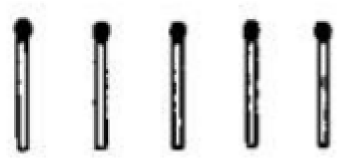
Inteligência Artificial

Thiago Henrique Leite da Silva, RA: 139920

AULA8: Exercício teórico jogos minimax e poda alfa-beta

1) (0,3) Temos um conjunto com 5 palitos de fósforos. No jogo Nim (conhecido como resta 1) cada jogador deve remover no mínimo 1 palito e no máximo 3 da fila. Quem retirar o último palito perde o jogo.



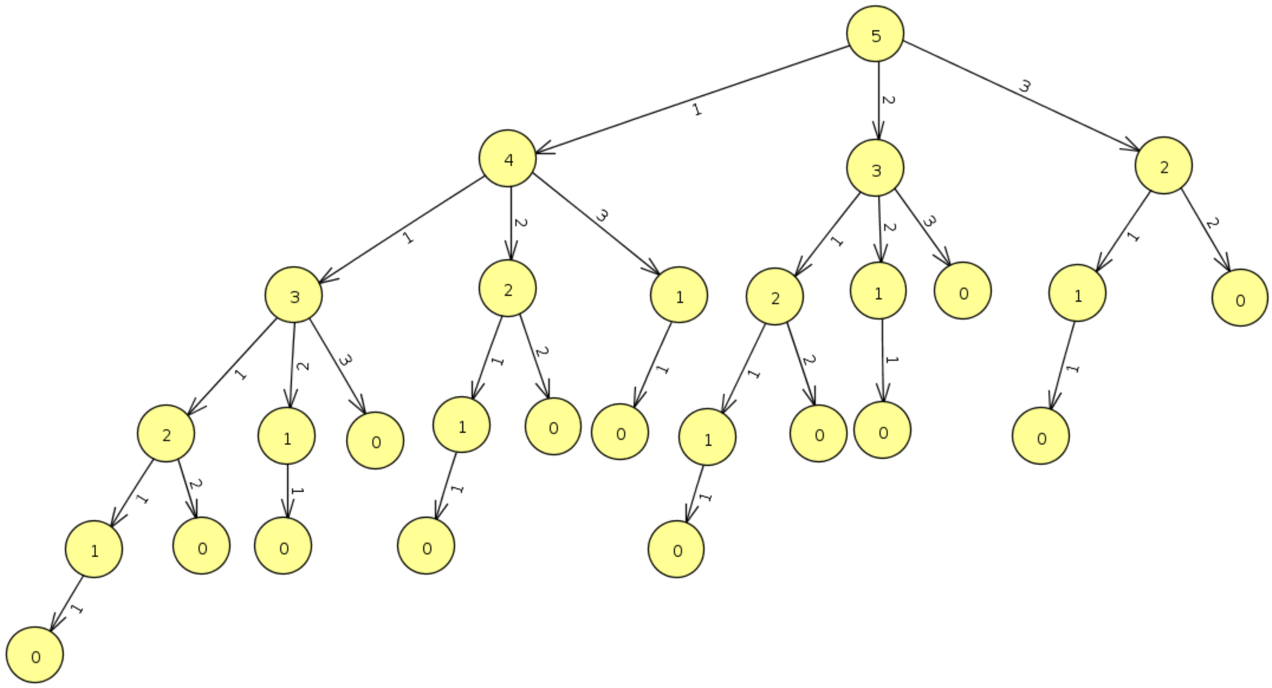
a) Desenhe a árvore minimax deste jogo. Que número de palitos o primeiro jogador deve retirar para ganhar o jogo?

O jogo de palitos de fósforo é jogado pelos jogadores A e B. Iremos determinar a jogada de Max para o jogador A (primeiro jogador), ou seja, queremos descobrir qual caminho fará com que jogador A vença o game.

Nossa função de avaliação será da seguinte maneira:

- +1 se o jogador A ganhar;
- -1 se o jogador B ganhar

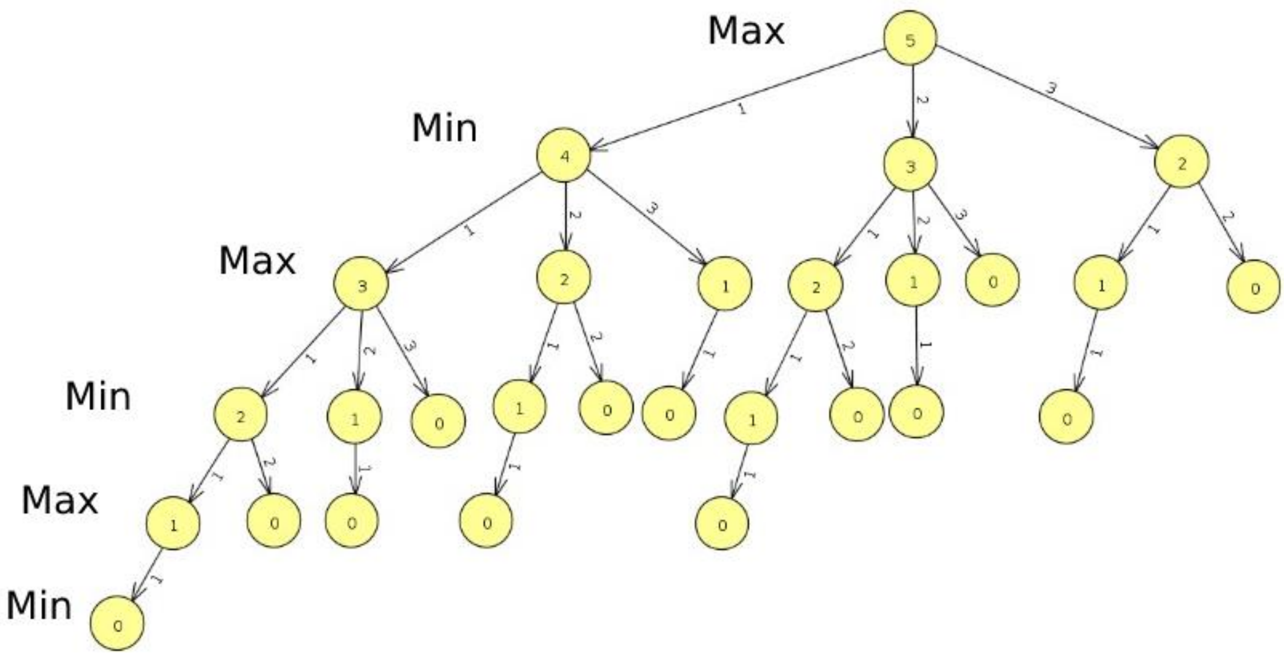
Para fazermos a árvore minimax do jogo, o primeiro passo é construirmos todos os estados possíveis, que são os estados abaixo:



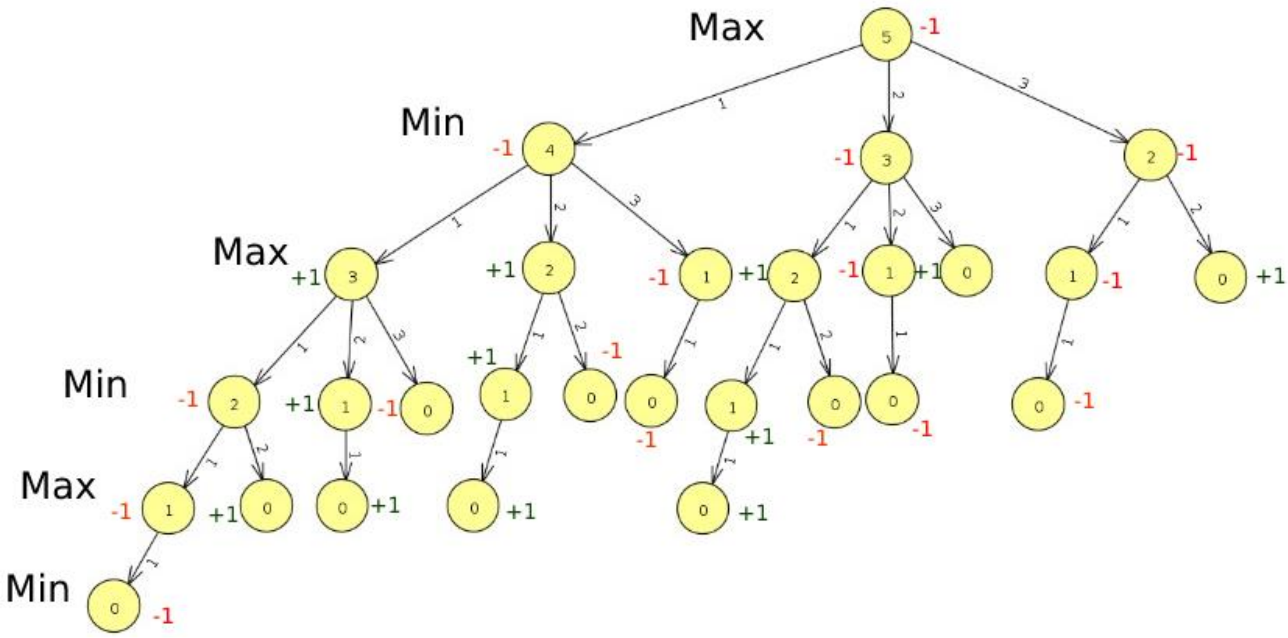
Os estados foram representados da seguinte forma:

- O número dentro das circunferências é a quantidade de palitos que restam ser retirados na jogada;
- O número ao lado das arestas é a quantidade de palitos retirados de um estado para outro;

Elencamos então nossos níveis Max e Min:



Agora, partindo de baixo para cima, da esquerda para direita, iremos aplicar a função de avaliação e selecionar os nós máximos e mínimos de acordo com o nível:



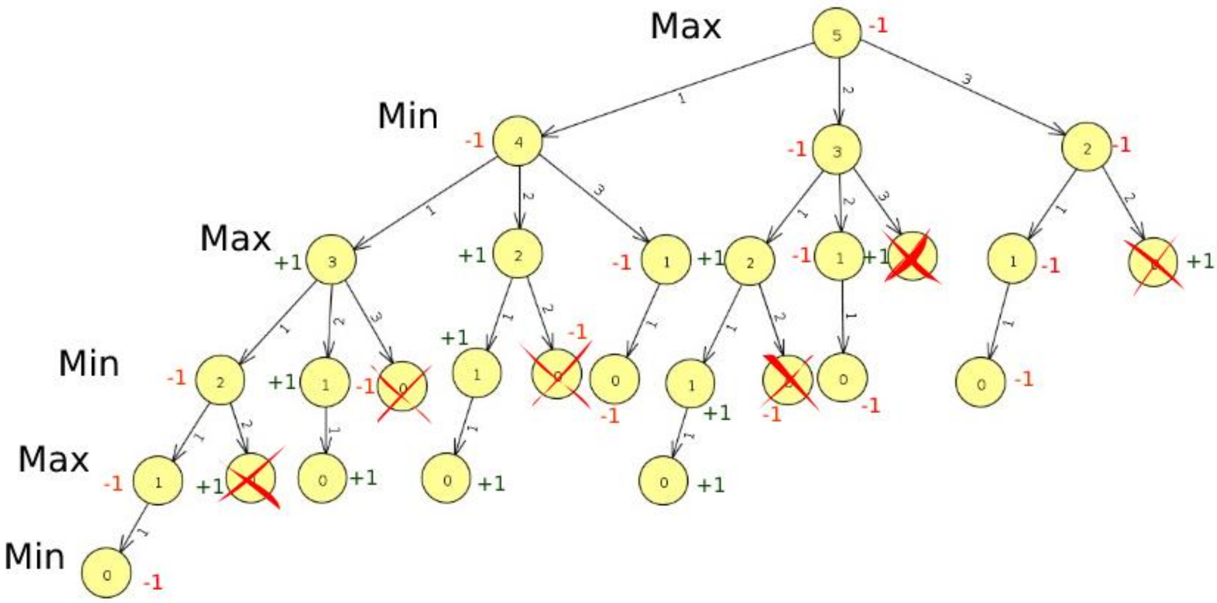
Nos níveis pares, B ganha, pois A retirou o último palito, e nos níveis ímpares o contrário é verdadeiro, após análise da árvore por completo, podemos perceber que qualquer jogada no início faz com que A (primeiro jogador) perca o jogo, a não ser que B cometa um erro.

É fácil provar esta hipótese, A deve tirar na primeira jogada no mínimo 1 palito, com isso, restam 4 palitos para B, caso B retire três na sua vez, A já está perdido, ou seja, independentemente do número de palitos que A retirar na primeira rodada, sempre sobra a possibilidade de B retirar uma quantia que faça com que reste apenas um palito para A, fazendo com que A perca o jogo.

b) Há como executar poda alfa-beta?

Sim. Neste jogo, temos apenas dois valores como possibilidade, +1 (vitória de A) ou –1 (vitória de B), já que não existe empate, portanto, quando estivermos em um nível de MAX, ao achar algum descendente com valor +1, não precisamos verificar mais os nós adjacentes, pois não teremos valores maiores; e quando estivermos em um nível de MIN, ao achar algum descendente com valor –1, também não precisamos verificar mais os nós adjacentes, pois não existirá valor menor.

Seguindo a explicação acima para fazer a poda alfa-beta na árvore minimax do jogo, conseguiríamos reduzir a árvore do seguinte modo:



Logo, conseguimos notar que teríamos um ganho de tempo aceitável com a aplicação da poda.

2) (0,3) Imagine um jogo de dois jogadores de perseguição e evasão. Assumamos agora que os jogadores se movem por vez. O jogo só termina quando os jogadores estão no mesmo nó; o resultado final para o perseguidor é menos o tempo total necessário (o evasor "ganha" por nunca perder).

