강화학습 원리 스터디

4장 – 동적 프로그래밍

강경민

목차

- 동적 프로그래밍과 정책 평가
- 정책 반복법
- 가치 반복법

지금까지...

- 상태 전이 확률 $p(s'|s,a)$
- 보상 함수 $r(s,a,s')$
- 정책 $\pi(a|s)$

이걸로 벨만 방정식을 만들고, 연립방정식을 계산하여 $v_\pi(s)$를 얻음

지금까지...

how?

만들고, 연립방정식을 계산하여 $v_{\pi}(s)$

어떻게 계산할?

정책 평가

정책 평가 - 정책 π가 주어졌을 때 그 정책의 가치 함수 $v_\pi(s)$ 또는 $q_\pi(s, a)$를 구하는 문제

정책 제어 - 정책을 조정하여 최적 정책을 만들기

벨만 방정식 복습

- 벨만 방정식

$$v_\pi(s) = \Sigma_{a,s'}\, \pi(a|s)\, p(s'|s,a)\, \{r(s,a,s') + \gamma v_\pi(s')\}$$

- 어떤 상태에서 수익의 기댓값은,

(s에서 a를 취할 확률) * (s,a가 일어났을 때 s'으로 전이할 확률) * {s,a,s'에서의 보상 + 할인율 * 다음 상태에서 수익의 기댓값}

-> 확률에 따른 가중치 * (보상 + 할인율 * 다음 상태의 기댓값)을 모든 s',a에 대해 전부 더한 것

추정치?

$v_\pi(s) = \Sigma_{a,s'} \pi(a|s)\, p(s'|s,a) \left\{ r(s,a,s') + \gamma v_\pi(s') \right\}$ 벨만 방정식을 갱신식으로 변형하면,

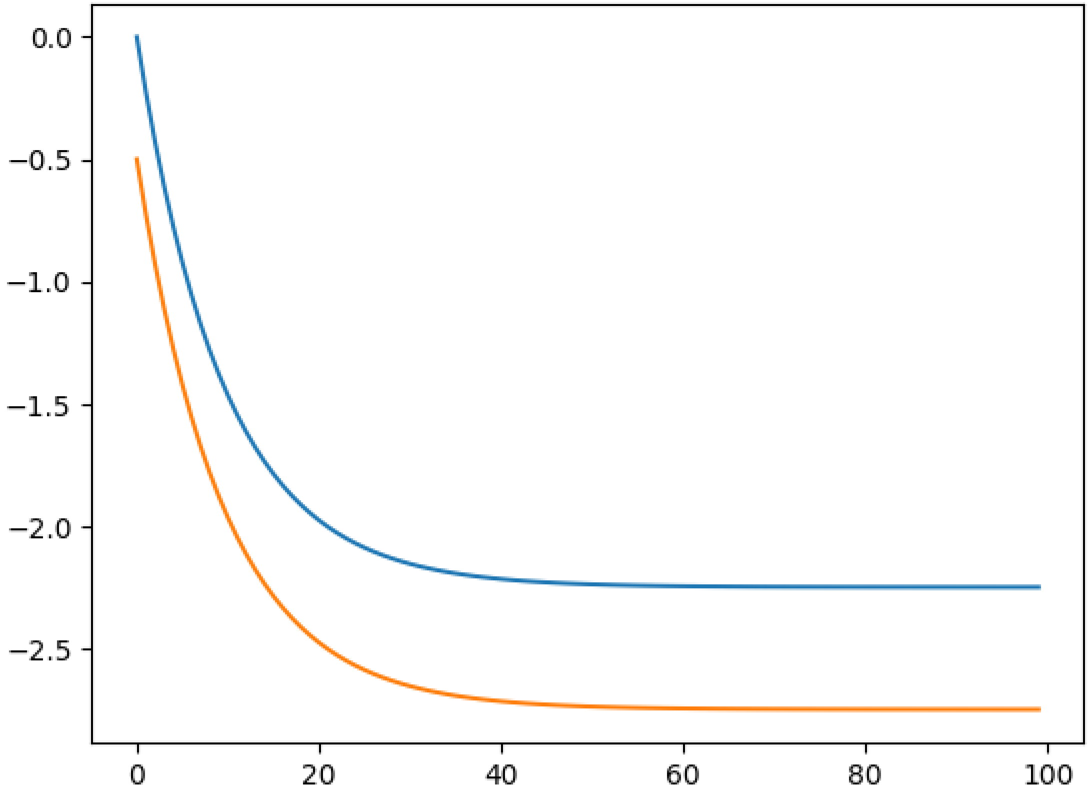
$V_{k+1}(s) = \Sigma_{a,s'} \pi(a|s)\, p(s'|s,a) \left\{ r(s,a,s') + \gamma V_k(s') \right\}$

