기초부터 시작하는 강화학습

최적정책, 몬테카를로, 시간차학습

최우진

2020/05/07

0. 목차

1. 동적계획법: 최적 정책 선택

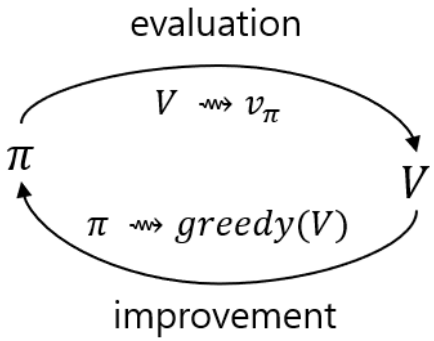
- 강화학습에서 가장 중요한 것은 최적 정책(Optimal policy)을 찾는 것이다.
- 최적 정책: 어떤 상태에서 수익 G가 최대가 되는 행동을 선택하는 정책
- 정책: 어떤 상태에서 행동을 선택할 확률

- 여러 가지 정책에 따라 상태가치함수와 행동가치함수를 이용해 상태가치와 행동가치를 계산한 후 최적의 행동을 계산한 정책이 최적 정책이다.
- 최적상태가치함수: $V^*(s) = \max_{\pi} V_{\pi}(s)$
- 최적행동가치함수: $Q^*(s,a) = \max_{\pi} Q_{\pi}(s,a)$

1. 동적계획법: 최적 정책 선택

- 학습과정:
- 1) 임의의 정책 π_0에서 학습시작
- 2) 여러 가지 시행착오를 통해 π_1, π_2로 보상을 통해 정책이 평가되고 개선
- 3) 높은 가치함수를 갖는 정책 탄생 (최적 정책)

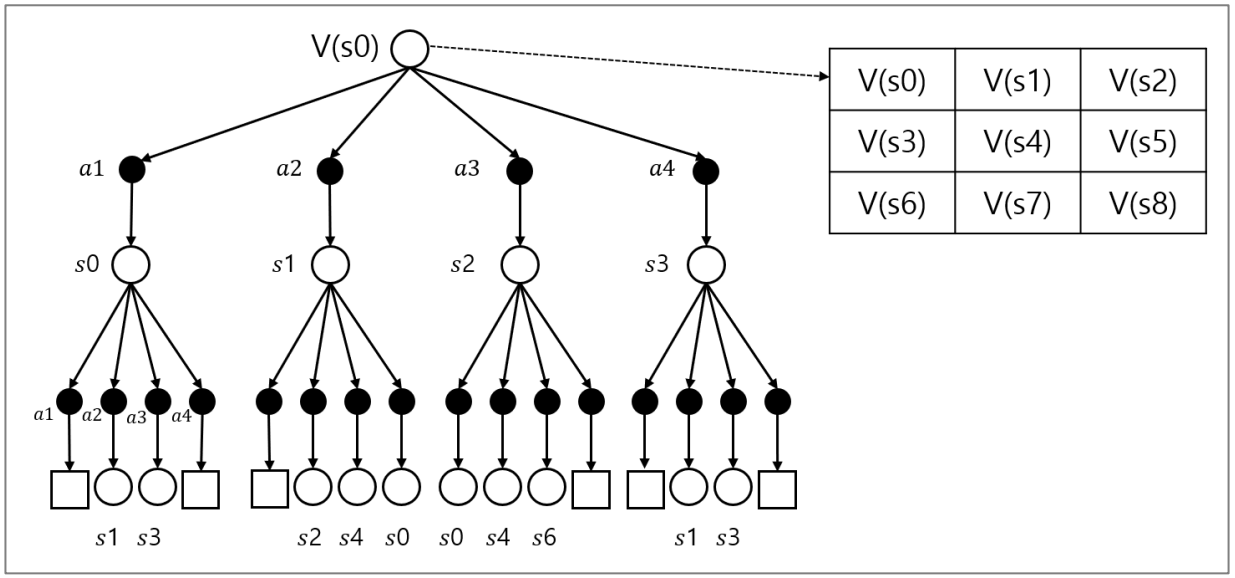
- 평가: 정책 π를 이용해 각 상태의 새로운 상태가치 $V_\pi(s)$를 계산
- 개선: 각 상태의 상태가치 $V_\pi(s)$를 이용해 새로운 정책 π로 개선



학습과정 그림

1.1 정책 평가

- 평가: 정책 π를 이용해 각 상태의 새로운 상태가치 $V_\pi(s)$를 계산
- 상태가치를 아래 그림과 같이 재귀 함수를 이용해 구하면 몇 단계 아래의 상태가치까지 참조하는 지에 따라 계산량이 달라진다.
- 너무 많아지게 되면 계산을 하지 못한다.

