Método Aproximado para Resolução de Processos Markovianos de Decisão

Luis França[1], **Solon Carvalho**[2], **Rita Rodrigues**[2]

[1]Programa de Doutorado em Computação Aplicada – CAP
Instituto Nacional de Pesquisas Espaciais – INPE

[2]Laboratório Associado de Computação e Matemática Aplicada – LAC
Instituto Nacional de Pesquisas Espaciais – INPE

lfamorim@gmail.com {solon,rita}@lac.inpe.br

Abstract. *Markov Decision Process (MDP) is a model for sequential decision problems under uncertainty, whose solution is an optimal policy that defines an action for each state in which the system may step in. For complex models, where the state space is large, the application of classical techniques for solving MPDs, and finding optimal policies, may be computationally infeasible. Several approaches have been developed to find near optimal policies efficiently using, for example, function approximation, online search methods and simulation. In this work, one approximate approach for solving large MDPs, called approximate dynamic programming, is described.*

Resumo. *Processo Markoviano de Decisão (PMD) é um modelo para problemas de decisão sequencial sob incerteza, cuja solução é uma política ótima que define uma ação para cada estado que o sistema possa assumir. Para modelos complexos, em que o espaço de estados é grande, a aplicação de técnicas clássicas para resolver PMDs, e encontrar políticas ótimas, pode ser computacionalmente inviável. Diversas abordagens vêm sendo desenvolvidas para encontrar políticas próximas às ótimas de maneira eficiente utilizando, por exemplo, aproximação de função, métodos de busca online e simulação. Neste trabalho, uma abordagem aproximada para resolver PMDs de grande porte, denominada programação dinâmica aproximada, é descrita.*

Palavras-chave: *Processo Markoviano de Decisão, Programação Dinâmica Aproximada, Simulação*

1. Introdução

O problema de se tomar uma sequência de decisões em um ambiente que apresenta incerteza surge em diversas áreas, como economia, inteligência artificial, teoria de controle e pesquisa operacional. Deve-se levar em conta que tais decisões apresentam consequências a curto e longo prazo, e o desafio é conseguir prever o impacto de cada uma delas mesmo que seus resultados muitas vezes não possam ser determinados com certeza.

Uma ferramenta comumente utilizada para modelar esse problema é o Processo Markoviano de Decisão (PMD), no qual o sistema tratado pode assumir um conjunto de estados que estão sujeito a transformações de origem estocástica, e deve-se escolher ações dentre as possíveis a fim de que a solução seja boa de acordo com algum critério. Tal solução é uma política ótima que define uma ação para cada um dos estados que o sistema possa

assumir. Há mais de 50 anos são estudados e empregados métodos capazes de determinar uma política ótima para um PMD, como os algoritmos de iteração de valores e iteração de políticas [Puterman 2005].

Aplicações reais modeladas por PMDs vêm crescendo em vários domínios, como gerenciamento de processos industriais, robótica, e telecomunicações ([Simão et al. 2009], [Mosharaf et al. 2003], [Tachibana et al. 2007]). Busca-se maneiras eficientes de solucionar o problema para sistemas cada vez mais complexos, cujo espaço de estados pode ser muito grande [Sigaud and Buffet 2010]. Para esses PMDs de grande porte, computar uma política ótima é geralmente impraticável.

Nesse contexto, novas abordagens vêm sendo desenvolvidas para desenvolver soluções próximas às ótimas em que não é preciso avaliar exaustivamente o espaço de estados e as ações. Para tanto, utiliza-se aproximações de função, métodos de busca especializados e simulação.

O intuito desde trabalho é apresentar uma técnica aproximada existente na literatura, denominada Programação Dinâmica Aproximada (PDA) [Powell 2007], que combina simulação e otimização para obtenção de boas soluções. Para aplicá-la, no entanto, é preciso se ter um conhecimento sobre a estrutura do problema tratado para conseguir capturar características que auxiliem na aproximação da função.

