.

CEFET/RJ
Inteligência Artificial (2017.2)
Professor: Eduardo Bezerra (ebezerra@cefet-rj.br)
Lista de exercícios 03

Créditos: essa lista de exercícios contém a tradução dos exercícios disponibilizados na disciplina CS188 - Artificial Intelligence, da Universidade de Berkeley. Os exercícios originais podem ser encontrados em `http://ai.berkeley.edu/section_handouts.html`.

1. No jogo de baralho denominado `micro-blackjack`, você repetidamente seleciona (compra) uma carta. Essa seleção é feita com reposição, i.e., após a seleção, você retorna a carta ao monte de cartas, de tal modo que ela pode ser selecionada novamente. É igualmente provável que a pontuação da carta selecionada seja 2, 3 ou 4. A cada jogada, você pode ou *Comprar* (`Draw`) ou *Parar* (`Stop`) se a pontuação total das cartas já compradas for menor do que 6. Caso contrário, você deve parar. Quando você parar, sua utilidade é igual a sua pontuação total (até 5), ou igual a zero, se você obtiver pontuação total igual ou superior a 6. Cada vez que você *Compra*, você não recebe utilidade alguma. Não há desconto (i.e., $\gamma = 1$).

   (a) Quais são os estados e as ações para este MDP?

   Solução
   O conjunto de estados S é formado pelas possíveis pontuações totais relativas às cartas selecionadas. Há também um estado terminal, que corresponde ao fim do jogo.

   $$S = \{0, 2, 3, 4, 5, \texttt{Done}\}$$

   São também aceitáveis respostas que incluam 1, 6, 7, 8, 9, um estado `Bust`, ou apenas "'a soma atual das cartas mais um estado terminal"'.)
   O conjunto de ações possíveis A é o seguinte:

   $$A = \{\texttt{Draw}, \texttt{Stop}\}$$

   (b) Quais são a função de transição e a função de recompensa para este MDP?

   Solução
   A função de transição pode ser definida em duas partes, uma para cada ação possível.

Em primeiro lugar, qualquer que seja o estado atual s, se o agente selecionar a ação `Stop`, o jogo termina (i.e., ocorre a transição para o estado terminal `Done`).

$$T(s, \texttt{Stop}, \texttt{Done}) = 1, \forall\, s \in S$$

$$T(s, \texttt{Stop}, s') = 0, \forall\, s \in S,\ s' \neq \texttt{Done}.$$

Para a ação `Draw`, temos que:

$$T(s, \texttt{Draw}, s') = \begin{cases} 1/3, & \text{se } s' - s \in \{2,3,4\} \\ 1/3, & \text{se } s = 2 \text{ e } s' = \texttt{Done} \\ 2/3, & \text{se } s = 3 \text{ e } s' = \texttt{Done} \\ 1, & \text{se } s \in \{4,5\} \text{ e } s' = \texttt{Done} \\ 0, & \text{caso contrário} \end{cases}$$

A função de recompensa $R(s,\, a,\, s')$ pode ser definida da seguinte forma:

$$\begin{aligned} R(s, \texttt{Stop}, s') &= \quad s, s \leq 5 \\ R(s,\, a,\, s') &= \quad 0,\ \text{caso contrário.} \end{aligned}$$

(c) Determine a política ideal (ótima) para este MDP.

Solução

Em geral, para encontrar a política ideal (ótima) para um MDP, usaríamos algum algoritmo, como o `Iteração de Valor` (*Value Iteration*) seguido pelo `Extração de Política` (*Policy Extraction*). No entanto, neste caso particular, é simples constatar que a política ótima é `Draw` se $s \leq 2$, e `Stop` em caso contrário.

$$\pi^*(s) = \begin{cases} \texttt{Draw}, & \text{se } s \leq 2 \\ \texttt{Stop}, & \text{caso contrário} \end{cases}$$

Por completude, são apresentados na tabela abaixo as etapas do algoritmo `Iteração de Valor` com base nos estados e funções de transição descritos acima. A política ótima é dada tomando o argmax em vez do max, na iteração final do algoritmo `Iteração de Valor`.

V	0	2	3	4	5	Done
V_0	0	0	0	0	0	0
V_1	0	2	3	4	5	0
V_2	3	3	3	4	5	0
V_3	10/3	3	3	4	5	0
Policy Extraction	$10/3_{Draw}$	3_{Draw}	3_{Stop}	4_{Stop}	5_{Stop}	0_{Stop}

(d) Qual é o menor número de rodadas (i.e., qual o valor de k) do algoritmo `Iteração de Valor` para o qual este MDP terá seus valores exatos (se você acha que o `Iteração de Valor` nunca irá convergir, declare isso).

Solução

O algoritmo `Iteração de Valor` converge para $k = 3$. Ver valores na tabela acima.

2. O Pacman está preso no seguinte labirinto 2×2 com um fantasma com fome!

Quando é sua vez de agir, o Pacman deve se mover um passo horizontal ou verticalmente para um quadrado vizinho. Quando é a vez do fantasma, ele também deve se mover um passo horizontalmente ou verticalmente. O fantasma e o Pacman alternam movimentos. Depois de cada movimento (pelo Fantasma ou pelo Pacman) se o Pacman e o fantasma ocupam o mesmo quadrado, o Pacman é comido e recebe a utilidade -100. Caso contrário, ele recebe uma utilidade igual a 1. O fantasma tenta minimizar a utilidade que o Pacman recebe. Suponha que o fantasma faz o primeiro movimento na configuração apresentada no labirinto acima. Por exemplo, com um fator de desconto de $\gamma = 1{,}0$, se o fantasma se move para baixo, então o Pacman se move para a esquerda, ganha uma recompensa de 1 após o movimento do fantasma e -100 após a sua movimentação, com uma utilidade total de -99. Note que não há garantias de que esse jogo termine.

