

Jogos com adversários: Minimax

Inteligência Artificial
PCS3438

Anna Helena Reali Costa
Escola Politécnica da USP
Engenharia de Computação (PCS)

Jogos: considerações gerais

- Aplicações atrativas para métodos IA desde o início:
 - Formulação simples do problema (ações bem definidas)
 - Ambiente totalmente observável (geralmente);
 - Abstração (representação simplificada de problemas reais);
 - Sinônimo de “inteligência”;
 - Primeiro algoritmo para xadrez foi proposto por Claude Shannon na década de 50.
- Porém desafiador:
 - Tamanho + limitação de tempo (35^{100} nós para xadrez);
 - Incerteza devido ao outro jogador;
 - Problema “contingencial”: agente deve agir antes de completar a busca

Formulando e resolvendo o problema

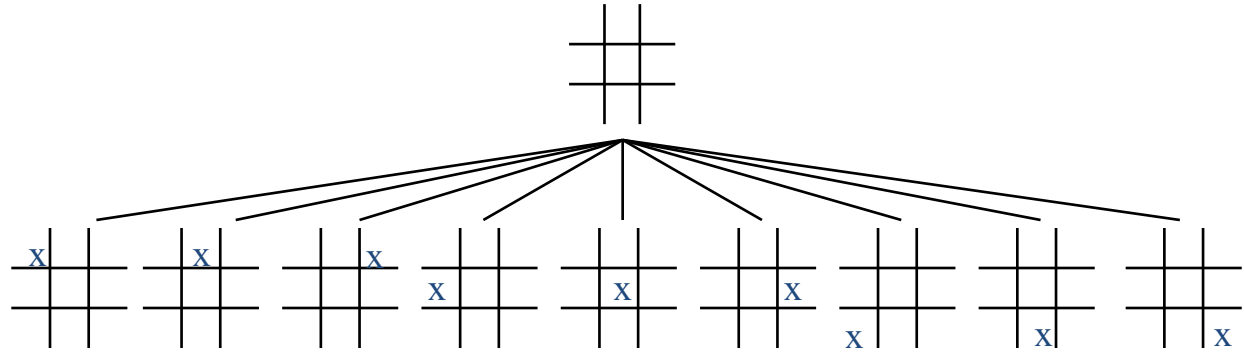
- 2 jogadores, revezam o lance, são adversários
- Formulação
 - **Estado inicial:** posições do tabuleiro + de quem é a vez
 - **Estado final:** posições em que o jogo acaba
 - **Operadores:** jogadas legais (=válidas)
 - **Função de utilidade:** valor numérico do resultado (pontuação)

Busca: algoritmo minimax

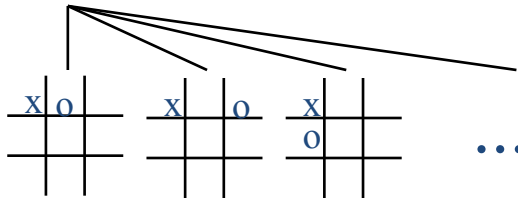
- Idéia: maximizar a utilidade (ganho) supondo que o adversário vai tentar minimizá-la.
- Minimax faz busca cega em profundidade.
- O agente é MAX e o adversário é MIN.

Jogo da velha (min-max)

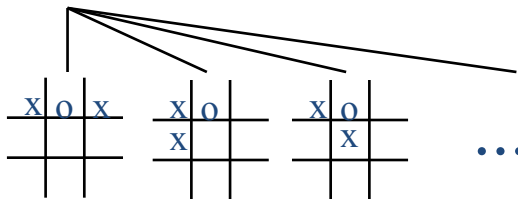
Max(X)



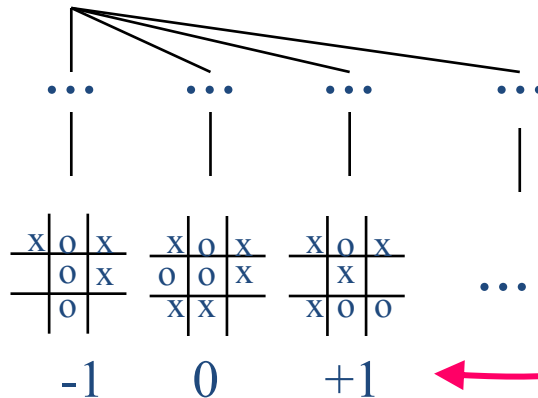
Min(O)



Max(X)



Min(O)



Função utilidade:

0 → empata

-1 → perde

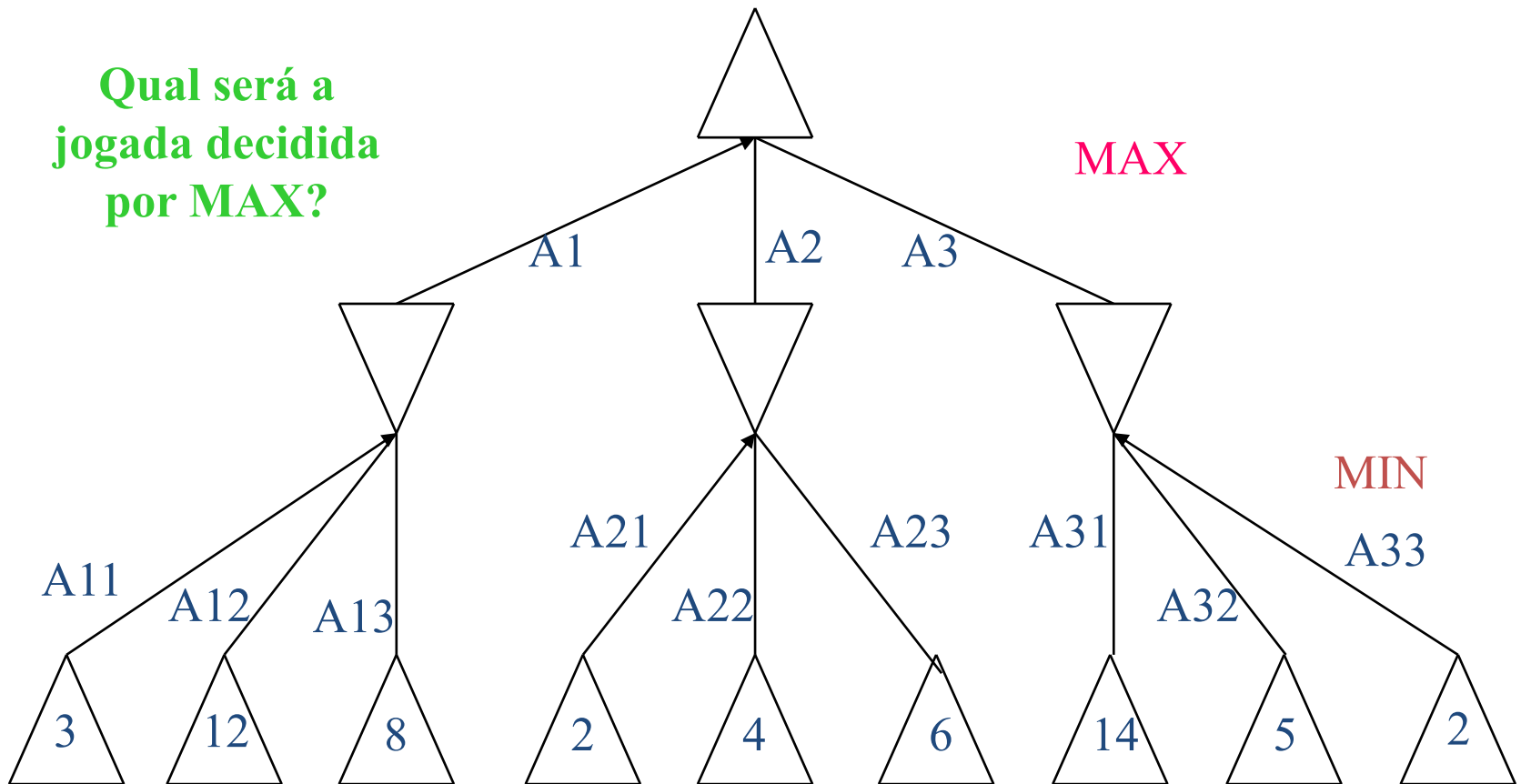
+1 → ganha

Minimax: passos

- Gera a árvore inteira até os estados terminais (ganha, perde ou empata).
- Aplica a função de utilidade nas folhas.
- Propaga os valores dessa função subindo a árvore através do minimax.
- Determinar qual a ação que será escolhida por MAX

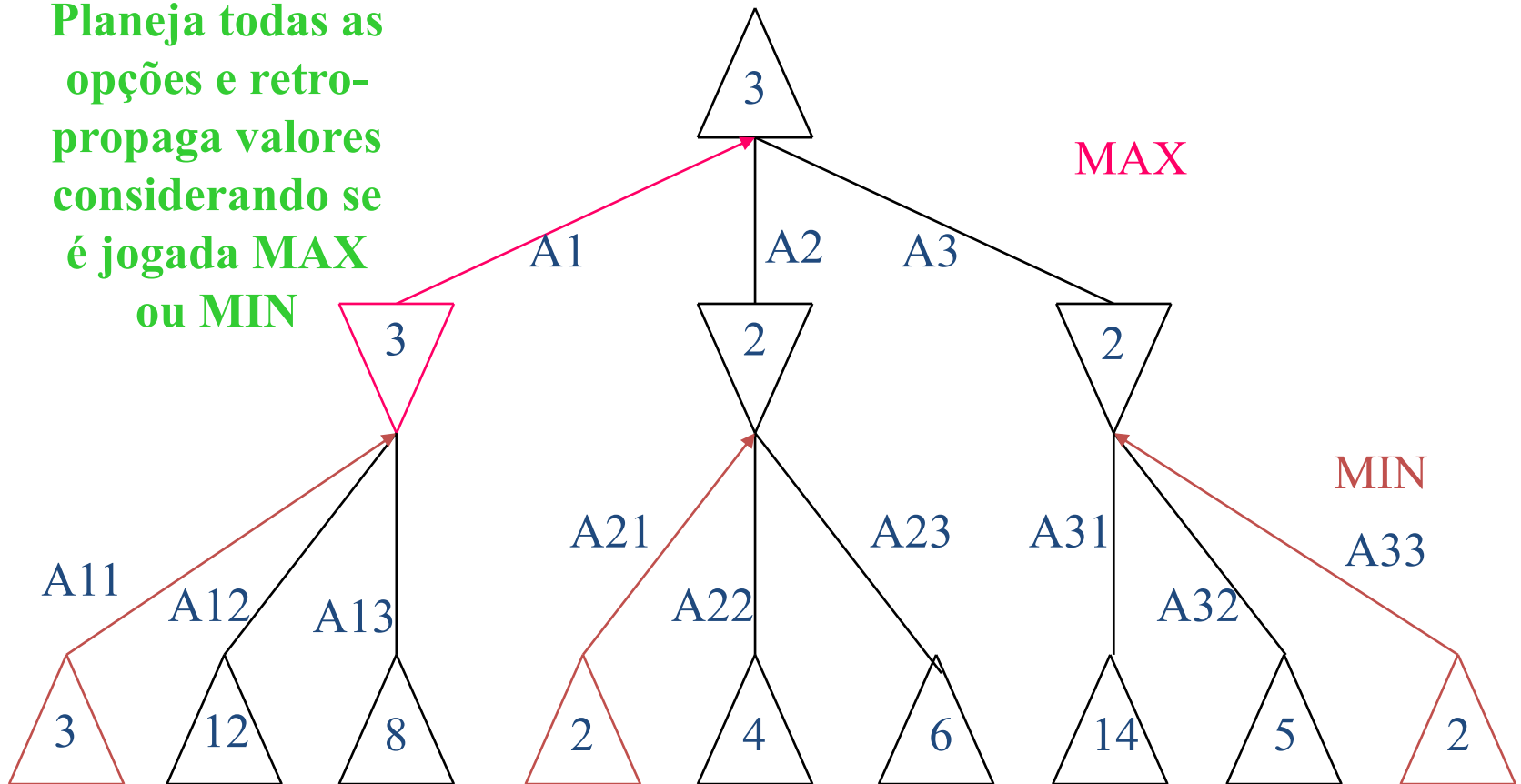
Minimax

Qual será a
jogada decidida
por MAX?