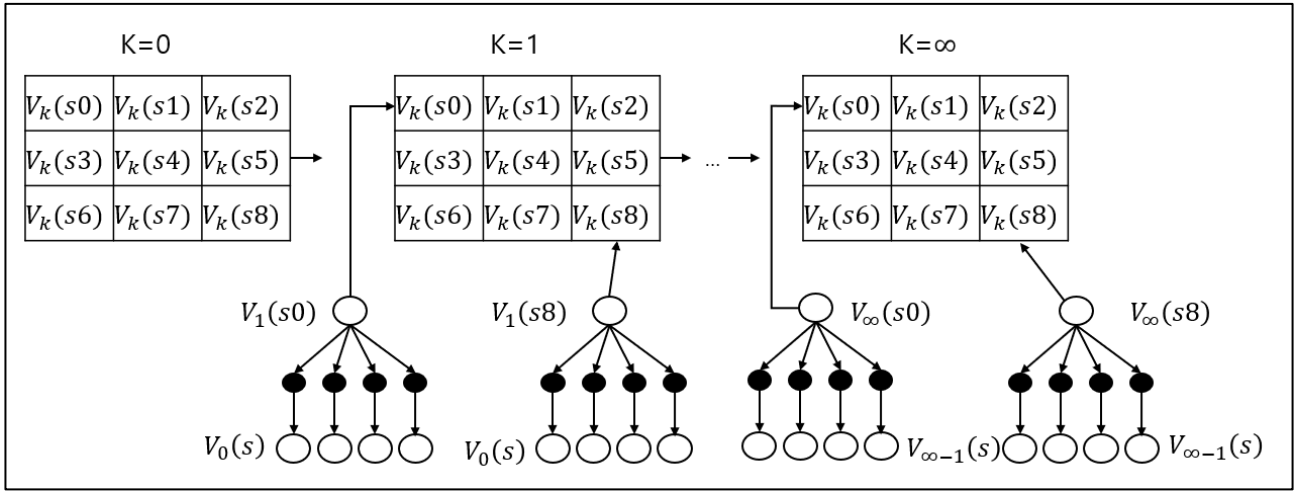


1.2 반복 정책 평가

- => 연결된 모든 상태를 재귀적으로 탐색하지 않고, 오직 연결된 다음 상태의 상태가치만 이용해 상태가치를 계산하는 과정을 반복

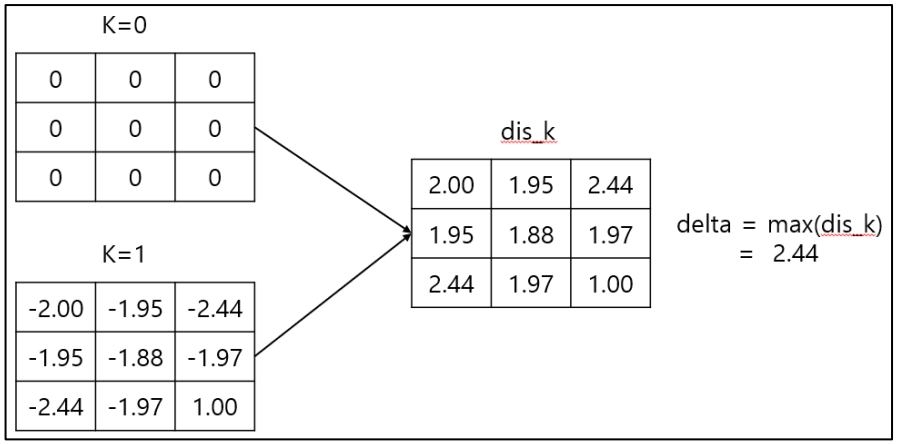
- 반복정책평가의 상태가치함수

- $V_{k+1}(s) = \sum_a \pi(a|s) \sum_{s_{t+1}} P(s'|s,a)[r(s,a,s') + \gamma V_k(s')$ ($\mathbf{k}$ = 계산 횟수)



1.2 반복 정책 평가

- 반복 정책 평가는 delta가 설정값(0.000001)보다 작은 경우 종료
- delta: (k - 1)일 때와 (k)일 때의 모든 상태가치의 차이 중 절댓값이 가장 큰 값을 저장
- delta가 0에 가깝다 => $V_{k-1}(s) = V_k(s)$ => 현재 상태가 이전 상태와 동일