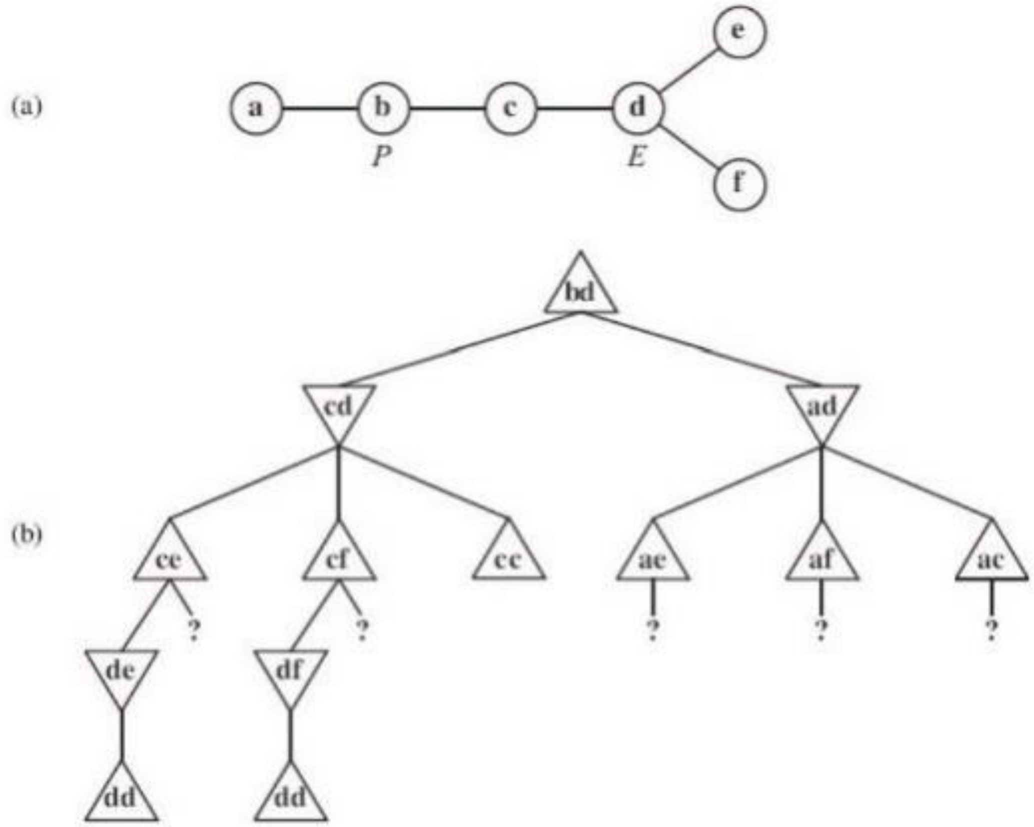
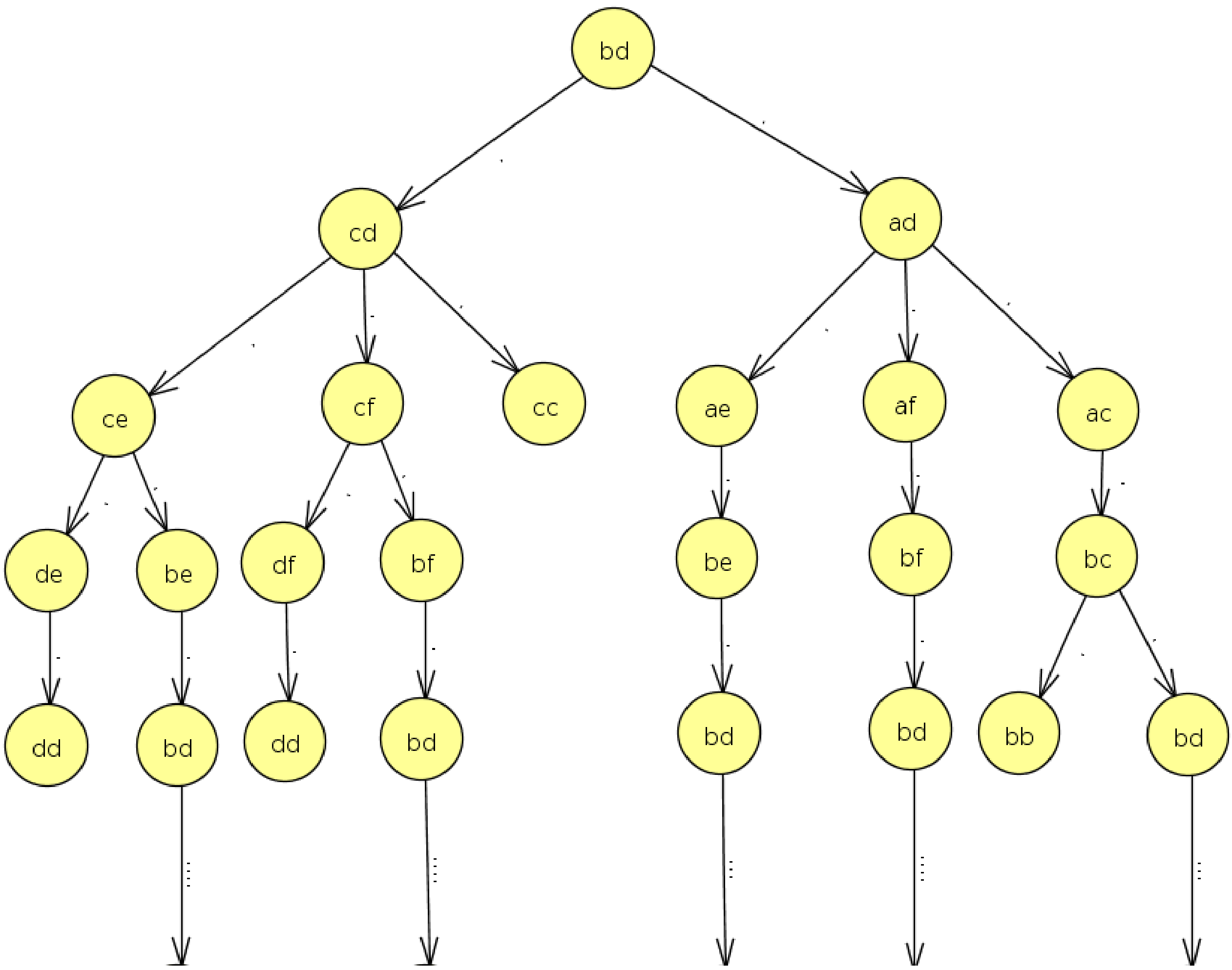