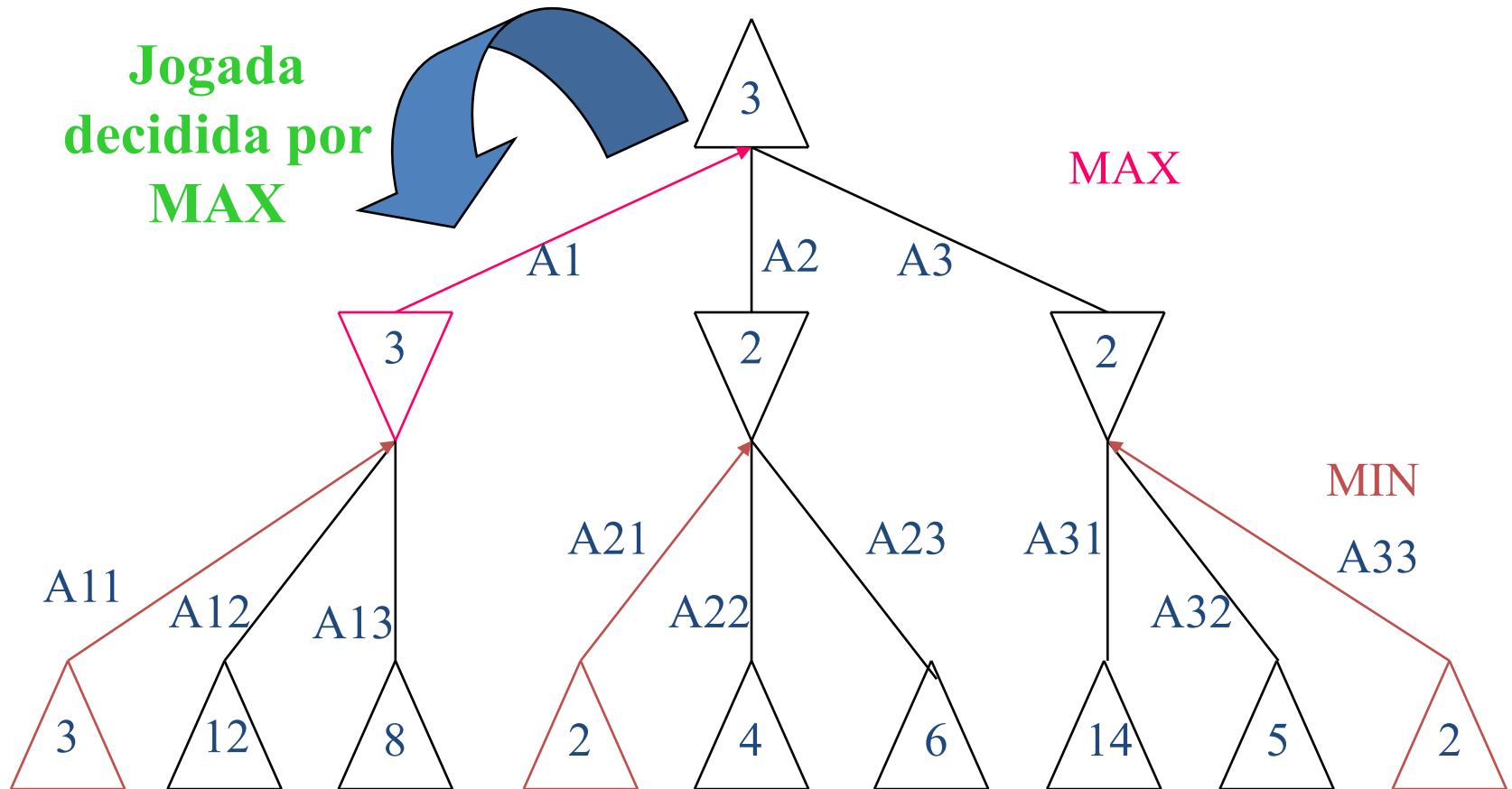


Minimax

Planeja todas as opções e retro-propaga valores considerando se é jogada MAX ou MIN



Minimax



Críticas

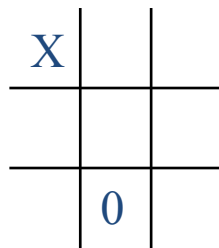
- Problemas

- Tempo gasto para determinar a decisão ótima é totalmente impraticável (ir até as folhas), porém o algoritmo serve como base para outros métodos mais realísticos.
- Complexidade: $O(b^m)$ – idem Busca em Profundidade.

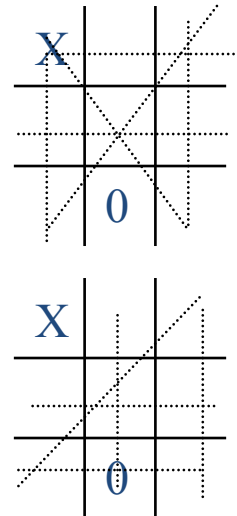
- Para melhorar

- 1) **Limitar** a profundidade da busca e substituir função de utilidade por **função de avaliação** (heurística);
- 2) Podar a árvore onde a busca seria irrelevante: **poda alfa-beta**