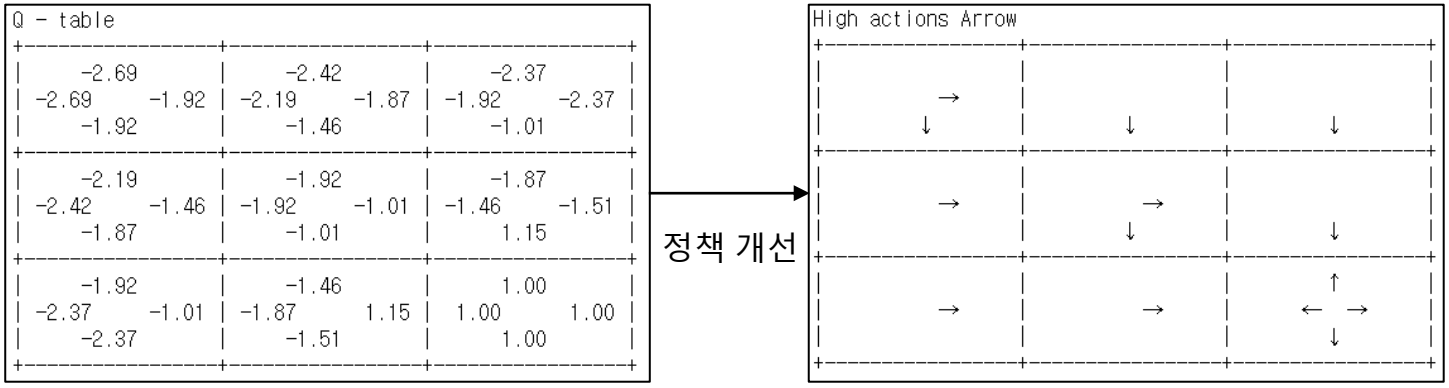
V133(S) : k = 133 delta = 0.000001 total_time = 0.0

-11.72	-9.88	-8.87
-9.88	-6.76	-2.93
-8.87	-2.93	10.00

수렴한 상태 가치

1.3 정책 개선

- 개선: 각 상태의 상태 가치 $V_\pi(s)$를 이용해 새로운 정책 π로 개선
- 정의: 정책 평가를 통해 계산된 새로운 상태가치를 이용해 최적의 행동(행동가치가 가장 큰 행동)을 선택하는 것
- 식: $\pi^*(s) = \underset{a}{\operatorname{argmax}}\, Q_\pi(s,a)$
- $\Rightarrow Q_\pi(s,a)$ 중에서 최대값이 되는 행동 a이 최적 정책 $\pi^*(s)$가 된다.



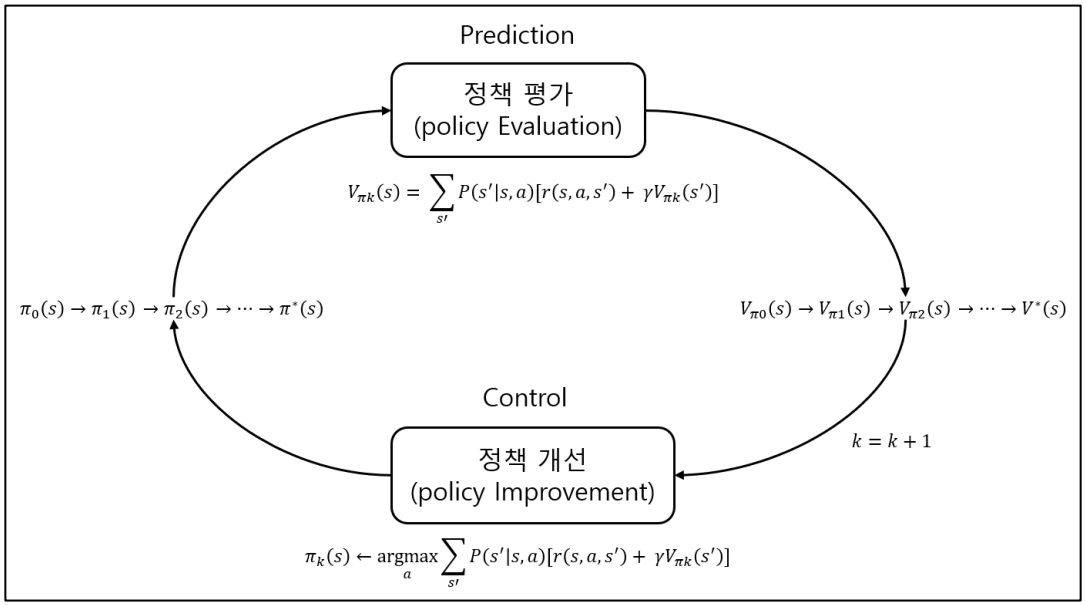
행동 가치 테이블 새로운 정책

1.4 정책 반복

- 정책 반복: Prediction과 Control을 반복하면서 최적의 상태가치 V'와 최적의 정책 π^*을 찾아가는 알고리즘
- 종료 조건: 이전 정책 = 개선된 정책

