Procura com adversários (jogos)

Ano Lectivo 2013/2014 - 1^o Semestre – Adaptado de http://aima.eecs.berkeley.edu/slides-tex/

Resumo

- ♢ Estratégias óptimas
- ♢ Recursos Limitados
- ♢ Corte α–β
- ♢ Jogos com factor sorte
- ♢ Jogos com informação imperfeita

Jogos vs. problemas de procura

Adversário "imprevisível" $\Rightarrow$ solução é uma estratégia
especificando uma jogada para cada resposta possível do oponente

Limites temporais $\Rightarrow$ pouco provável encontrar estratégia óptima, tem de se aproximar

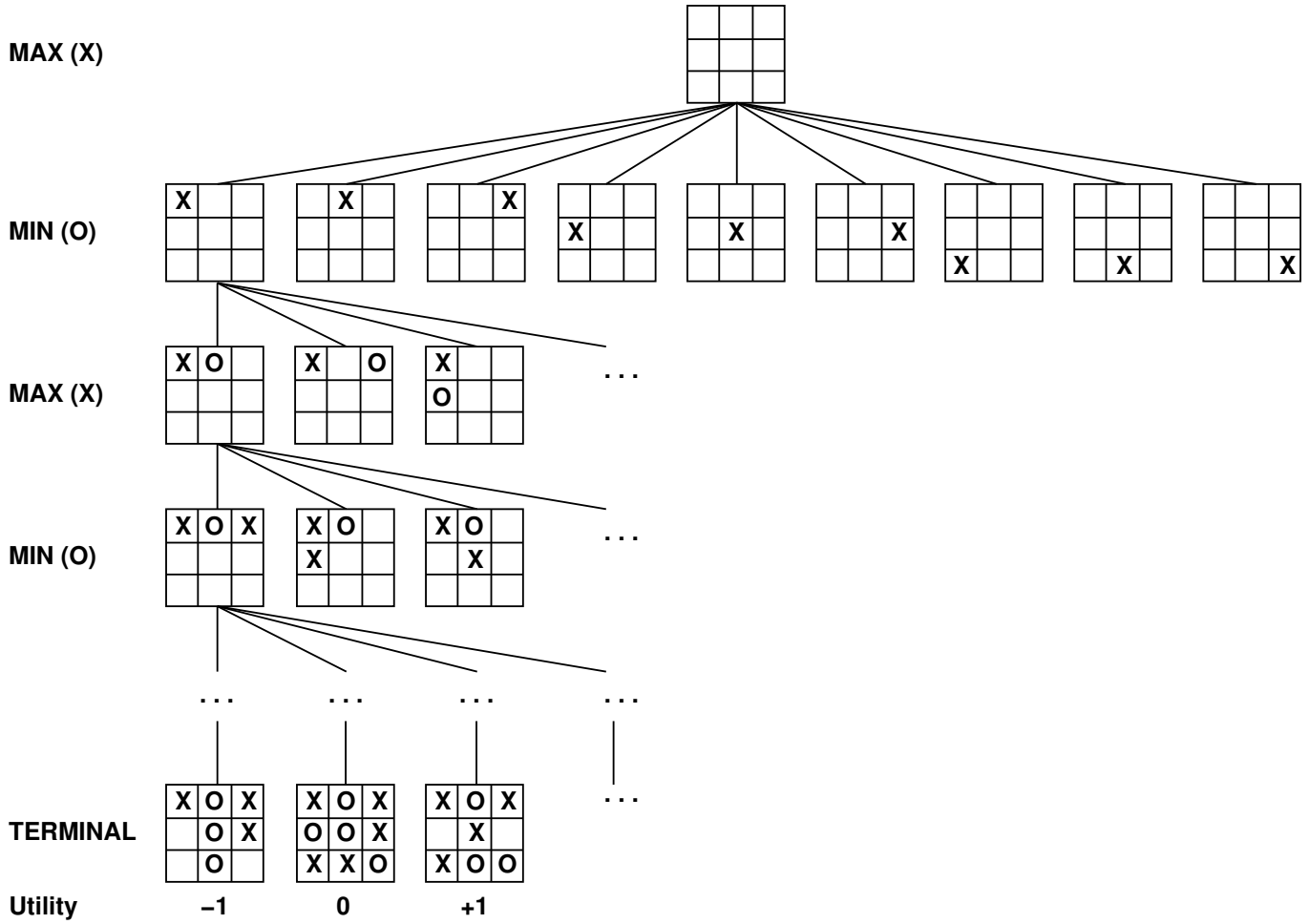
Plano de ataque:

- Computador considera as possíveis jogadas (Babbage, 1846)
- Algoritmo para jogador perfeito (Zermelo, 1912; Von Neumann, 1944)
- Horizonte finito, avaliação aproximada (Zuse, 1945; Wiener, 1948; Shannon, 1950)
- Primeiro jogador de Xadrez (Turing, 1951)
- Aprendizagem automática para melhor precisão da avaliação (Samuel, 1952–57)
- Cortes permitem procura mais profunda (McCarthy, 1956)

Tipos de jogos

	deterministic	chance
perfect information	chess, checkers, go, othello	backgammon monopoly
imperfect information	battleships, blind tictactoe	bridge, poker, scrabble nuclear war

Árvore de Jogo (2 jog., determ., alternados)

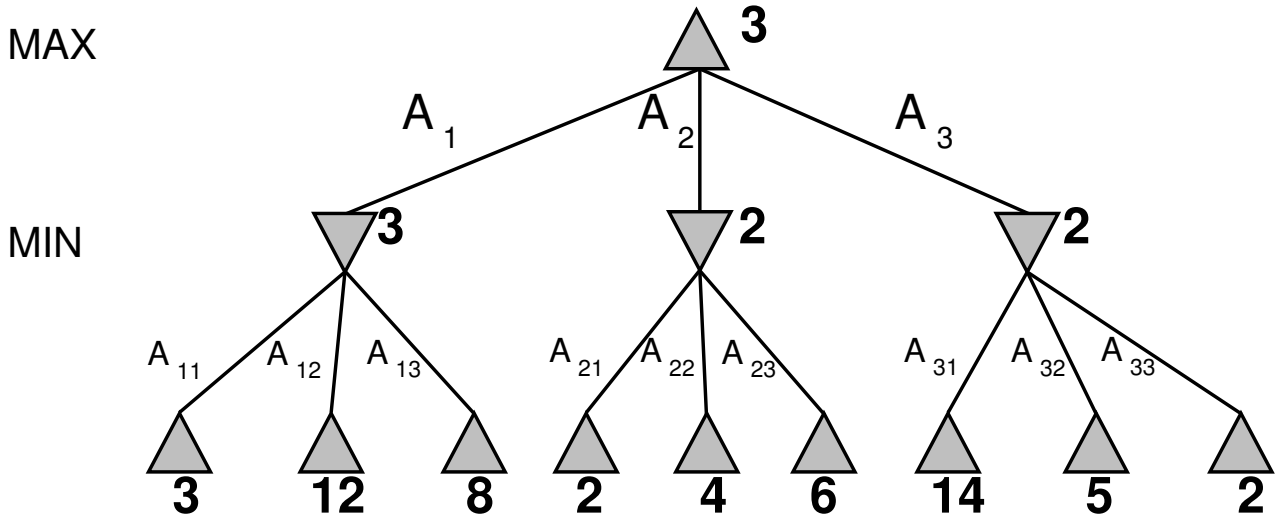


Minimax

Estratégia perfeita para jogos deterministas e com informação perfeita

Ideia: escolher a jogada para posição com maior *valor minimax*
= melhor recompensa alcançável com estratégia óptima do adversário

E.g., jogo com 2 jogadas:



Algoritmo Minimax

function MINIMAX-DECISION(*state*) **returns** *an action*
  inputs: *state*, current state in game

  $v \leftarrow$ MAX-VALUE(*state*)
  return the *action* in SUCCESSORS(*state*) with value v

---

function MAX-VALUE(*state*) **returns** *a utility value*
  if TERMINAL-TEST(*state*) **then return** UTILITY(*state*)
  $v \leftarrow -\infty$
  for *a, s* in SUCCESSORS(*state*) **do** $v \leftarrow$ MAX(v, MIN-VALUE(s))
  return v

---

function MIN-VALUE(*state*) **returns** *a utility value*
  if TERMINAL-TEST(*state*) **then return** UTILITY(*state*)
  $v \leftarrow \infty$
  for *a, s* in SUCCESSORS(*state*) **do** $v \leftarrow$ MIN(v, MAX-VALUE(s))
  return v

Propriedades do minimax

Completo??

Propriedades do minimax

Completo?? Apenas se a árvore é finita (o Xadrez tem regras específicas para o efeito). **NB** uma estratégia finita pode existir mesmo em árvores infinitas!

Óptima??

Propriedades do minimax

Completo?? Apenas se a árvore é finita (o Xadrez tem regras específicas para o efeito). **NB** uma estratégia finita pode existir mesmo em árvores infinitas!

Óptima?? Sim, contra um adversário perfeito. Caso contrário??

Complexidade Temporal??

Propriedades do minimax

Completo?? Apenas se a árvore é finita (o Xadrez tem regras específicas para o efeito). **NB** uma estratégia finita pode existir mesmo em árvores infinitas!

Óptima?? Sim, contra um adversário perfeito. Caso contrário??

Complexidade Temporal?? $O(b^m)$

Complexidade Espacial??

Propriedades do minimax

Completo?? Apenas se a árvore é finita (o Xadrez tem regras específicas para o efeito). **NB** uma estratégia finita pode existir mesmo em árvores infinitas!

Óptima?? Sim, contra um adversário perfeito. Caso contrário??

Complexidade Temporal?? $O(b^m)$

Complexidade Espacial?? $O(bm)$ (exploração pelo melhor primeiro)

Para o Xadrez, $b \approx 35$, $m \approx 100$ para jogos "razoáveis"

$\Rightarrow$ solução exacta completamente impossível

Recursos Limitados

Suponha-se que temos 100 segundos, explorando 10^4 nós/segundo

$\Rightarrow 10^6$ nós por jogada

Aproximação Standard:

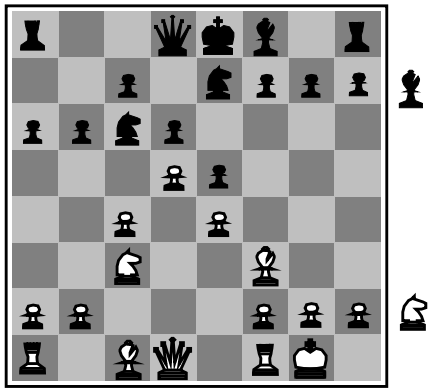
- *teste de corte*

  e.g., limitar profundidade (possivelmente com *procura quiescente*)

- *função de avaliação*

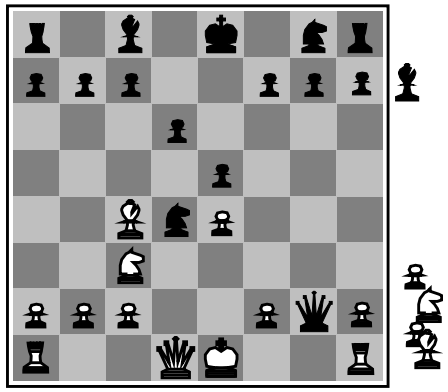
  = estimativa da razoabilidade da posição

Funções de avaliação



Black to move

White slightly better



White to move

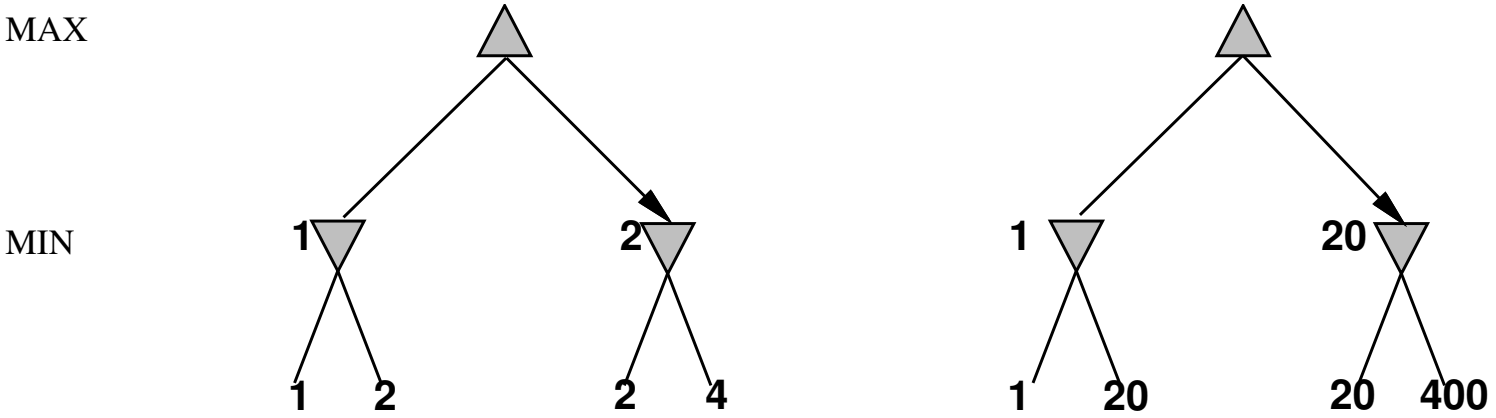
Black winning

Para o Xadrez, habitualmente soma pesada *linear* de características

$$Eval(s) = w_1 f_1(s) + w_2 f_2(s) + \ldots + w_n f_n(s)$$

e.g., $w_1 = 9$ com
$f_1(s) =$ (núm. de rainhas brancas) – (núm. de rainhas pretas), etc.

