Ch4

MDP를 알 때의 플래닝

Introduction

1. 상태집합 s나 액션의 집합 a의 크기가 작은 경우
2. MDP를 알 때 = 보상함수와 전이확률을 알고 있을 경우

가정

정책 개선 과정(플래닝)

Tabular method

(s, a)에 대한 테이블을 업데이트하는 방식

prediction

정책 π가 주어졌을 때 각 상태의 밸류를 평가하는 문제

control

최적의 정책 함수를 찾는 문제

planning

prediction 문제

ex> 정책 함수 π(4방향 랜덤)가 주어진 상태에서 각 상태 s에 대한 가치함수 v(s) 구하기

출발	s_1	s_2	s_3
s_4	s_5	s_6	s_7
s_8	s_9	s_{10}	s_{11}
s_{12}	s_{13}	s_{14}	종료

보상 : 스텝마다 -1

정책 : 4방향 uniform random

반복적 정책 평가

prediction 문제의 해결 – 반복적 정책 평가

1 테이블 초기화

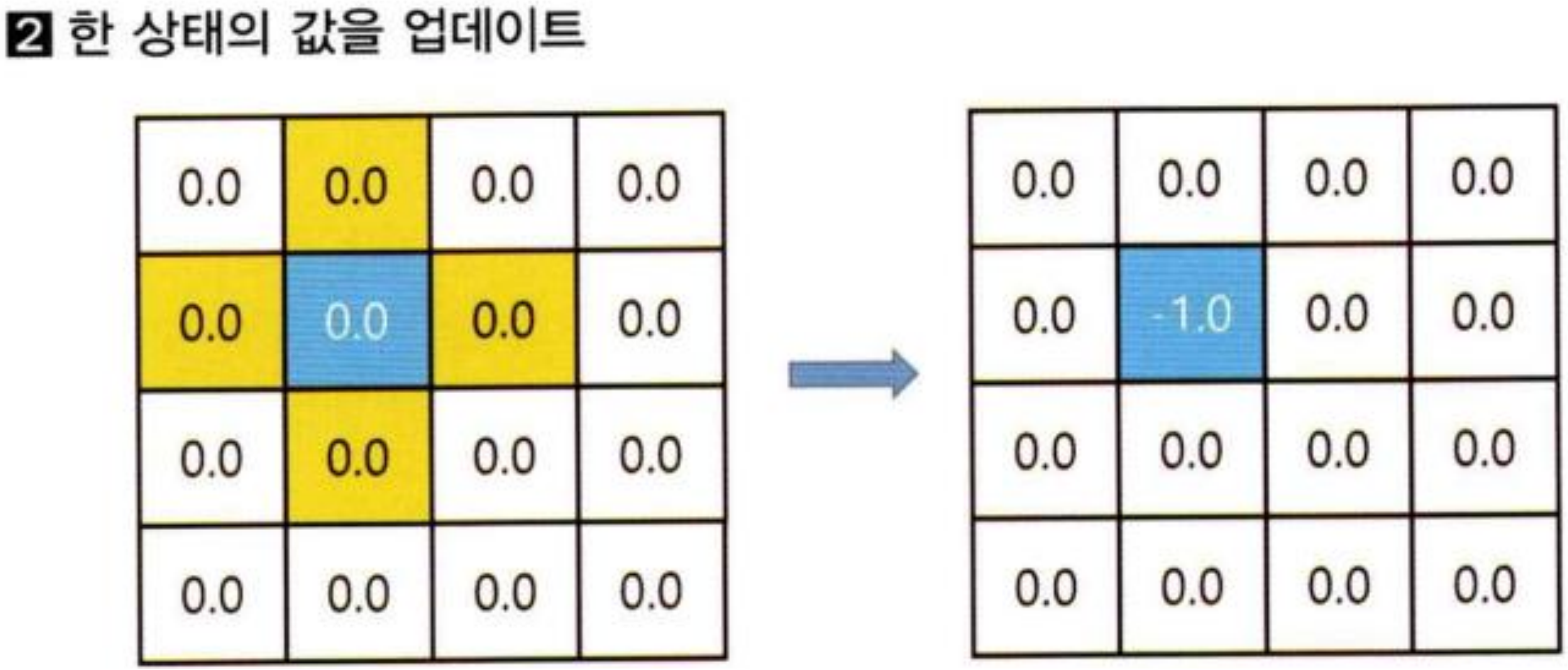
s_0	s_1	s_2	s_3
s_4	s_5	s_6	s_7
s_8	s_9	s_{10}	s_{11}
s_{12}	s_{13}	s_{14}	종료