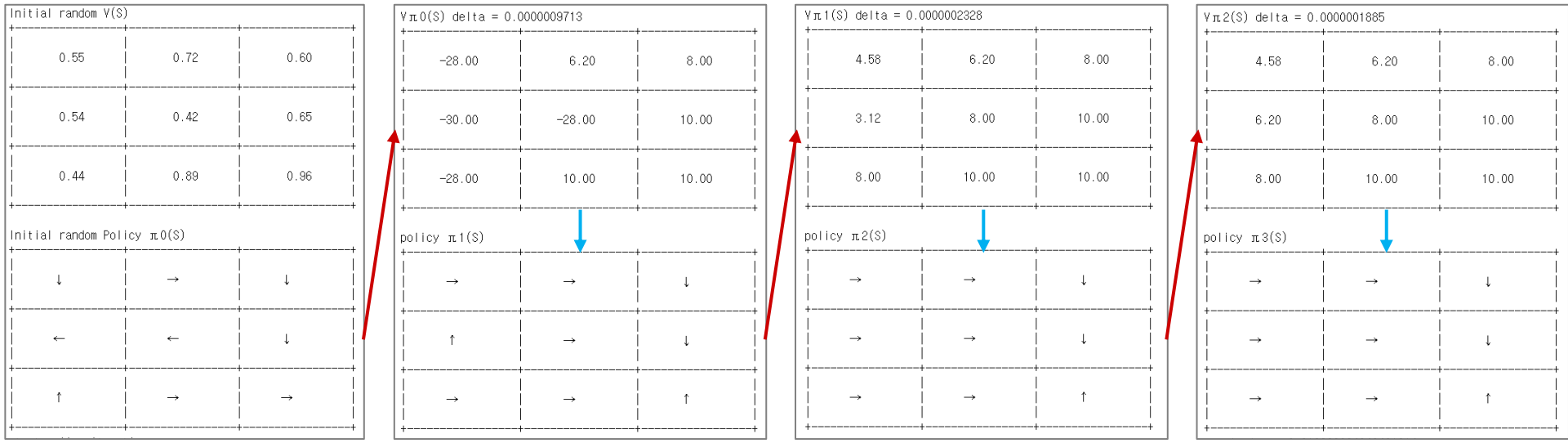
Prediction → 새로운 상태가치 $V_\pi(S)$ 계산

Control → 새로운 정책 π 생성



1.4 정책 반복

- 정책 반복: Prediction과 Control을 반복하면서 최적의 상태가치 V'와 최적의 정책 π^*을 찾아가는 알고리즘
- 정책 반복 예제

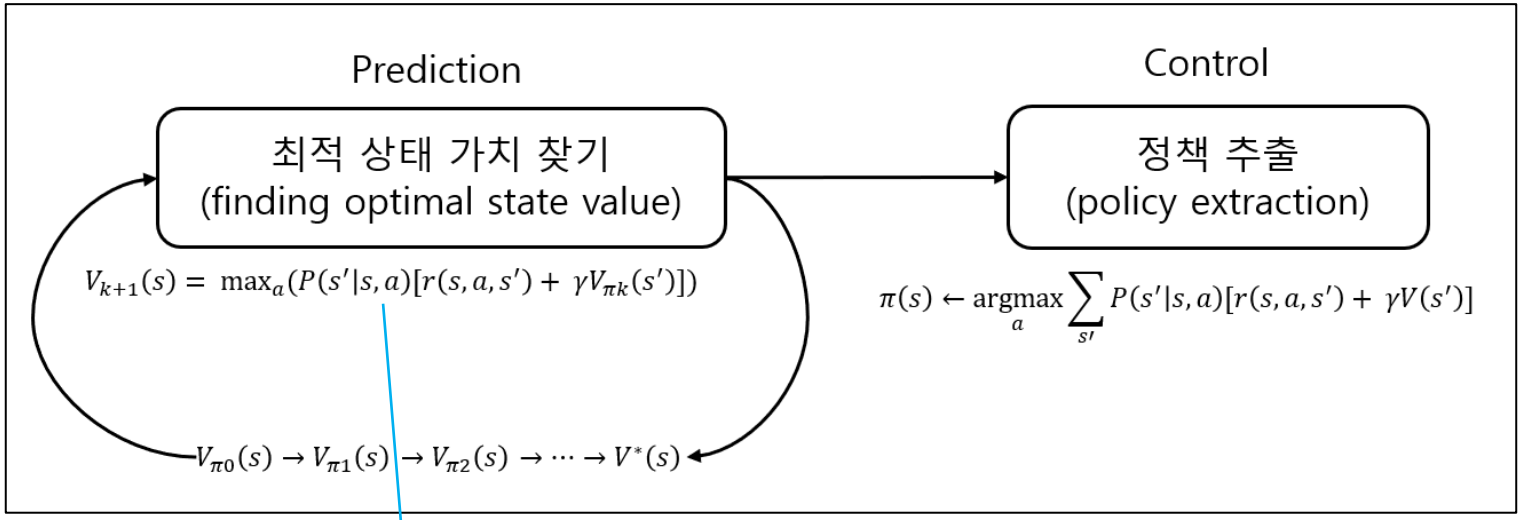


정책 평가
정책 개선

1.5 가치 반복

- 1) Prediction에서 최적의 상태가치를 찾을 때까지 반복
- 2) 최적의 상태가치를 찾으면, Control에서는 최적의 상태가치로부터 최적 정책을 추출

- 진행과정 (종료조건: 이전 상태가치 = 현재 상태가치)



가능한 행동 a와 연결된 모든 상태가치 중에서 max를 이용해 가장 가치가 큰 상태가치만 고려

1.5 가치 반복

- **가치 반복 예제**

Initial random V0(S)

0.55	0.72	0.60
0.54	0.42	0.65
0.44	0.89	0.96

[이동방향] 상태가치= 보상 + 감가율 * 이동한 곳의 상태가치

s(0):