Digressão: Valores exactos não são importantes



O comportamento é preservado com qualquer transformação *monótona* de EVAL

Apenas a ordem interessa:

recompensa em jogos deterministas comporta-se como uma função de *utilidade ordinal*

Cortar a procura

MinimaxCutoff é idêntico a MinimaxValue excepto

1. Terminal? é substituído por Cutoff?
2. Utilidade é substituído por Eval

Funciona na prática?

$$b^m = 10^6, \quad b = 35 \quad \Rightarrow \quad m = 4$$

4-jogadas de previsão é um jogador de Xadrez iniciante!

4-jogadas $\approx$ jogador iniciante
8-jogadas $\approx$ PC típico, mestre
12-jogadas $\approx$ Deep Blue, Kasparov

Como cortar um nó?

♢ Limitar a profundidade a um número fixo (naïf...)

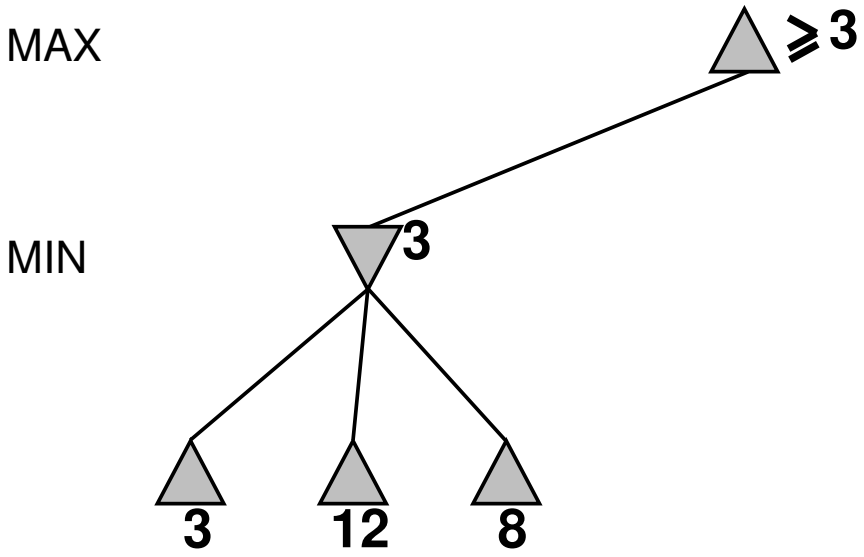
♢ Efectuar aprofundamento progressivo enquanto houver tempo.

♢ Efectuar procura quiescente

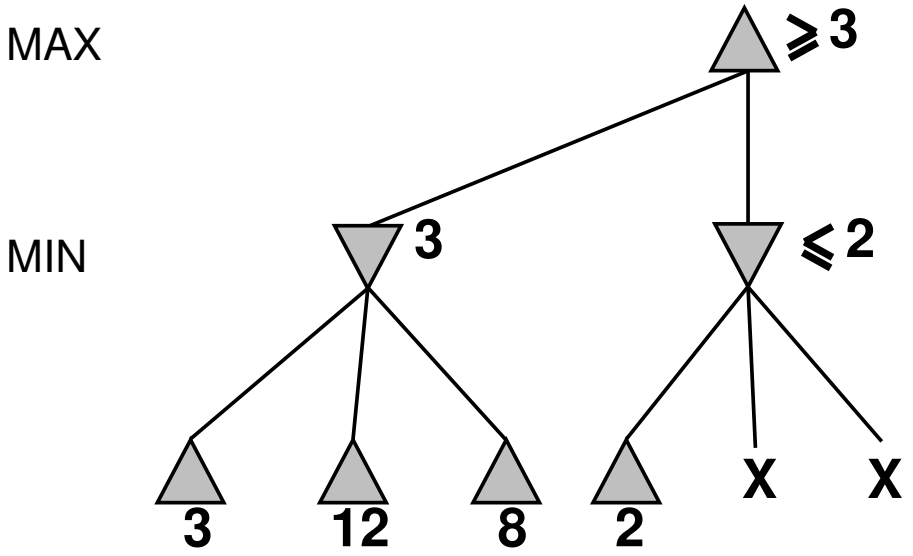
Só cortar nós em que o valor da função de avaliação estabilizou (sem grandes alterações previsíveis nas próximas jogadas)

Mias difícil de resolver é o problema da morte adiada (ou efeito horizonte). Uma jogada terrível pode ter os seus efeitos protelados pelo o outro jogador, de maneira a ficarem invisíveis.

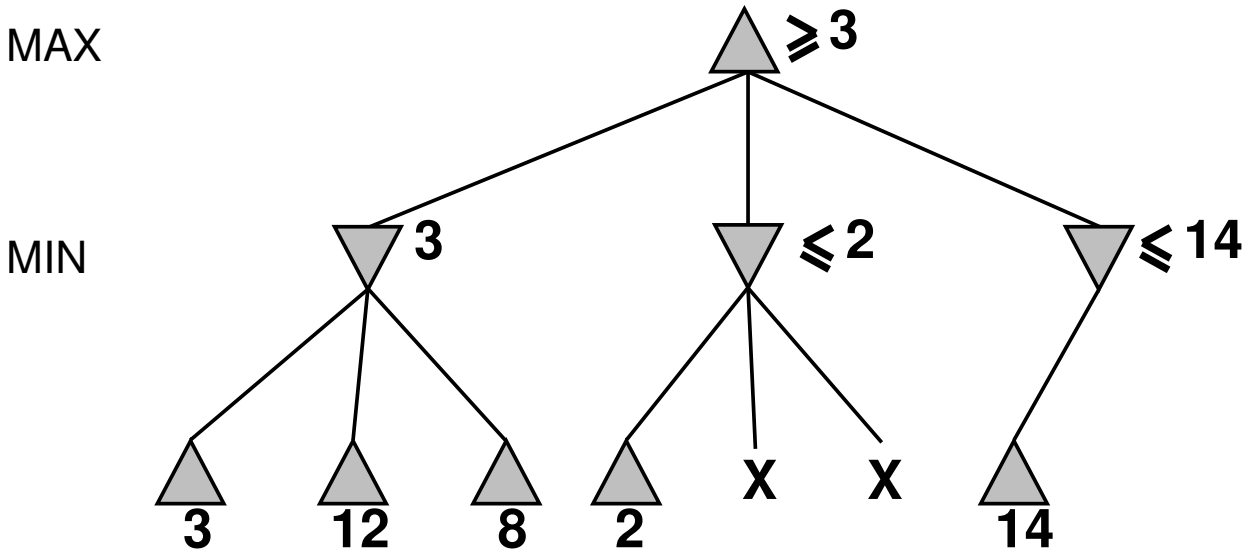
Exemplo de corte α–β



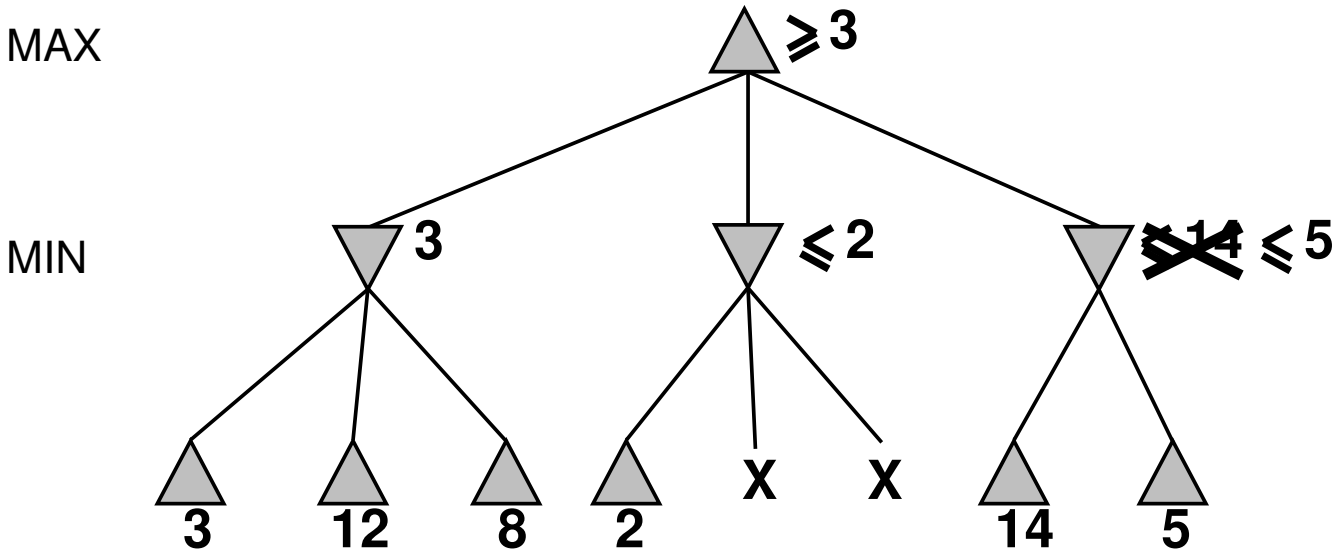
Exemplo de corte α–β



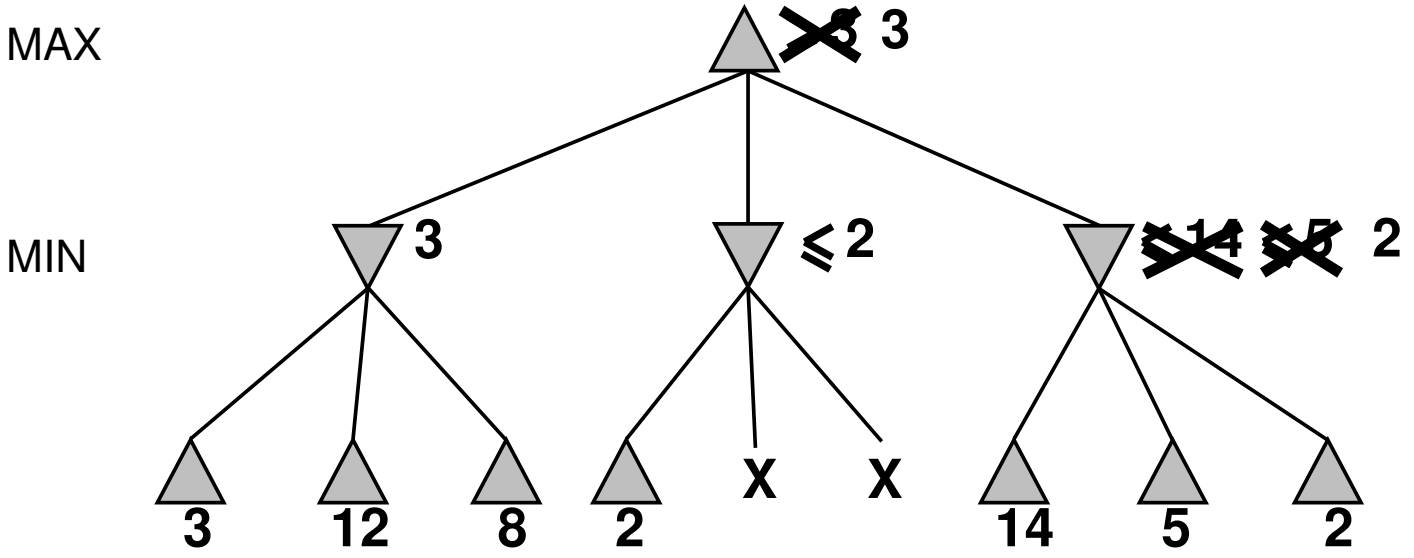
Exemplo de corte α–β



Exemplo de corte α–β



Exemplo de corte α–β



Propriedades do α–β

Corte *não* afecta o resultado final

Uma boa ordenação das jogadas melhora o efeito do corte

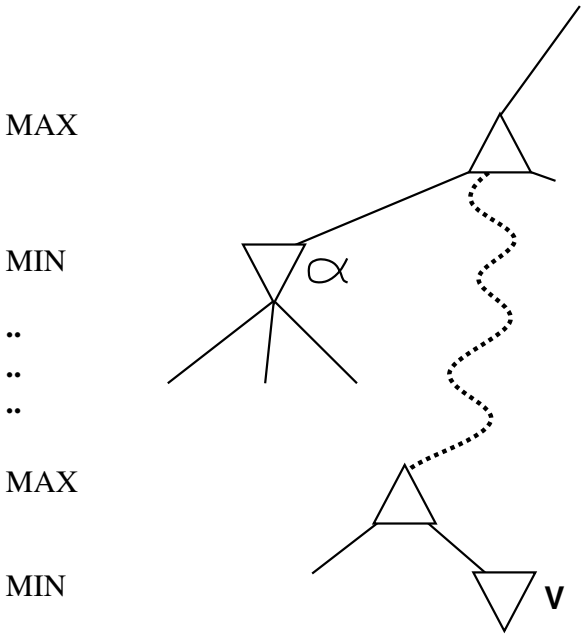
Com "ordenação perfeita," complexidade temporal $= O(b^{m/2})$

$\Rightarrow$ *duplica* profundidade da procura

$\Rightarrow$ pode facilmente atingir profundidade 8 e jogar bem Xadrez

Um exemplo simples do valor do raciocínio sobre quais as computações relevantes (uma forma de *metaraciocínio*)

Caso geral α–β?