→

0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0

- 테이블 셀 개수 = 상태 s의 개수
- 적당한 값으로 초기화

prediction 문제의 해결 – 반복적 정책 평가



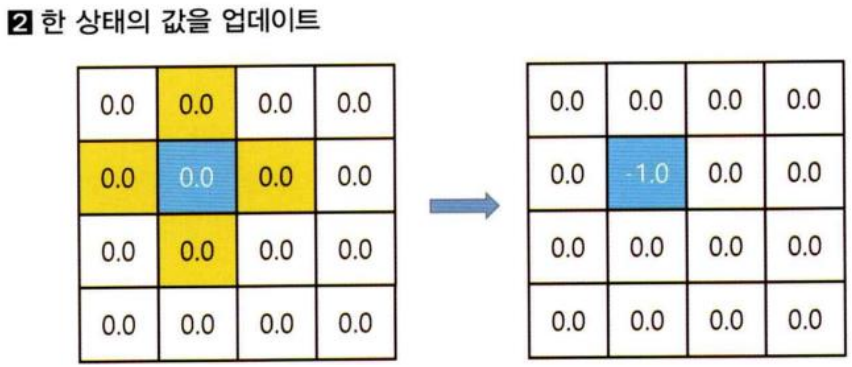
현재 상태(파란색)에서 다음 상태의 후보(노란색)를
정확히 알고 있음

벨만 기대 방정식

$$\underbrace{v_\pi(s)}_{\text{테이블에 담겨있는 임의의 값}} = \sum_{a \in A} \pi(a|s)\left(\underbrace{r_s^a}_{\text{환경이 주는 실제 값}} + \gamma \sum_{s' \in S} P_{ss'}^a \underbrace{v_\pi(s')}_{\text{테이블에 담겨있는 임의의 값}} \right)$$

prediction 문제의 해결 – 반복적 정책 평가

2. 한상태의 값을 업데이트 cont.



Q1. 무의미한 초기화 값에서 어떻게 실제 가치를 담을 수 있게 될까?

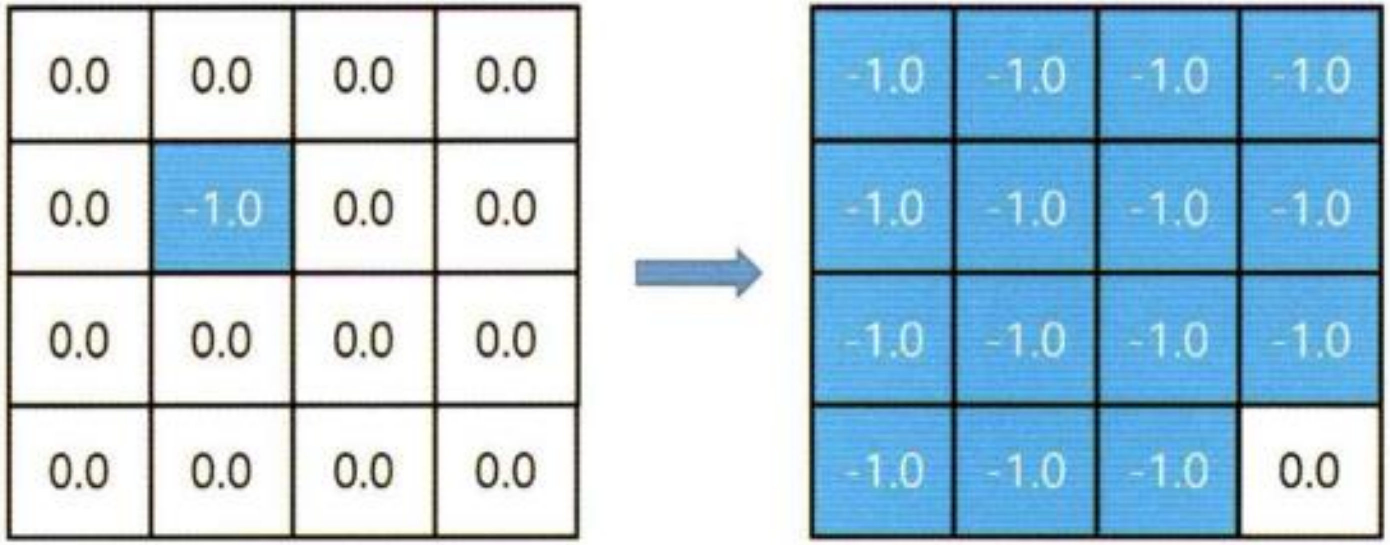
'보상'