[↑] -2.51 = -3.00 + 0.90 * 0.55

[→] -0.36 = -1.00 + 0.90 * 0.72

[↓] -0.51 = -1.00 + 0.90 * 0.54

[←] -2.51 = -3.00 + 0.90 * 0.55

V(0): max = -0.36

V1(S) : k = 1 delta = 1.221402

-0.36	-0.46	-0.42
-0.61	-0.20	1.87
-0.20	1.87	1.87

V153(S) : k = 153 delta = 0.000000

4.58	6.20	8.00
6.20	8.00	10.00
8.00	10.00	10.00

최적 상태 가치

Optimal policy

→	→	↓
→	→	↓
→	→	↑

최적 정책 추출

2. 몬테카를로 방법

- 상태가치함수: $V_\pi(s) = \sum_a \pi(a|S) \sum_{s'} P(s'|s,a)[r(s,a,s') + \gamma V_\pi(s')]$
- 상태가치를 구하기 위해 알아야할 요소들
  - 1) 정책 $\pi(a|S)$
  - 2) 상태전이확률 $P(s'|s,a)$
  - 3) 보상 $r(s,a,s')$
  - 4) 감가율 γ
  - 5) 다음 상태의 상태가치 $V_\pi(s')$

  → 앞에서 제시한 방법들로 해결
- 위와 같은 환경에 대한 정보를 모두 알고 있는 상태에서 강화학습을 푸는 알고리즘: 모델 기반 알고리즘(Model-based algorithm)
- 상태전이확률을 몰라도 되는 알고리즘: 모델 프리 알고리즘(Model-free algorithm)
- 모델 프리 알고리즘 종류: 몬테카를로 방법, 시간차 학습

2.1 몬테카를로 방법의 Prediction

- 몬테카를로 방법은 탐색적인 방법을 이용해 상태가치함수와 행동가치함수를 학습
- 또한, 경험으로 상태전이확률을 대신한다.

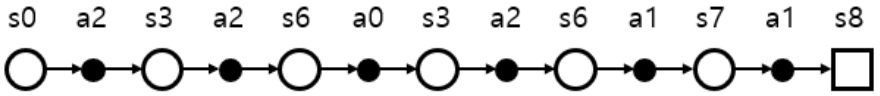
- 몬테카를로 방법
  - => 모든 상태(도착지점 제외)에서 에피소드를 시작하고, 에피소드별로 얻은 수익 G를 저장
- 1) 모든 단계에서 행동은 가능한 행동들 중 무작위 선택
- 2) 지정된 횟수(n번)만큼 에피소드가 끝나면 수익 G들의 평균을 각 상태마다 계산
- 3) 각 상태마다 계산된 평균 수익 G를 그 상태의 상태가치로 저장
- 몬테카를로 방법의 상태가치함수

$$V(s) = average(G_1, G_2, \dots, G_n)$$

2.1 몬테카를로 방법의 Prediction

- **몬테카를로 방법이 제대로 학습하기 위한 전제조건**
  - 1. 모든 상태에서 시작할 수 있어야 한다.
  - 2. 에피소드는 반드시 끝이 있어야 한다.

- **에피소드에서 같은 상태를 2번 지나가는 경우**
- **1. First-visit 몬테카를로 방법**
  - 첫 번째로 도착한 상태의 보상만 참고
- **2. Every-visit 몬테카를로 방법**
  - 모든 상태의 보상을 수익에 참고



중복 상태를 가지는 에피소드

- **1번과 2번 모두 동일한 결과로 수렴하지만, 계산시간에서 차이가 난다.**

2.1 몬테카를로 방법의 Prediction

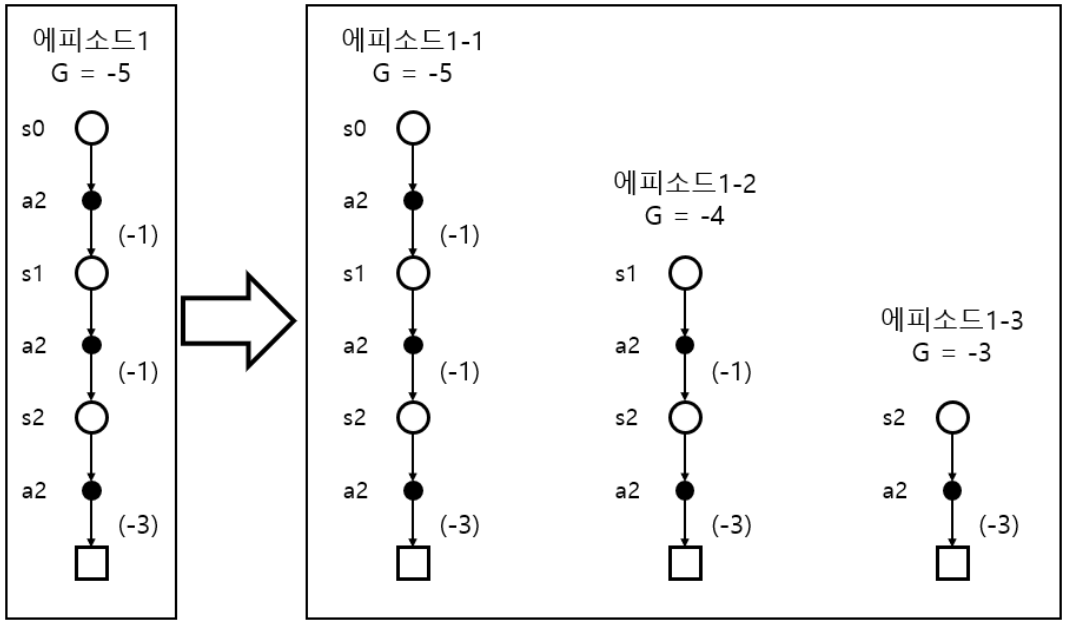
- 앞에서 상태가치를 구할 때, 얻은 전체 수익의 평균으로 계산하였다.
- 샘플링이 많아지는 경우 메모리가 많이 차지 하기 때문에 아래와 같이 변경