α é o melhor valor (para o jogador MAX) encontrado até ao momento

Se v é pior do que α, MAX evita-o $\Rightarrow$ corta esse ramo

Define-se β similarmente para MIN

O algoritmo α–β

function ALPHA-BETA-SEARCH(*state*) **returns** an action

inputs: *state*, current state in game

$v \leftarrow$ MAX-VALUE($state, -\infty, +\infty$)

return the *action* in SUCCESSORS(*state*) with value v

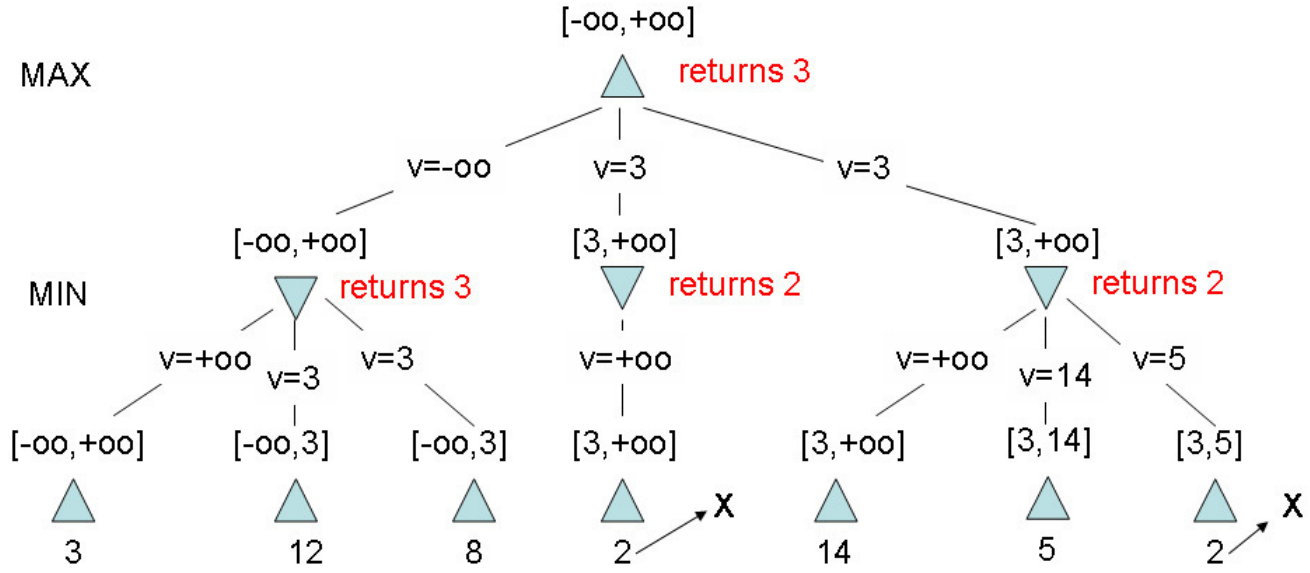
O algoritmo α–β

function MAX-VALUE($state, \alpha, \beta$) **returns** *a utility value*
  if CUTOFF-TEST($state$) **then return** EVAL($state$)
  $v \leftarrow -\infty$
  for each a, s **in** SUCCESSORS($state$) **do**
    $v \leftarrow \mathsf{MAX}(v,$ MIN-VALUE$(s, \alpha, \beta))$
    if $v \geq \beta$ **then return** v
    $\alpha \leftarrow \mathsf{MAX}(\alpha, v)$
  return v

---

function MIN-VALUE($state, \alpha, \beta$) **returns** *a utility value*
  if CUTOFF-TEST($state$) **then return** EVAL($state$)
  $v \leftarrow +\infty$
  for each a, s **in** SUCCESSORS($state$) **do**
    $v \leftarrow \mathsf{MIN}(v,$ MAX-VALUE$(s, \alpha, \beta))$
    if $v \leq \alpha$ **then return** v
    $\beta \leftarrow \mathsf{MIN}(\beta, v)$
  return v

Execução do algoritmo α–β



Legenda:

$[\alpha, \beta]$	Valor dos parâmetros na invocação
$\mathtt{v} = u$	Valor da variável v antes da chamada recursiva
$\mathbf{X}$	Corte (após retorno da chamada recursiva)
`returns` u	valor MINIMAX de saída da chamada

Jogos deterministas na prática

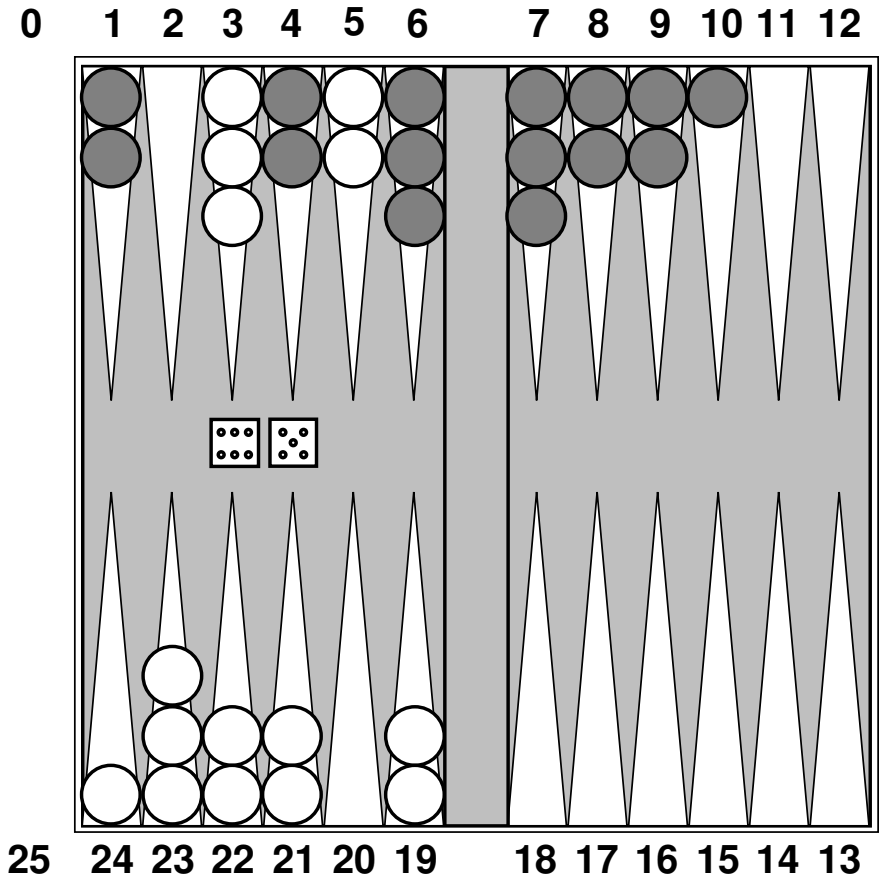
Damas: Chinook terminou com o reinado de 40 anos do campeão mundial Marion Tinsley em 1994. Utilizou uma base de dados de final de jogo definindo a estratégia perfeita para todas as posições com 8 ou menos peças no tabuleiro, num total de 443,748,401,247 posições.

Xadrez: Deep Blue derrotou o campeão mundial humano Gary Kasparov num encontro a 6 partidas em 1997. Deep Blue procura 200 milhões de posições por segundo, utiliza avaliação muito sofisticada, e recorres a métodos para estender algumas linhas de procura até 40 jogadas.

Othello: campeões humanos recusam-se a competir contra computadores, que são demasiado bons.

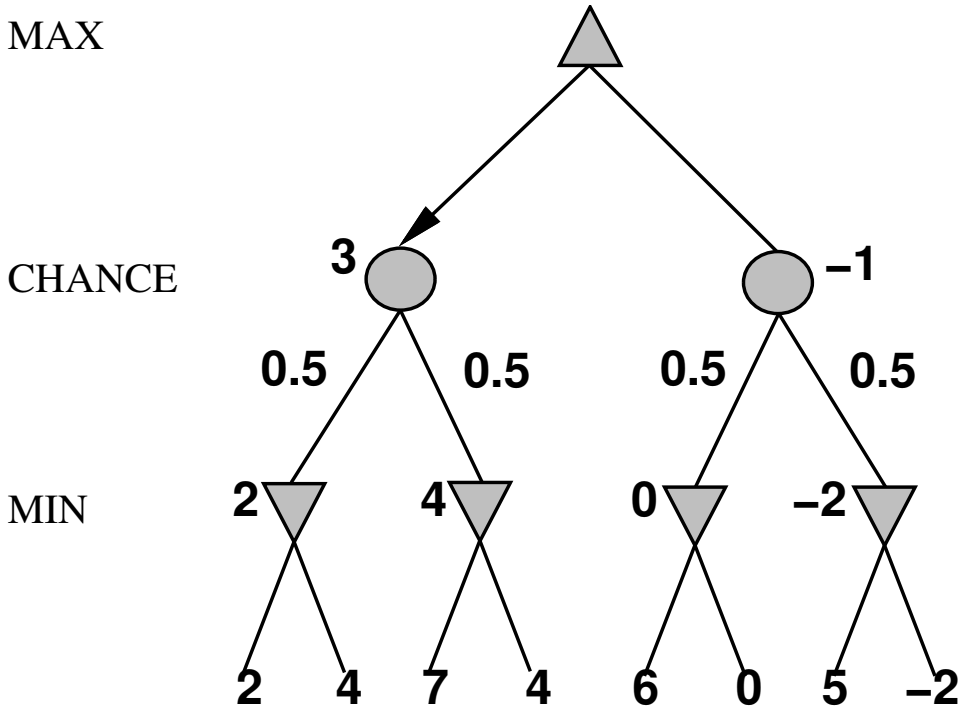
Go: campeões humanos recusam-se a competir contra computadores, que são péssimos jogadores. No go, $b > 300$, portanto a maioria dos programas recorrem a bases de conhecimento de padrões para sugerir jogadas plausíveis.