Fig. 1: (a) Um mapa onde o custo de cada aresta é 1. Inicialmente, o perseguidor P está no nó b e o evasor E está no nó d. (b) Uma árvore de jogo parcial para esse mapa. Cada nó é rotulado com as posições P e E. P move-se em primeiro lugar. Ainda não foram explorados os ramos marcados com "?".

a. Finalize a árvore do jogo.



b. Ao lado de cada nó interno, escreva o fato mais forte que você pode inferir sobre o seu valor (um número, uma ou mais desigualdades, tais como "≥14" ou um "?").

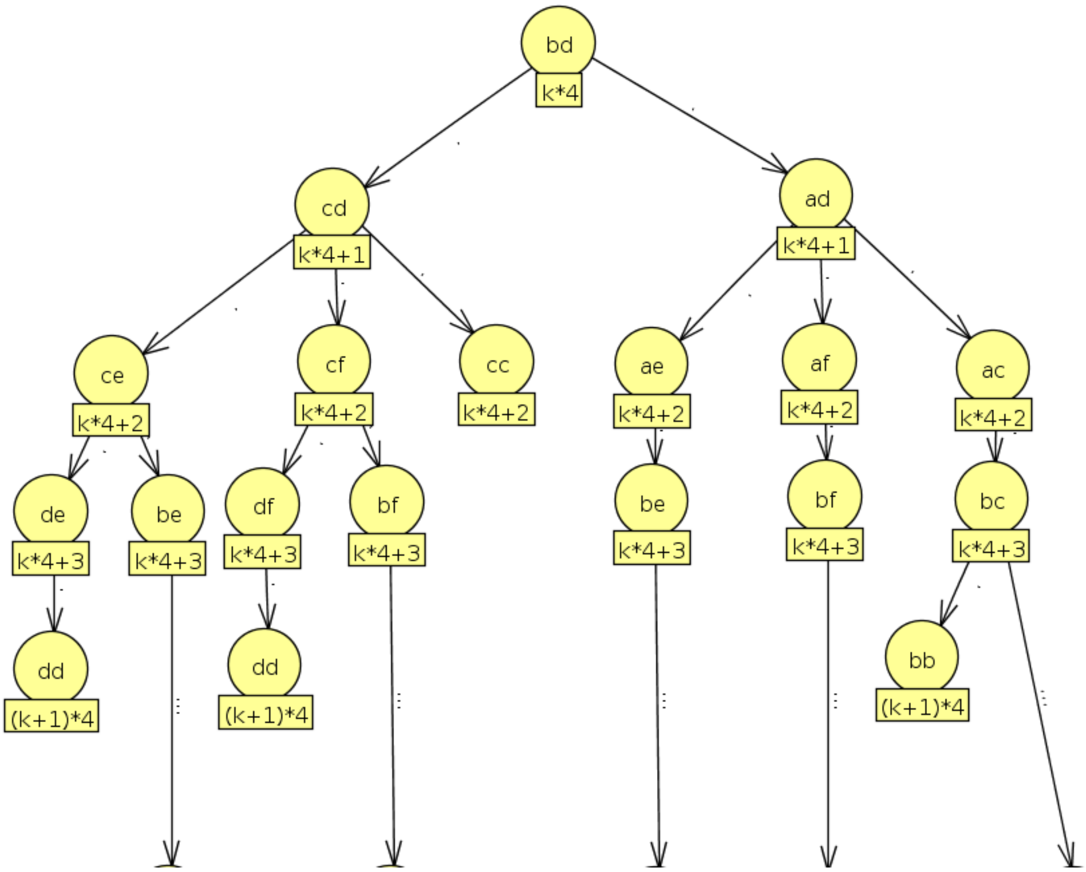
Conforme dito no enunciado do jogo, cada aresta tem peso 1, então podemos procurar inferir sobre cada um dos nós internos da árvore qual o custo para se chegar até ele.