$$V(S_t) \leftarrow V(S_t) + \frac{1}{n+1}[G_t - V(S_t)]$$

- 새로운 상태가치를 기준으로 $[G_t - V(S_t)] = 0$이 되도록 상태가치 $V(S_t)$를 학습

2.1 몬테카를로 방법의 Prediction

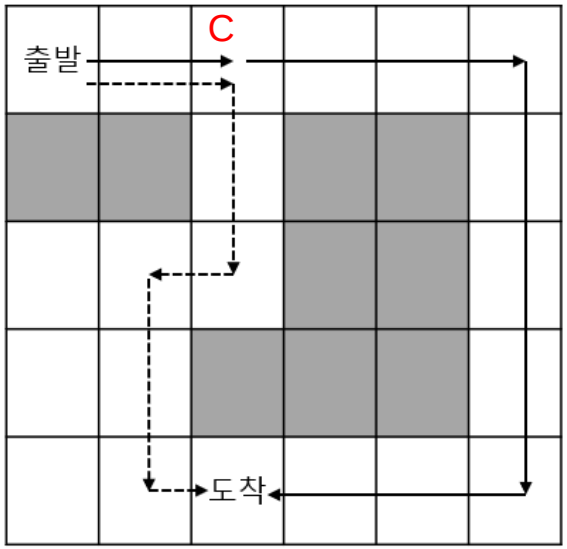
- **모든 상태에서 시작을 할 수 없고 처음 상태에서만 시작이 가능한 문제인 경우**
  - 에피소드 분리를 통해 해결



2.2 몬테카를로 방법의 Control

- **어떤 상태에서 행동을 선택하는 정책**
  - 1. 무작위로 행동을 선택하는 정책
  - 2. 행동가치를 이용해 확률로 행동을 선택하는 정책
  - 3. 행동가치 중 가치가 가장 높은 행동을 선택하는 정책(greedy policy)

- **탐욕정책(greedy policy)의 단점**
- **=> 학습 중간에 local minimum에 빠질 수 있다.**
- **실선의 에피소드가 먼저 도착지점에 도착하게 되면,**
  계속 C에서 오른쪽으로만 가게 된다.



2.2 몬테카를로 방법의 Control

- 탐욕정책(greedy policy)을 해결한 방법: ϵ - greedy 정책

- $\pi(s,a) = \begin{cases} 1 - \epsilon + \frac{\epsilon}{|A(s)|} & (a = A^*) \\ \frac{\epsilon}{|A(s)|} & (a \neq A^*) \end{cases}$, $|A(s)|$: 상태 s에서 가능한 행동의 개수, $0 \leq \epsilon \leq 1$

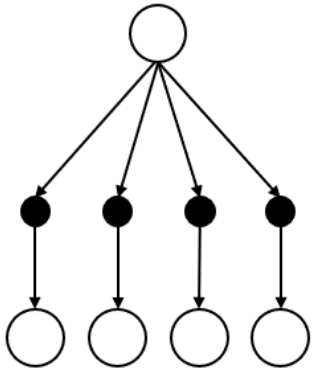
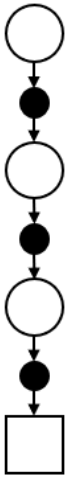
- 예) $|A(s)|$ = 4개(동서남북), A^* = 서쪽 이동(최적 행동)
- $\epsilon = 1$, 최적 행동과 상관없이 모든 행동의 선택될 확률이 0.25로 동일 (즉, 무작위 선택)
- $\epsilon = 0$, 최적 행동만 선택되어 탐욕정책

2.2 몬테카를로 방법의 Control

- 몬테카를로 방법의 Control 알고리즘
  - 1) 임의의 정책 $\pi(s,a)$을 이용해 에피소드 생성
  - 2) 생성된 에피소드를 이용해 행동가치 $Q(s,a)$를 학습
  - 3) 학습된 행동가치 $Q(s,a)$를 이용해 각 상태의 최적 행동 A^* 추출
  - 4) 각 상태의 최적 행동 A^*와 ϵ-greedy 정책을 이용해 새로운 정책 $\pi(s,a)$을 생성
  - 5) 최대 에피소드 수 만큼 (1) ~ (4) 반복