Jogos não deterministas: Gamão



Jogos não deterministas em geral

Jogos não deterministas, factor sorte introduzido pelos dados, baralhar das cartas.
Exemplo simplificado com moeda-ao-ar:



Algoritmo para jogos não deterministas

EXPECTIMINIMAX fornece estratégia óptima

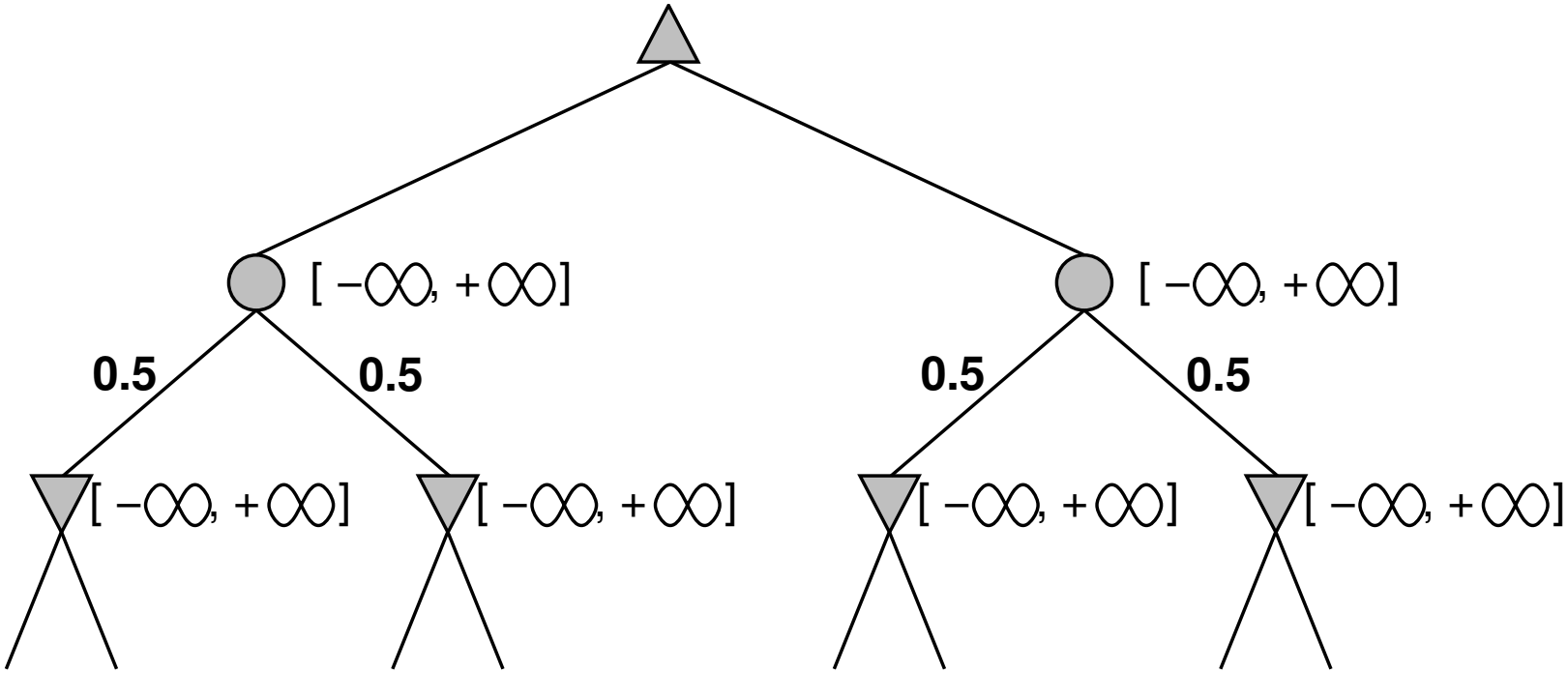
Como o MINIMAX, excepto que se tem de ter em conta nós de sorte:

EXPECTIMINIMAX(n) =

$$\begin{cases} & \text{UTILITY}(n) & \text{se } n \text{ é um estado terminal} \\ \max_{s \in Successors(n)} & \text{EXPECTIMINIMAX}(s) & \text{se } n \text{ é um nó MAX} \\ \min_{s \in Successors(n)} & \text{EXPECTIMINIMAX}(s) & \text{se } n \text{ é um nó MIN} \\ \sum_{s \in Successors(n)} & \text{P(s)} \times \text{EXPECTIMINIMAX}(s) & \text{se } n \text{ é um nó CHANCE} \end{cases}$$

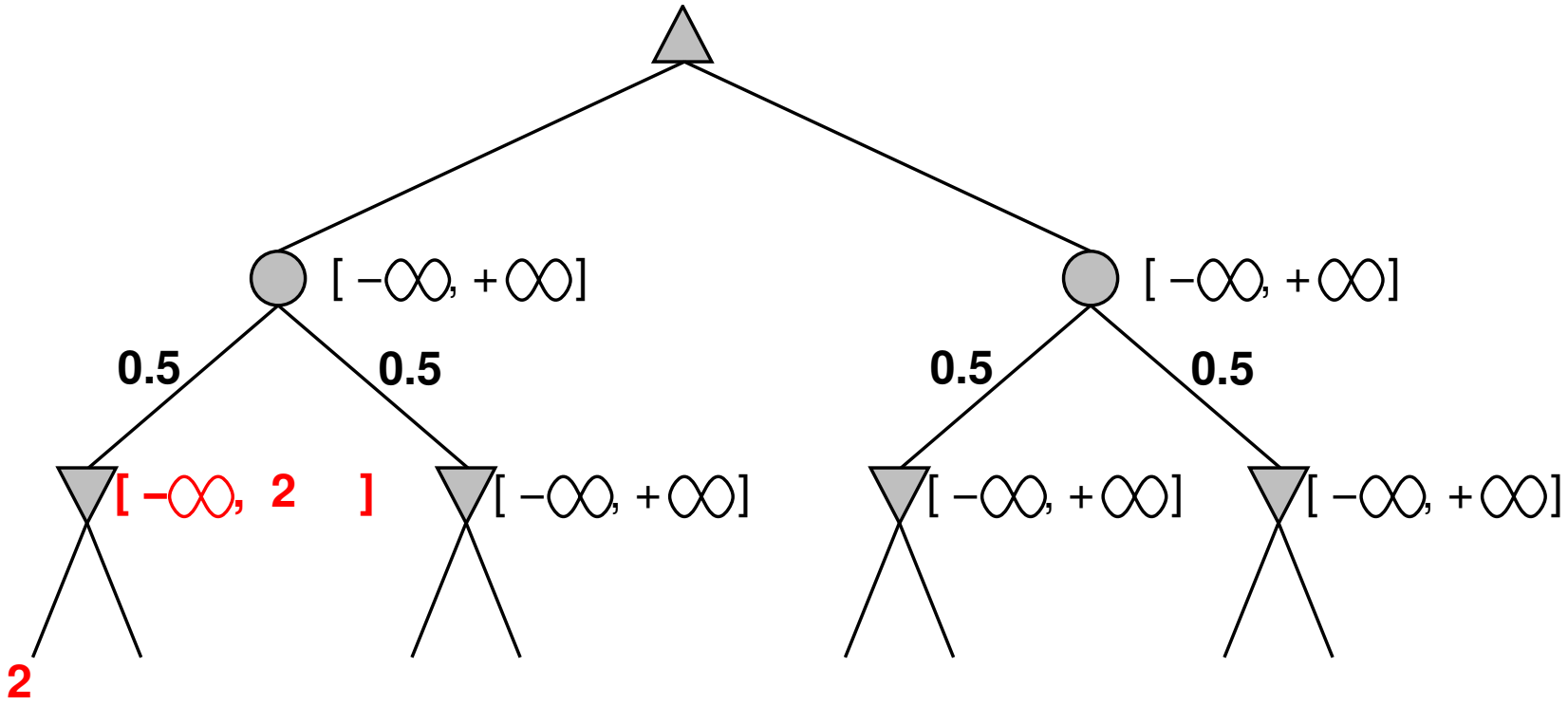
Corte em árvores de jogos não deterministas

Uma adaptação do corte α-β é possível:



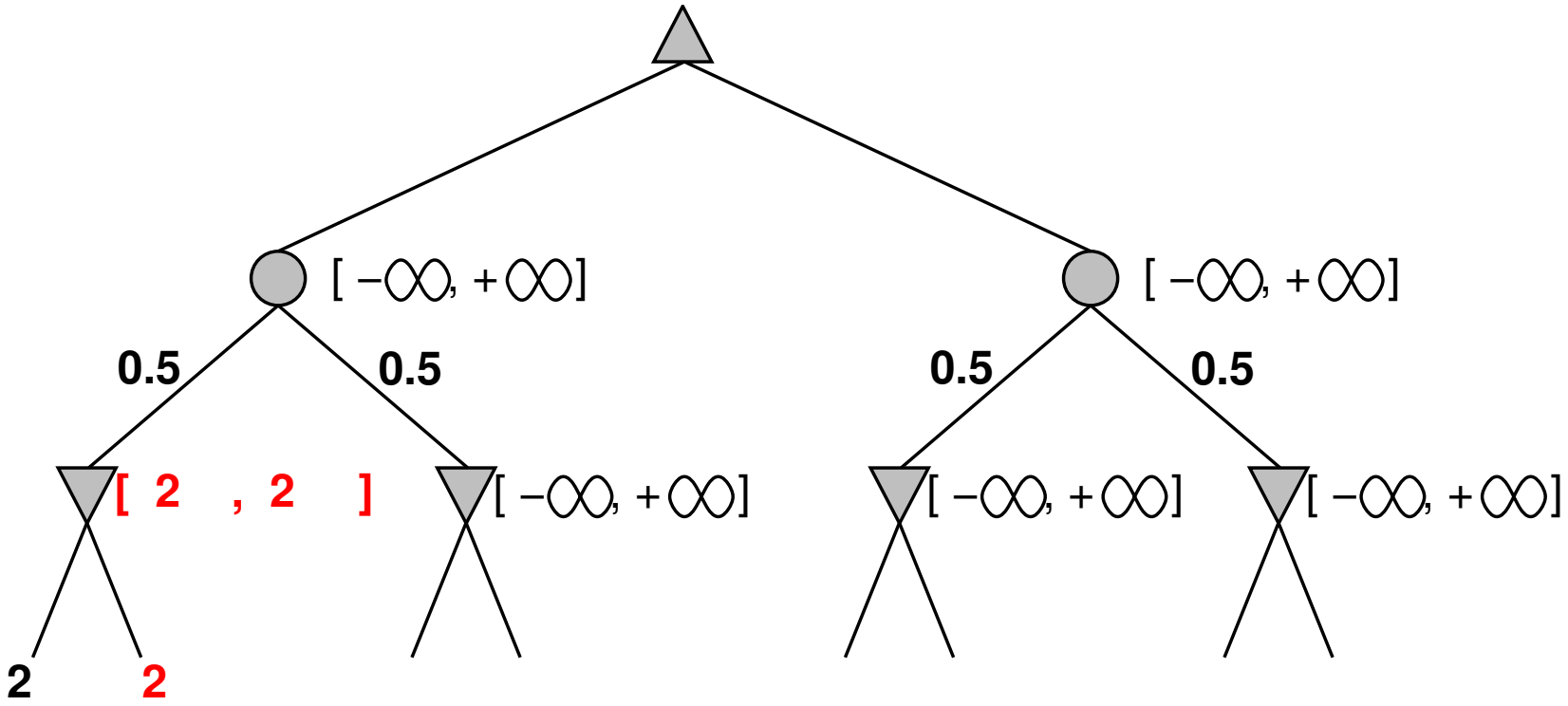
Corte em árvores de jogos não deterministas

Uma adaptação do corte α-β é possível:



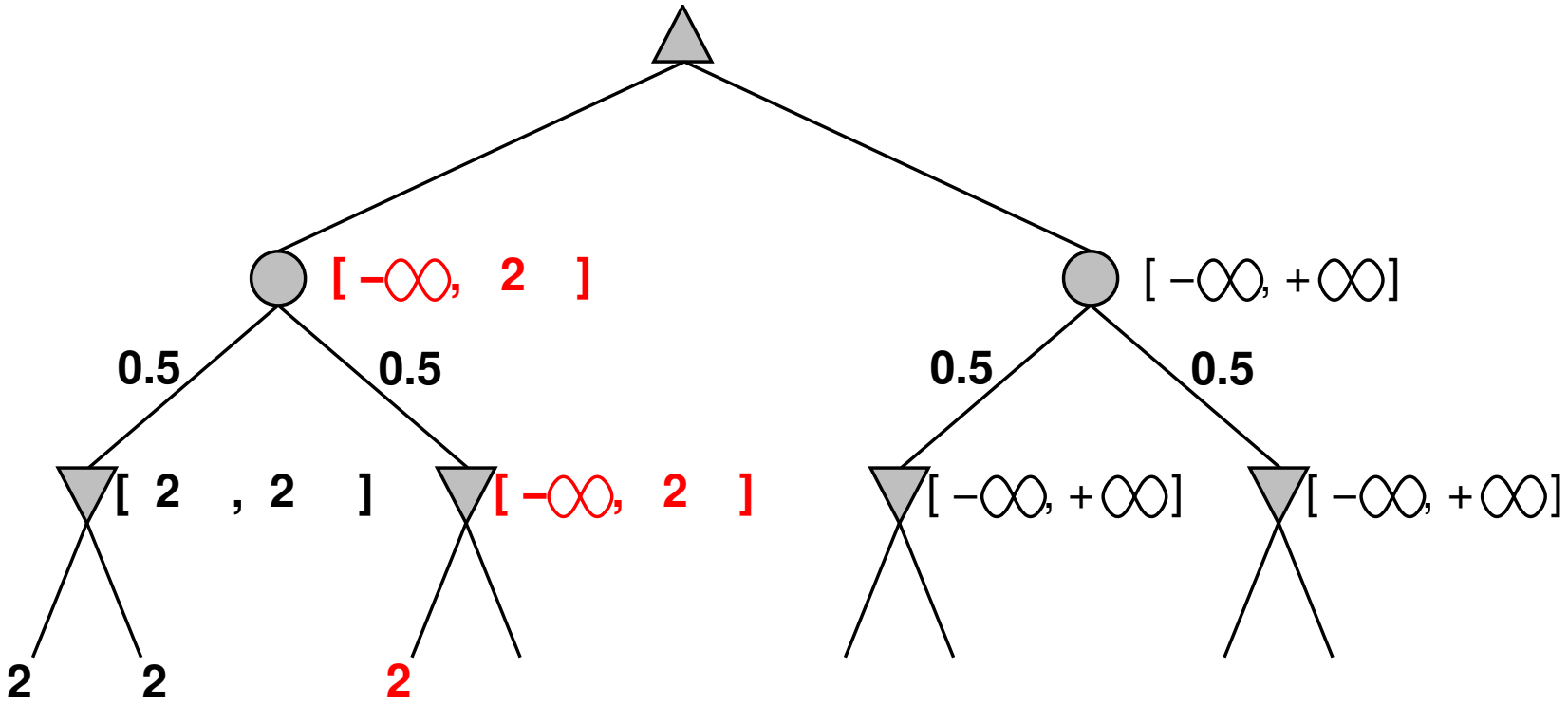
Corte em árvores de jogos não deterministas