(a) Considere um fator de desconto $\gamma = 0{,}5$, aplicado cada vez que o Pacman ou o fantasma se move. Qual é o valor minimax do jogo truncado após 2 movimentos do fantasma e 2 movimentos do Pacman? Dica: você não precisa construir a árvore minimax.

Solução

Com duas jogadas do PacMan e duas jogadas do fantasma, se jogar racionalmente, o PacMan recebe a seguinte sequência de recompensas: $(1, 1, 1, 1)$. Entretanto, é preciso aplicar o fator de amortização ($\gamma = 0{,}5$) para computar a recompensa total:

$$1 \times \gamma^0 + 1 \times \gamma^1 + 1 \times \gamma^2 + 1 \times \gamma^3 = 1 + 0.5 + 0.25 + 0.125 = 1.875$$

(b) Considere um fator de desconto $\gamma = 0{,}5$. Qual é o valor minimax do jogo completo (infinito)? Dica: você não precisa construir a árvore minimax.

Solução

Nesse caso, devemos computar o valor da soma de infinitas recompensas amortizadas que o PacMan recebe, quando age racionalmente:

$$\sum_{i=0}^{\infty} \left(1 \times \gamma^i = \gamma^0 + \gamma^1 + \gamma^2 + \gamma^3 + \gamma^4 + \ldots\right) = \frac{1}{1-\gamma}$$

A expressão acima corresponde à soma de uma progressão geométrica infinita em que a razão (γ neste caso) é menor do que 1. De fato, para $\gamma = 0{,}5$, temos que o valor total corresponde às infinitas recompensas amortizadas é igual a 2.

(c) Por que o algoritmo `Iteração de Valor` (*Value Iteration*) é superior ao `minimax` para resolver este jogo?

Solução

O algoritmo `Iteração de Valor` aproveita estados repetidos para computar com facilidade a política ótima. No caso do algoritmo `minimax`, mesmo uma árvore minimax truncada aumenta em tamanho exponencialmente à medida que a profundidade da pesquisa aumenta.

(d) Este jogo é semelhante a um MDP porque as recompensas são obtidas em cada passo de tempo. No entanto, é também um jogo adversarial envolvendo decisões alternadas de dois agentes.

Seja s um estado (por exemplo, a posição de Pacman e do fantasma), e considere que $A_P(s)$ seja o espaço de ações disponível para o Pacman no estado s. Da mesma forma, considere que $A_G(s)$ é o conjunto de ações disponíveis para o fantasma em s. Seja $N(s, a) = s'$ a função sucessora (dado um estado inicial s, esta função retorna o estado s' que resulta depois de tomar a ação a). Finalmente, considere que $R(s)$ denotar a utilidade recebida depois de passar para o estado s.

Escreva uma expressão para $P^*(s)$, o valor do jogo para o Pacman em função do estado corrente s (análogo às equações de Bellman). Use um fator de desconto $\gamma = 1{,}0$. Dica: sua resposta deve ser uma equação que inclui $P^*(s)$ no lado direito.

Solução

$$P^*(s) = \max_{a \in A_p(s)} R(N(s,a)) + \min_{a' \in A_C(N(s,a))} R(N(N(s,a),a')) + P^*(N(N(s,a),a'))$$

3. Considere um PDM sem descontos e composto por três estados, $(1, 2, 3)$, cujas recompensas são -1, -2 e 0, respectivamente. O estado 3 é o único estado terminal. Nos estados 1 e 2 há duas ações possíveis: a e b. O modelo de transição é o seguinte:

   - No estado 1, a ação a move o agente para o estado 2 com probabilidade 0,8 e faz com que o agente fique parado com probabilidade 0,2.
   - No estado 2, a ação a move o agente para o estado 1 com probabilidade 0,8 e faz com que o agente fique parado com probabilidade 0,2.
   - Tanto no estado 1 quanto no estado 2, ação b move o agente para o estado 3 com probabilidade 0,1 e faz com que o agente fique parado com probabilidade 0,9.

   Lembre-se de que o valor ótimo para um estado s, $V^*(s)$, é obtido pela equação de Bellman:

$$V^*(s) = \max_a \sum_{s'} T(s,a,s') \left(R(s,a,s') + \gamma\, V^*(s')\right)$$

   Lembre-se também de que essa equação é adaptada pelo algoritmo `Iteração de Valor` como uma equação de atualização para computar $V^*(s)$:

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} T(s,a,s') \left(R(s,a,s') + \gamma\, V_k(s')\right)$$

   (a) O que pode ser determinado qualitativamente sobre a política ótima nos estados 1 e 2? Ou seja, que ações um agente racional deve tomar nesses estados? Justifique sua resposta.

   (b) Aplique o algoritmo `Iteração de Valor` para determinar as utilidades dos estados 1 e 2 na primeira iteração, isto é, determine $V_1(1)$ e $V_1(2)$. Lembre-se de que a iniciação desse algoritmo é fazer com que $V_0(s) = 0$, $\forall s$.