Uma função heurística para o jogo da velha

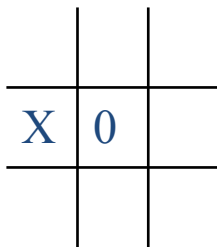


$$H = 6 - 5 = 1$$

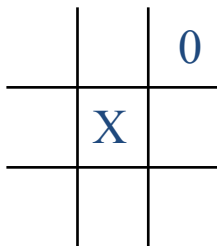


X tem 6 possibilidades de fazer trinca

0 tem 5 possibilidades de fazer trinca



$$H = 4 - 6 = -2$$



$$H = 5 - 4 = 1$$

Uso da Função de Avaliação

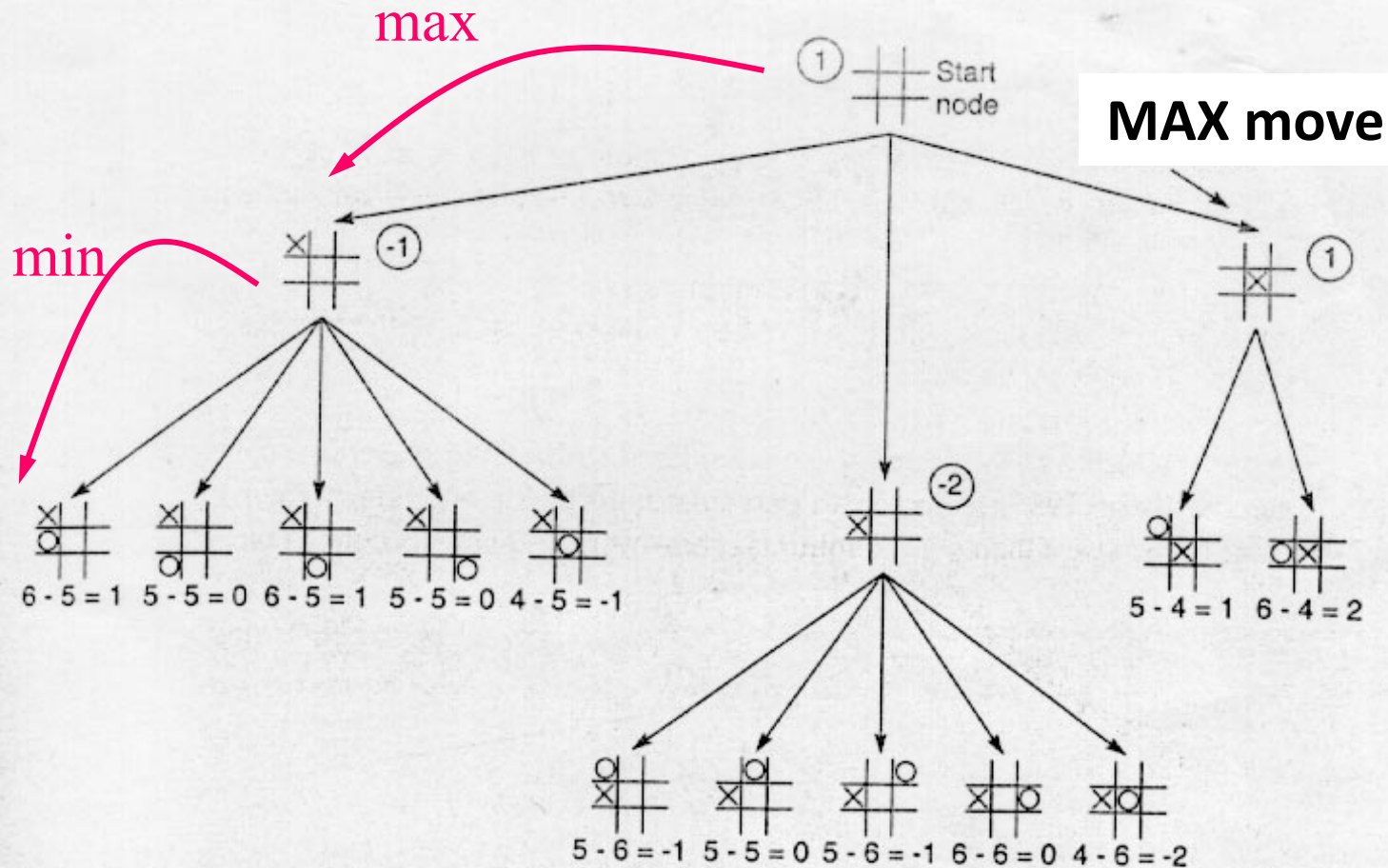


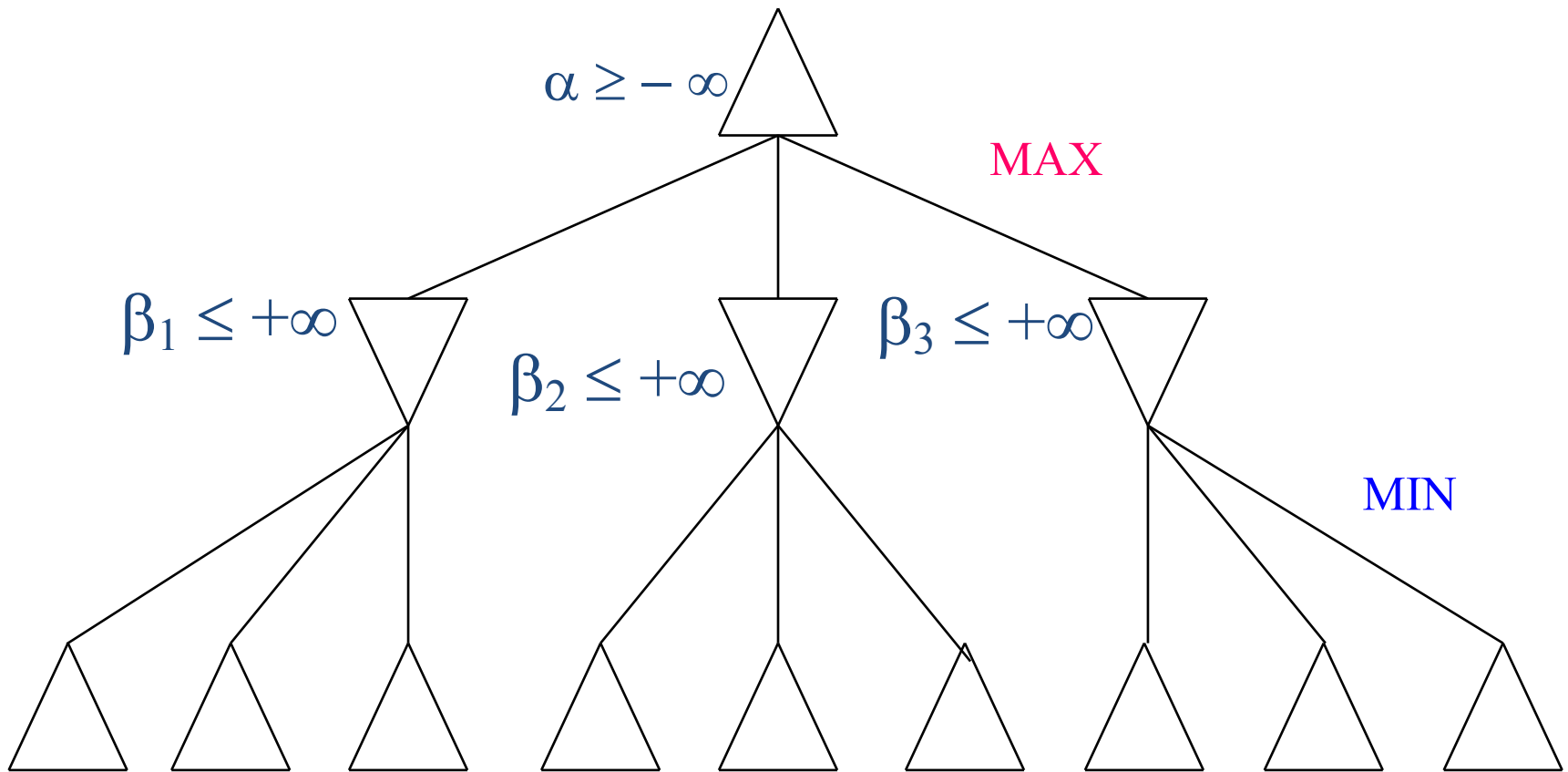
Figure 4.17 Two-ply minimax applied to the opening move of tic-tac-toe.