3. 시간차 학습

- **학습방법별 비교**

	동적계획법	몬테카를로법	시간차 학습
환경 정보	필요(model-based)	불필요(model-free)	불필요(model-free)
가치함수 계산	상태전이확률	샘플링	샘플링
학습 단위	Time Step	Episode	Time Step
백업 다이어그램			

3.1 시간차 학습의 Prediction

- 1) 상태가치함수: $V(S_t) \leftarrow V(S_t) + \alpha[G_t - V(S_t)]$
- 2) 시간차 학습의 수익: $G_t = r_{t+1} + \gamma V(S_{t+1})$
- 3) 시간차 학습의 상태가치함수: $V(S_t) \leftarrow V(S_t) + \alpha[r_{t+1} + \gamma V(S_{t+1}) - V(S_t)]$
- 시간차 학습의 상태가치 $V(S_t)$가 수렴하는 조건

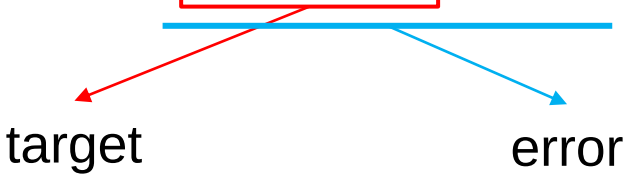
$$r_{t+1} + \gamma V(S_{t+1}) = V(S_t)$$

- 시간차 학습의 여러 방법이 있지만, 여기서는 오직 연결된 다음 상태만의 상태가치를 이용해 상태가치를 구하는 방법을 이용한다.

3.2 시간차 학습의 Control: SARSA(On-policy)

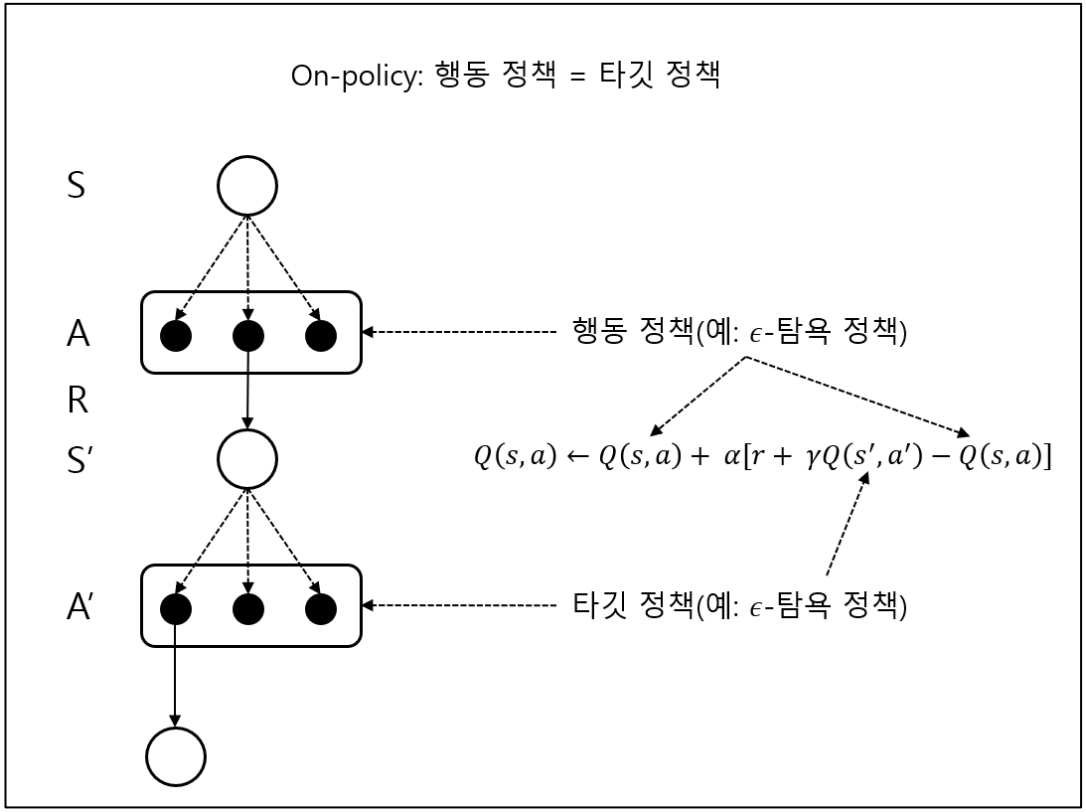
- **최적 정책을 학습하는 TD(0) Control에서는 두 가지의 행동을 선택하는 정책이 있다.**
  - 1. 행동 정책(Behavior Policy): 현재 상태 s에서 가능한 행동들 중에서 a를 선택하는 정책
  - 2. 타깃 정책(Target Policy): 행동가치함수를 학습하기 위해 다음 상태 s'에서 가능한 행동들 중에서 a'를 선택하는 정책