Uma adaptação do corte α-β é possível:



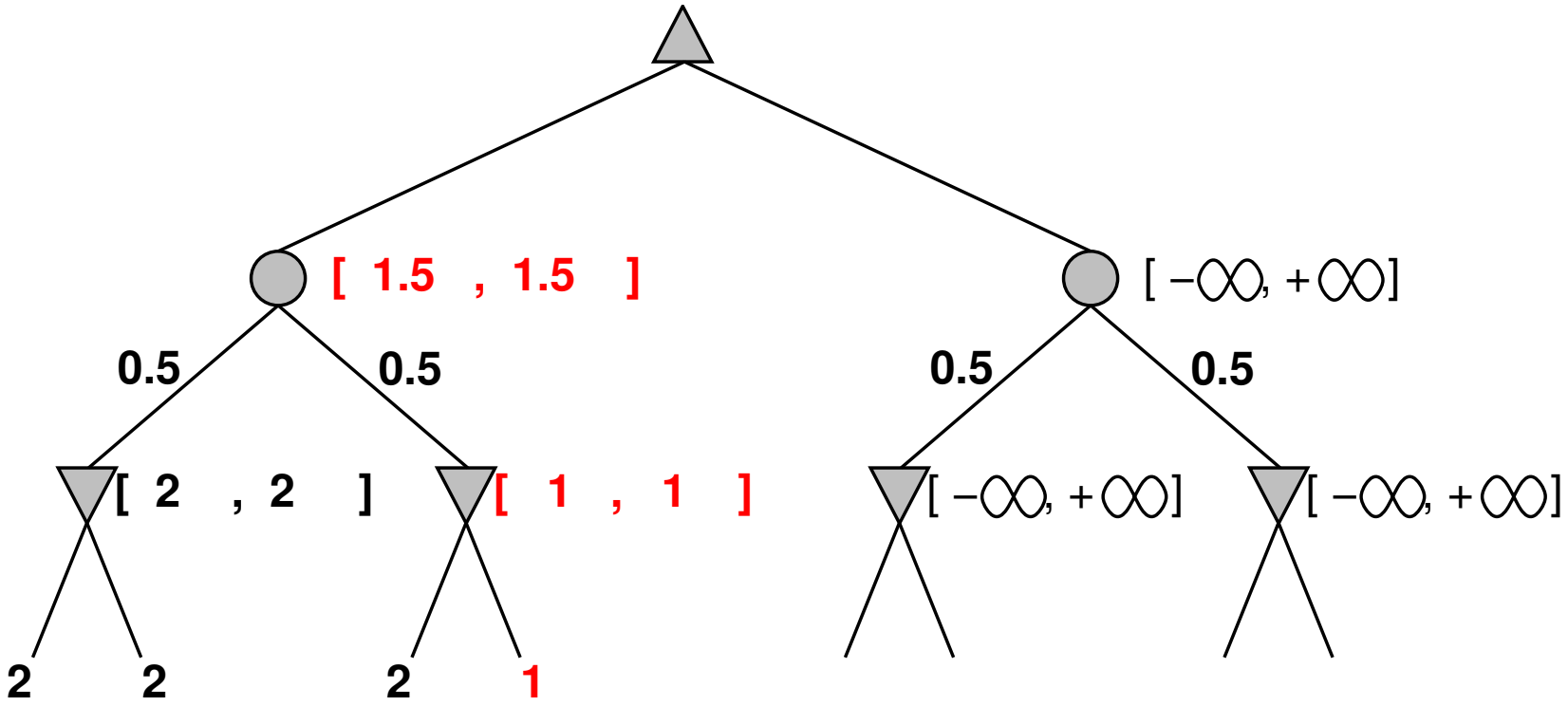
Corte em árvores de jogos não deterministas

Uma adaptação do corte α-β é possível:



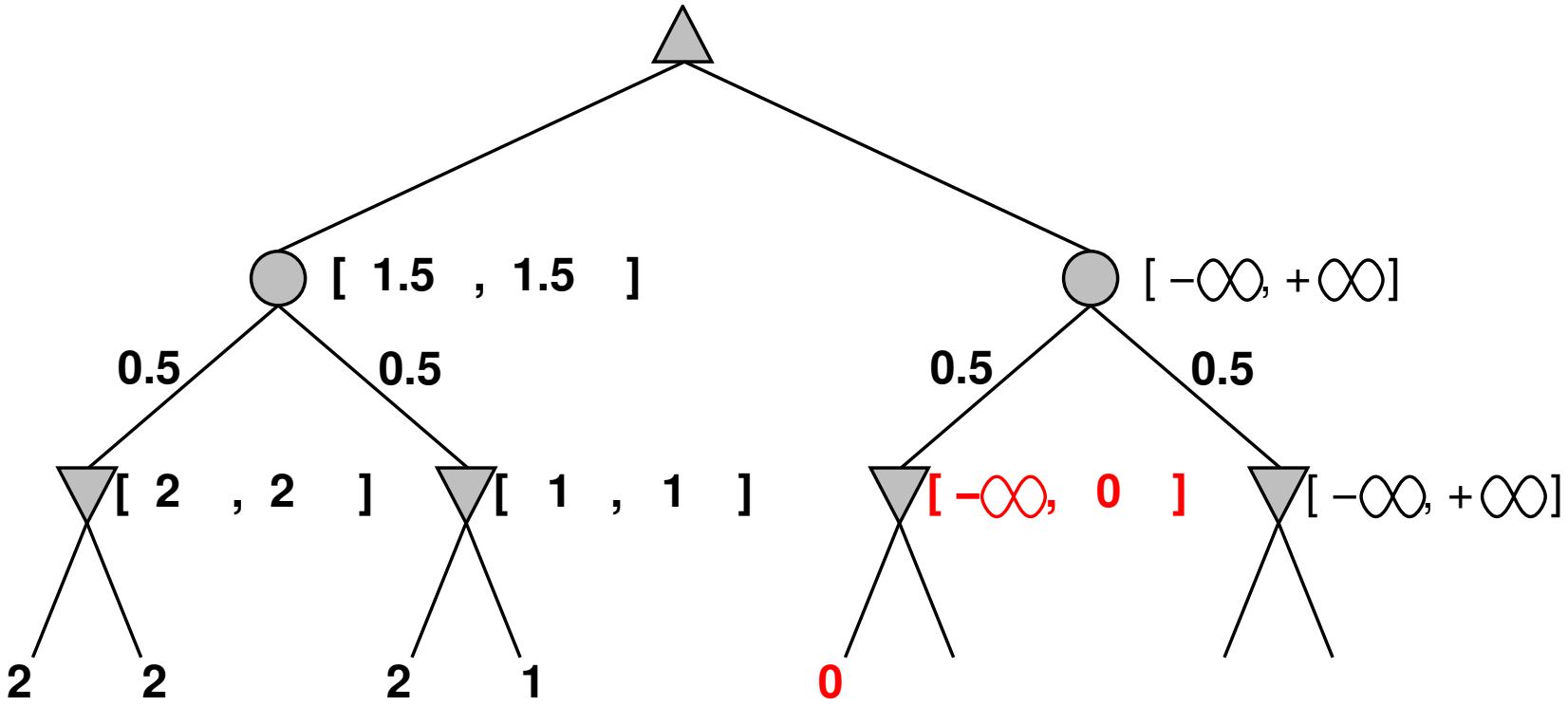
Corte em árvores de jogos não deterministas

Uma adaptação do corte α-β é possível:



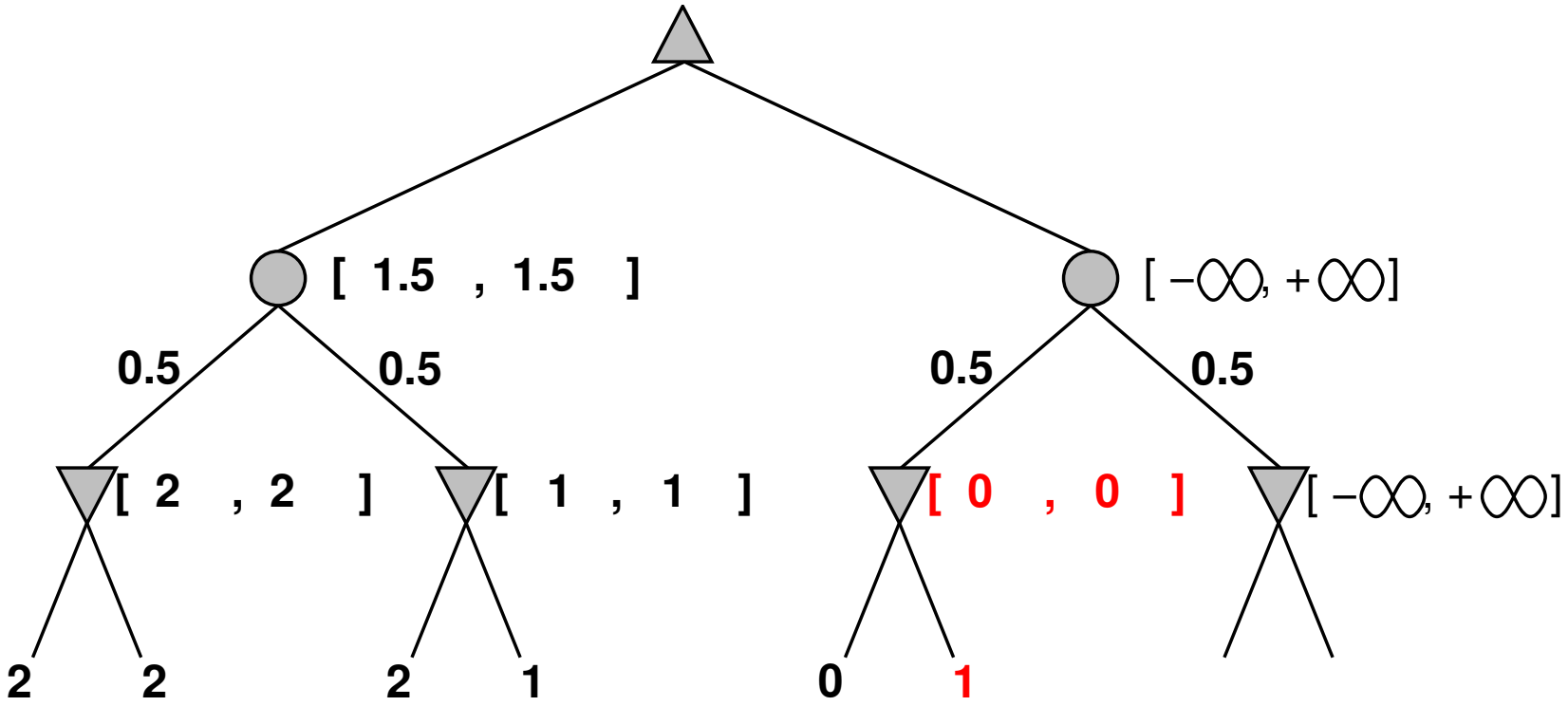
Corte em árvores de jogos não deterministas

Uma adaptação do corte α-β é possível:



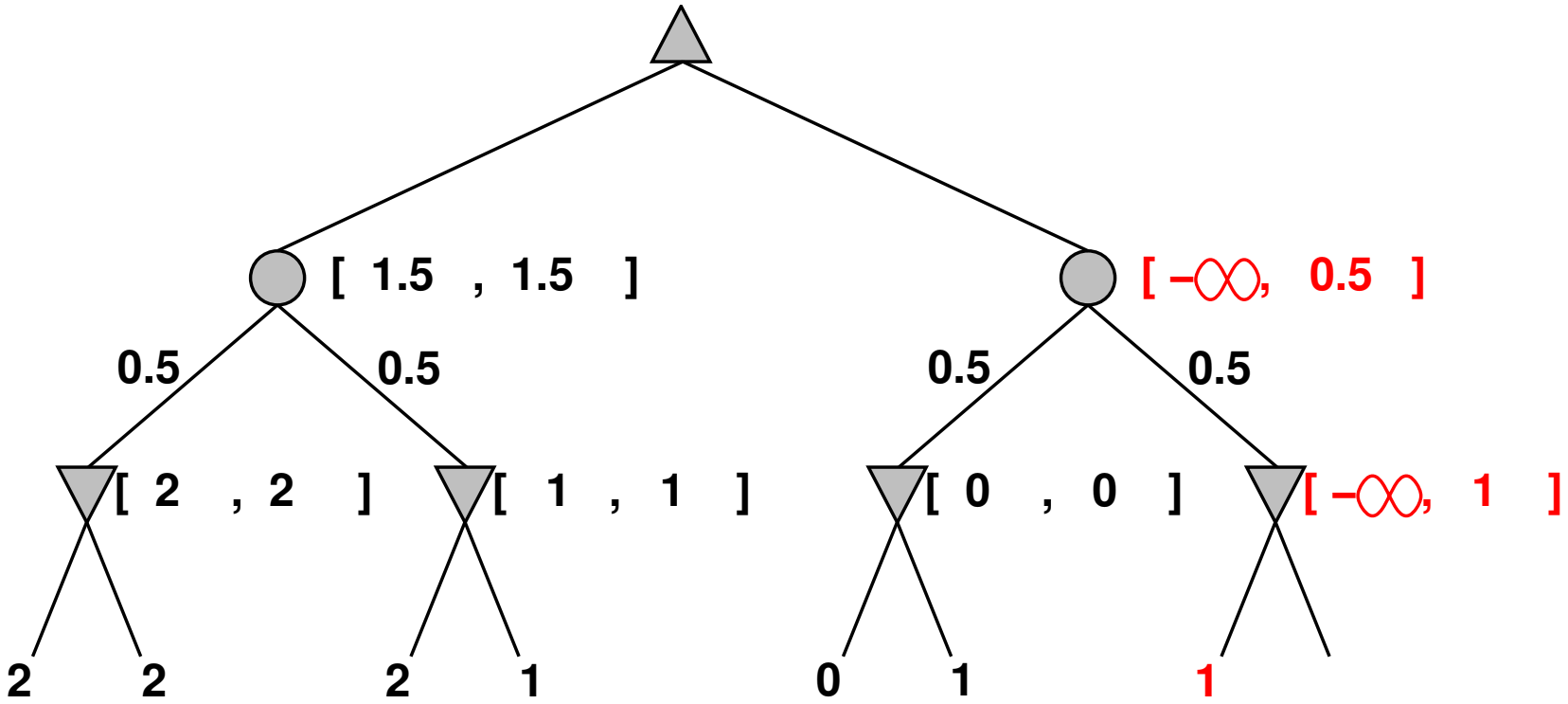
Corte em árvores de jogos não deterministas

Uma adaptação do corte α-β é possível:



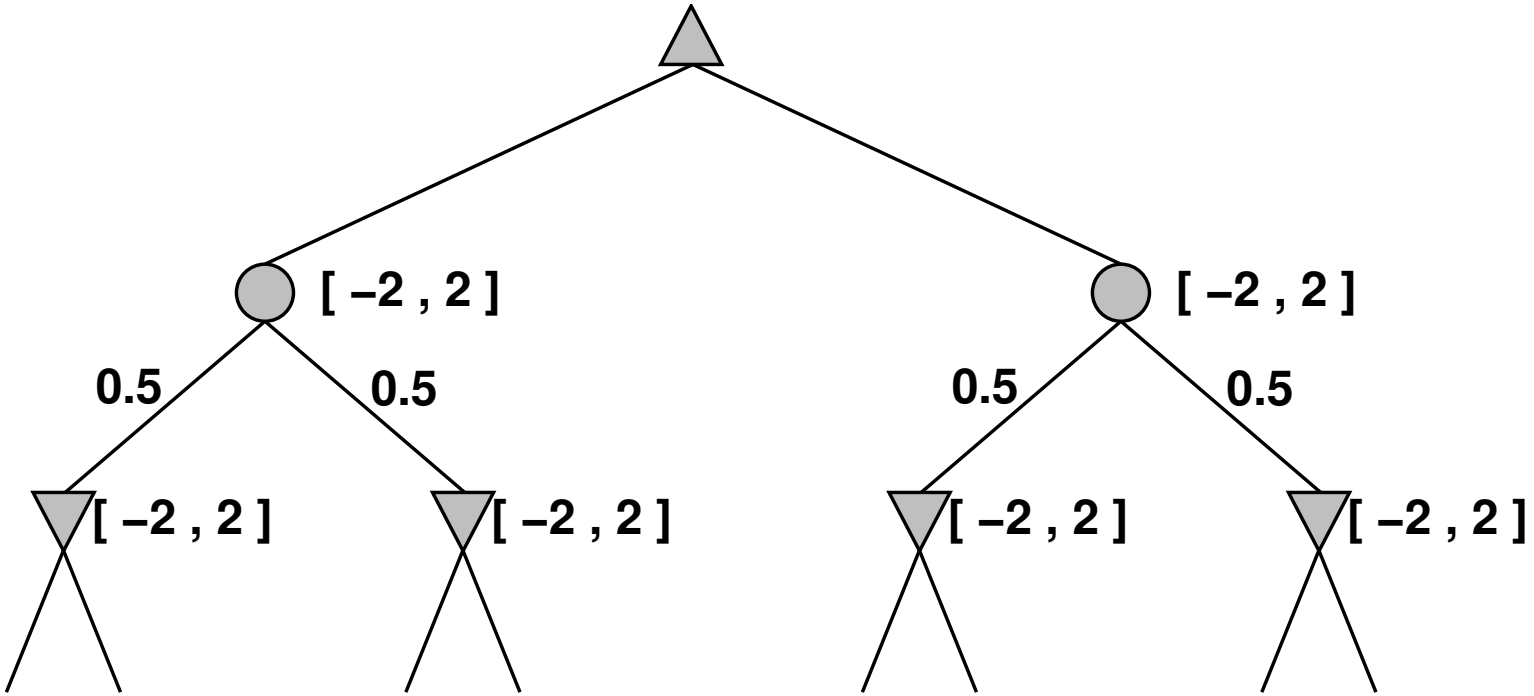
Corte em árvores de jogos não deterministas

Uma adaptação do corte α-β é possível:



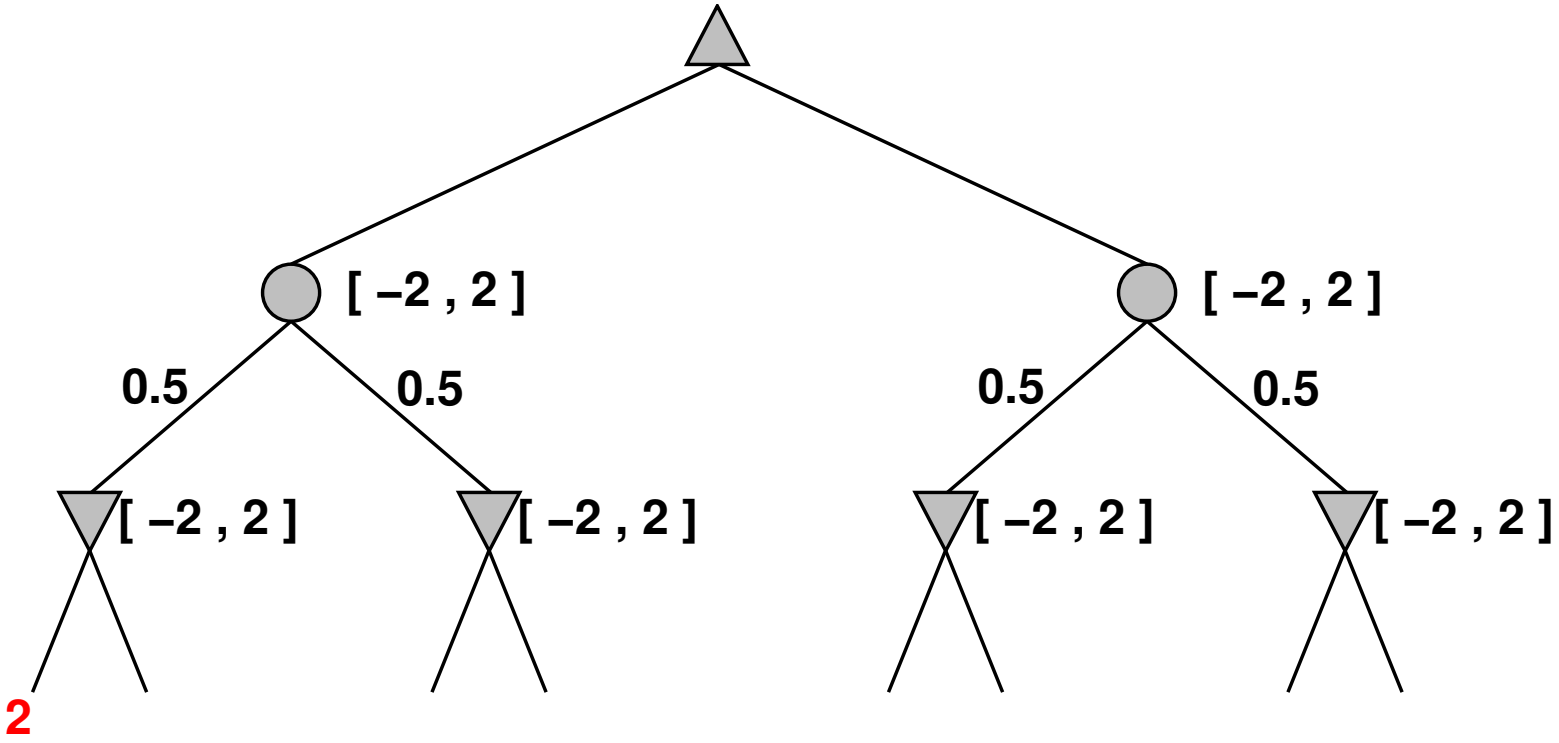
Corte (cont.)

Pode-se cortar mais se os valores das folhas forem limitados



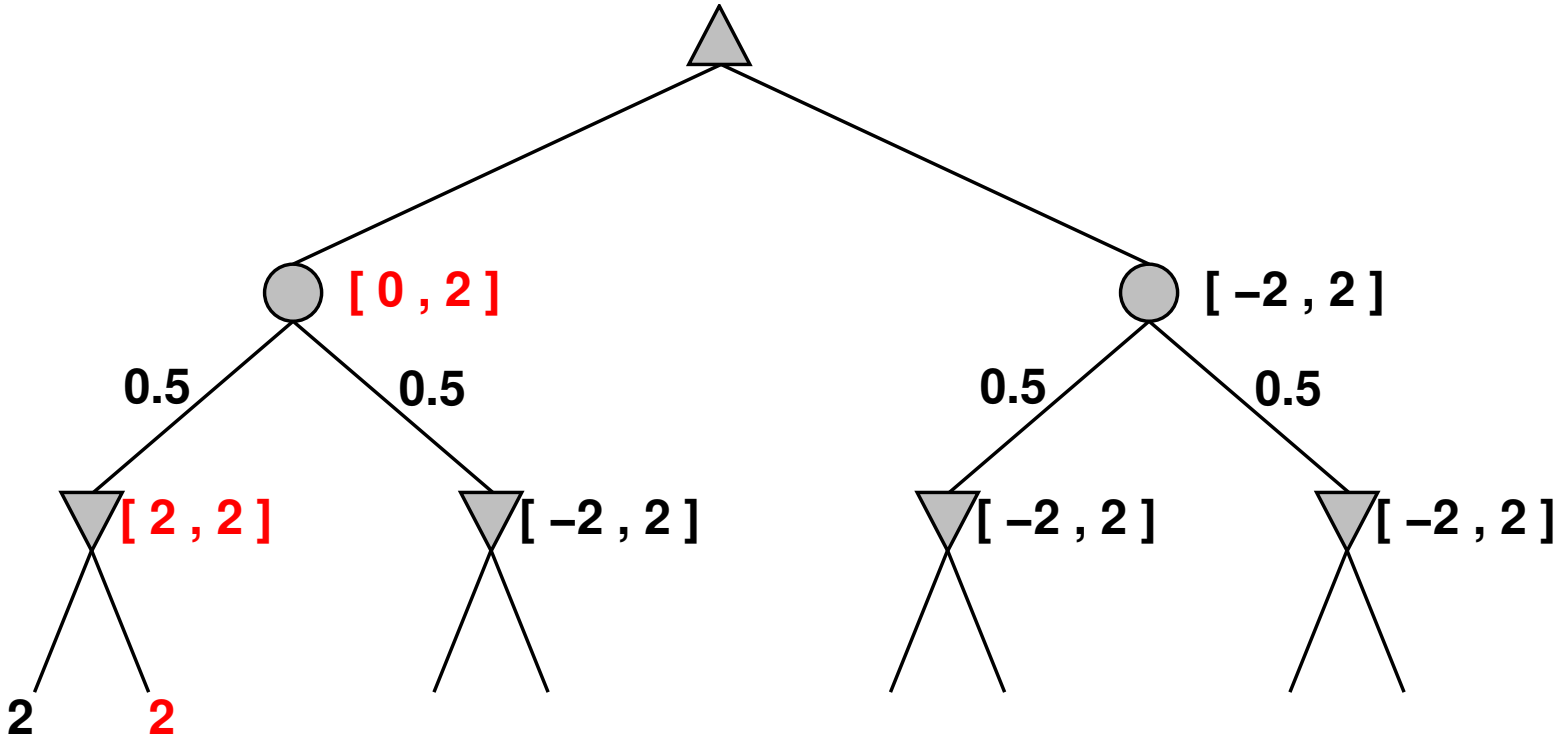
Corte (cont.)