O restante do trabalho é organizado da seguinte forma: na seção 2 é apresentado o PMD bem como um dos métodos clássicos utilizado para resolvê-lo; na seção 3 é descrita a PDA e seus algoritmos básicos, a partir dos quais se desenvolvem a pesquisa nessa área; e, por fim, na seção 4 são feitas as considerações finais.

2. Processo Markoviano de Decisão

A Figura 1 ilustra um problema de decisão sequencial, no qual, em um determinado instante de tempo, o decisor observa o estado atual do sistema e escolhe uma ação entre um conjunto de ações possíveis para aquele estado. Essa ação gera uma recompensa (ou incorre em um custo) imediata, e pode fazer com que o sistema evolua para um novo estado em um determinado tempo. No próximo instante de decisão o processo se repete.

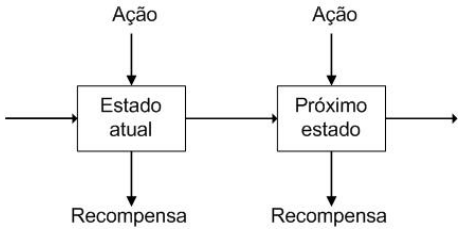

Figura 1. Representação de um problema de decisão sequencial (Adaptado de [Puterman 2005])

As consequências das ações disponíveis podem apresentar incerteza, ou seja, os efeitos das decisões tomadas podem não ser conhecidos antecipadamente de uma forma determinística. Dessa forma, tal problema passa ser denominado *problema de decisão sequencial sob incerteza*.

Uma solução para esse problema deve especificar qual ação deve ser escolhida em cada instante do processo de decisão para qualquer estado que o sistema possa alcançar. Busca-

se uma solução, denominada política, ótima que gere a máxima (mínimo) recompensa (custo) esperada.

Um modelo particular para o problema de tomada de decisão sequencial sob incerteza é o Processo Markoviano de Decisão, que vem sendo utilizado em várias áreas, dentre elas telecomunicações, processamento de sinais, inteligência artificial, gestão e economia [Hu and Yue 2007].

Em um PMD apenas as informações proporcionadas pelo estado e ação atuais são necessários para o cálculo das recompensas, conjunto de ações possíveis e probabilidades de transição do sistema. Busca-se então uma boa descrição de cada estado do sistema que respeite a propriedade de Markov, que garante a ausência de memória, ou seja, a possibilidade de se prever o "futuro" do processo dado o "presente", independentemente do "passado" [Tijms 2003].

Quando as decisões são tomadas apenas em um conjunto de pontos fixos no tempo pode-se modelar o sistema como um Processo Markoviano de Decisão a Tempo Discreto, descrito como uma quíntupla

$$\{T,\ S,\ A(i),\ p_t(j|i,a),\ r_t(i,a)\},$$

em que o conjunto de instantes de decisão é representado por $T = \{1, 2, ...\}$; S é o espaço de estados que o sistema pode assumir; $A(i)$ o conjunto de ações possíveis para o estado $i \in S$; $p_t(j|i,a)$ a probabilidade de transição do estado $i \in S$ para o estado $j \in S$ ao se escolher a ação $a \in A(i)$ no tempo t; e, por fim, $r_t(i,a)$ é a recompensa ao se tomar a ação $a \in A(i)$ no estado $i \in S$ no tempo t.

No instante t, ao se observar o sistema no estado i, uma ação a deve ser escolhida. Feito isso, tem-se:

1) uma recompensa $r_t(i,a)$;

2) uma probabilidade $p_t(j|i,a)$ de transição para o estado j em que:

$$\sum_{j \in S} p_t(j|i,a) = 1$$

Em algumas aplicações, o tempo entre os instantes de decisão pode não ser constante, mas sim aleatório com $T = (0, \infty]$. Nesse caso, o sistema pode ser modelado como um Processo Markoviano de Decisão a Tempo Contínuo.

