동적계획법과 강화학습 강의 노트

작성자 : 정재원(2021321388)

작성일시 : 2022.10.08

Lecture 6. Model Free Policy Control

1. Model Free Control

   MDP model을 모를 때나 알려져 있더라도 사용하기 어려울 때 sampling 기법을 사용하는 Model-free control을 사용한다.

2. On and Off Policy Learning

   구하고자 하는 policy와 trace의 policy가 동일한 상황에서 학습하는 방법이 On policy Learning이다. Off policy Learning은 구하고자 하는 target policy와 갖고 있는 sample의 policy가 다른 상황에서 학습하는 방법이다.

3. Generalized Policy Iteration

   Policy iteration은 주어진 policy에서 value function을 추정하는 Policy evaluation과 가장 큰 value를 가져다 주는 action을 선택하는 Policy improvement로 구성된다.

4. Model Free policy iteration using Action-value Function

   Transition matrix를 알고 있는 경우, MDP를 이용해 value domain에서 policy improvement를 진행할 수 있다. 그러나, Model free 상황에서는 transition matrix를 모르므로, q(state, action) domain에서 policy improvement를 진행한다.

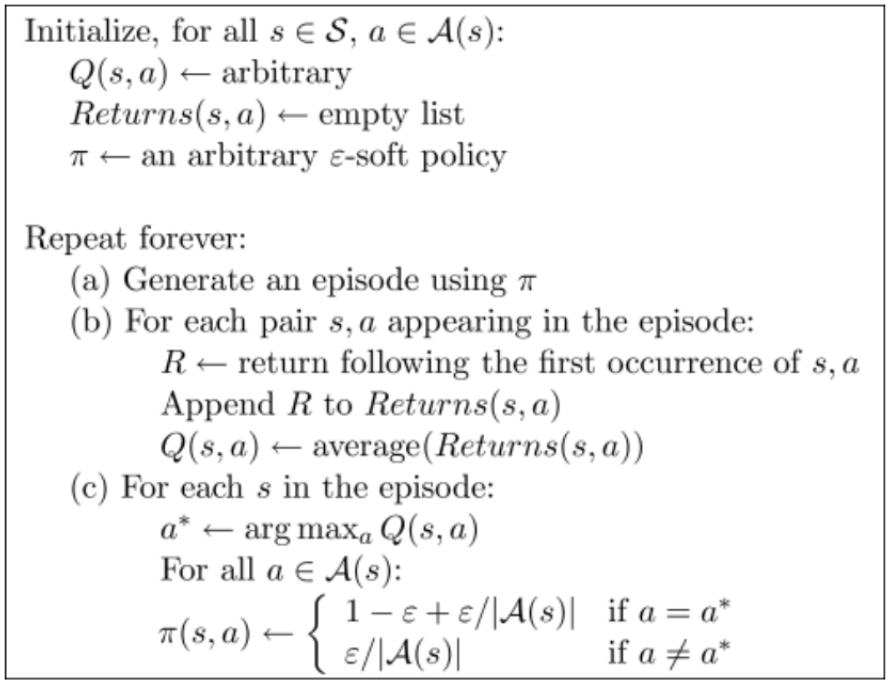
5. ϵ – greedy exploration

   Greedy Exploration을 할 경우, 항상 Q값이 가장 큰 action을 선택하므로, Policy evaluation이 충분히 수행되지 않았을 때, 낮은 value를 갖는 action은 선택되지 않는다. 그러나, 이런 경우 Exploration이 잘 되지 않고 value가 낮은 action은 evaluation을 충분히 받지 못하므로, ϵ – greedy exploration을

사용한다. ϵ – greedy exploration은 $1-\epsilon$의 확률로 현재 state에서 가장 큰 q값을 갖는 action을 선택하고 ϵ의 확률로 random하게 action을 선택해서 모든 action들이 0이 아닌 선택될 확률을 갖는다.

6. Monte-Carlo Control

Monte-Carlo Control은 매 episode마다 Monte-Carlo policy evaluation을 하고 ϵ – greedy policy improvement를 한다.



[Figure 1. An ϵ-soft on-policy Monte-Carlo Control Algorithm]

7. GLIE

학습 방법이 아래의 두 가지 성질을 만족시킬 경우, 그 학습방법은 GLIE라고 말할 수 있다.

- Time step이 무한대로 발산하여 충분히 많이 시행될 경우, 모든 state-action의 쌍은 무한대의 횟수로 방문된다. (ϵ – greedy exploration에서 ϵ이 0이 아닐 경우 이 성질이 만족된다.)
- Time step이 무한대로 발산하여 충분히 많이 시행될 경우, policy가 greedy policy로 수렴한다.

  (ϵ – greedy exploration에서 time step이 진행됨에 따라 ϵ의 값이 점점 줄여들면 이 성질이 만족된다.)

8. Sarsa

TD를 이용해 action-value function을 추정하는 방법이 Sarsa이다. Value function이 아닌 action-value function $Q(s_t, a_t)$을 추정하므로 $(s_t, a_t, r_{t+1}, s_{t+1}, a_{t+1})$의 tuple을 이용한다. 따라서, Sarsa라는 이름을 갖고 있다.

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

[Figure 2. Sarsa update]

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)
    $Q(s, a) \leftarrow Q(s, a) + \alpha\big[r + \gamma Q(s', a') - Q(s, a)\big]$
    $s \leftarrow s'$; $a \leftarrow a'$;
  until s is terminal

TD target

[Figure 3. Sarsa Algorithm for On-Policy Control]

8.1 n-step Sarsa

Figure 3에 나온 Sarsa는 1-step Sarsa라고 불릴 수 있다. Action-value fuction을 추정하는 데, 더 많은 time step을 고려하는 것이 n-step Sarsa이다. n이 무한대로 발산하면 n-step Sarsa는 MC와 동일하다.

9. Off-Policy Learning

Action을 생성하는 policy(behavior policy)와 evaluate되고 improved되는 policy(estimation policy)가 다른 학습 방법이 Off-Policy Learning이다.

9.1 Importance Sampling

Sample policy에서 생성된 return을 estimation(target) policy를 평가하기 위해 두 policy의 유사

도를 활용하여 보정하는 방법이 importance sampling이다. 그러나, importance sampling은 sample policy의 확률값이 0인 action이 있으면 사용될 수 없어서 실제로 사용하기는 힘들다.

9.2 Q-Learning

Q-Learning은 target policy는 greedy하게 action을 선택하고 behavior policy는 ϵ – greedy 하게 action을 선택한다. Sarsa는 target policy와 behavior policy 모두 ϵ – greedy 를 따른다. Q-Learning은 target policy와 behavior policy가 다르므로, Off-policy Learning이다.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\Big[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)\Big]$$

[Figure 4. Q-Learning update]

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'
    $Q(s, a) \leftarrow Q(s, a) + \alpha\big[r + \gamma \max_{a'} Q(s', a') - Q(s, a)\big]$
    $s \leftarrow s'$;
  until s is terminal

[Figure 4. Q-Learning algorithm]

[참고문헌]

- Sutton, Richard S., and Andrew G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.