Pode-se cortar mais se os valores das folhas forem limitados



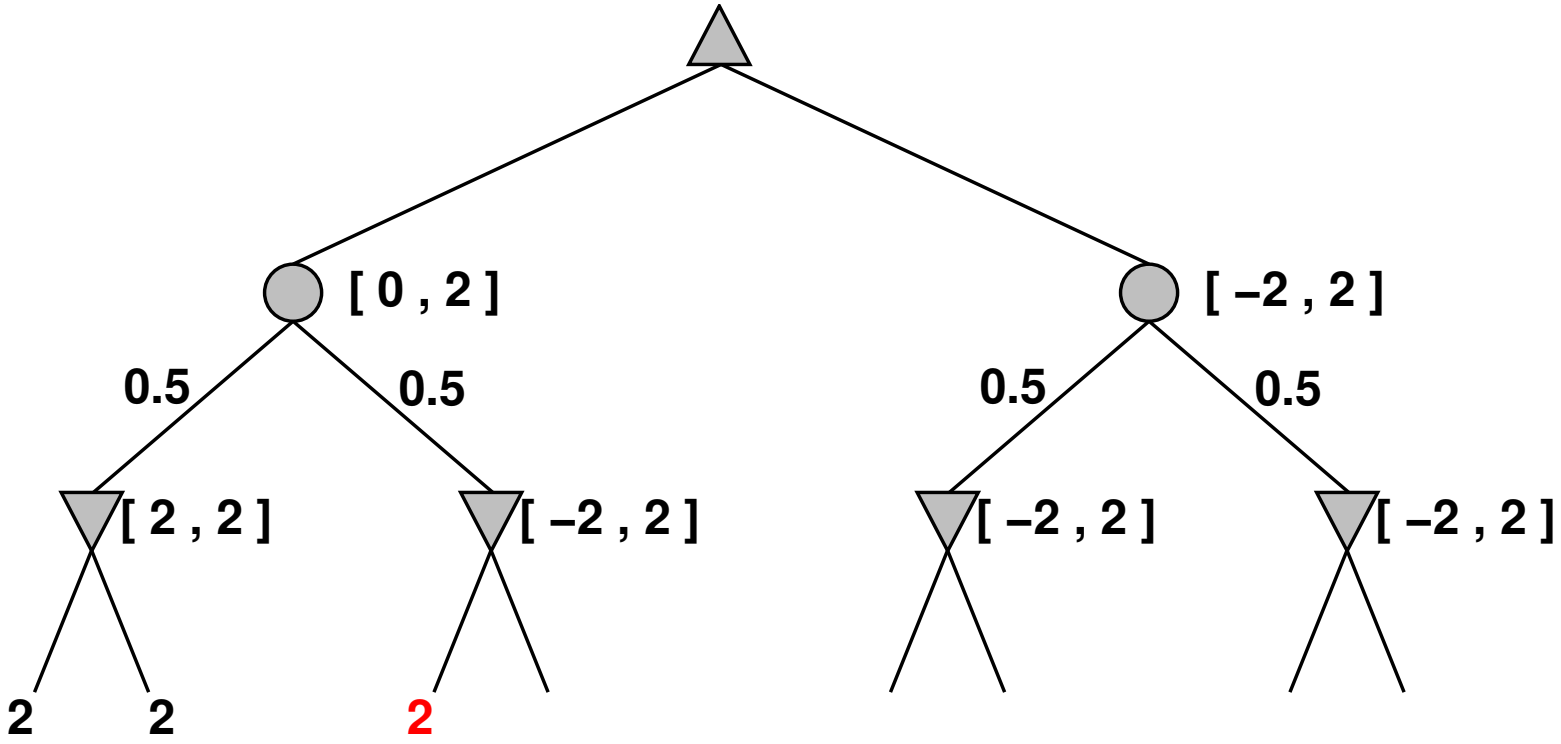
Corte (cont.)

Pode-se cortar mais se os valores das folhas forem limitados



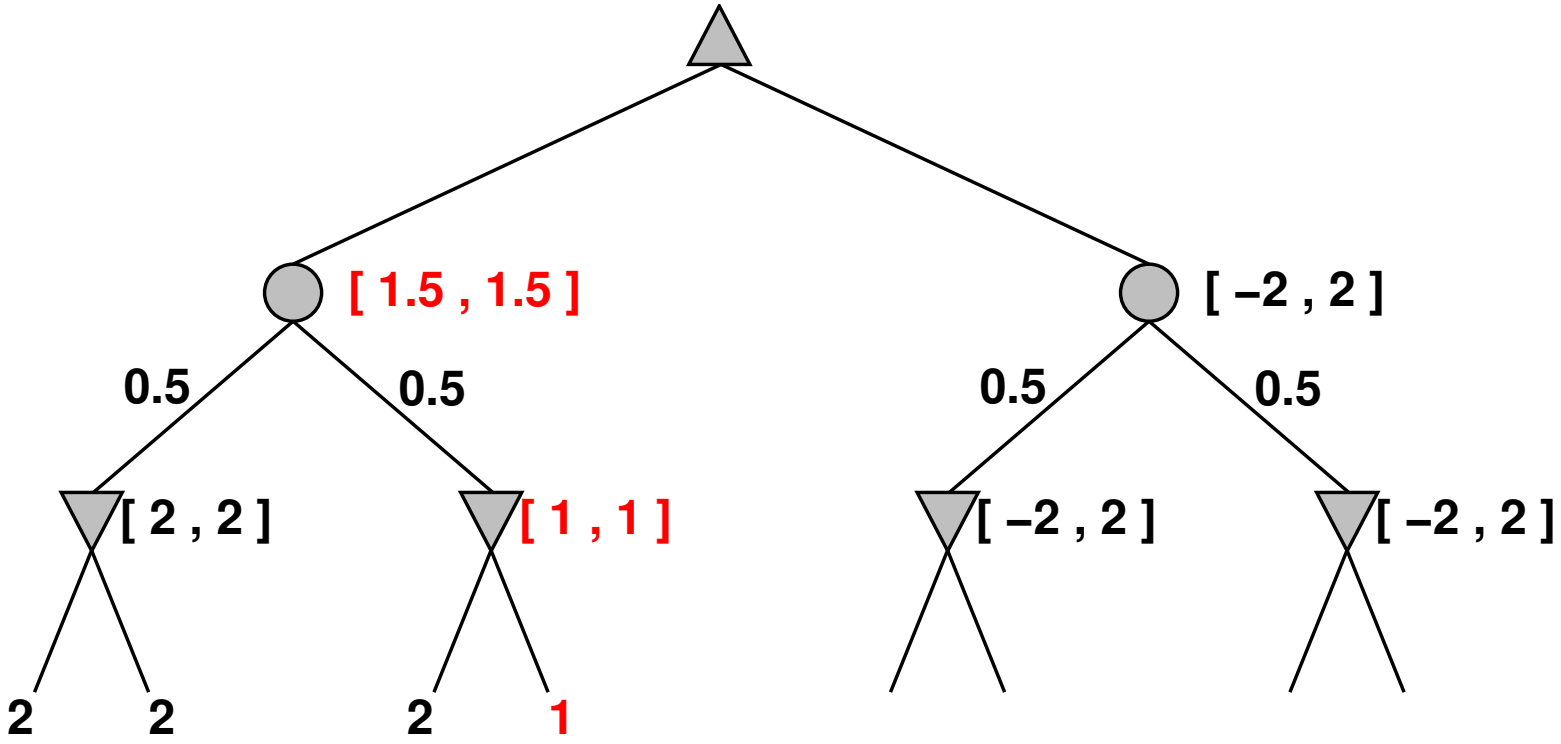
Corte (cont.)

Pode-se cortar mais se os valores das folhas forem limitados



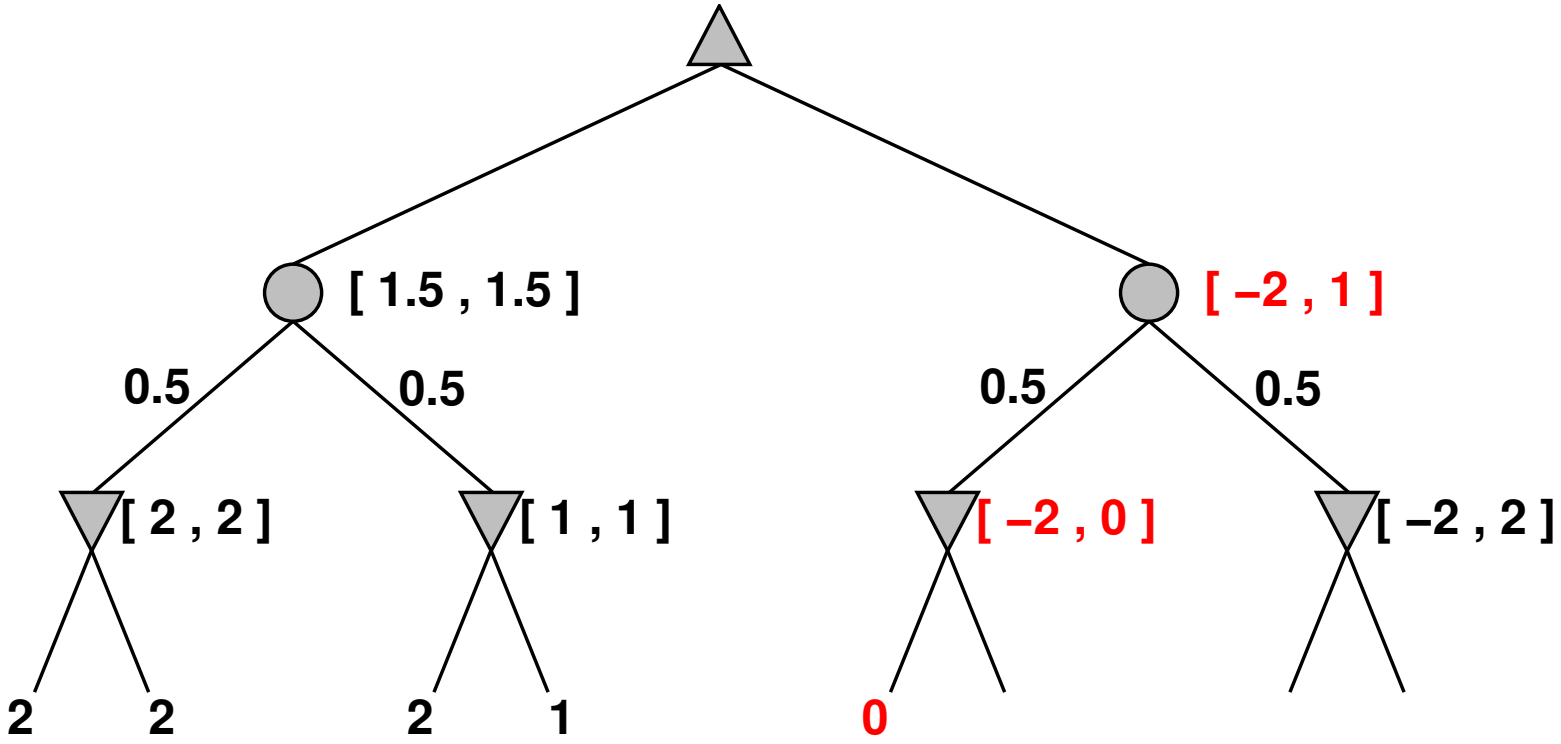
Corte (cont.)

Pode-se cortar mais se os valores das folhas forem limitados



Corte (cont.)