부트스트래핑

$V_k(s')$을 이용하여 V_{k+1} 갱신

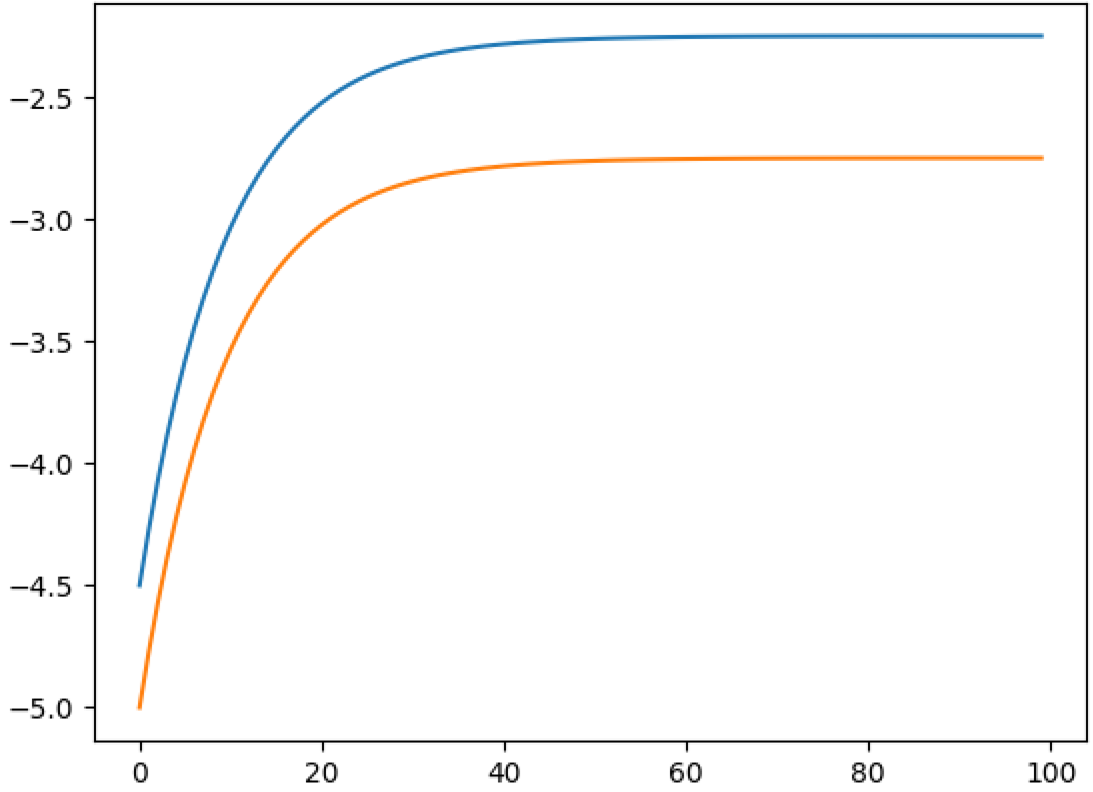
추정치를 사용하여 추정치를 개선하는 과정을 부트스트래핑이라 함

DP 알고리즘 실행