Q2. 다음 상태의 값은 항상 무의미할까?

'마지막 상태로부터 시작하여 정확한 값의 시그널이 점차 전파되어 나간다'

prediction 문제의 해결 – 반복적 정책 평가

3 모든 상태에 대해 **2**의 과정을 적용



- 가장자리에서 바깥을 향하는 액션은 그 자리 그대로를 의미

prediction 문제의 해결 – 반복적 정책 평가

4 앞의 **2**~**3** 과정을 계속해서 반복

s_0	s_1	s_2	s_3
s_4	s_5	s_6	s_7
s_8	s_9	s_{10}	s_{11}
s_{12}	s_{13}	s_{14}	종료

$k = 0$

0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0

$k = 1$

-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	0.0

$k = 2$

-2.0	-2.0	-2.0	-2.0
-2.0	-2.0	-2.0	-2.0
-2.0	-2.0	-2.0	-1.75
-2.0	-2.0	-1.75	0.0

$k = 3$

-3.0	-3.0	-3.0	-3.0
-3.0	-3.0	-3.0	-2.94
-3.0	-3.0	-2.88	-2.4
-3.0	-2.94	-2.4	0.0

$\times \infty$

$k = \infty$

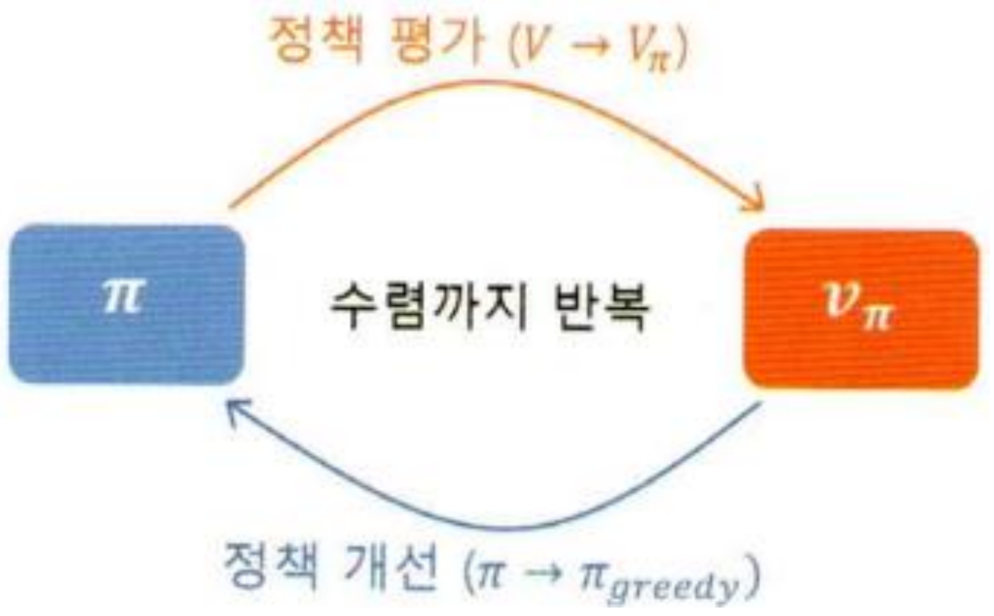
-59.4	-57.4	-54.3	-51.7
-57.4	-54.6	-49.7	-45.1
-54.3	-49.7	-40.9	-30
-51.7	-45.1	-30	0.0

- 수렴하는 값이 실제 밸류 (증명 생략)