Pode-se cortar mais se os valores das folhas forem limitados



Jogos não deterministas na prática

Lançamentos de dados aumentam b: 21 lançamentos possíveis com 2 dados
Gamão $\approx$ 20 jogadas legais (podem ser 4000 com lançamentos de doubles)

$$\text{profundidade } 4 = 20 \times (21 \times 20)^3 \approx 1.2 \times 10^9$$

Com o aumento da profundidade, probabilidade de chegar a um determinado nó diminui

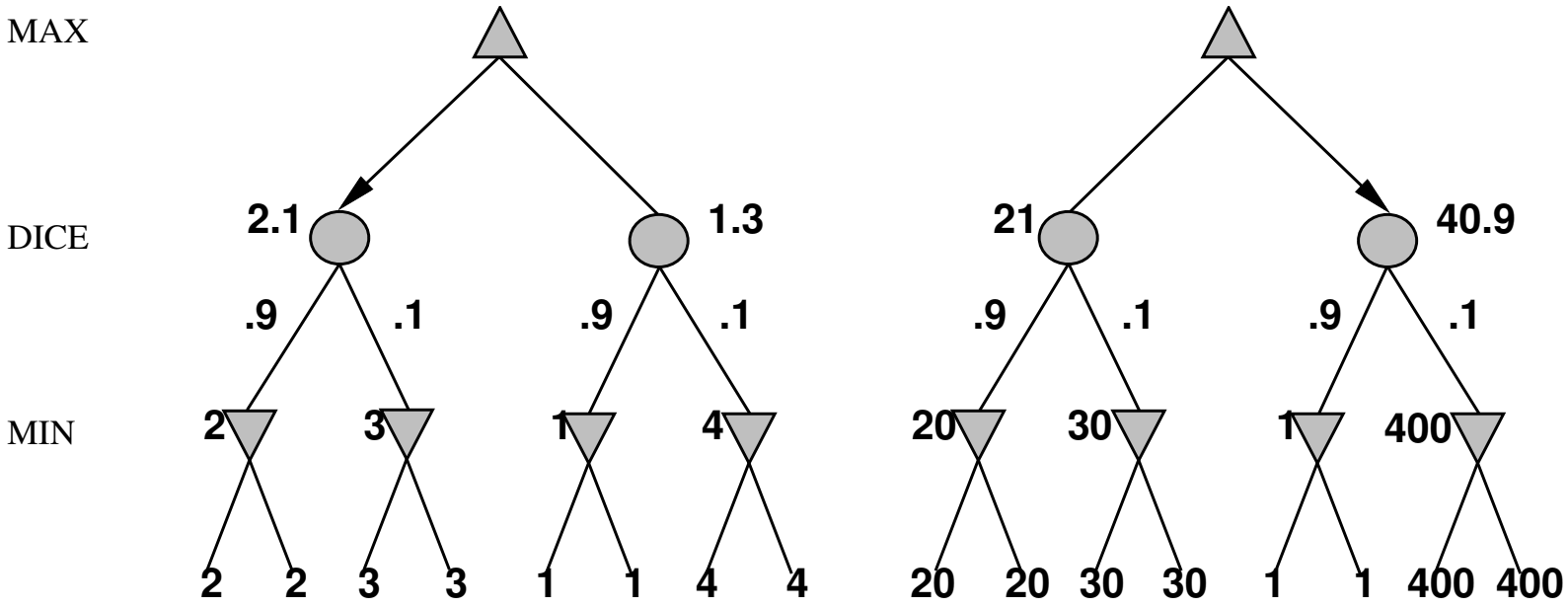
$\Rightarrow$ valor do "lookahead" diminui

Corte α–β é muito menos eficaz

TDGammon usa procura profundidade 2 + Eval muito bom

$\approx$ nível de campeão mundial

Digressão: Valores exactos SÃO importantes



O comportamento é preservado apenas com transformações *lineares positivas* de EVAL

Logo EVAL deve ser proporcional à recompensa esperada

Jogos com informação imperfeita

Exemplo: jogos de cartas, em que as cartas iniciais do adversário são desconhecidas

Tipicamente pode-se calcular a probabilidade de cada distribuição de cartas pelos jogadores

Aparentemente semelhante a um lançamento de dados no início do jogo

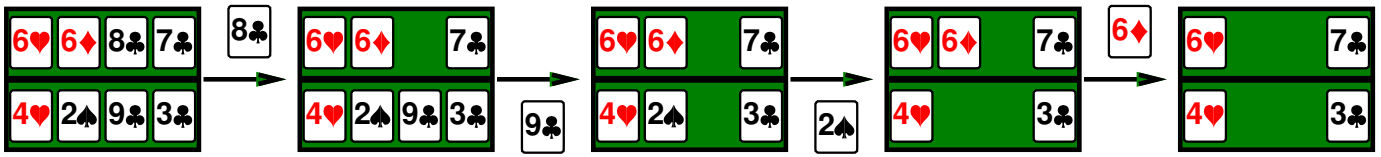
Ideia: calcular o valor minimax de cada acção em cada mão,
então escolher a acção com maior valor esperado de entre todas as mãos*

Caso especial: Se uma acção é óptima para todas as mãos então é óptima.*

GIB, melhor programa de bridge actualmente, aproxima esta ideia
1) gerando 100 mãos consistentes com a informação de apostas actuais
2) escolhe a acção que ganha mais vazas em média

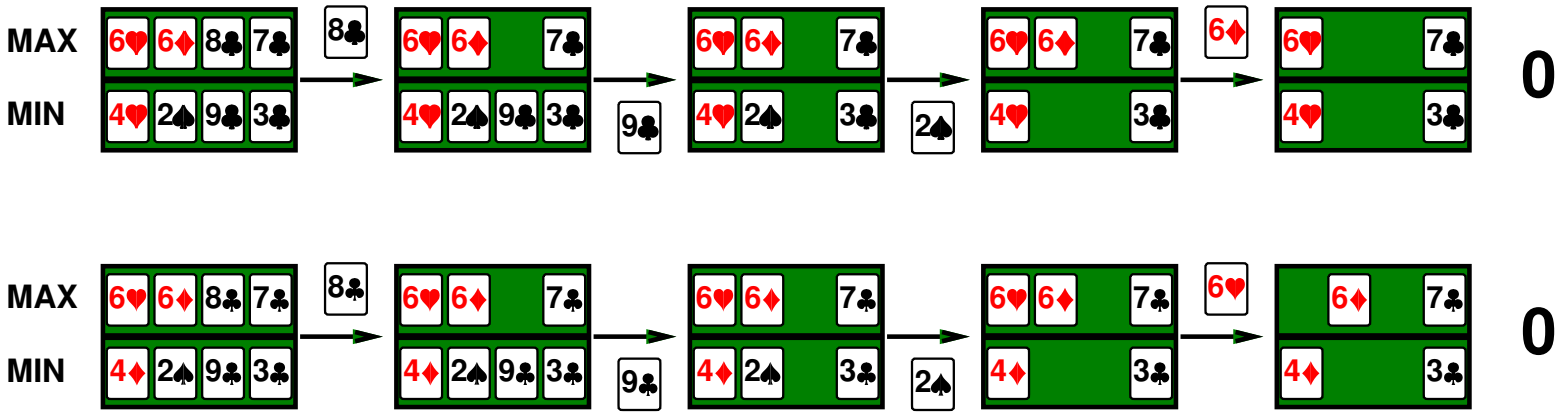
Exemplo

Mão de quarto cartas, MAX joga primeiro



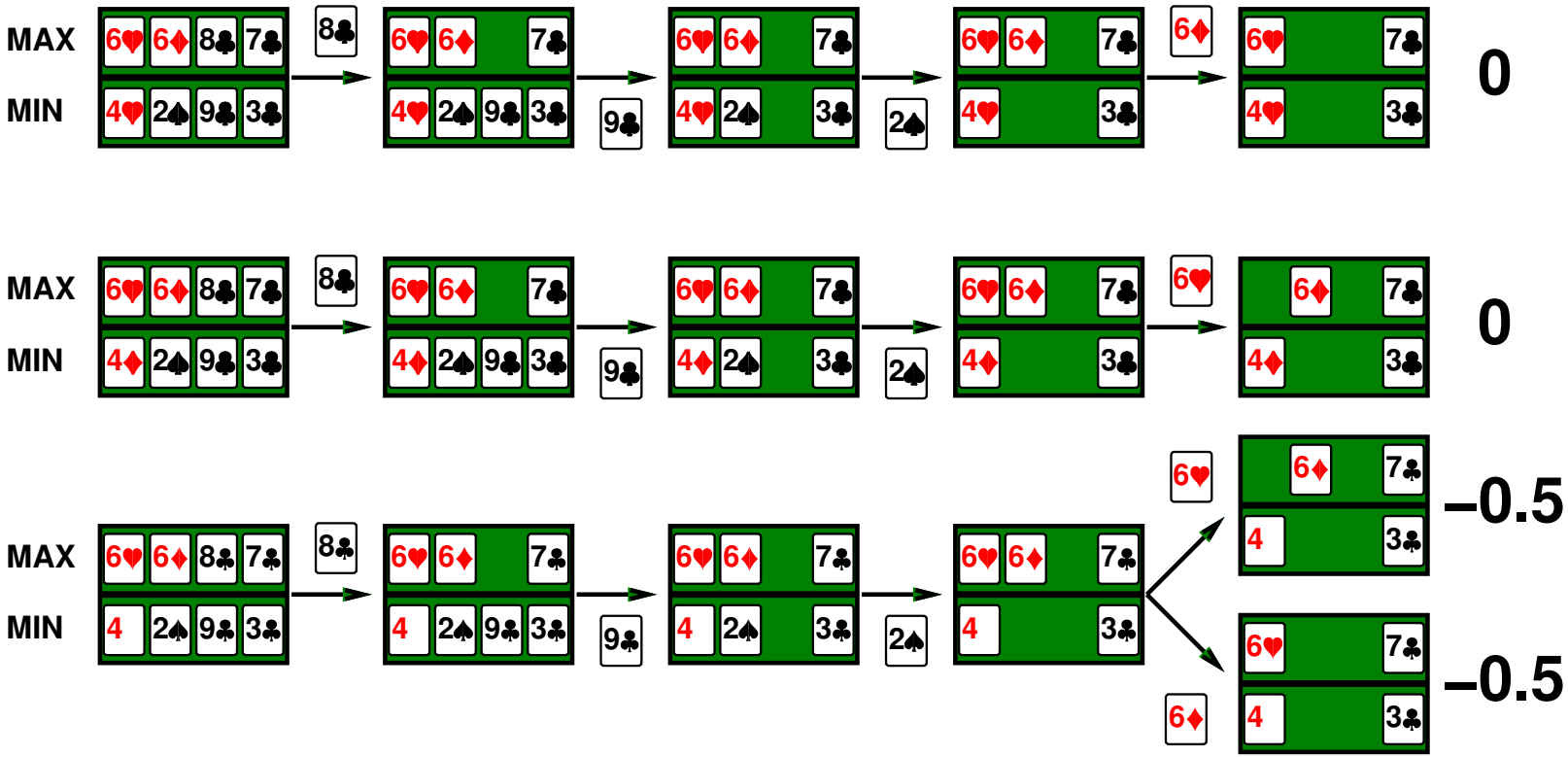
Exemplo

Mão de quarto cartas, MAX joga primeiro



Exemplo

Mão de quarto cartas, MAX joga primeiro



Exemplo de senso comum

Estrada A leva a um pequeno pote de moedas de ouro

Estrada B leva a um cruzamento:

se for pela esquerda encontra um monte de jóias;

se for pela direita é atropelado por um autocarro.

Exemplo de senso comum

Estrada A leva a um pequeno pote de moedas de ouro
Estrada B leva a um cruzamento:
se for pela esquerda encontra um monte de jóias;
se for pela direita é atropelado por um autocarro.

Estrada A leva a um pequeno pote de moedas de ouro
Estrada B leva a um cruzamento:
se for pela esquerda é atropelado por um autocarro.
se for pela direita encontra um monte de jóias;

Exemplo de senso comum

Estrada A leva a um pequeno pote de moedas de ouro
Estrada B leva a um cruzamento:
se for pela esquerda encontra um monte de jóias;
se for pela direita é atropelado por um autocarro.

Estrada A leva a um pequeno pote de moedas de ouro
Estrada B leva a um cruzamento:
se for pela esquerda é atropelado por um autocarro.
se for pela direita encontra um monte de jóias;

Estrada A leva a um pequeno pote de moedas de ouro
Estrada B leva a um cruzamento:
adivinhe correctamente e encontra um monte de jóias;
adivinhe incorrectamente e é atropelado por um autocarro.

Análise correcta

* Intuição que o valor de uma acção é a média de todos os seus valores em todos os estados possíveis é ERRADA

Com obervação parcial, valor de uma acção depende do estado de informação ou estado de crença em que o agente se encontra

Pode gerar e procurar a árvore de estados de informação

Origina os seguintes comportamentos racionais

- ♢ Agir para obter informação
- ♢ Fazer sinais ao parceiro
- ♢ Agindo aleatoriamente para minimizar a descoberta de informação

Sumário

Jogos ilustram vários aspectos importantes da IA

♢ perfeição é inatingível ⇒ tem de se aproximar

♢ boa ideia pensar sobre o que se vai pensar

♢ a incerteza limita a atribuição de valores a estados

Os jogos estão para IA como a Fórmula I está para a indústria automóvel