2.1. Programação Dinâmica

Visto que o problema é sequencial, [Bellman 1957] sugeriu uma abordagem para resolvê-lo por meio de equações funcionais, conhecida como Programação Dinâmica. Define-se então uma função $V_t(i)$, conhecida como função de valor, que representa o valor máximo do retorno esperado no instante de decisão t até o final do horizonte de planejamento a partir de um estado inicial i. $V_t(i)$ pode ser escrita em função de $V_{t+1}(j)$, com $j \in S$, da seguinte maneira

$$V_t(i) = \max_{a \in A(i)} \left\{ r_t(i,a) + \sum_{j \in S} p_t(j|i,a) V_{t+1}(j) \right\}. \quad (1)$$

A Equação 1, conhecida como equação de Bellman, permite a obtenção do retorno máximo esperado para um horizonte finito de forma retrógrada no tempo, e, para isso, é inicializada com um conjunto de valores $V_T(i)$, com $i \in S$. Tais valores correspondem aos retornos recebidos no final do horizonte de planejamento em função do último estado i observado.

Se existem $|S|$ estados possíveis, existem $|S|$ equações de Bellman a serem resolvidas, uma para cada estado. A Figura 2 ilustra o algoritmo utilizado para resolução.

Algoritmo Programação Dinâmica
  Inicialize $V_T(i)$, $\forall\, i \in S$
  Para $t = T - 1$ **até** $t = 0$ **faça**
    Para cada $i \in S$ **faça**

$$V_t(i) = \max_{a \in A(i)} \left\{ r_t(i,a) + \sum_{j \in S} p_t(j|i,a) V_{t+1}(j) \right\}$$

$$a_t(i) = \operatorname*{argmax}_{a \in A(i)} \left\{ r_t(i,a) + \sum_{j \in S} p_t(j|i,a) V_{t+1}(j) \right\}$$

    Fim Para
  Fim Para
Fim Algoritmo

Figura 2. Algoritmo para PMD a horizonte finito

Para PMDs a horizonte infinito utiliza-se, tradicionalmente, três estratégias para a obtenção de uma política ótima: iteração de valores, iteração de políticas e programação linear.

Neste trabalho, apenas o algoritmo de iteração de valores com critério descontado é detalhado, visto que este é largamente utilizado e serve como base para a técnica abordada neste trabalho. No critério descontado, o fator de desconto γ, com $0 \leq \gamma < 1$, descreve a preferência em recompensas atuais sobre recompensas futuras. Quando mais próximo γ está de 0, menores os valores das recompensas no futuro.

2.2. Algoritmo de Iteração de Valores

O algoritmo de iteração de valores assemelha-se ao algoritmo apresentado na Figura 2. A diferença reside em se incrementar t iterativamente e calcular $V_t(i)$ até que um critério de convergência seja satisfeito. Seja

$$||V_t - V_{t-1}|| = \max_{i \in S} |V_t(i) - V_{t-1}(i)|,$$

segundo [Tijms 1995], para um t suficientemente grande, quando $||V_t - V_{t-1}||$ resulta em um valor menor que uma tolerância pré-estabelecida ϵ, um estado de equilíbrio é alcançado. Ao se alcançar tal equilíbrio, os valores encontrados são soluções para as equações de Bellman. Além disso, essas soluções são únicas, e a política obtida é ótima.

A Figura 3 mostra o algoritmo para o caso discreto, em que, para cada estado do sistema, calcula-se a função $V_t(i)$ e a ação $a_t(i)$ que a maximiza, com o intuito de se obter uma política ótima.

