Universidade Federal de Viçosa
Centro de Ciências Exatas e Tecnológicas
Departamento de Informática

Métodos Monte Carlo
Levi H. S. Lelis
levi.lelis@ufv.br

Métodos Monte Carlo

Na aula passada vimos métodos que requerem conhecimento da dinâmica do processo de decisão de Markov para resolve-lo. Nessa aula iremos estudar métodos que aprendem com a experiência do agente no ambiente. Ou seja, não é necessário ter conhecimento da função de transição $p(s', r|s, a)$.

Mesmo se for possível obter a função $p(s', r|s, a)$, muitas vezes o seu cálculo não é trivial. Por exemplo, qual a probabilidade de aparecer um Rei de Ouros na próxima rodada de uma mesa de pôquer? Um programa de computador pode realizar o cálculo, mas a sua implementação poder ser trabalhosa e sujeita a erros.

Previsão com Monte Carlo

Dada uma política π, podemos aproximar os valores de V através de simulações de episódios com π:

$$S_0, A_0, R_1, S_1, A_1, R_2, \cdots, S_{T-1}, A_{T-1}, R_T\,.$$

O valor de $V(s_t)$ é a média dos valores de recompensa acumulada G_t obtida após estado s ser visitado.

Existem duas possíveis implementações dessa ideia:

- Primeira visita: a média é calculada para os valores G_t da primeira visita a um estado s em um episódio. Iremos focar nessa implementação nessa aula.
- Todas visitas: a média é calculada para todos os valores G_t de todas as visitas a um estado em um episódio. Essa ideia é útil quando utilizando **traços de elegibilidade**.

O algoritmo abaixo mostra o pseudocódigo de um método MC para previsão utilizando a estratégia de "primeira visita". O código abaixo assume que o agente interage infinitamente com o ambiente, obtendo episódios que são utilizados para estimar $V(s)$.

```
1  def Primeira-Visita MC Previsão(π):
2    inicialize V(s) de forma arbitrária.
3    (V(s) = 0 para estados terminais)
4    inicializa Retorno[s] = [] para todos s
5    while True:
6      gera um episódio com π: S_0, A_0, R_1, S_1, A_1, R_2, ..., S_{T-1}, A_{T-1}, R_T
7      G = 0
8      for t in range(T-1, 0):
9        G = γG + R_{t+1}
10       if S_t not in {S_0, S_1, ..., S_{t-1}}:
11         Retorno[S_t].append(G)
12         V(S_t) =média(Retorno[S_t])
```

Implementação Incremental

A implementação acima não é eficiente em termos de tempo e memória, pois requer o armazenamento de todos os valores de G para cada estado. Além disso, o algoritmo calcula a média desses valores de G toda vez que um novo valor é adicionado à lista. Uma forma de solucionar esse problema é fazendo uma implementação incremental das médias.

$$\begin{aligned} V_{k+1} &= \frac{1}{k}\sum_{i=1}^{k} G_i \\ &= \frac{1}{k}\left(G_k + \sum_{i=1}^{k-1} G_i\right) \\ &= \frac{1}{k}\left(G_k + (k-1)\frac{1}{(k-1)}\sum_{i=1}^{k-1} G_i\right) \\ &= \frac{1}{k}\left(G_k + (k-1)V_k\right) \\ &= \frac{1}{k}\left(G_k + kV_k - V_k\right) \\ &= V_k + \frac{1}{k}\left(G_k - V_k\right) \end{aligned}$$

Assim, podemos utilizar a última equação acima no lugar das linhas 11 e 12 do pseudocódigo.

Monte Carlo para Estimar Valores de Ações

Se um modelo do ambiente não está disponível, então estimar os valores de pares estado-ação é muito mais útil que estimar os valores de estados. Nos algoritmos de programação dinâmica, como o Iteração de Valor, uma vez obtida uma estimativa dos valores de V, uma política pode ser extraída através do processo de *lookahead*. Na ausência de um modelo, o *lookahead* não é possível.

O procedimento para estimar os valores $q_\pi(s, a)$ é o mesmo discutido acima. A diferença é que as médias são calculadas para valores G_t obtidos após visitar estado s e tomar ação a.

O problema é que, dependendo de π, alguns pares estado-ação nunca serão visitados e seus valores não serão aproximados corretamente. Uma forma de lidar com esse problema é através dos **inícios exploratórios**: todo par estado-ação pode ser escolhido como início de um episódio com probabilidade maior que zero.