$k = \infty$

-59.4	-57.4	-54.3	-51.7
-57.4	-54.6	-49.7	-45.1
-54.3	-49.7	-40.9	-30
-51.7	-45.1	-30	0.0

control 문제- 정책 이터레이션



π'= π인 지점에 수렴하면
최적 정책, 최적 가치

정책 평가: 반복적 정책평가, v(s)
정책 개선: 그리디 정책 π ' 생성, π'> π

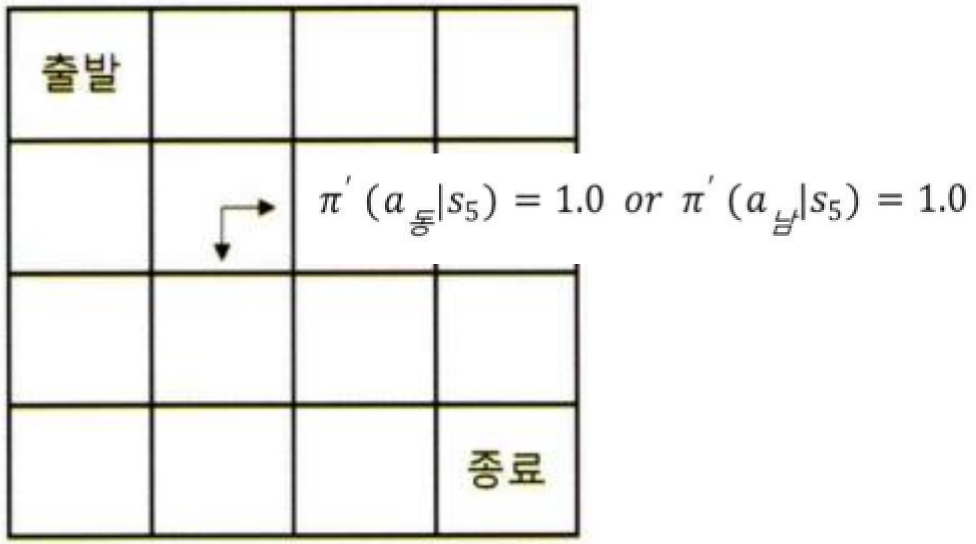
control 문제- 정책 이터레이션

ex> 그리드 월드

$k = \infty$

-59.4	-57.4	-54.3	-51.7
-57.4	-54.6	-49.7	-45.1
-54.3	-49.7	-40.9	-30
-51.7	-45.1	-30	0.0

그리디 정책



랜덤 정책 π에 대해 밸류 평가 후 그리디 정책 π’,
π’ > π로 보다 개선되었음.

control 문제- 정책 이터레이션

Q1. π′ > π가 항상 보장되는가?

- π_{greedy} : 딱 한 칸만 그리디 정책으로 움직이고 나머지 모든 칸은 원래 정책으로 움직이는 정책
- π : 원래 정책

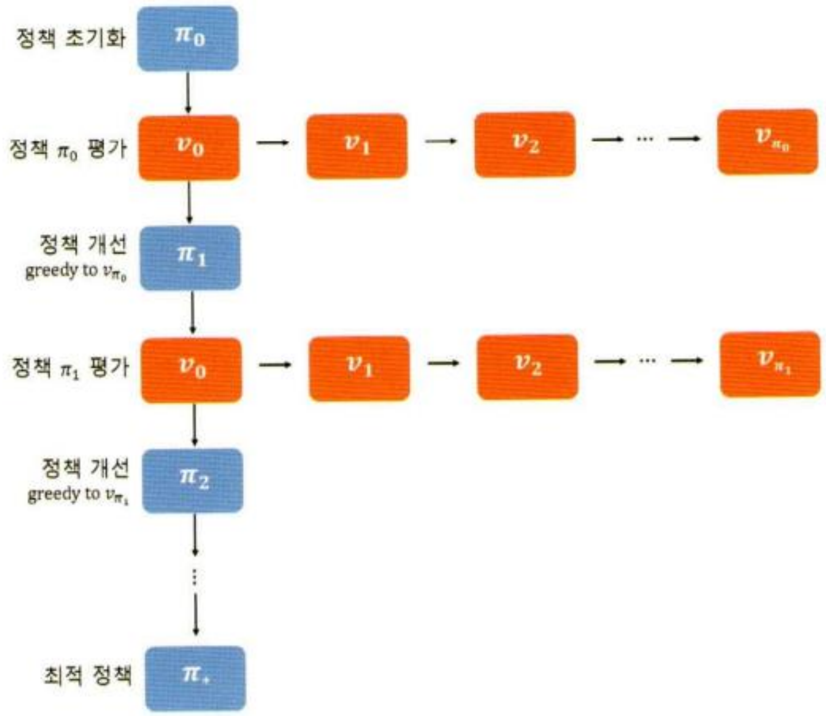
-59.4	-57.4	-54.3	-51.7
-57.4	-54.6	-49.7	-45.1
-54.3	-49.7	-40.9	-30
-51.7	-45.1	-30	0.0