Um aspecto importante de nossa árvore é que ela possui uma recursividade, ou seja, em determinado nó ao meio dela, estamos retornando ao nó inicial.

Se analisarmos atentamente, para retornarmos ao nó inicial em todos os pontos que permitem essa possibilidade, precisamos passar obrigatoriamente por 4 arestas.

Sendo assim, fixei uma constante K, inicialmente igual a zero, que irá incrementando a cada vez que retornarmos ao nó inicial (BD). Desta forma, em cada um dos níveis da árvore teremos os seguintes custos:

- Nível 0 (BD): 4K
- Nível 1 (CD, AD): 4K+1
- Nível 2 (CE, CF, CC, AE, AF, AC): 4K+2
- Nível 3 (DE, DF, BF, BE, BC): 4K+3
- Nível 4 (DD, BB): 4K+4 = 4(K+1)



Chegamos por fim na árvore acima, onde abaixo de cada um dos nós, podemos observar a função que nos leva ao seu custo, basta sabermos em qual iteração estamos.

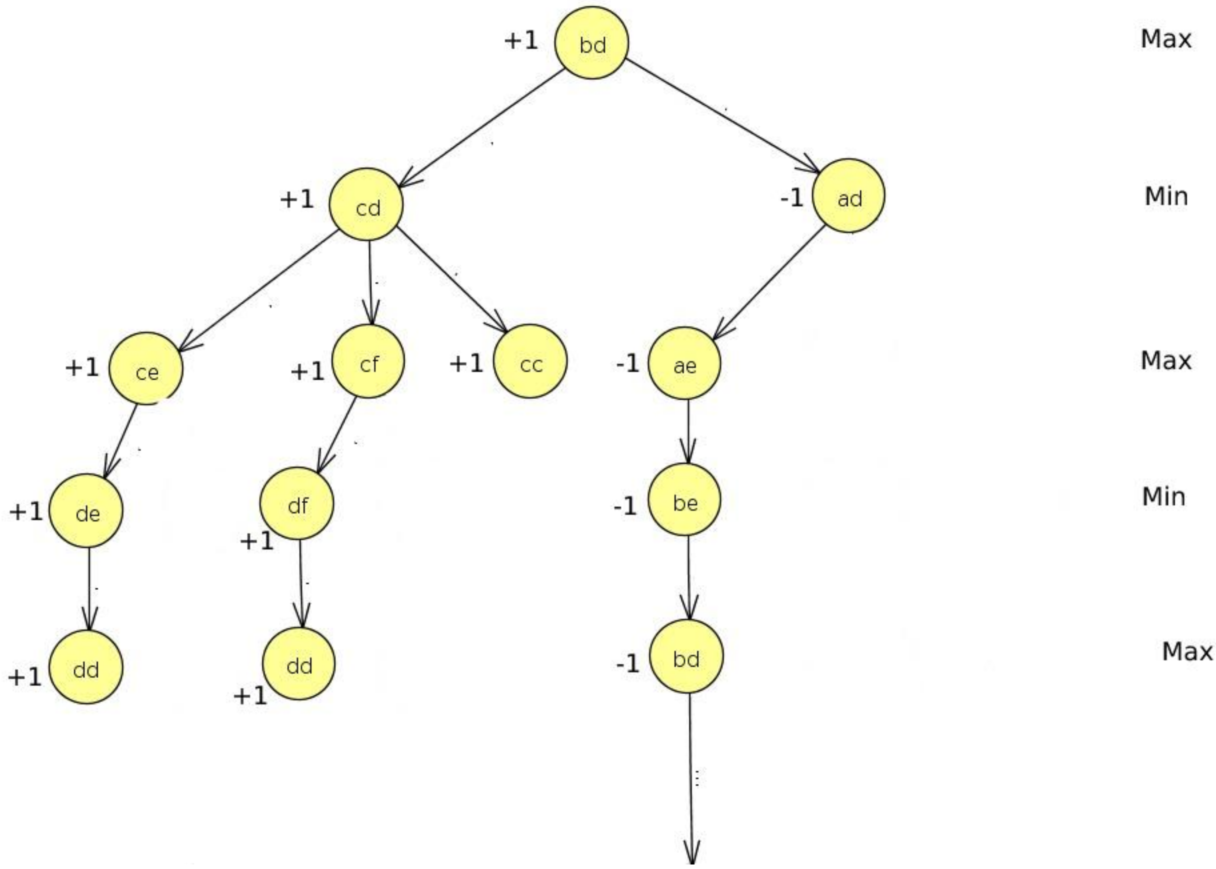
c. Aplique a poda alfa-beta na árvore.

Para aplicar a poda na árvore, utilizaremos uma metodologia similar a utilizada no exercício 1. Nossa função de avaliação será +1 para a vitória do perseguidor P, e –1 para a vitória do evasor E.