- **SARSA의 행동가치함수:** $Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma Q(s',a') - Q(s,a)]$



3.2 시간차 학습의 Control: SARSA(On-policy)

- SARSA의 특징
  - 행동 정책과 타깃 정책이 동일

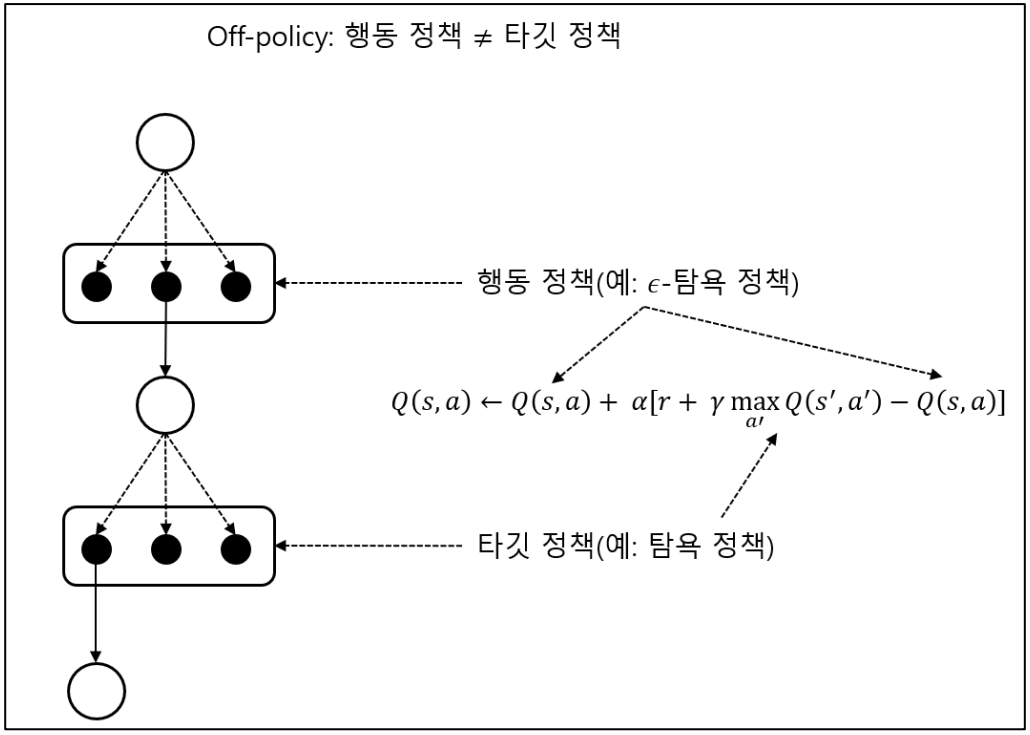


3.3 시간차 학습의 Control: Q-learning(Off-policy)

- Q-learning의 행동가치함수: $Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)]$

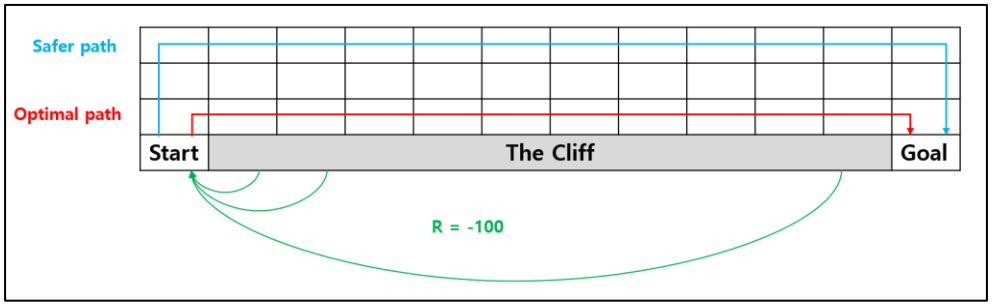
target error

- Q-learning의 특징
  - 행동 정책과 타깃 정책이 다름



3.4 SARSA와 Q-learning의 차이점

- Sutton 교수의 책(Reinforcement Learning 2nd 예제)



- 4 x 12 격자 존재, 양쪽 끝에 Start 지점과 Goal 지점 존재
- 아래쪽에 절벽이 있어 에이전트가 떨어지면 -100의 보상을 받고 Start 지점으로 이동.
- SARSA는 안전한 경로를 학습하고 Q-learning은 최적 경로를 학습한다.
- Max 값을 추구하는 Q-learning은 절벽에 떨어지는 것을 감수하고 최단 경로를 탐색하지만, SARSA는 절벽 바로 위의 상태들은 R=-100의 영향을 받아 멀리 돌아가는 안정적인 경로 탐색

따라서, 풀고자 하는 문제에 따라 잘 선택해서 사용하는 것이 좋다.

최적정책, 몬테카를로, 시간차학습

감사합니다