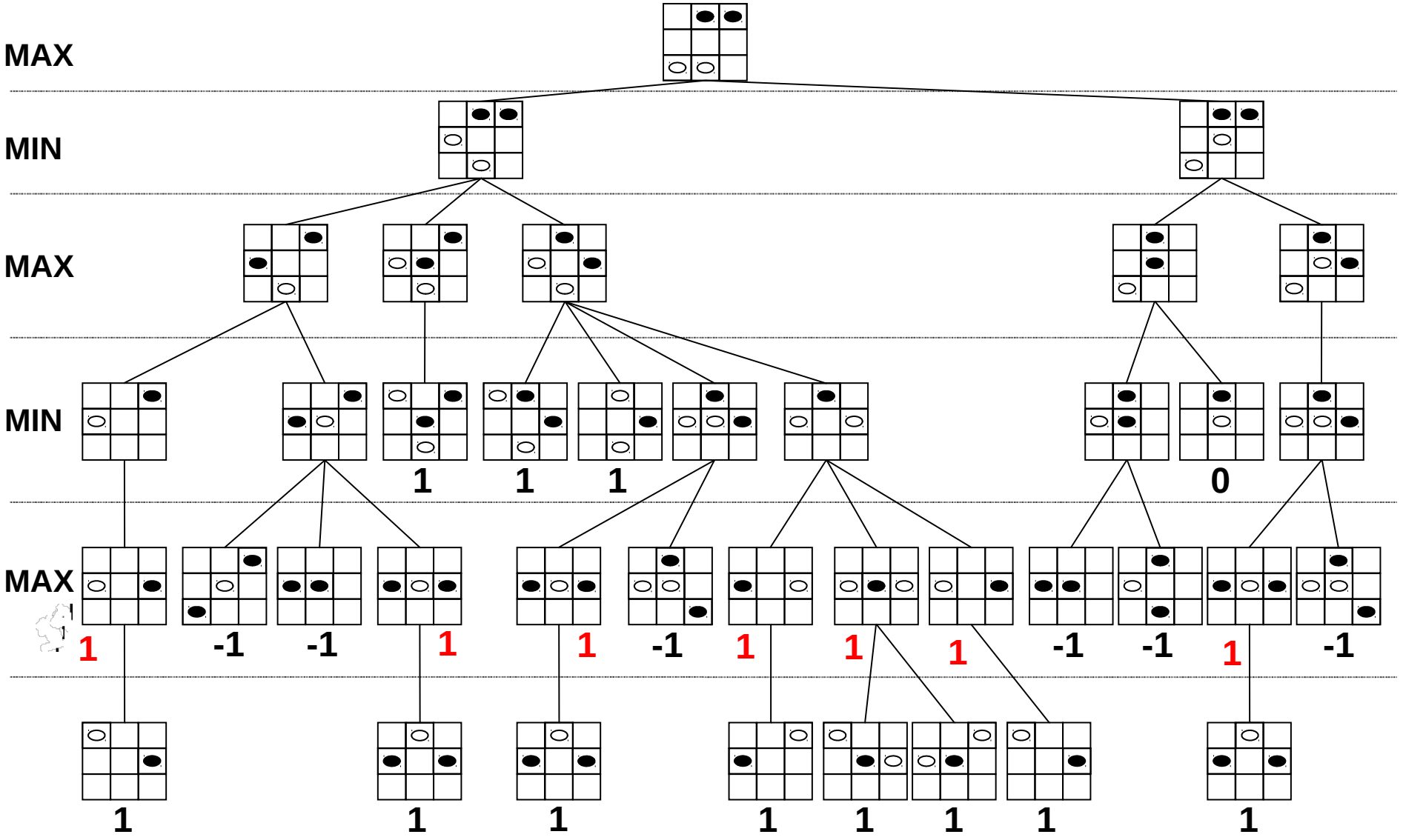
- verificar qual é a ordenação da árvore de procura de modo a que a poda alfa-beta corte o máximo de nós possível