Algoritmo Iteração de Valores
Inicialize $V_0(i), \forall\, i \in S$
t = 1
Repita
Para cada $i \in S$ **faça**

$$V_t(i) = \max_{a \in A(i)} \left\{ r(i,a) + \gamma \sum_{j \in S} p(j|i,a) V_{t-1}(j) \right\}$$

$$a_t(i) = \operatorname*{argmax}_{a \in A(i)} \left\{ r(i,a) + \gamma \sum_{j \in S} p(j|i,a) V_{t-1}(j) \right\}$$

Fim Para
t = t + 1
Até que $||V_t - V_{t-1}|| < \epsilon$
Fim Algoritmo

Figura 3. Algoritmo de iteração de valores

3. Programação Dinâmica Aproximada

Para a maioria dos problemas reais, a quantidade de estados do sistema é tão grande que uma representação perfeita da função de valor para cada um destes torna-se impraticável. É necessário, então, utilizar técnicas que aproximem essa função e desenvolver métodos que façam uso dessas técnicas.

Para tanto avança-se no tempo, diferentemente da programação dinâmica, utilizando algoritmos iterativos para estimar a função de valor. Essa estratégia, de acordo com [Powell 2007], já foi desenvolvida sob os nomes de programação dinâmica adaptativa (*adaptive dynamic programming*), aprendizado por reforço (*reinforcement learning*) e programação neuro-dinâmica (*neuro-dynamic programming*). Um outro termo que vem sido adotado, neste trabalho inclusive, é programação dinâmica aproximada - PDA (*approximate dynamic programming*).

Ao se avançar no tempo, as funções de valor ainda não foram computadas, então surge a necessidade de se substituir a função de valor por uma aproximação estatística, dada por $\bar{V}_t(i)$. O desafio ao se utilizar a PDA é encontrar aproximações que sejam boas o suficiente para o problema tratado.

Existem diversas maneiras de se aproximar a função de valor. A mais simples é a chamada tabela de pesquisa (*lookup table*), na qual, para cada estado i, tem-se uma estimativa $\bar{V}_t(i)$. Estima-se $\bar{V}_t(i)$ iterativamente assumindo-se que existe uma aproximação inicial $\bar{V}_t^0$ para todo t. Seja $\bar{V}_t^{n-1}$ a aproximação da função de valor após $n-1$ iterações e considerando a n-ésima iteração, tem-se o seguinte problema de otimização

$$\bar{v}_t^n = \max_{a_t \in A(i_t^n)} \left\{ r(i_t^n, a_t) + \gamma \sum_{j \in S} p(j|i_t^n, a_t).\bar{V}_{t+1}^{n-1}(j) \right\}, \tag{2}$$

em que i_t^n é o estado atual no tempo t na iteração n, $A(i_t^n)$ o espaço de ações para o estado i_t^n, e $\bar{v}_t^n$ uma estimativa do valor de se estar nesse estado. A ação ótima a_t^n é determinada por

$$a_t^n = \operatorname*{argmax}_{a_t \in A(i_t^n)} \left\{ r(i_t^n, a_t) + \gamma \sum_{j \in S} p(j|i_t^n, a_t).\bar{V}_{t+1}^{n-1}(i) \right\}. \tag{3}$$

Ao se utilizar a tabela de pesquisa, a atualização da estimativa da função de valor do estado i_t é dada por

$$\bar{V}_t^n(i_t) = \bar{v}_t^n \ , \tag{4}$$

enquanto para os demais estados, a função de valor no tempo t é a mesma da iteração anterior.

Escolhida a ação pela Equação 3, deve-se conhecer a informação externa que chegará ao sistema entre os instantes t e $t+1$, a qual pode ser, por exemplo, a nova demanda por um determinado produto. Como no tempo t tal informação é desconhecida, a estratégia adotada é utilizar um processo artificial para escolher uma informação, ou observação, aleatória da população de informações possíveis, ou seja, uma simulação.

A partir da ação escolhida e da informação obtida pelo processo de simulação, pode-se computar o próximo estado i_{t+1}^n a ser visitado. Essa computação é feita pela função de transição

$$i_{t+1}^n = i^M(i_t^n, a_t^n, W_{t+1}) \ ,$$

em que W_{t+1} é um conjunto de variáveis aleatórias que representam a informação que chegou ao sistema entre t e $t+1$. Feito isso, repete-se o processo até que o horizonte de planejamento seja alcançado.

