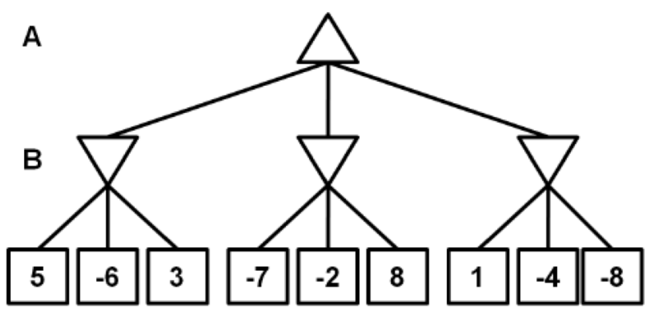
.

CEFET/RJ
Disciplina: Inteligência Artificial
Professor: Eduardo Bezerra
Lista de exercícios 02

Créditos: alguns itens desta lista são adaptados do material da disciplina *CS188 - Artificial Intelligence*[1], da University of Berkeley. Outros são adaptações de exercícios propostos no livro-texto da disciplina, *AIMA*[2].

1. (Busca Competitiva) Considere a seguinte árvore de um jogo de soma zero, no qual as utilidades mostradas nos nós-folha são para o primeiro jogador (A) que é um MAXimizador. Suponha que o segundo jogador (B) é um MINimizador.



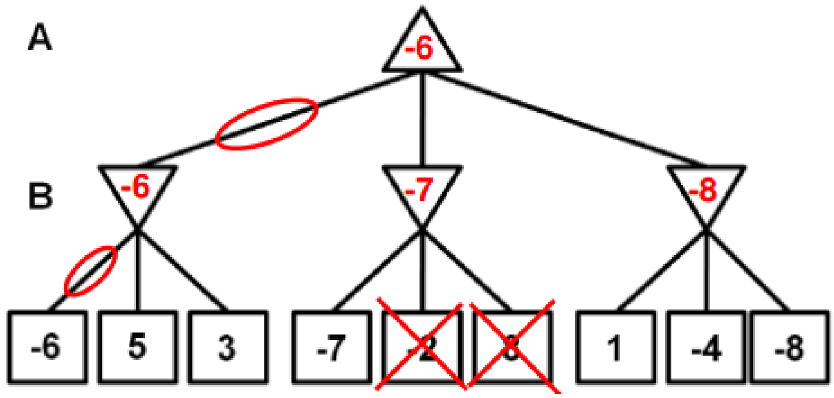
   (a) Escreva nos nós internos da árvore o valor da utilidade $U_A(s)$ do jogador A (isto é, o valor minimax desses nós).

   (b) Circule as arestas da árvore correspondentes às jogadas escolhidas por A e por B de acordo com o valor minimax.

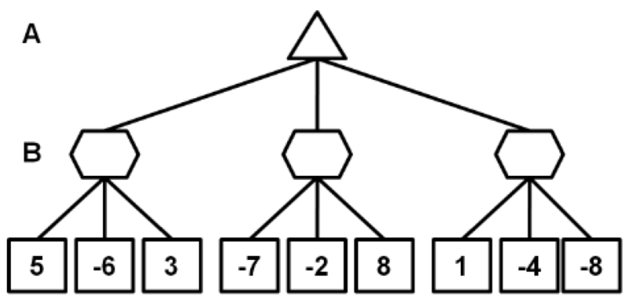
   (c) Faça um X em cima dos nós que seriam podados pela poda alfa-beta, supondo que os nós são percorridos da esquerda para a direita.

Solução

[1] `http://ai.berkeley.edu/home.html`
[2] `http://aima.cs.berkeley.edu/`



2. Ainda considerando o jogo de soma zero do exercício anterior, suponha agora que o segundo jogador (B) é um equilibrador. Um equilibrador não tenta minimizar a utilidade de A. Ao invés disso, ele quer que o resultado do jogo seja o mais equilibrado possível, isto é, o mais próximo possível de zero. A figura abaixo mostra a árvore do jogo com hexágonos indicando os nós de equilíbrio.