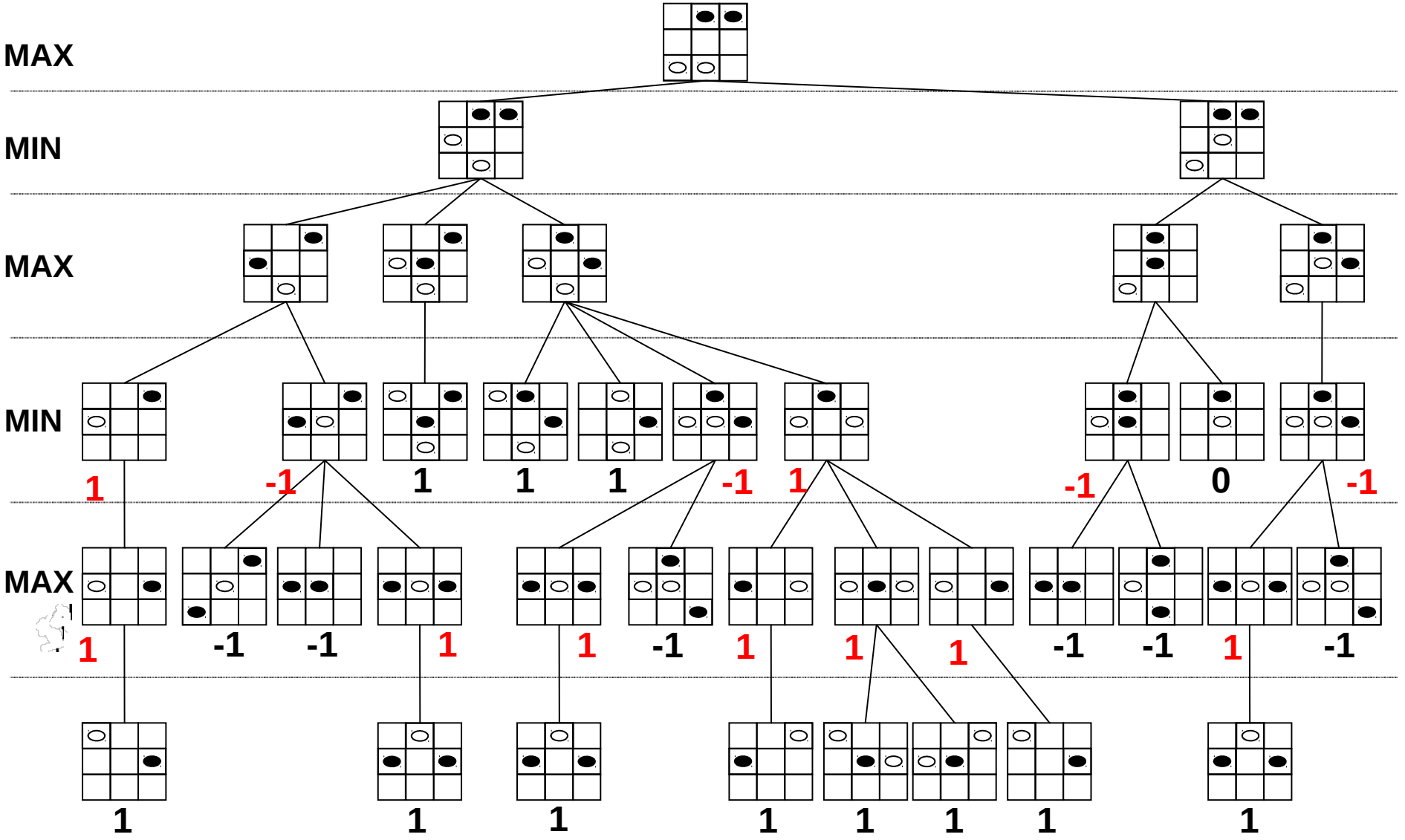
Ciências ULisboa

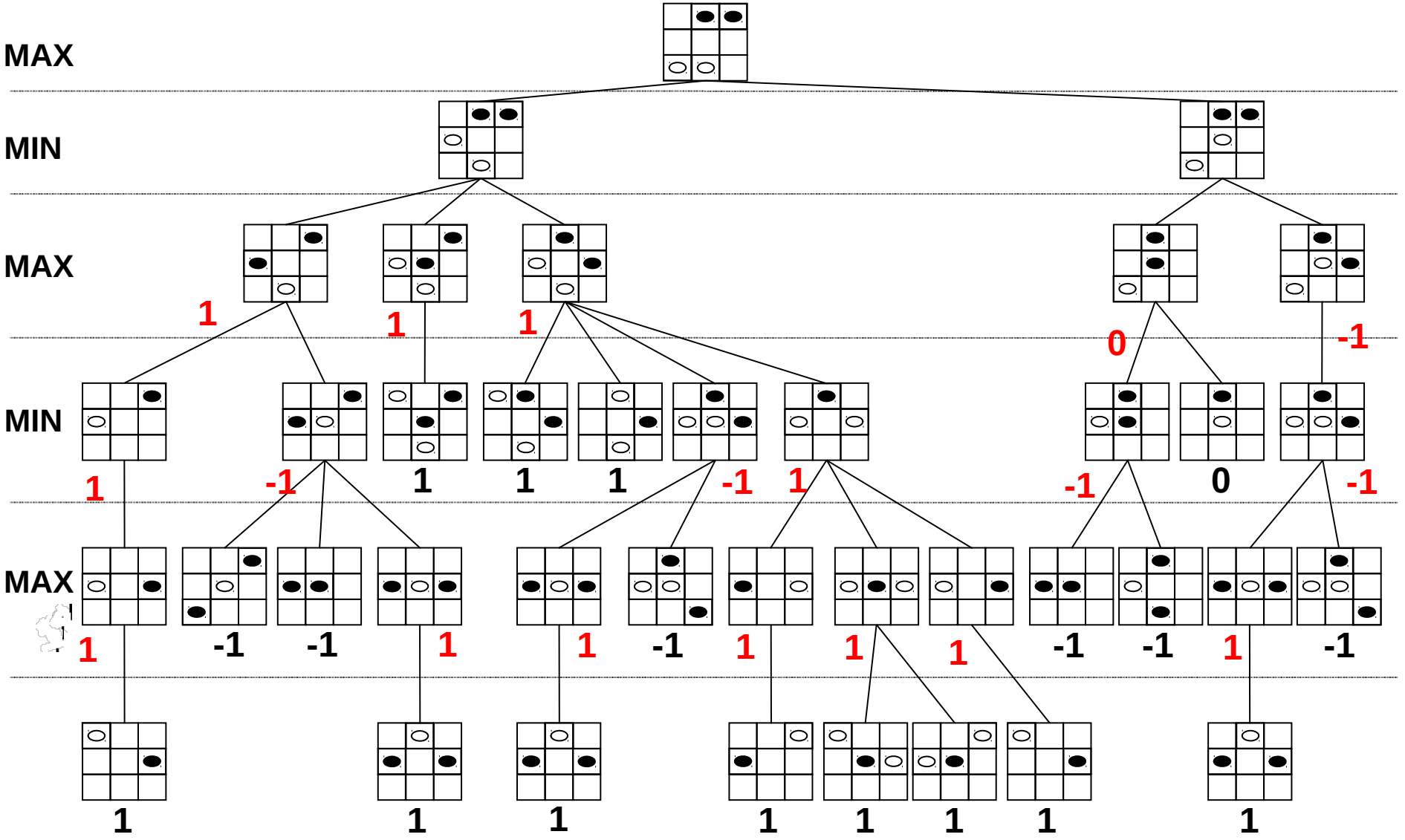
jogo dos peões

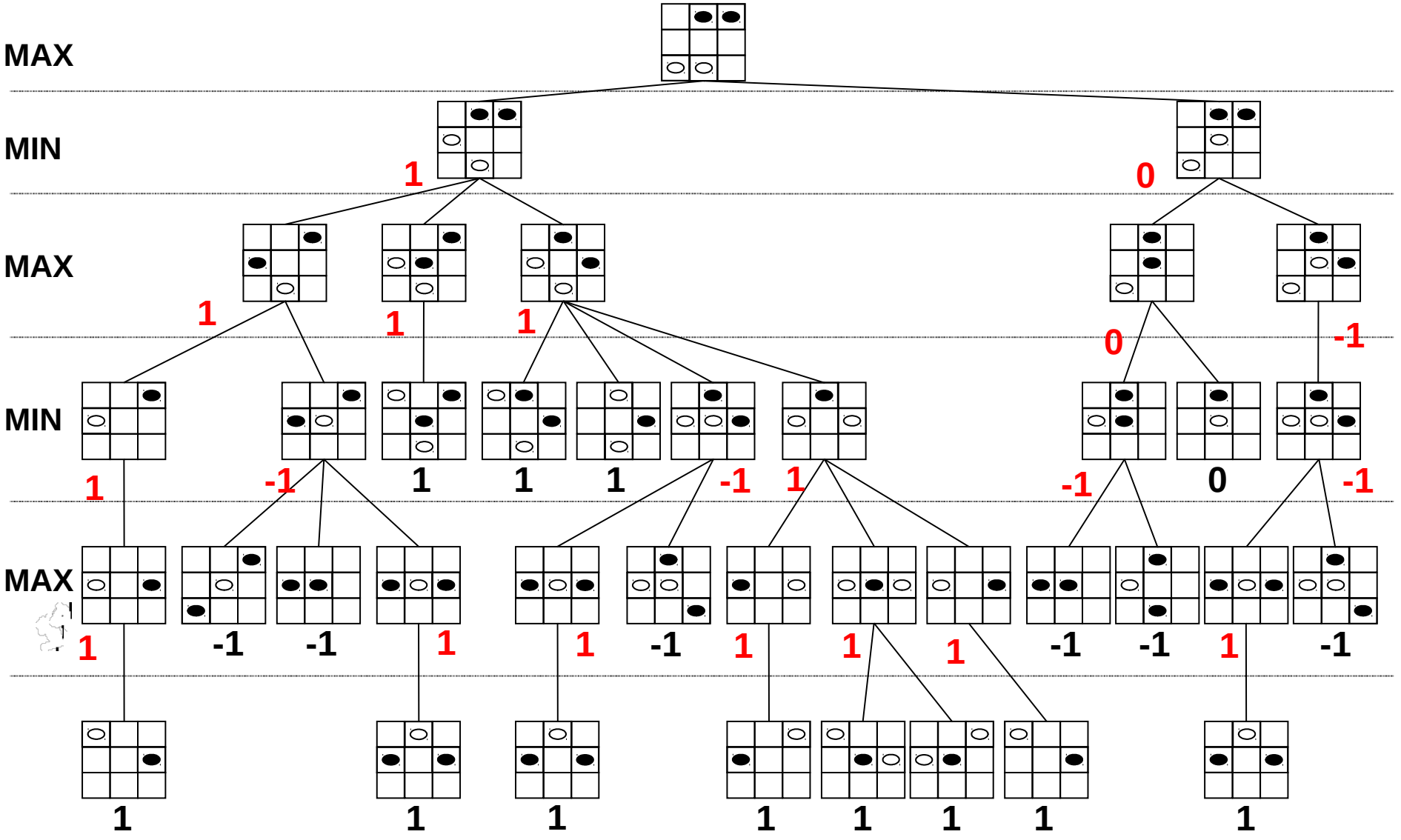


