SCE5880

Algoritmos de Estimação para Robótica Móvel

Processo de Decisão de Markov

Percepção e Ação

O objetivo final de um algoritmo de controle robótico é escolher as ações corretas para executar uma tarefa determinada

Problemas:

- **Incerteza na percepção**
- **Incerteza na ação**

Incerteza nas ações de um robô

Situações possíveis:

Ações determinísticas
X
Ações estocásticas

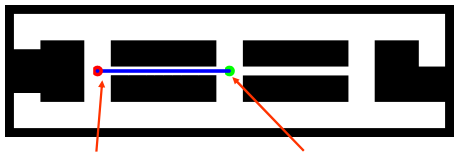
Ambiente parcialmente observável
X
Ambiente totalmente observável

Incerteza nas ações de um robô

- **Diversos resultados teóricos na robótica consideram determinístico o efeito das ações executadas**
- **Na maioria das situações reais, é insuficiente planejar uma seqüência de ações e executá-las "cegamente".**
- **É necessário considerar a incerteza quando se planeja uma seqüência de ações.**
- **Estratégia: antecipar a incerteza e planejar ações para qualquer situação possível**

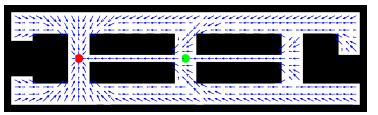
Situação ideal

Ações determinísticas e ambiente totalmente observável

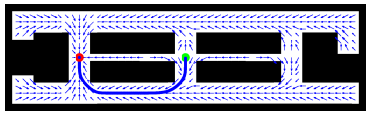


Plano global de ações

Ações determinísticas e ambiente totalmente observável

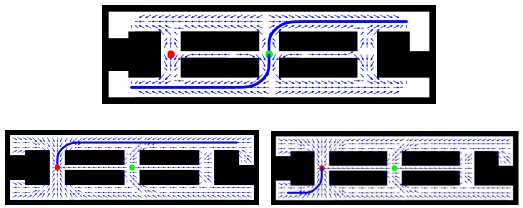


Ações estocásticas e ambiente totalmente observável

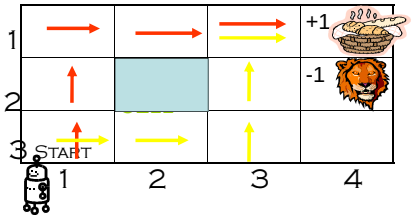


Plano global de ações

Ações estocásticas e ambiente parcialmente observável



Qual o melhor plano para chegar a recompensa?



Processo de decisão de Markov PDM

- **Dados:**
  - Estados x
  - Ações u
  - Probabilidades de transição $p(x'|u,x)$
  - Função de Recompensa $r(x,u)$
- **Obter:**
  - Plano $\pi(x)$ que maximiza a recompensa experada

D1

Recompensas e planos

- Plano (estado observável): $\pi: \; x_t \rightarrow u_t$
- Plano (caso geral): $\pi: \; z_{1t-1}, u_{1t-1} \rightarrow u_t$
- Recompensa experada: $R_T = E\left[\sum_{\tau=1}^{T} \gamma^{\tau} r_{t+\tau}\right]$
  - T=1: plano greedy
  - T>1: horizonte finito, y=1
  - T=infinity: horitzonte infinito, tipicamente y < 1

Recompensas e planos

- Plano ótimo para 1 passo:

$$\pi_1(x) = \underset{u}{\operatorname{argmax}} \; r(x,u)$$

- Função de valor para o melhor plano de 1 passo:

$$V_1(x) = \gamma \max_u r(x,u)$$

Plano de 2 etapas

- Plano ótimo:

$$\pi_2(x) = \underset{u}{\operatorname{argmax}} \left[r(x,u) + \int V_1(x') p(x'|u,x) dx' \right]$$

- Função de valor:

$$V_2(x) = \gamma \max_u \left[r(x,u) + \int V_1(x') p(x'|u,x) dx' \right]$$

Slide 9

D1 Denis; 03/06/2009

Plano de T etapas

- Plano ótimo:

$$\pi_T(x) = \underset{u}{\operatorname{argmax}} \left[r(x,u) + \int V_{T-1}(x') p(x'|u,x) dx' \right]$$