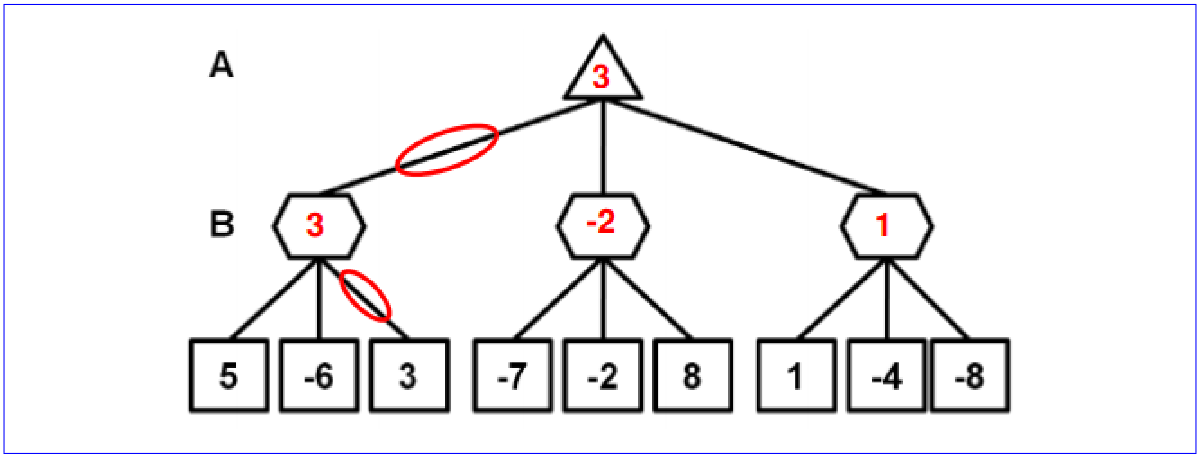


(a) Escreva nos nós internos da árvore o valor da utilidade $U_A(s)$ do jogador A, supondo que B é um equilibrador.

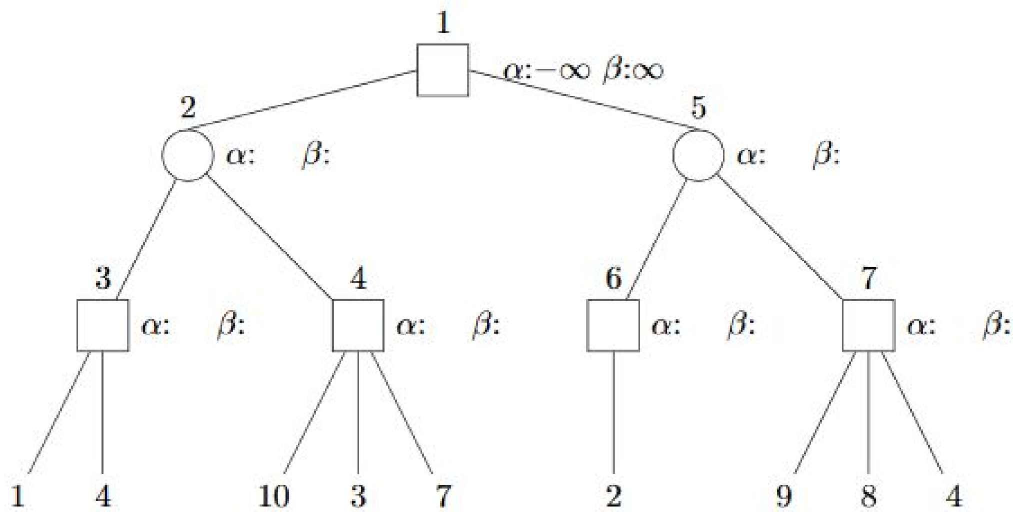
(b) Circule as arestas da árvore correspondentes às jogadas escolhidas por A e por B.

(c) Repita esse exercício considerando agora que a utilidade de B é dada por $U_B(s) = -|U_A(s)|$.

Solução



3. A árvore a seguir representa todos os resultados possíveis de um jogo hipotético de soma zero. Esta árvore foi construída da perspectiva do jogador MAX; Os nós MAX são representados por quadrados e os nós MIN são representados por meio de círculos. As folhas da árvore representam o valor do jogo para o jogador MAX. O número em cada nó indica a ordem em que são considerados pelo algotimos Minimax e $\alpha - \beta$.



(a) Calcule os valores de *back-up* de cada nó na árvore usando a estratégia Minimax, e escreva esses valores no espaço dentro de cada nó.