Assim como no exercício 1, o maior valor de Max que podemos alcançar é +1, e o menor valor de Min é -1, portanto, quando estivermos em um nível de Max, e um dos nós descendentes do nó que estamos possuir valor +1, não precisamos mais buscar adjacentes, pois já achamos o Max. O contrário é verdadeiro para o nível de Min; se um dos nós descendentes ao nó que estamos analisando for igual a –1, não precisamos mais verificar seus adjacentes pois achamos o valor de Min.

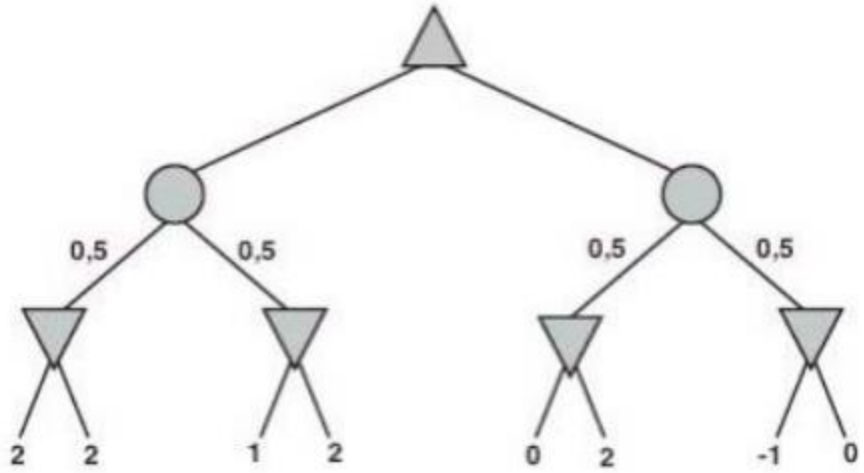
Outro fato importante é que estamos considerando o retorno ao nó inicial como vitória do Evasor, pois considera-se que entramos em um loop sem fim, ou seja, ele nunca é pego pelo perseguidor.

Assim, ao passarmos nossa árvore na função de avaliação partindo da esquerda para direita, de baixo para cima, temos a seguinte árvore com a poda alfabeta aplicada



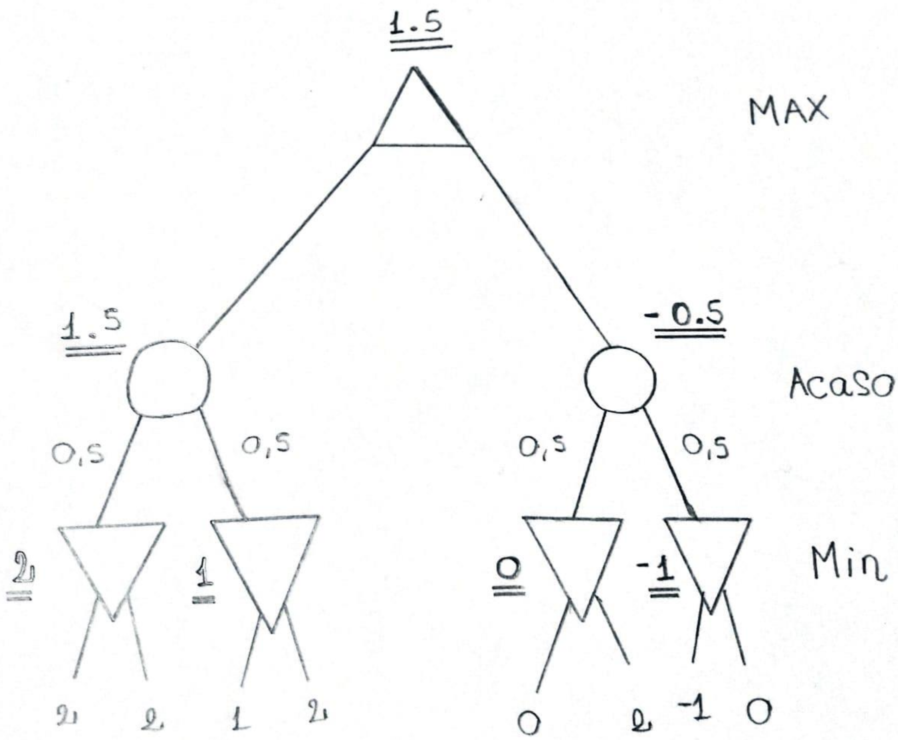
Deste modo, é fácil notar que, da perspectiva do perseguidor, o melhor movimento inicial para ele é ir para casa C, se aproximando do evasor; pois caso vá para casa A, está dando a chance do evasor entrar em um loop onde nunca é pego.

3) (0,2) A Figura abaixo mostra a árvore de jogo completa para um jogo trivial. Suponha que os nós folha estejam para ser avaliados na ordem da esquerda para a direita e que, antes de um nó folha ser avaliado, não sabemos nada sobre o seu valor — a faixa de valores possíveis é –∞ até ∞.



a. Copie a figura, marque o valor de todos os nós internos e indique a melhor jogada na raiz.