- Função de valor:

$$V_T(x) = \gamma \max_u \left[r(x,u) + \int V_{T-1}(x') p(x'|u,x) dx' \right]$$

Infinitas etapas

- Plano ótimo (equação de **Bellman**):

$$V_\infty(x) = \gamma \max_u \left[r(x,u) + \int V_\infty(x') p(x'|u,x) dx' \right]$$

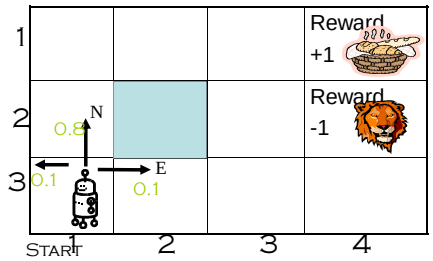
Value Iteration

- for all x do
  $\hat{V}(x) \leftarrow r_{\min}$
- endfor
- repeat until convergence
  - for all x do
    $\hat{V}(x) \leftarrow \gamma \max_u \left[r(x,u) + \int \hat{V}(x') p(x'|u,x) dx' \right]$
  - endfor
- endrepeat

$$\pi(x) = \underset{u}{\operatorname{argmax}} \left[r(x,u) + \int \hat{V}(x') p(x'|u,x) dx' \right]$$

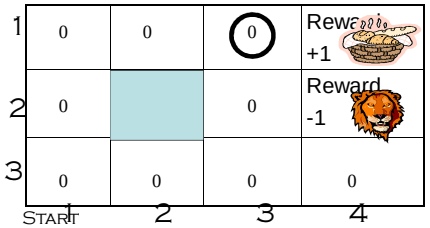
Exemplo: Qual o melhor plano de ação?

- O robô pode andar nas seguintes direções: N, E, S, W
- Cada ação tem um custo de –1/25

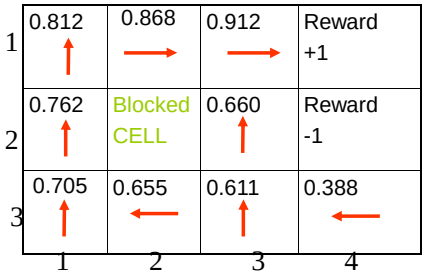


1ª Iteração: célula (3,1)

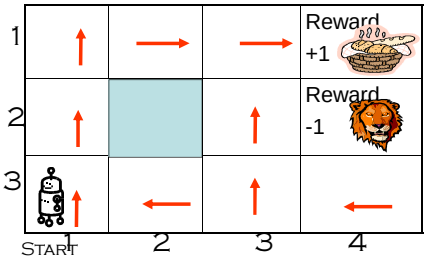
- East: -1/25 + [0.8 * 1 + 0.1 * 0 + 0.1 * 0] = 0.76
- North/South: -1/25 + [0.8 * 0 + 0.1 * 1 + 0.1 * 0] = 0.06
- West: -1/25
- Portanto, célula (3,1) assume o valor 0.76 (East)



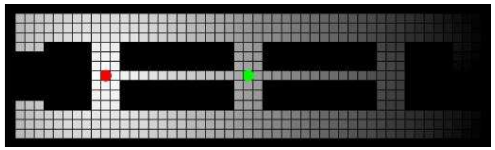
Resultado após convergência



Resultado após convergência

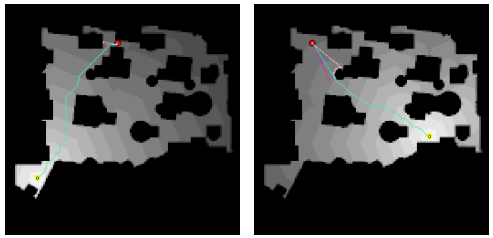


Value Iteration for Motion Planning



Exemplo de plano obtido utilizando Value Iteration

Value Iteration for Motion Planning



Processo de Decisão de Markov Parcialmente Observável - PDMPO

PDMPO

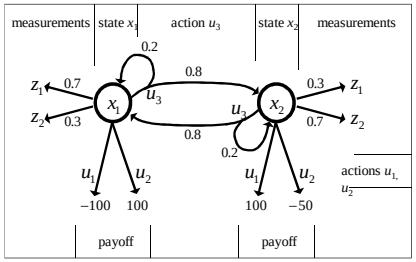
- **O princípio básico é o mesmo do PDM, porém o estado não é diretamente observável**
- **As decisões devem ser tomadas com base em uma distribuição probabilística (*belief* - *b*) do estado**
- **PDMPO:**

$$V_T(b) = \gamma \max_u \left[r(b,u) + \int V_{T-1}(b') p(b' \mid u, b)\, db' \right]$$