Observa-se que uma parte fundamental da PDA é a escolha de uma sequência de informações externas ao sistema, representada por ω. Desse modo, $W_t(\omega)$ representa uma informação externa escolhida aleatoriamente que chegou ao sistema no instante t seguindo a sequência ω. Em um algoritmo iterativo, na n-ésima iteração, escolhe-se uma sequência ω^n, e tem-se as informações $W_1(\omega^n), W_2(\omega^n), ..., W_T(\omega^n)$. Na próxima iteração escolhe-se um novo ω, e assim sucessivamente.

Um algoritmo básico de PDA é ilustrado pela Figura 4 . Nota-se que esse algoritmo itera N vezes, sendo que a cada iteração um ω é escolhido e encontram-se as melhores ações para os estados visitados pelo sistema de acordo com as informações geradas por ele.

Algoritmo Programação Dinâmica Aproximada
  Inicialize $\bar{V}_t^0(i_t)$, $\forall\, i \in S$, $\forall\, t \in \{0, 1, ..., T\}$
  Escolha um estado inicial i_0^1
  Para $n = 1$ **até** $n = N$ **faça**
    Escolha um ω^n
    Para $t = 0$ **até** $t = T - 1$ **faça**

$$\bar{v}_t^n = \max_{a_t \in A(i_t^n)} \left\{ r(i_t^n, a_t) + \gamma \sum_{j \in S} p(j|i_t^n, a_t) \bar{V}_{t+1}^{n-1}(j) \right\}$$

$$\text{Seja } a_t^n \text{ o valor de } a_t \text{ que maximiza } \bar{v}_t^n$$

$$V_t^n(i_t) = \begin{cases} \bar{v}_t^n & \text{se } i_t = i_t^n \\ \bar{V}_t^{n-1}(i_t) & \text{caso contrário} \end{cases}$$

$$\text{Compute } i_{t+1}^n = i^M(i_t^n, a_t^n, W_{t+1}(\omega^n))$$

    Fim Para
  Fim Para
Fim Algoritmo

Figura 4. Algoritmo básico de programação dinâmica aproximada

Não é preciso calcular a função de valor para todos os estados possíveis, problema encontrado na programação dinâmica, mas, de acordo com [Powell 2007], alguns novos desafios são introduzidos:

- Precisa-se de uma aproximação da função de valor dos estados visitados pelo sistema, e daqueles que possam ser visitado por meio desses estados iniciais no cálculo envolvendo a matriz de probabilidades de transição. Tal número pode ainda ser muito grande para se tratar computacionalmente;
- Apenas as funções de valor dos estados visitados pelo sistema são atualizadas. Assumindo que as aproximações iniciais tenham valor 0, deve-se encontrar uma maneira de estimar o valor de estados que não foram visitados pelo algoritmo;
- Pode-se cair numa situação em que estados que foram visitados apresentem uma função de valor alta, tendo mais preferência em futuras decisões; fazendo com que estados que possam ser bons para o sistema nunca sejam visitados.

A aproximação das funções de valor também é aplicada a problemas cujo horizonte de planejamento é infinito. Tais modelos tendem a ser computacionalmente mais fáceis, visto que não é preciso aproximar funções de valor em cada instante de tempo, reduzindo o número de parâmetros a serem estimados. A Figura 5 ilustra um algoritmo básico de PDA para horizonte infinito.