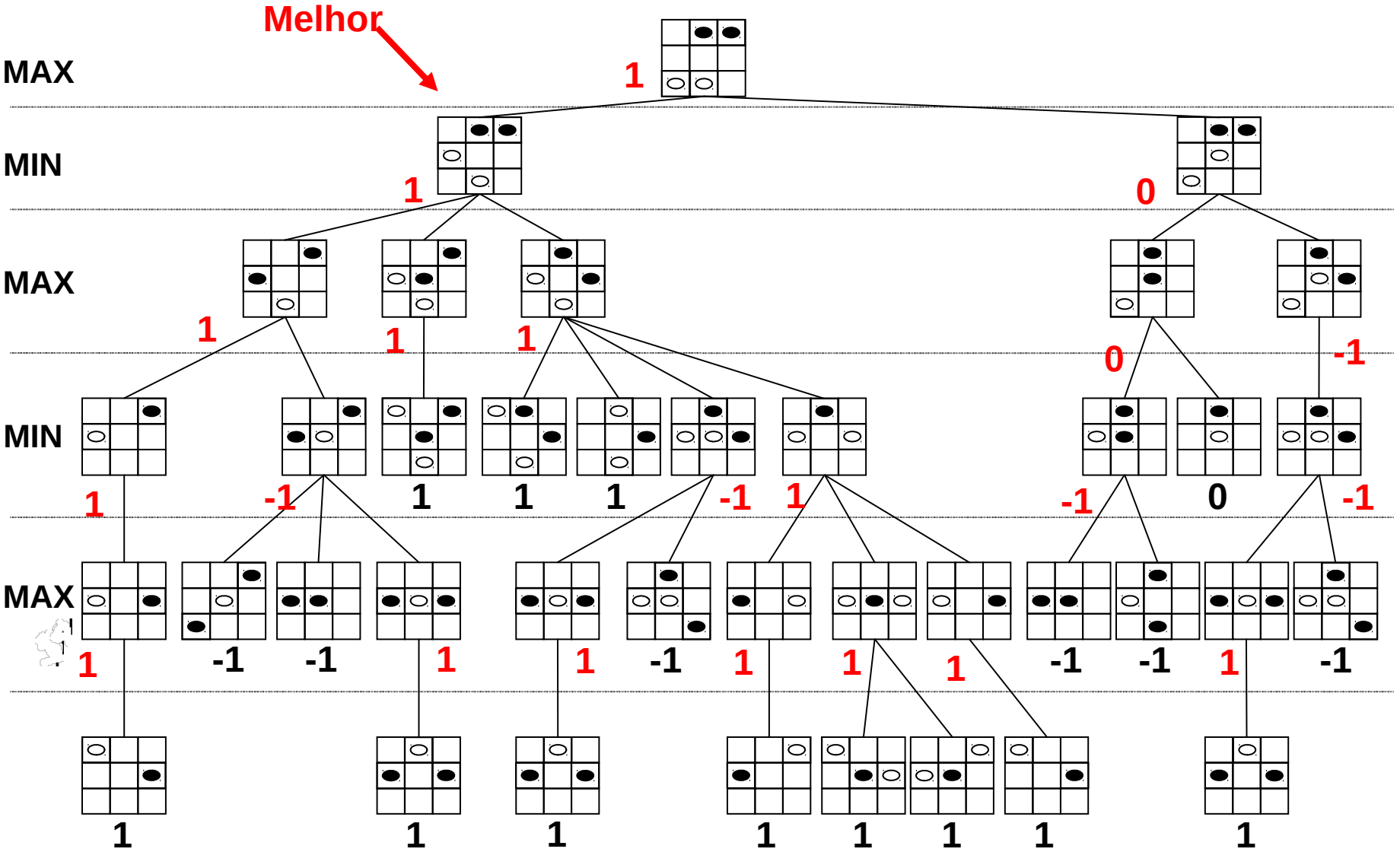

MAX
MIN
MAX
MIN
MAX
1
1
1
0
-1
-1
-1
-1
-1
-1
1
1
1
1
1
1
1


MAX
MIN
MAX
MIN
MAX
1
1
1
0
1
-1
-1
1
1
-1
1
1
1
-1
-1
1
-1
1
1
1
1
1
1
1
1


MAX
MIN
MAX
MIN
MAX
1
-1
1
1
1
-1
1
-1
0
-1
1
-1
-1
1
1
-1
1
1
1
1
-1
-1
1
-1
1
1
1
1
1
1
1
1
1
1
1


MAX
MIN
MAX
MIN
MAX
1
1
1
0
-1
1
-1
1
1
1
-1
1
-1
0
-1
1
-1
-1
1
1
-1
1
1
1
1
-1
-1
1
-1
1
1
1
1
1
1
1
1
1
1
1


MAX
MIN
MAX
MIN
MAX
1
0
1
1
1
0
-1
1
-1
1
1
1
-1
1
-1
0
-1
1
-1
-1
1
1
-1
1
1
1
-1
-1
1
-1
1
1
1
1
1
1
1


Melhor
MAX
MIN
MAX
MIN
MAX