PDMPO

- **PDMPO é uma função de uma distribuição probabilística.**
- **PDMPO é um problema complexo, uma vez que distribuições probabilísticas são funções contínuas**
- **Para cenários finitos, com um número finito de ações, estados e horizontes, é possível representar as funções de valores (value functions) por segmentos de funções lineares.**

PDMPO - Exemplo



Parâmetros do exemplo

- As ações u_1 e u_2 são ações terminais.
- A ação u_3 é uma ação de percepção que potencialmente leva a uma mudança de estados.
- O horizonte é finito e γ=1.

$$
\begin{aligned}
r(x_1, u_1) &= -100 & r(x_2, u_1) &= +100 \\
r(x_1, u_2) &= +100 & r(x_2, u_2) &= -50 \\
r(x_1, u_3) &= -1 & r(x_2, u_3) &= -1
\end{aligned}
$$

$$
\begin{aligned}
p(x_1'|x_1, u_3) &= 0.2 & p(x_2'|x_1, u_3) &= 0.8 \\
p(x_1'|x_2, u_3) &= 0.8 & p(z_2'|x_2, u_3) &= 0.2
\end{aligned}
$$

$$
\begin{aligned}
p(z_1|x_1) &= 0.7 & p(z_2|x_1) &= 0.3 \\
p(z_1|x_2) &= 0.3 & p(z_2|x_2) &= 0.7
\end{aligned}
$$

Payoff in POMDPs

- Em um PDM, a recompensa depende do estado do sistema.
- Em um PDMPO, o estado do sistema não é conhecido.
- A recompensa esperada é calculada através da integração de todos os estados:

$$
\begin{aligned}
r(b, u) &= E_x[r(x, u)] \\
&= \int r(x, u) p(x)\, dx \\
&= p_1\, r(x_1, u) + p_2\, r(x_2, u)
\end{aligned}
$$

Recompensa - exemplo

- Se há 100% de certeza que o estado é x_1 então, quando executada a ação u_1, recebemos a recompensa de -100
- Se há 100% de certeza que o estado é x_2 então, quando executada a ação u_1, recebemos a recompensa de +100.
- Para valores intermediários, é calculada uma função linear:

$$
\begin{aligned}
r(b, u_1) &= -100\, p_1 + 100\, p_2 \\
&= -100\, p_1 + 100\, (1 - p_1)
\end{aligned}
$$

$$
r(b, u_2) = 100\, p_1 - 50\, (1 - p_1)
$$

$$
r(b, u_3) = -1
$$

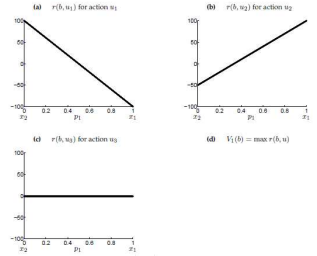
Função resultante

- A value function resultante $V_1(b)$ corresponde ao máximo das 3 funções a cada ponto.

$$
\begin{aligned}
V_1(b) &= \max_u\, r(b, u) \\
&= \max \left\{ \begin{array}{cc} -100\, p_1 & +100\, (1 - p_1) \\ 100\, p_1 & -50\, (1 - p_1) \\ & -1 \end{array} \right\}
\end{aligned}
$$

- A função linear é composta por segmentos de retas em um formato convexo.

Recompensa - exemplo



Plano para T=1

- Dado que o PDMPO tem horizonte finito T=1, utiliza-se $V_1(b)$ para determinar o melhor plano.
- No exemplo utilizado, o melhor plano para T=1 é:

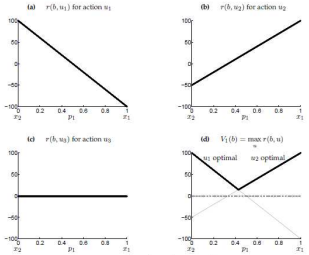
$$\pi_1(b) = \begin{cases} u_1 & \text{if } p_1 \leq \frac{3}{7} \\ u_2 & \text{if } p_1 > \frac{3}{7} \end{cases}$$