Poda Alfa-Beta

- Objetivo: não expandir desnecessariamente nós durante o minimax.
 - **Idéia: não vale a pena piorar, se já achou algo melhor.**
- Mantém 2 parâmetros:
 - α : melhor valor para MAX
 - β : melhor valor para MIN
- Teste de expansão:
 - α : **não pode diminuir** (não pode ser menor que um ancestral)
 - β : **não pode aumentar** (não pode ser maior que um ancestral)

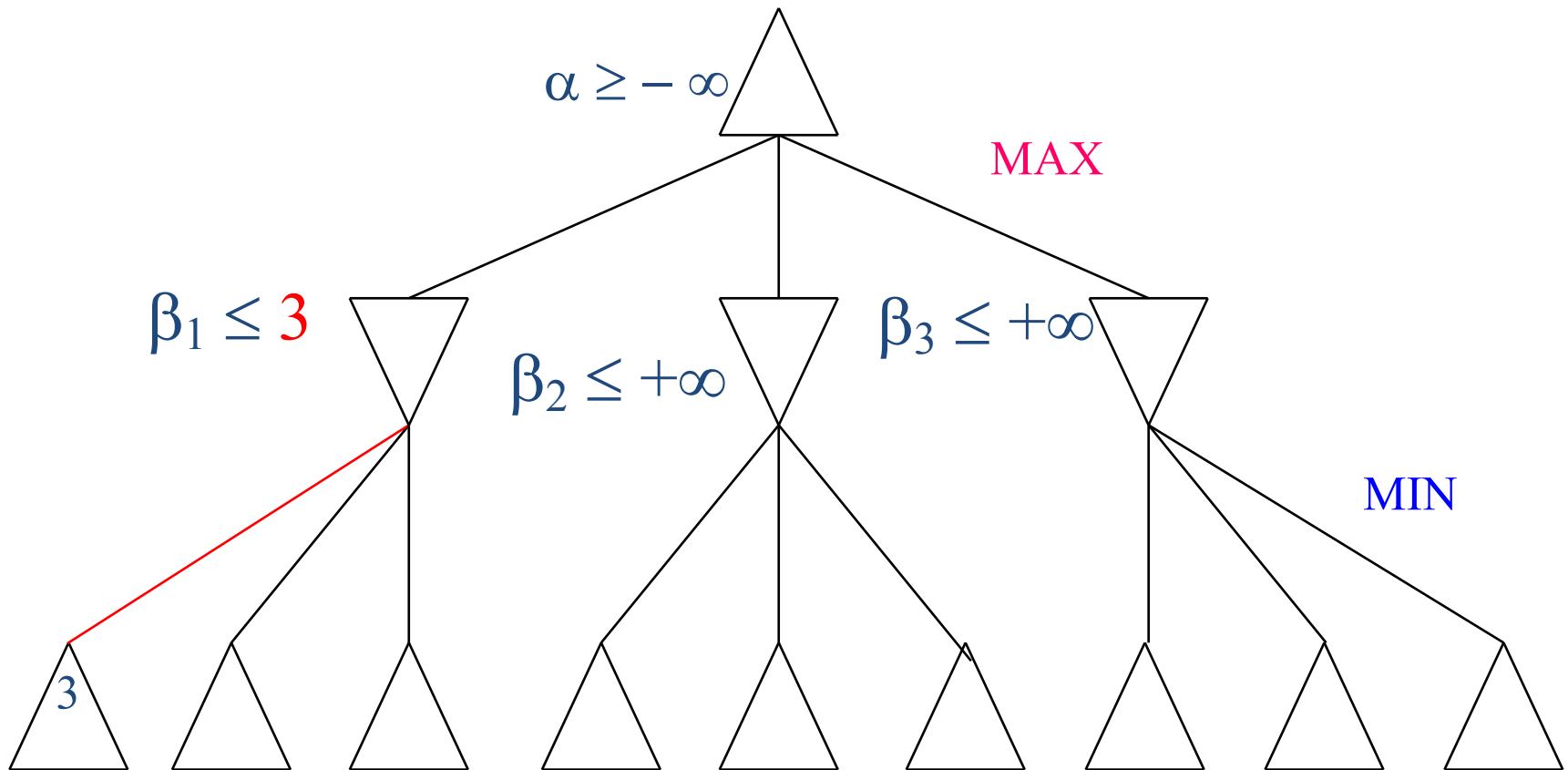
Poda Alpha-Beta: exemplo

Inicia valores α (max) e β (min).