π_greedy가 π보다 더 좋은 정책,

귀납적 논리에 따라 모든 상태에서 greedy한 정책 π′ > 원래 정책π

control 문제- 정책 이터레이션 개선

Q1. 정책 평가(반복적 정책 평가)를 얼마나 많이 반복해야 하는가?

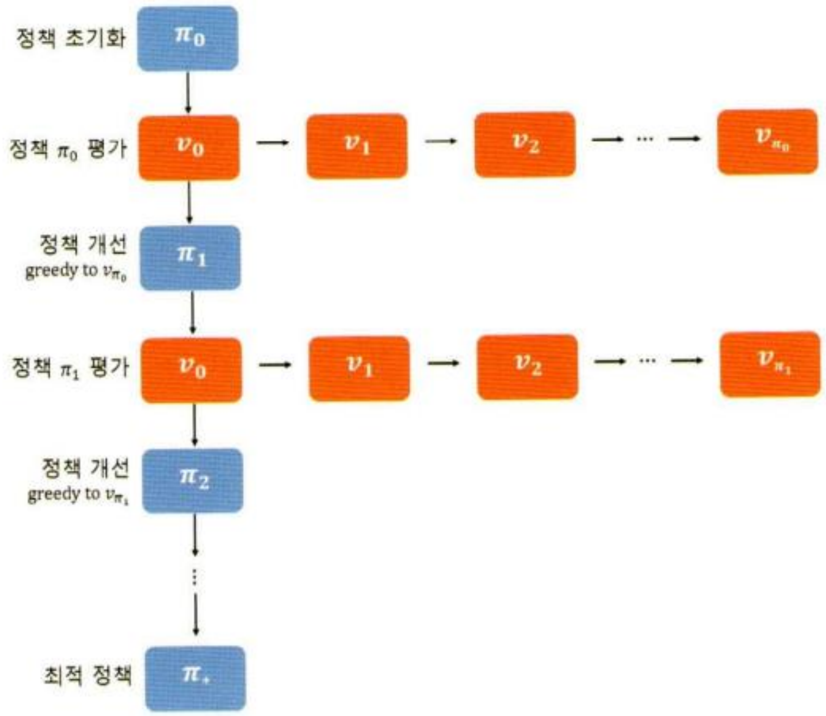


정책 평가를 무한반복하면 실제 밸류 값이 도출되지만, 현실적으로 이는 불가능

→ Early stopping

control 문제- 정책 이터레이션 개선안

Q1. 정책 평가(반복적 정책 평가)를 얼마나 많이 반복해야 하는가?



정책 평가를 무한반복하면 실제 밸류 값이 도출되지만, 현실적으로 이는 불가능

→ Early stopping

control 문제- 정책 이터레이션 개선안

Q1. 정책 평가(반복적 정책 평가)를 얼마나 많이 반복해야 하는가?(cont.)

$k = 6$

-6	-5.99	-5.96	-5.90
-5.99	-5.95	-5.80	-5.55
-5.96	-5.80	-5.27	-4.22
-5.90	-5.55	-4.22	0.0

→

그리디 정책

↓→	↓→	↓→	↓
↓→	↓→	↓→	↓
↓→	↓→	↓→	↓
→	→	→	종료

$k = \infty$

-59.4	-57.4	-54.3	-51.7
-57.4	-54.6	-49.7	-45.1
-54.3	-49.7	-40.9	-30
-51.7	-45.1	-30	0.0

현재 단계에서 구한 greedy 정책이 현재의 정책과 다르기만 하면 업데이트됨.