Pruning

- Ao analisar $V_1(b)$, pode-se notar que apenas os 2 primeiros componentes contribuem para a solução final.
- O terceiro componente pode ser eliminado (pruned) de $V_1(b)$ sem interferir no resultado final.

$$V_1(b) = \max \begin{Bmatrix} -100\, p_1 & +100\,(1-p_1) \\ 100\, p_1 & -50\,(1-p_1) \end{Bmatrix}$$

Recompensa - exemplo



Integrando a percepção

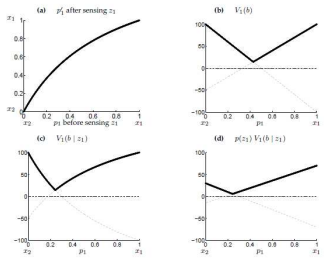
- Considerar que o robô pode fazer uma observção antes de decidir pela ação.
- $p'_1 = p(x_1 \mid z) = p(z \mid x_1).\ p(x_1) / p(z)$
- Supondo que o robô observa z_1 para o qual $p(z_1 \mid x_1)=0.7$ e $p(z_1 \mid x_2)=0.3$.
- Dada a observação z_1 o sistema é atualizado usando a regra de Bayes|.

$$p'_1 = \frac{0.7\, p_1}{p(z_1)}$$

$$p'_2 = \frac{0.3(1-p_1)}{p(z_1)}$$

$$p(z_1) = 0.7\, p_1 + 0.3(1-p_1) = 0.4\, p_1 + 0.3$$

PDMPO - Exemplo



Valor esperado depois da percepção

- Como não se sabe qual será o próxima observação, deve-se computar o belief esperado:

$$\begin{aligned} \bar{V}_1(b) &= E_z[V_1(b \mid z)] \\ &= \sum_{i=1}^{2} p(z_i)\, V_1(b \mid z_i) \\ &= \max \begin{Bmatrix} -70\, p_1 & +30\,(1-p_1) \\ 70\, p_1 & -15\,(1-p_1) \end{Bmatrix} \\ &\quad + \max \begin{Bmatrix} -30\, p_1 & +70\,(1-p_1) \\ 30\, p_1 & -35\,(1-p_1) \end{Bmatrix} \end{aligned}$$

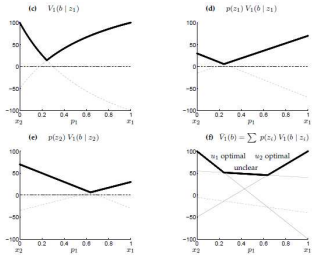
Valor esperado depois da percepção

- As 4 possibilidades consistem em:

$$\bar{V}_1(b) = \max \left\{ \begin{array}{cccc} -70\,p_1 & +30\,(1-p_1) & -30\,p_1 & +70\,(1-p_1) \\ -70\,p_1 & +30\,(1-p_1) & +30\,p_1 & -35\,(1-p_1) \\ +70\,p_1 & -15\,(1-p_1) & -30\,p_1 & +70\,(1-p_1) \\ +70\,p_1 & -15\,(1-p_1) & +30\,p_1 & -35\,(1-p_1) \end{array} \right\}$$

$$= \max \left\{ \begin{array}{cc} -100\,p_1 & +100\,(1-p_1) \\ +40\,p_1 & +55\,(1-p_1) \\ +100\,p_1 & -50\,(1-p_1) \end{array} \right\}$$

PDMPO - Exemplo



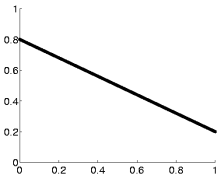
Transição de estados

- Quando a ação u_3 o estado potencialmente muda, o que deve ser levado em consideração na predição.

$$\begin{aligned} p_1' &= E_x[p(x_1 \mid x, u_3)] \\ &= \sum_{i=1}^{2} p(x_1 \mid x_i, u_3) p_i \\ &= 0.2p_1 + 0.8(1-p_1) \\ &= 0.8 - 0.6p_1 \end{aligned}$$

Transição de estados

$$\begin{aligned} p_1' &= E_x[p(x_1 \mid x, u_3)] \\ &= \sum_{i=1}^{2} p(x_1 \mid x_i, u_3) p_i \\ &= 0.2p_1 + 0.8(1-p_1) \\ &= 0.8 - 0.6p_1 \end{aligned}$$