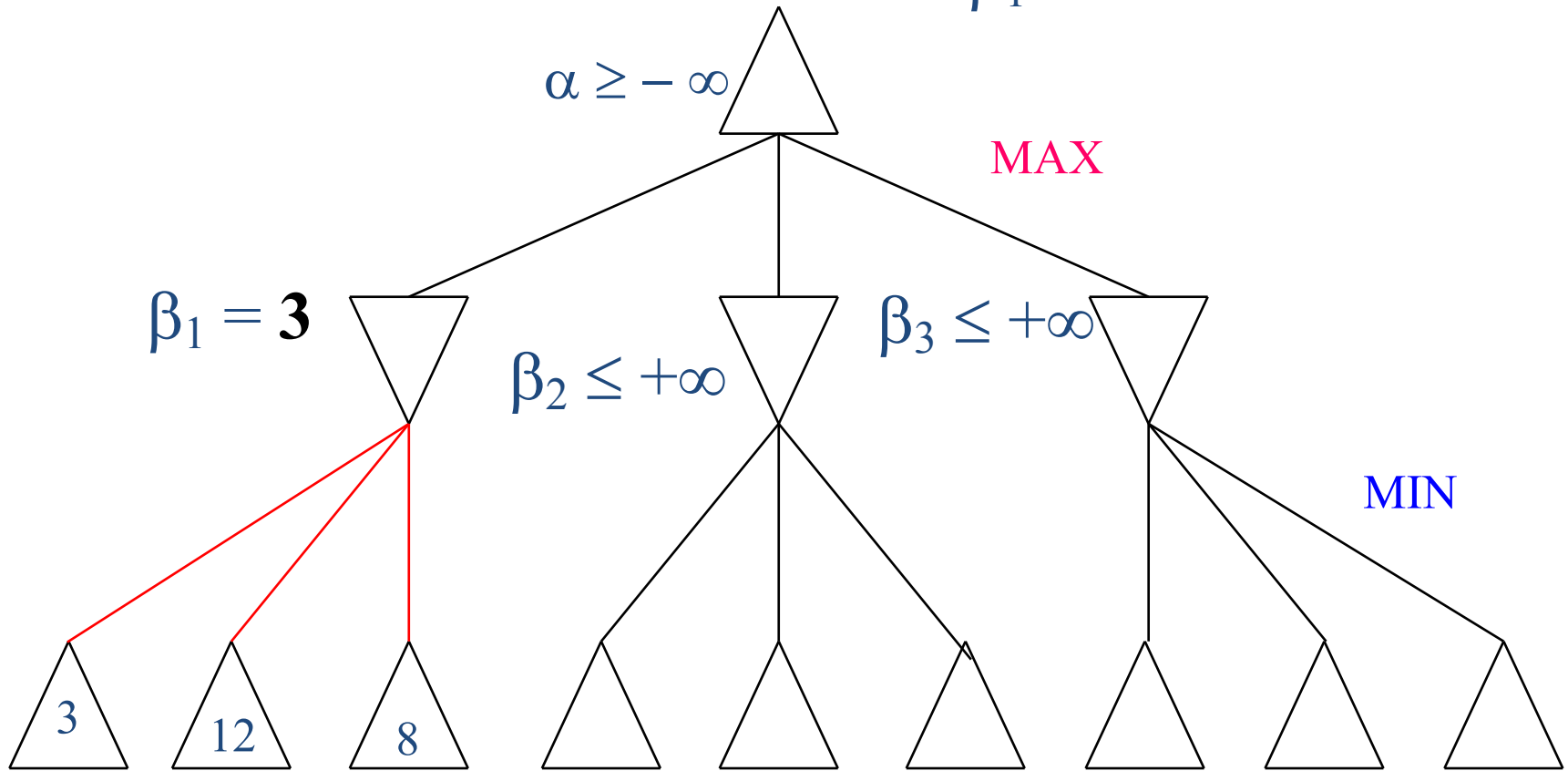
Poda Alpha-Beta: exemplo

Expande até o primeiro nó da profundidade desejada. Aplica a heurística neste nó. Atualiza β_1



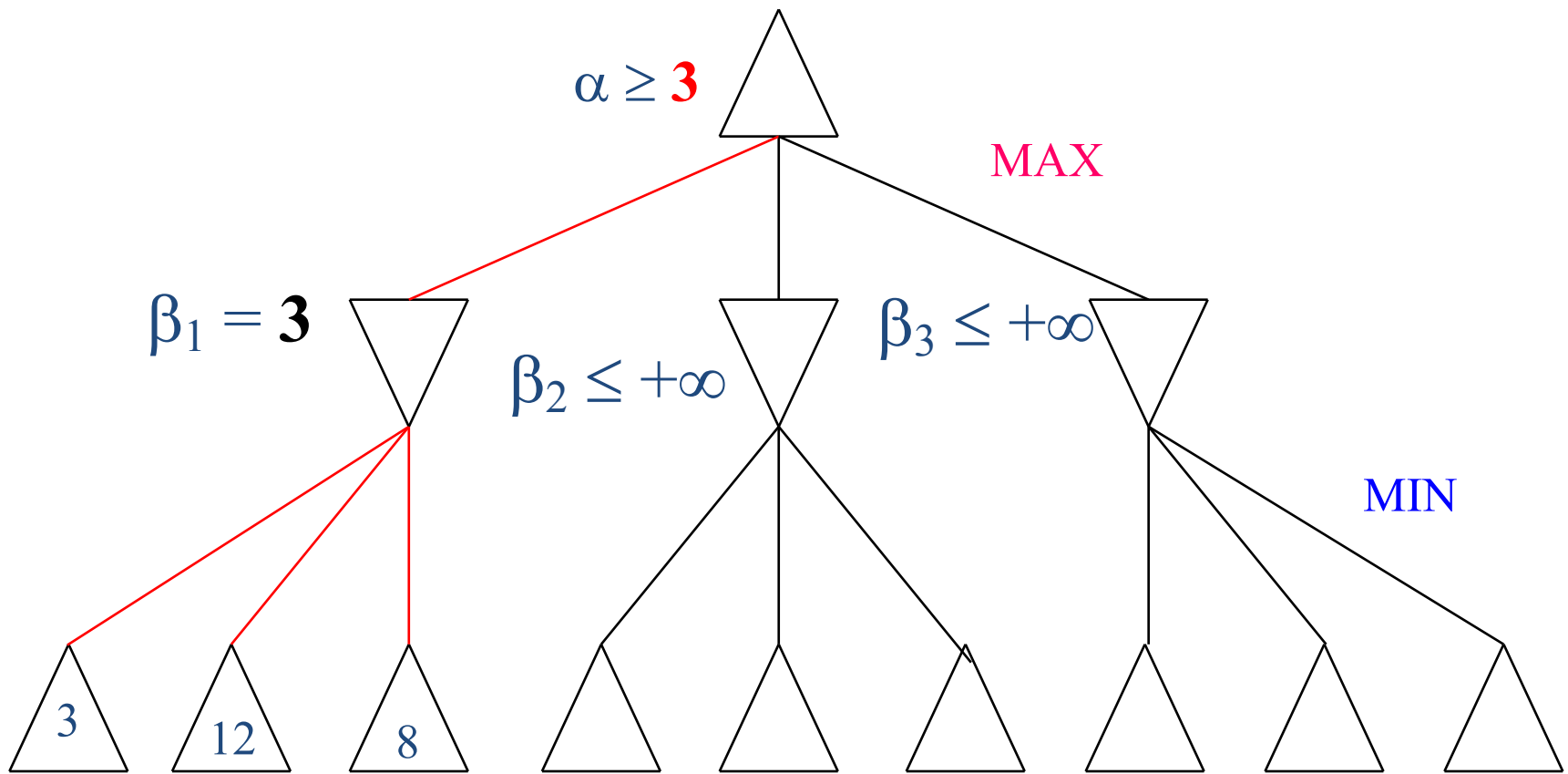
Poda Alpha-Beta: exemplo

Para ter certeza do valor de β_1 deve expandir todos
filhos do nó de β_1 .



