Atividade 9 **Yan A. S. Duarte**
PEL 208 Tópicos Especiais em Aprendizagem
Prof. Reinaldo A. C. Bianchi Entrega: 20/12/2017

Introdução

Esse relatório tem como objetivo detalhar a teoria, a implementação, os resultados e a conclusão da sétima atividade do curso. A propósta do exercício é implementar aprendizado por reforço utilizando os algoritmos sarsa e q-learning no problema de small grid world.

Teoria

Aprendizado por reforço

O aprendizado por reforço é um tipo de aprendizado de máquina inspirado na psicologia comportamental. É utilizado em problemas de otimização e controle, quando não se conhece o modelo do problema e também quando se pode treinar com testes e erros. Seu objetivo é aprender como um determinando agente autônomo deve-se comportar em um ambiente, tentando sempre executar as melhores ações possíveis a fim de atingir um objetivo. Os passos do aprendizado por reforço são:

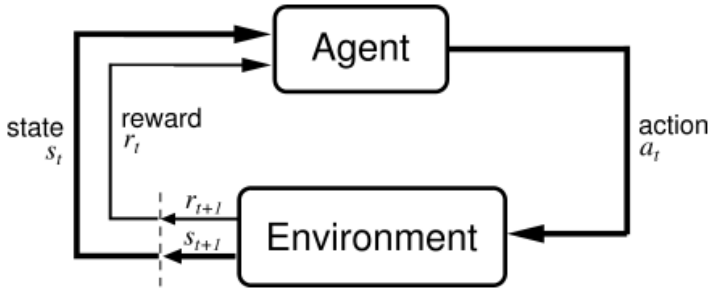

Figura 1: Interface de agente e ambiente.

- O agente e o ambiente interagem em passos de tempo discreto: $t = 0, 1, 2..$
- Agente observa o estado em que se encontra: $s_t \in S$
- Executa uma possíveis ação no momento t: $a_t \in A(s_t)$
- Recebe uma recompensa após executar a ação: $r_{t+1} \in \mathbb{R}$
- O estado em que o agente vai após tomar uma ação: s_{t+1}

A função que mapeia cada estados do ambiente em relação as ações que o agente irá tomar é definida como a política do agente π. Essa política deve escolher tomar ações que maximizem o valor final da soma das recompensas recebidas em um intervalo de tempo. A maneira que a política de comportamento é obtida se dá através de um processo de tentativa e erro, guiado por diferentes algoritmos.

Geralmente o sistema do problema é não-determinístico, ou seja, uma mesma ação tomada a partir de um estado pode resultar em diferentes estados e diferentes valores de recompensas recebidos.

Diferente dos algoritmos de aprendizado supervisionado, a forma com que o aprendizado por reforço trabalha não leva em consideração amostras de entrada ou saida para serem utilizadas nas etapas de treinamento e testes da classificação. Sua metodologia faz com que, após executada uma ação, o agente receba uma recompensa e não fique ciente se a ação foi a melhor possível para alcançar o objetivo. Somente após obter experiências das possíveis ações, estados, transições e recompensas que o sistema consegue atingir um resultado consideravelmente bom.

Sarsa

Sarsa (State Action Reward State Action) é um algoritmo utilizado para aprender uma politica de um problema de aprendizado por reforço. Sua principal funcionalidade é atualizar o Q-value de acordo com o estado atual do agente, a ação que o agente irá escolher e a recompensa que o mesmo irá receber por ter escolhido essa ação. Podemos expressar sua ecuação da seguinte maneira:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \beta[r_{t+1} + \gamma Q(s_{t_1}, a_{t+1}) - Q(s_t, a_t)]$$

Q-Learning

Q-learning é um algoritmo utilizado para otimizar a politica de seleção de ações de um processo de decisão markoviano. Seu funcionamento se da através do aprendizado de uma função de valor de ação ($Q(s, a)$), que retorna o valor esperado de tomar uma determinada ação em um determinado estado formando assim uma politica.

$$Q(s_t, a_t) \leftarrow (1 - \alpha) \cdot \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\left(\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}} \right)}^{\text{learned value}}$$

Implementação

Sarsa

Algoritmo Sarsa

```
Initialize Q(s, a) arbitrarily
Repeat (for each episode):
   Initialize s
   Choose a from s using policy derived from Q
         (e.g., ε-greedy)
   Repeat (for each step of episode):
      Take action a, observe r, s'
      Choose a' from s' using policy derived from Q
         (e.g., ε-greedy)
      Q(s, a) <-- Q(s, a) + α[r + γQ(s', a')- Q(s, a)]
      s <-- s'; a <-- a';
   until s is terminal
```

Q-Learning

Algoritmo Q-learning

```
Initialize Q(s, a) arbitrarily
Repeat (for each episode):
   Initialize s
   Repeat (for each step of episode):
      Choose a from s using policy derived from Q
         (e.g., ε-greedy)
      Take action a, observe r, s'
      Q(s, a) <-- Q(s, a) + α [r + γ max_α,Q(s', a') - Q(s, a)]
      s <-- s';
   until s is terminal
```

Testes

O teste foi realizado em um small grid world com obstáculo (penhasco) disponível no material da aula. O objetivo do agente é sair do ponto inicial e ir para o final sem cair no penhasco.

Resultados

Podemos observar que o algoritmo sarsa fez com que o agente caisse mais vezes no penhasco.

Figura 2: Resultado com algoritmo Sarsa.

Sarsa

Q-Learning

Já no algoritmo Q-learning, o caminho percorrido pelo agente foi mais seguro, fazendo com que o mesmo não caisse no penhasco

Figura 3: Resultado com algoritmo q-learning.

Conclusão

O relatório propós a implementação de dois algorítmos de aprendizagem por reforço (sarsa e q-learning) em liguagem c++. Para testar os algorítmos foi utilizado um small grid world com um obstaculo (penhasco) presente no material da disciplina. O resultado de ambos algoritmos conseguiram chegar no objetivo, porém, o algoritmo q-learning demonstrou melhor resultado pois o agente, por percorrer um caminho mais seguro, não caia no penhasco.

Referências

[1] S. Russell, and P. Norvig. *Artificial Intelligence: A Modern Approach. Series in Artificial Intelligence Prentice Hall, Upper Saddle River, NJ, terceira edition, 2010*

[2] Q-learning Disponível em (https://en.wikipedia.org/wiki/Q-learning). Acesso em: 20 de dez. de 2017

[3] *State–action–reward–state–action Disponível em (https://en.wikipedia.org/wiki/State–action–reward–state–a Acesso em: 20 de dez. de 2017*