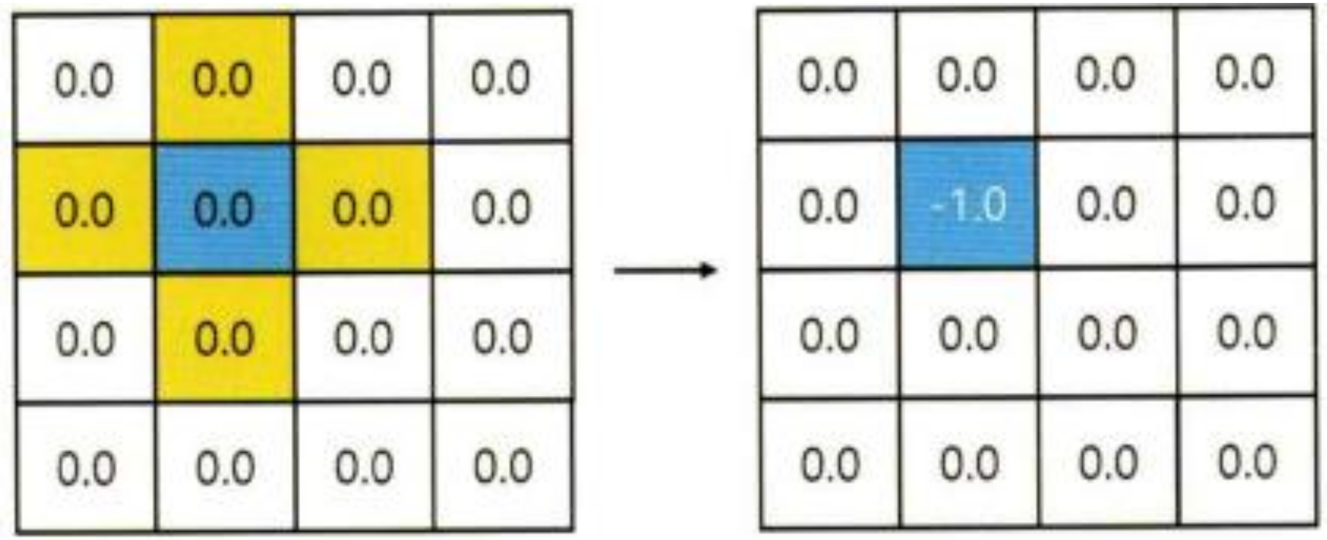
control 문제- 밸류 이터레이션

1 테이블 초기화

s_0	s_1	s_2	s_3
s_4	s_5	s_6	s_7
s_8	s_9	s_{10}	s_{11}
s_{12}	s_{13}	s_{14}	종료

0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0

control 문제- 밸류 이터레이션



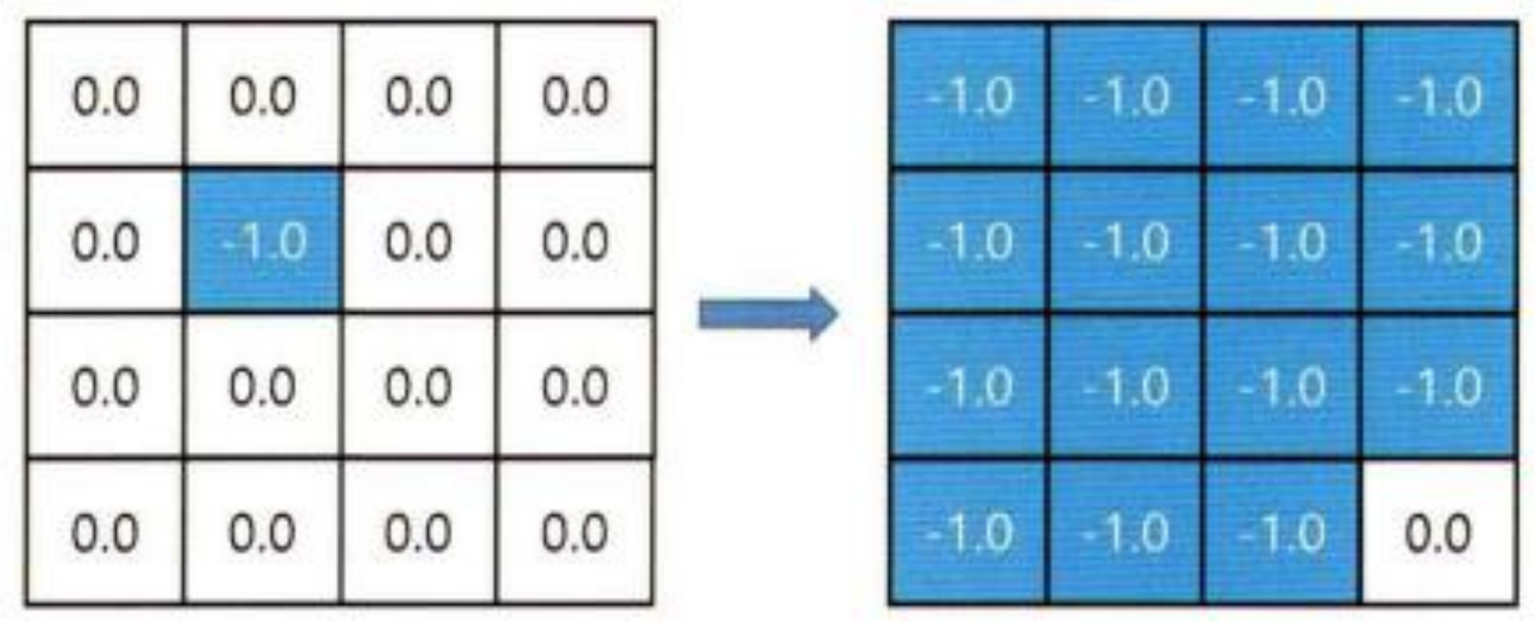
$$v_*(s) = \max_a \left[r_s^a + \gamma \sum_{s' \in S} P_{ss'}^a \, v_*(s') \right]$$