Value Function depois de executar u_3

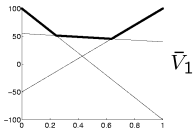
- Considerando a possível transição de estados obtemos:

$$\bar{V}_1(b) = \max \left\{ \begin{array}{cccc} -70\,p_1 & +30\,(1-p_1) & -30\,p_1 & +70\,(1-p_1) \\ -70\,p_1 & +30\,(1-p_1) & +30\,p_1 & -35\,(1-p_1) \\ +70\,p_1 & -15\,(1-p_1) & -30\,p_1 & +70\,(1-p_1) \\ +70\,p_1 & -15\,(1-p_1) & +30\,p_1 & -35\,(1-p_1) \end{array} \right\}$$

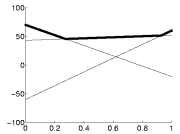
$$= \max \left\{ \begin{array}{cc} -100\,p_1 & +100\,(1-p_1) \\ +40\,p_1 & +55\,(1-p_1) \\ +100\,p_1 & -50\,(1-p_1) \end{array} \right\}$$

$$\bar{V}_1(b \mid u_3) = \max \left\{ \begin{array}{cc} 60\,p_1 & -60\,(1-p_1) \\ 52\,p_1 & +43\,(1-p_1) \\ -20\,p_1 & +70\,(1-p_1) \end{array} \right\}$$

Value Function depois de executar u_3



$\bar{V}_1(b)$



$\bar{V}_1(b \mid u_3)$

Porque Pruning é essencial

- Cada passo adiciona novos componentes lineares para V.
- Cada observação evela ao quadrado o número de componentes lineares.
- Exemplo: Uma função “un-pruned” com T=20 possui mais de $10^{547,864}$ componentes lineares.
- Com T=30, temos $10^{561,012,337}$ componentes lineares.
- A função pruned com T=20, contém ontem somente 12 componentes lineares.
- A explosão combinatória de componentes lineares torna os PDMPO impraticáveis em diversas aplicações.

PDMPO - Algoritmo

Algorithm POMDP(Γ):

$\Upsilon = (0, \ldots, 0)$
for $\tau = 1$ *to* T *do*
  $\Upsilon' = \emptyset$
  for all $(a', v_1^k, \ldots, v_N^k)$ *in* Υ *do*
    for all control actions u *do*
      for all measurements z *do*
        for $j = 1$ *to* N *do*

$$v_{j,a,z}^k = \sum_{i=1}^{N} v_i^k \; p(z \mid x_i) \, p(x_i \mid u, x_j)$$

        endfor
      endfor
    endfor
  endfor
  for all control actions u *do*
    for all $k(1), \ldots, k(M) = (1, \ldots, 1)$ *to* $(|\Upsilon|, \ldots, |\Upsilon|)$ *do*
      for $i = 1$ *to* N *do*

$$v_i' = \gamma \left[r(x_i, u) + \sum_z v_{u,x_i}^{k(z)} \right]$$

      endfor
      add $(u, v_1', \ldots, v_N')$ *to* Υ'
    endfor
  endfor
  optional: prune Υ'
  $\Upsilon = \Upsilon'$
endfor
return Υ

PDMPO - Sumário

- PDMPOs computam ações ótimas para ambientes estocásticos e parcialmente observáveis.
- Para problemas de horizonte finito, a função resultante é composta por segmentos de reta convexos..
- A cada iteração, o número de componentes lineares cresce exponencialmente.
- PDMPOs vem sendo aplicados para situações com poucos estados possíveis e poucas ações e observações possíveis.

Point-based Value Iteration

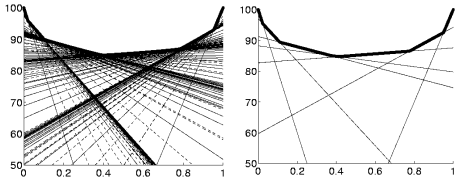
- Mantém apenas alguns possívelis estados válidos
- Somente considera situações que maximizem as funções de valor de pelo menos um desses estados

Ex: B={p1=0.0, p1=0.1, p1=0.2…, p1=1.0}

Para T=30 PDMPO = $10^{561,012,337}$ e PBVI = 120

Point-based Value Iteration

Value functions for T=30



PDMPO PBVI