Poda Alpha-Beta: exemplo

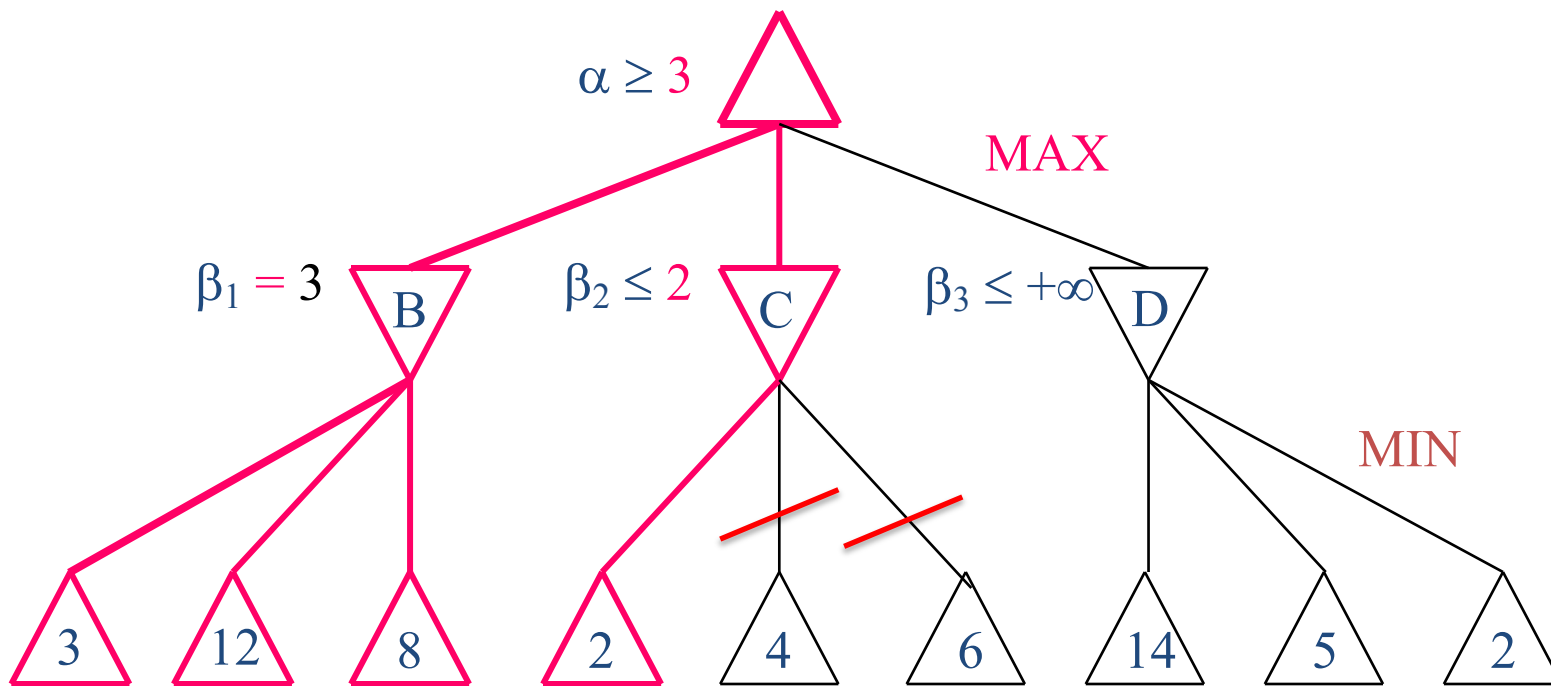
Propaga este valor para α .



Poda Alpha-Beta: exemplo

Expande C: folha com 2, atualiza $\beta_2 \leq 2$. Como $2 < \alpha$ do nó pai, C não precisa mais ser expandido.

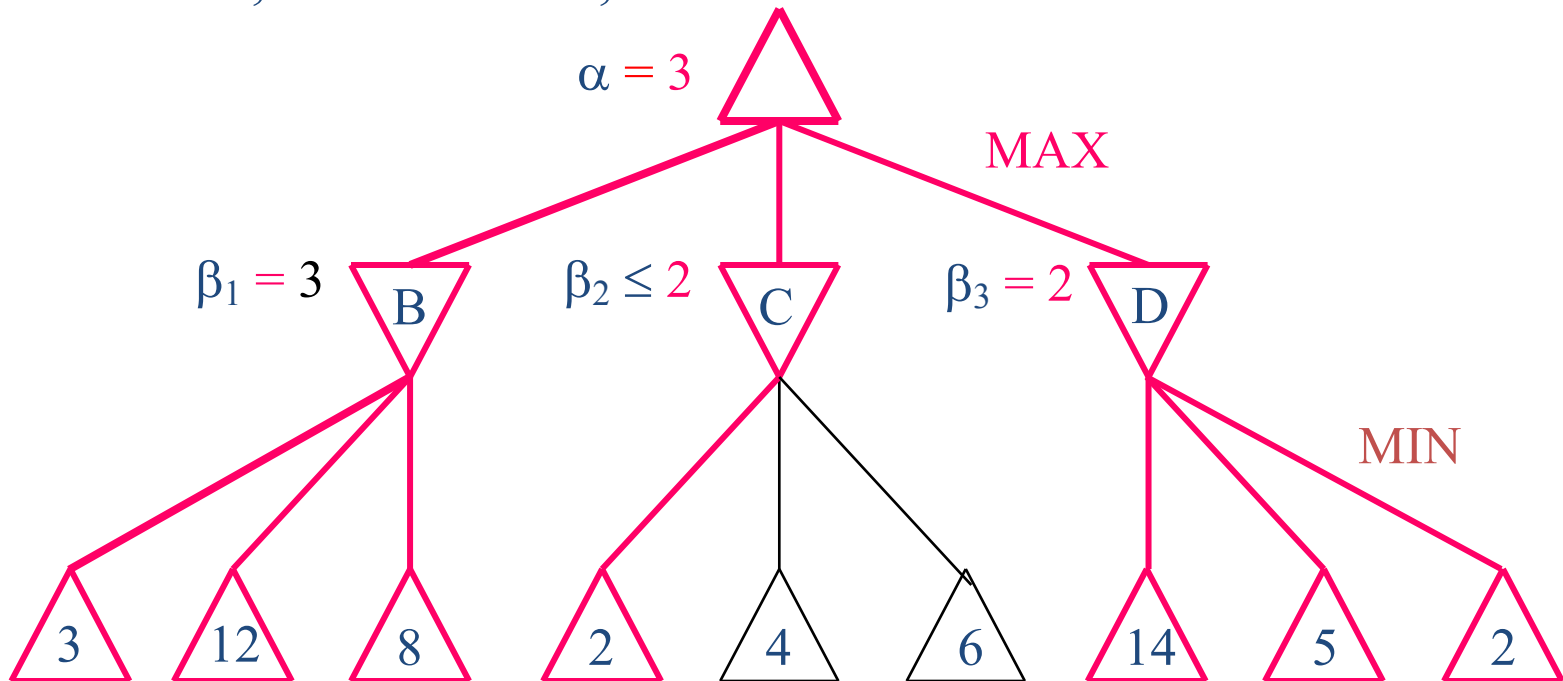
Justificativa: se novo filho de C tiver valor MAIOR que 2, como β_2 não pode aumentar, nada será mudado; se novo filho tiver valor MENOR que 2, β_2 reduzirá mas não afetará α , que só pode aumentar e já está com valor 3.