Monte Carlo para Controle

Agora que já sabemos como utilizar o método Monte Carlo para estimar os valores de $q_\pi(s, a)$, podemos utilizar o método de iteração de política para encontrar uma política ótima π_*. A ideia é alternar avaliação de uma política inicialmente arbitrária e a melhoria gulosa da mesma. A melhoria gulosa com valores de q são feitos da seguinte forma:

$$\pi(s) = \arg\max_a q(s, a)\,,$$

o que não requer o uso de um modelo do problema.

Intuitivamente, o algoritmo acima não pode convergir para uma política sub-ótima. Se ele convergisse, então o valor de V iria convergir para o valor correto daquela política, que faria a política mudar. No entanto, ainda não existe nenhuma prova formal de que o algoritmo converge para uma política ótima.

```
def Controle MC com Início Exploratório(π):
  inicialize π(s) de forma arbitrária
  inicialize Q(s,a) de forma arbitrária.
  inicialize N(s,a) = 0
  while True:
    escolha S_0 e A_0 aleatoriamente de forma que todos os pares tenham
        probabilidade maior que zero.
    gera um episódio a partir de S_0, A_0 seguindo π:
        S_0, A_0, R_1, S_1, A_1, R_2, ··· , S_{T-1}, A_{T-1}, R_T
    G = 0
    for t in range(T-1, 0):
      G = γG + R_{t+1}
      if S_t not in {S_0, S_1, ··· , S_{t-1}}:
        N(S_t, A_t) = N(S_t, A_t) + 1
        Q(S_t, A_t) = Q(S_t, A_t) + 1/N(S_t, A_t) (G − Q(S_t, A_t))
        π(S_t) = arg max_a Q(S_t, a)
```

Monte Carlo para Controle sem Inícios Exploratórios

Os início exploratórios nos permitem avaliar os valores de pares estado-ação. No entanto, é pouco provável que os inícios exploratórios podem ser implementados no mundo real.

Uma forma de dispensar o uso dos inícios exploratórios é derivando uma política que assinala uma probabilidade, mesmo que pequena, de executar todas as ações. Uma dessas políticas é a ϵ-gulosa, que em um estado s,

- executa a melhor ação a com probabilidade $1 - \epsilon + \frac{\epsilon}{|A(s)|}$
- executa uma ação aleatória com probabilidade $\frac{\epsilon}{|A(s)|}$

O algoritmo acima converge para a melhor política ϵ-gulosa, que não necessariamente é ótima, mas é próxima de ótima.

Previsão com Monte Carlo Off-Policy

Uma outra alternativa para remover o requisito de inícios exploratórios é aproximar os valores de uma **política alvo** π agindo como uma **política de comportamento** b. Assim, b pode ser uma política ϵ-gulosa, que garante que todos os estados e ações são visitados, e π pode ser uma política determinística.

O requisito é que se π executa uma ação com probabilidade maior que zero, b também tem que executar a mesma ação com probabilidade maior que zero. Do contrário não seria possível aprender os valores de v_π através de episódios gerados por b, pois tais ações nunca seriam experimentadas.

Amostragem por Importância

Em estatística, **amostragem por importância** nos permite amostrar valores de uma distribuição para aproximar o valor de uma outra distribuição. A ideia é amostrar valores de $v_c(s)$ para estimar $v_\pi(s)$.

```
def Controle MC On-Policy (ϵ):
  inicialize π(s) com uma política ϵ-gulosa arbitrária
  inicialize Q(s,a) de forma arbitrária.
  inicialize N(s,a) = 0
  while True:
    gera um episódio com π: S_0, A_0, R_1, S_1, A_1, R_2, ··· , S_{T-1}, A_{T-1}, R_T
    G = 0
    for t in range(T-1, 0):
      G = γG + R_{t+1}
      if t not in {S_0, S_1, ··· , S_{t-1}}:
        N(S_t, A_t) = N(S_t, A_t) + 1
        Q(S_t, A_t) = Q(S_t, A_t) + 1/N(S_t, A_t) (G − Q(S_t, A_t))
        A* = arg max_a Q(S_t, a)
        for a in A(S_t):
          if a == A*:
            π(a|S_t) = 1 − ϵ + ϵ/|A(S_t)|
          else:
            π(a|S_t) = ϵ/|A(S_t)|
```