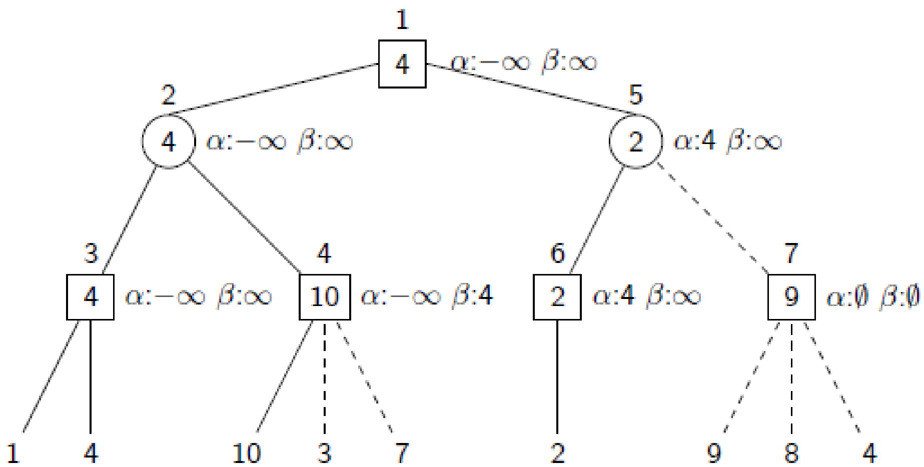
(b) Nota: esta pergunta tem duas partes; Leia as duas partes completamente antes de começar a responder.

i. Execute o algoritmo $\alpha - \beta$ e circule cada folha e nó que NÃO for considerado pelo algoritmo. (Suponha que as folhas são consideradas na ordem da esquerda para a direita.)

ii. Além disso, ao fazê-lo, escreva os valores de α e de β que são passados como argumentos para a chamada recursiva em cada nó no espaço fornecido. (Por exemplo, o algoritmo de poda $\alpha - \beta$ é inicializado com $\alpha = -\infty$ e $\beta = -\infty$, portanto estes são os valores passados como argumentos para a primeira chamada no nó raiz.) Nota: Se algum nó não receber valores α ou β devido à poda, basta escrever X para seus valores α e β.

Solução

Na figura a seguir, as linhas pontilhadas correspondem às arestas podadas durante a busca.

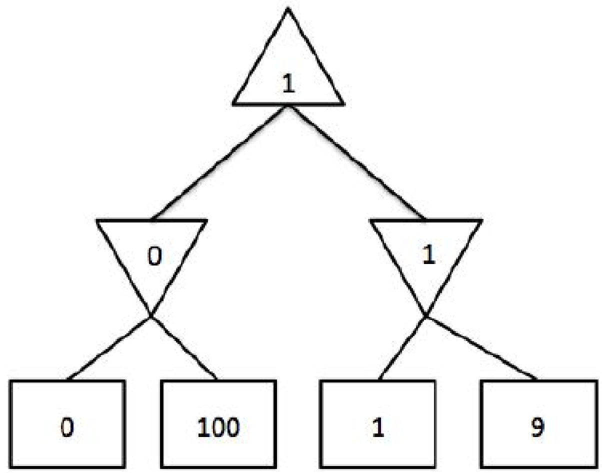


Um ponto importante neste problema é lembrar que α e β são calculados a partir de valores de backup, mas que α e β só são passados para baixo na árvore. Assim, por definição, o valor de backup escolhido por um nó MIN ou MAX pode afetar os valores do seu nó pai, mas apenas para chamadas subsequentes em outros nós irmãos do mesmo pai. Assim, o nó 3 retorna 4 e portante $\beta = 4$ para o nó 4, mas esse valor para β não é passado de volta ao nó 2. Em vez disso, o nó 2 retorna 4 de volta ao nó 1 e, portanto, o nó 1 define $\alpha = 4$ para as avaliações no nó 5 e no nós subsequentes. O outro ponto importante é lembrar que a tarefa solicitada no exercício era escrever os valores de α e β foram dados como *entrada* par algoritmo de poda $\alpha - \beta$ em cada nó.

4. Neste problema, você vai investigar a relação entre as árvores expectimax e árvores minimax para jogos de soma-zero com dois jogadores. Imagine que você tem um jogo que alterna entre o jogador 1 (MAX) e o jogador 2. O jogo começa no estado s_0, com o jogador 1. O jogador 1 pode escolher um movimento usando ou a busca minimax, ou a busca expectimax, onde os nós do jogador 2 são nós de acaso em vez de serem nós min.

   (a) Desenhe uma árvore de jogo (pequena) na qual o nó raiz tem um valor maior se a busca **expectimax** for usada do que se a busca **minimax** for usada, ou argumente por que não seria possível.

Solução



Podemos ver aqui que a árvore de jogo acima tem um valor na raiz igual a 1 para a estratégia minimax. Se nós mudarmos para expectimax e substituirmos os nós min com nós de acaso, a raiz da árvore assume um valor igual a 50 e a ação ótima muda para MAX.