$$\begin{aligned} v_*(s_5) = \max(&-1 + 1.0 * 0, \\ &-1 + 1.0 * 0, \\ &-1 + 1.0 * 0, \\ &-1 + 1.0 * 0) \\ = \ & -1.0 \end{aligned}$$

최적 밸류 사이의 관계를 알기 위해 벨만 최적 방정식을 이용

<-> 정책 π를 업데이트했던 경우 벨만 기대 방정식

control 문제- 밸류 이터레이션



control 문제- 밸류 이터레이션

s_0	s_1	s_2	s_3
s_4	s_5	s_6	s_7
s_8	s_9	s_{10}	s_{11}
s_{12}	s_{13}	s_{14}	종료

$k = 0$

0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

$k = 1$

-1	-1	-1	-1
-1	-1	-1	-1
-1	-1	-1	-1
-1	-1	-1	0

$k = 2$

-2	-2	-2	-2
-2	-2	-2	-2
-2	-2	-2	-1
-2	-2	-1	0

$k = 3$

-3	-3	-3	-3
-3	-3	-3	-2
-3	-3	-2	-1
-3	-2	-1	0

$\times \infty$

$k = \infty$

-6	-5	-4	-3
-5	-4	-3	-2
-4	-3	-2	-1
-3	-2	-1	0

DP의 한계와 강화학습

실제 상황을 DP로 푸는데에는 현실적인 문제들이 존재

1. 계산 복잡도와 차원의 저주: 전체 상태의 개수 n에 대해 DP의 계산복잡도는 O(n^3)
2. 환경에 대한 완벽한 정보 필요: 보상과 상태변환확률을 이미 정확히 알아야한다.

강화학습, GPI (Generalized Policy Iteration)

Prediction

몬테카를로 예측
(에이전트가 에피소드를 진행(샘플링) ->몬테카를로 근사를 통해 가치함수 추정)

시간차 예측 $V(S_t) \leftarrow V(S_t) + \alpha(R + \gamma V(S_{t+1} - V(S_t)))$
(다음 state의 추정치를 기반으로 현재 state의 추정치를 업데이트 = bootstrapping)

Control

ϵ−greedy policy improvement
(에이전트가 특정 확률로 비탐욕적인 행동 선택하도록 함) $\pi(a|s) = \begin{cases} \epsilon/m + 1 - \epsilon & \text{if } a^* = \underset{a \in \mathcal{A}}{\operatorname{argmax}}\, Q(s,a) \\ \epsilon/m & \text{otherwise} \end{cases}$

Q-learning, off-policy 학습
(on-policy 문제 해결, 행동하는 정책은 ϵ−greedy, 학습 정책은 벨만 최적 방정식)

Further subjects