Poda Alpha-Beta: exemplo

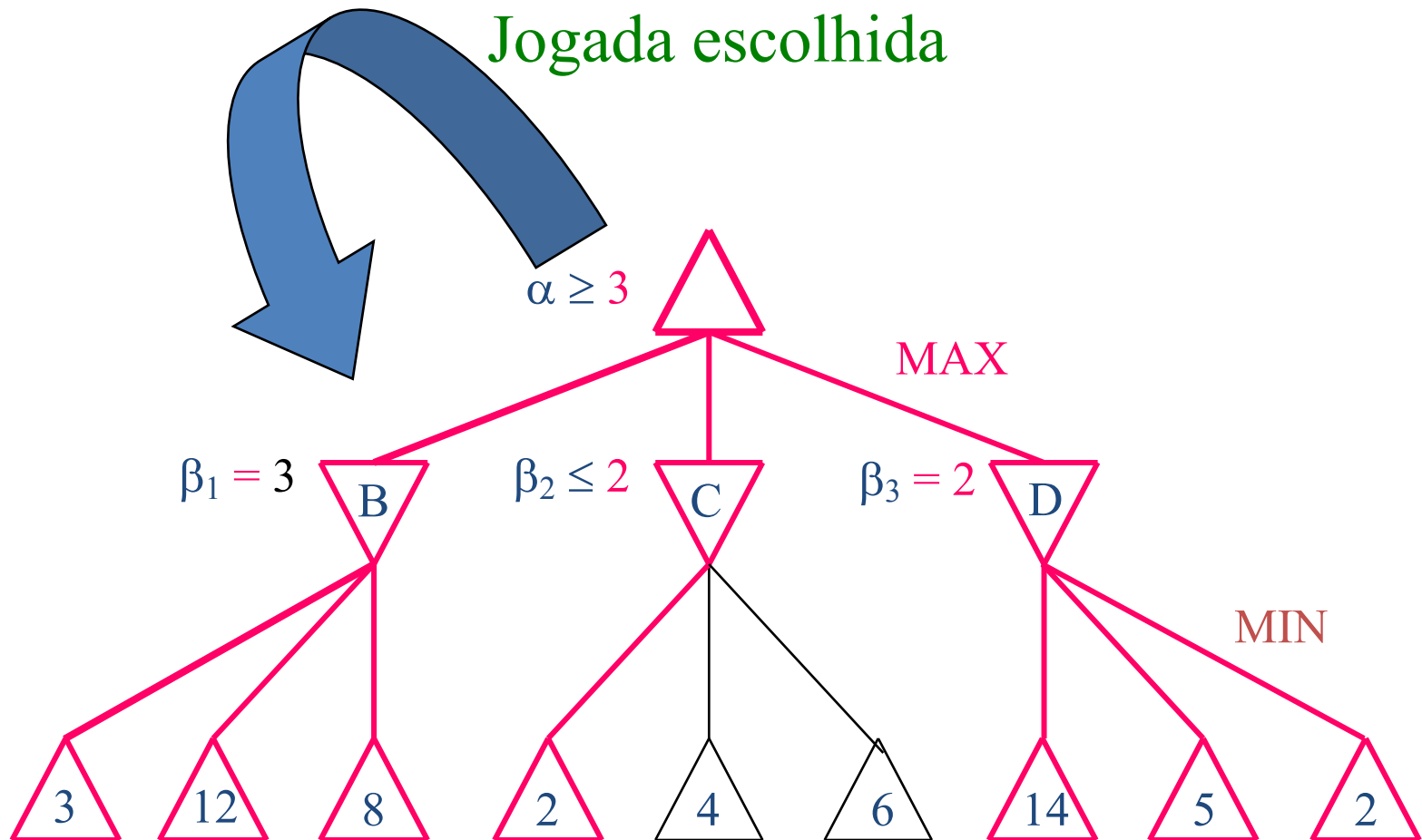
Expande D:

- 1) folha com 14, atualiza $\beta_3 \leq 14$ e como $14 > \alpha$ e β_3 ainda pode diminuir, continua a expansão de D.
- 2) Folha com 5, reduz β_3 e como $5 > \alpha$ e β_3 ainda pode diminuir, continua a expansão de D.
- 3) Última folha tem 2: define valor de $\beta_3 = 2$ e verifica se atualiza α ; como $\alpha > 2$, ele não muda e a busca termina.



Poda Alpha-Beta: exemplo

Busca termina, com escolha da jogada B.



Exercício

Decidir a jogada de MAX (A, B ou C) considerando as utilidades fornecidas nas folhas. Adotando a poda alfa-beta, indicar quais arestas/subárvores serão podadas.