(b) Desenhe uma árvore de jogo (pequena) na qual o nó raiz tem um valor maior se a busca **minimax** for usada do que se busca **expectimax** é usada, ou argumente por que não seria possível.

Solução

O jogo ideal para MIN, por definição, significa os melhores movimentos para MIN para obter o menor valor possível. Jogar de forma aleatória inclui movimentos que não são ótimos. Supondo que não existem empates (i.e., não há duas folhas que possuam o mesmo valor), expectimax sempre i´ra calcular a média em movimentos subóptimos. A média de um movimento subótimo (para MIN) contra um movimento ótimo (para MIN) sempre aumentará o resultado esperado.

Com isso em mente, podemos ver como não há árvore de jogo para a qual o valor da raiz para expectimax é inferior ao valor da raiz para minimax. Um deles é o jogo ótimo, enquanto que o outro é o jogo sub-ótimo em que se faz a média com o jogo óptimo, que por definição conduz a um valor mais elevado para MIN.

(c) Sob quais suposições acerca do jogador 2 o jogador 1 deve usar busca minimax em vez da busca expectimax para selecionar um movimento?

Solução

O jogador 1 deve usar a busca minimax se ele espera que o jogador 2 se mova otimamente.

(d) Sob quais suposições acerca do jogador 2 o jogador 1 deve usar a busca de expectimax em vez da busca minimax?

Solução

Se o jogador 1 espera que o jogador 2 se mova de forma aleatória, então ele deve usar a busca expectimax. Isso irá otimizar a obtenção do valor máximo esperado.

(e) Imagine que o jogador 1 deseja agir de forma ótima (i.e., racionalmente), e o jogador 1 sabe que o jogador 2 também pretende de forma ótima. No entanto, o jogador 1 também sabe que o jogador 2 (equivocadamente) acredita que o jogador 1 está se movendo uniformemente ao acaso em vez de otimamente. Explique como o jogador 1 deve usar esse conhecimento para selecionar um movimento. Sua resposta deve ser um algoritmo preciso envolvendo uma pesquisa de árvore de jogo, e deve incluir um esboço de uma árvore de jogo apropriada com o movimento do jogador 1 na raiz. Seja claro acerca de que tipo de nós estão em cada camada (ply) e de quem é a vez em cada camada.

Solução

Use duas árvores de jogos:

- Árvore de jogo 1: max é substituído por um nó de acaso. Resolva esta árvore para encontrar a política para MIN.
- Árvore de jogo 2: a árvore original, mas MIN não tem nenhuma escolha agora; em vez disso, MIN é obrigado a seguir a política encontrada a no árvore de jogo 1.

5. Definimos X_n como o número de linhas, colunas ou diagonais com exatamente n valores de X e nenhum valor de O (análogo para O). A função de utilidade atribui $+1$ a qualquer posição com $X_3 = 1$ e -1 a qualquer posição com $O_3 = 1$. Todas as outras posições terminais tem utilidade 0. No caso de posições não-terminais, utilizamos uma função de avaliação linear definida como

$$Aval(s) = 3X_2(s) + X_1(s) - (3O_2(s) + O_1(s))$$

(a) Aproximadamente, quantas possibilidades de jogos existem no jogo da velha?

(b) Mostre a árvore de jogo inteira a partir de um tabuleiro vazio até a profundidade 2, levando em conta a simetria.

(c) Marque em sua árvore as avaliações de todas as posições na profundidade 2.

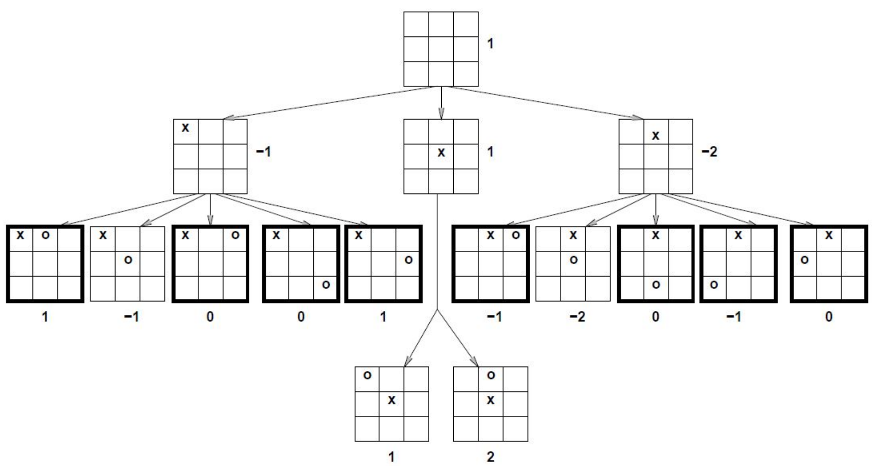
(d) Usando o algoritmo minimax, marque em sua árvore os valores propagados, e utilize esses valores para escolher o melhor movimento inicial.