De uma forma geral temos:

$$\begin{aligned}\mathbb{E}_{X\sim P}[f(X)] &= \sum P(X)f(X) \\ &= \sum Q(x)\frac{P(X)}{Q(X)}f(X) \\ &= \mathbb{E}_{X\sim Q}\left[\frac{P(X)}{Q(X)}f(X)\right]\end{aligned}$$

Para aplicarmos a amostragem por importância no problema de controle precisamos calcular o valor de probabilidade de observarmos um episódio dada a política π e dada a política b.

$$\begin{aligned}Pr\{A_t, S_{t+1}, A_{t+1}, \cdots, S_T|S_t, A_{t:T-1} \sim \pi\} &= \pi(A_t|S_t)p(S_{t+1}|S_t, A_t)\pi(A_{t+1}|S_{t+1})\cdots p(S_T|S_{T-1}, A_{T-1}) \\ &= \prod_{k=1}^{T-1}\pi(A_k|S_k)p(S_{k+1}|S_k, A_k)\end{aligned}$$

A fração de amostragem por importância é dada por:

$$\rho_{t:T-1} = \frac{\prod_{k=1}^{T-1}\pi(A_k|S_k)p(S_{k+1}|S_k, A_k)}{\prod_{k=1}^{T-1}b(A_k|S_k)p(S_{k+1}|S_k, A_k)} = \frac{\prod_{k=1}^{T-1}\pi(A_k|S_k)}{\prod_{k=1}^{T-1}b(A_k|S_k)}$$

Embora o episódio dependa da dinâmica do ambiente, como o ambiente é idêntico, os valores de p se cancelam e a fração ρ depende apenas de π e b.

Controle com Monte Carlo Off-Policy

O algoritmo de de controle de Monte Carlo Off-Policy abaixo aprende sobre uma política de comportamento b enquanto deriva uma política ótima π_*. A política b tem que ser do estilo ϵ-gulosa, para garantir que todos os estados e ações são amostrados um número infinito de vezes, para garantir convergência.

```
def MC Previsão Off-Policy(π):
  inicialize Q(s,a) de forma arbitrária.
  while True:
    b = política ε-gulosa arbitrária
    gera um episódio utilizando b: S_0, A_0, R_1, S_1, A_1, R_2, ··· , S_{T-1}, A_{T-1}, R_T
    G = 0
    W = 1
    for t in range(T-1, 0):
      G = γG + R_{t+1}
      if S_t not in {S_0, S_1, ··· , S_{t-1}}:
        N(S_t, A_t) = N(S_t, A_t) + 1
        Q(S_t, A_t) = Q(S_t, A_t) + 1/N(S_t, A_t) (WG − Q(S_t, A_t))
        W = W π(A_t|S_t)/b(A_t|S_t)
```

```
def MC Controle Off-Policy(γ):
  inicialize Q(s,a) de forma arbitrária.
  inicialize π de forma arbitrária
  π(S_t) = arg max_a Q(S_t, a)
  while True:
    b = política ε-gulosa arbitrária
    gera um episódio utilizando b: S_0, A_0, R_1, S_1, A_1, R_2, ··· , S_{T-1}, A_{T-1}, R_T
    G = 0
    W = 1
    for t in range(T-1, 0):
      G = γG + R_{t+1}
      if S_t not in {S_0, S_1, ··· , S_{t-1}}:
        N(S_t, A_t) = N(S_t, A_t) + 1
        Q(S_t, A_t) = Q(S_t, A_t) + 1/N(S_t, A_t) (WG − V(S_t))
        W = W π(A_t|S_t)/b(A_t|S_t)
        π(S_t) = arg max_a Q(S_t, a)
```

O método de Monte Carlo com amostragem por importância é não enviesado, no sentido estatístico de que ele converge para o valor exato de q_π da política π sendo avaliada. No entanto, a variância da abordagem pode ser muito grande, o que faz com que seja necessário um número muito grande de episódios para convergir para o valor de q_π. Veremos nas próximas aulas um outro método que é enviesado mas possui menor variância.

Vantagens dos métodos Monte Carlo com relação aos métodos de programação dinâmica.

1. Não é necessário um um modelo, o agente aprende através de interações diretas com o ambiente.
2. O agente aprende através de simulações, as equações de transição do problema não são necessárias.
3. É possível focar em estados que realmente interessam o agente.