Os valores dos nós internos são da seguinte forma:



O jogo em questão é um jogo estocástico, onde podemos ter o fator sorte, portanto, além dos nós de Max, onde o jogador maximiza seu ganho; nós de Min, onde o jogador minimiza o ganho do oponente; temos os nós de acaso (representados pelo círculo), onde será calculada a soma dos valores de todos os resultados, ponderada pela probabilidade de cada uma das ações.

Sendo assim, para chegarmos até os valores da árvore, seguimos os seguintes passos:

- Primeiro, como estamos em um nó de Min, pegamos os menores valores dentre os nós folhas em cada caso, que foram, respectivamente: 2, 1, 0 e –1.
- Depois, aplicamos o nó de acaso, fazendo a soma da multiplicação dos valores encontrados pela probabilidade dada (0,5). Assim, obtivemos, respectivamente:
  - Somatório: 2*0,5 + 1*0,5 = 1,5
  - Somatório: 0*0,5 + (-1) *0,5 = -0.5
- Por fim, queremos o máximo dentre os valores achados, que em nosso caso, é o **1,5**.

Logo, podemos concluir que a melhor jogada na raiz é optarmos pela ação que tem valor 1,5; que está à esquerda.

b. Dados os valores das primeiras seis folhas, precisamos avaliar a sétima e oitava folhas? Dados os valores das sete primeiras folhas, precisamos avaliar a oitava folha?

- **Dados os valores das primeiras seis folhas, precisamos avaliar a sétima e oitava folhas?**

Sim. Isso ocorre pois o nó mais acima de nossos nós folhas, sem contar o nó de acaso, é um nó de Max, ou seja, queremos maximiza o ganho do jogador, até o momento, tendo percorrido seis folhas, descobrimos que, após aplicada a probabilidade, temos 1,5 no nó da ramificação esquerda; sendo assim, precisamos verificar a ramificação da direita para termos certeza de que lá não gera um nó com valor maior. Avaliando só as primeiras 6 folhas, não conseguimos garantir isto, pois se a sétima e oitava folha tiverem valor maior que 3, o valor da ramificação da direita seria maior do que 1,5, logo, seria escolhido pelo nó raiz.

- **Dados os valores das sete primeiras folhas, precisamos avaliar a oitava folha?**

Não. Pois a sétima folha possui valor –1, e estamos em um nó de Min, ou seja, -1 será escolhido a não ser que a oitava folha seja menor do que –1, e tendo –1, já sabemos que após aplicarmos o nó de acaso, nunca conseguiremos ultrapassar os 1,5 que encontramos do outro lado da árvore.

4) (0,2) Descreva as descrições de estados, geradores de movimentos, testes de término funções utilidade / funções de avaliação para um jogo a sua escolha (diferente dos vistos em aula como jogo da velha).

Jogo: Xadrez

Estados: O xadrez possui cerca de 10^{121} estados, sendo que, no décimo movimento, já temos 69.352.859.712.417 de jogos possíveis.

Testes de término: O jogo pode acabar de quatro formas:

- Quando um dos jogadores abandona;
- Quando um dos jogadores dá um xeque-mate, ou seja, o Rei não tem mais saída;
- Quando ocorre um empate, um exemplo é quando só sobram os dois reis no tabuleiro, ou os jogadores repetem o mesmo movimento três vezes;
- Quando ocorre o empate por afogamento, ou seja, o rei adversário não está em xeque, mas qualquer jogada que ele efetuar colocará ele nesta posição, o que não é permitido.

Funções utilidade: Podemos utilizar a seguinte representação para nossos possíveis estados terminais 1 para vitória, 0 para empate e -1 para derrota.

Funções de avaliação: Podemos fazer uma avaliação estática do jogo, uma avaliação proposta por Turing, que é baseada na ideia de atribuirmos valores a cada um dos tipos de peça existentes no jogo; após feita as atribuições, somamos as peças brancas e também as pretas, depois fazemos uma subtração para sabermos qual está em vantagem. Logo, a avaliação é baseada nesta ponderação de valores, de acordo com os seguintes critérios estabelecidos:

- Peão: Valor 1, porém, se estiver na terceira ou quarta linha, recebe mais 0,2.
- Cavalos e Bispos: Valor 3.
- Torre: Valor 5.
- Dama: Valor 9.

Somente esta avaliação não é capaz de nos ajudar a avaliar o jogo como um todo, por isso poderíamos utilizar o algoritmo de Minimax da seguinte forma:

- Geramos todos os estados sucessores de um outro utilizando a poda alfa-beta;
- Aplicamos a avaliação estática vista acima para cada um deles;
- Comparamos os valores;
- Transmitimos os valores para os nós mais acima;
- Quando chegarmos na raiz, teremos um movimento que maximiza o ganho das peças brancas e minimiza o ganho das peças pretas; ou vice-versa, depende de qual delas queremos maximizar.