Algoritmo Programação Dinâmica Aproximada
Inicialize $\bar{V}^0(i), \forall\, i \in S$
Escolha um estado inicial i^1
Para $n = 1$ **até** $n = N$ **faça**

$$\bar{v}^n = \max_{a \in A(i^n)} \left\{ r(i^n, a) + \gamma \sum_{j \in S} p(j|i^n, a).\bar{V}^{n-1}(j) \right\}$$

Seja a^n o valor de a que maximiza $\bar{v}^n$

$$V^n(i) = \begin{cases} \bar{v}^n & \text{se } i = i^n \\ \bar{V}^{n-1}(i) & \text{caso contrário} \end{cases}$$

Escolha uma informação ω^{n+1}

Compute $i^{n+1} = i^M(i^n, a^n, W(\omega^{n+1}))$

Fim Para
Fim Algoritmo

Figura 5. Algoritmo básico de programação dinâmica aproximada para um problema a horizonte infinito

Nota-se que, diferentemente dos algoritmos para horizonte finito, ω^n é uma realização da informação que chega em um único período, e não uma sequência de informações.

4. Considerações Finais

Neste trabalho foi descrito um método existente na literatura cujo objetivo é tornar viável a resolução de PMDs de grande porte. Parte do poder da PDA está no fato que ela funciona como uma simulação clássica com regras de decisão mais inteligentes, já que nos seus algoritmos tem-se dois passos: um passo de otimização, onde se escolhe a melhor a ação, e um passo de simulação, onde são capturados os efeitos de informações aleatórias.

Após o desenvolvimento de um algoritmo de PDA é necessário testá-lo para verificar se as soluções são boas para o problema modelado. Neste caso, duas estratégias são geralmente adotadas: comparar com a solução exata de um PMD, quando for possível resolver uma instância pequena do problema de maneira exata; ou comparar com um problema determinístico, pois muitas vezes o problema possui uma formulação determinística que seja significativa para permitir, ao menos, uma comparação com uma solução aproximada.

Ressalta-se que os algoritmos apresentados nas Figuras 4 e 5 são genéricos, e a partir destes são desenvolvidos os algoritmos específicos para cada aplicação dos PMDs. Parte desse desenvolvimento envolve a escolha do método de representação da função de valor a ser utilizado, assim como a técnica de simulação empregada.

Como trabalho futuro pretende-se desenvolver um algoritmo de PDA para resolver PMDs aplicados na área de telecomunicações, mais especificamente em problemas de otimização em redes ópticas.

Referências

Bellman, R. (1957). *Dynamic Programming*. Princeton University Press, Princeton (NJ), 1 edition.

Guo, X. and Hernández-Lerma, O. (2009). *Continuous-time Markov decision processes*. Springer, Berlin, 1 edition.

Hu, Q. and Yue, W. (2007). *Markov decision processes with their applications*. Springer, New York, 1 edition.

Mosharaf, K., Talim, J., and Lambadaris, I. (2003). A markov decision process model for dynamic wavelength allocation in wdm networks. In *Proceedings...*, volume 5, pages 2590–2594, San Francisco. IEEE Global Telecommunications Conference, IEEE Communications Society.

Powell, W. B. (2007). *Approximate Dynamic Programming: Solving the Curses of Dimensionality (Wiley Series in Probability and Statistics)*. Wiley-Interscience, Hoboken, NJ.

Puterman, M. L. (2005). *Markov decision processes*. Wiley-Interscience, Hoboken, NJ.

Sigaud, O. and Buffet, O. (2010). *Markov Decision Processes in Artificial Intelligence*. Wiley-IEEE Press, Hoboken, NJ, 1 edition.

Simão, H. P., Day, J., George, A. P., Gifford, T., Nienow, J., and Powell, W. B. (2009). An approximate dynamic programming algorithm for large-scale fleet management: A case application. *Transportation Science*, 43(2):178–197.

Tachibana, T., Kasaharat, S., and Sugimoto, K. (2007). Dynamic lightpath establishment for service differentiation based on optimal mdp policy in all-optical networks with wavelength conversion. In *Proceedings...*, pages 2424–2429, Glasgow. IEEE International Conference on Communications, IEEE Communications Society.

Tijms, H. C. (1995). *Stochastic models*. John Wiley & Sons, Chichester (UK).

Tijms, H. C. (2003). *A first course in stochastic models*. Wiley, Chichester (UK), 2 edition.