(e) Destaque os nós na profundidade 2 que não seriam avaliados se a poda alfa-beta fosse aplicada, supondo que os nós fossem gerados na ordem ótima para poda alfa-beta.

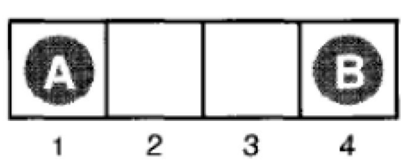
Solução

a) existem no máximo 9! jogos. (Este é o número de sequências de movimento necessárias para preencher completamente o tabuleiro, mas muitas vitórias e derrotas terminam antes que o tabuleiro esteja cheio.)

b-e) a figura abaixo mostra a árvore de jogo, com os valores da função de avaliação abaixo dos nós terminais e os valores minimax à direita dos nós não-terminais. Os valores implicam que o melhor movimento de partida para X é ocupar o centro do tabuleiro. Os nós terminais com um contorno em negrito são os que não precisam a ser avaliados, presumindo a ordem ideal.



6. Considere o seguinte jogo de dois jogadores:



- O jogador A joga primeiro.
- Os dois jogadores de revezam na movimentação.

- Cada jogador deve mover sua ficha para um espaço adjacente aberto em qualquer direção.
- Se o oponente ocupar um espaço adjacente, o jogador pode saltar sobre o oponente até o próximo espaço aberto, se houver.
- O jogo termina quando um jogador chega à extremidade oposta.

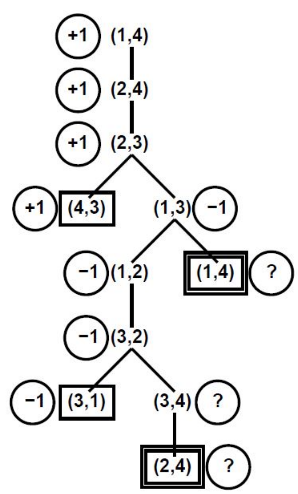
(a) Desenhe a árvore de jogo completa, usando as convenções a seguir:
   - Escreva cada estado com (sA, sB), onde sA e sB denotam as posições das fichas.
   - Coloque cada estado terminal em um quadrado e escreva o seu valor em um círculo.
   - Insira os estados repetidos em quadrados duplos. Tendo em vista que não esta clara a maneira de atribuir valores a esses estados, identifique cada um com um "?".

(b) Agora marque cada nó com seu valor minimax propagado. Explique como você tratou os valores "?" e por que.

(c) Explique por que o algoritmo minimax padrão falharia nessa árvore e faça um resumo de como corrigi-lo, baseando-se em sua resposta ao item (b).

Solução

a) A figura abaixo apresenta a árvore do jogo. Estados terminais correspondem aos retângulos com moldura simples. Estados repetidos estão em retângulos com moldura dupla. Cada estado está anotado com seu valor minimax em um círculo.



b) Os valores `?` são tratados considerando que um agente diante de uma escolha entre ganhar o jogo e entrar em um estado `?` sempre escolherá a vitória. Ou seja, $min(-1, \texttt{?})$ é -1 e $max(+1, \texttt{?})$ é $+1$. Se todos os sucessores forem `?`, o valor de propagado é `?`.

c) O minimax padrão usa DFS e consequentemente entraria em um *loop* infinito. Pode ser corrigido comparando o estado atual com a pilha (*stack*): se o estado for repetido, então retorna um valor `?`. A propagação dos valores `?` pode ser feita conforme descrito no item anterior. Embora funcione neste caso, nem sempre funciona porque não está claro como comparar `?` com uma posição de empate; nem é claro como lidar com a comparação quando há ganhos de diferentes graus (como no backgammon). Finalmente, em jogos com nós de acaso (*chance nodes*), não é claro como calcular a média entre um número e um `?`. Observe que não é correto tratar estados repetidos automaticamente como posições de empate; neste exemplo, ambos (1,4) e (2,4) se repetem na árvore, mas eles são posições de vitória.