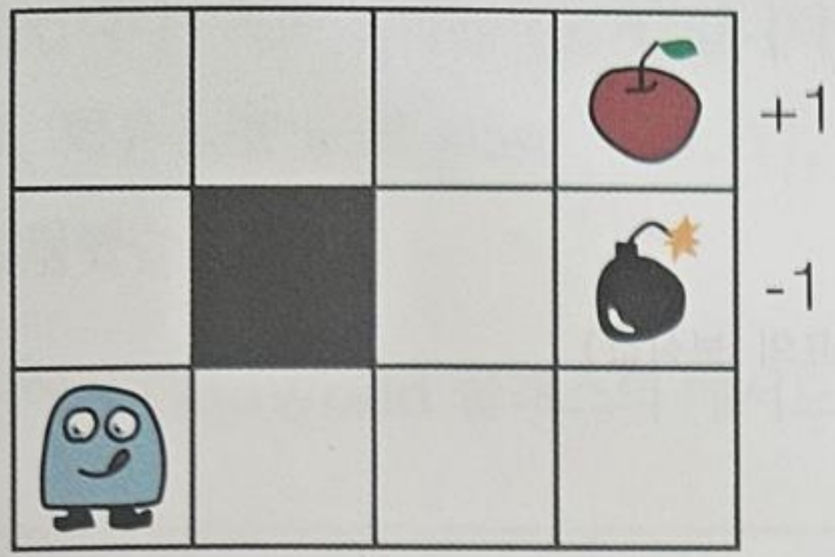
DP 알고리즘 실행

- 초깃값을 다르게 잡았을 때:



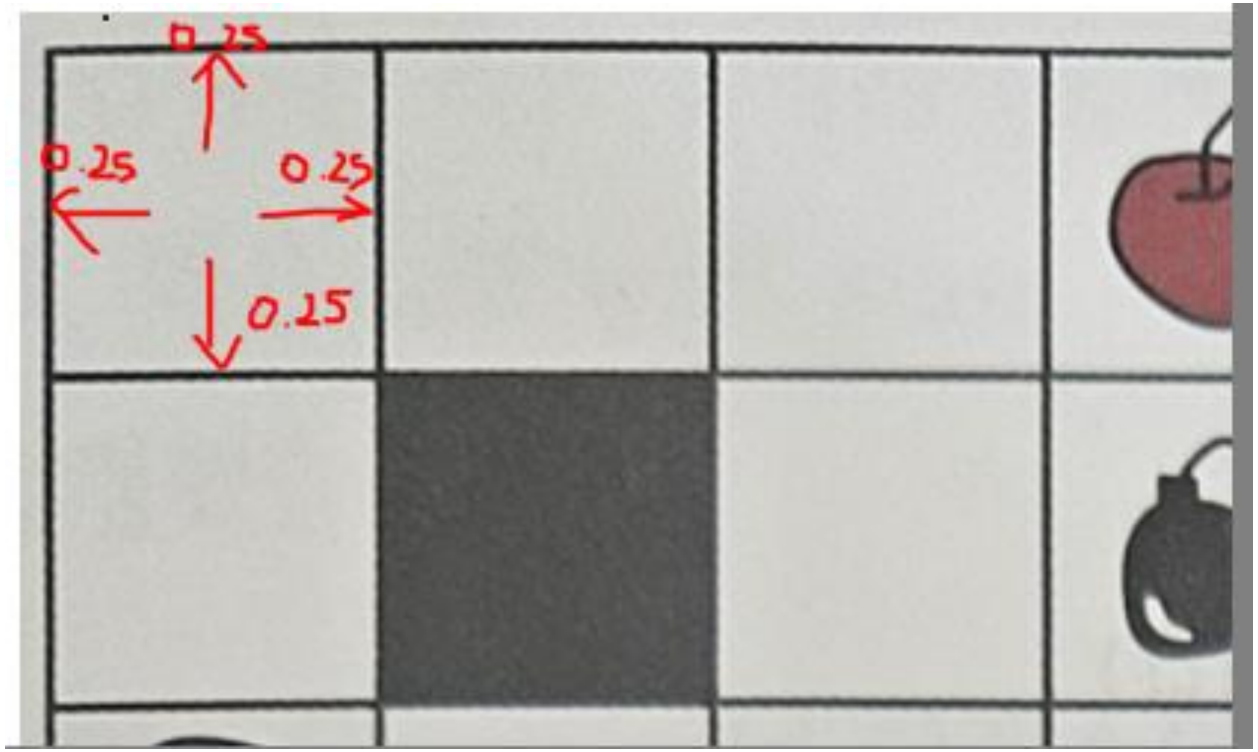
DP 예시 2

그림 4-8 3×4 그리드 월드

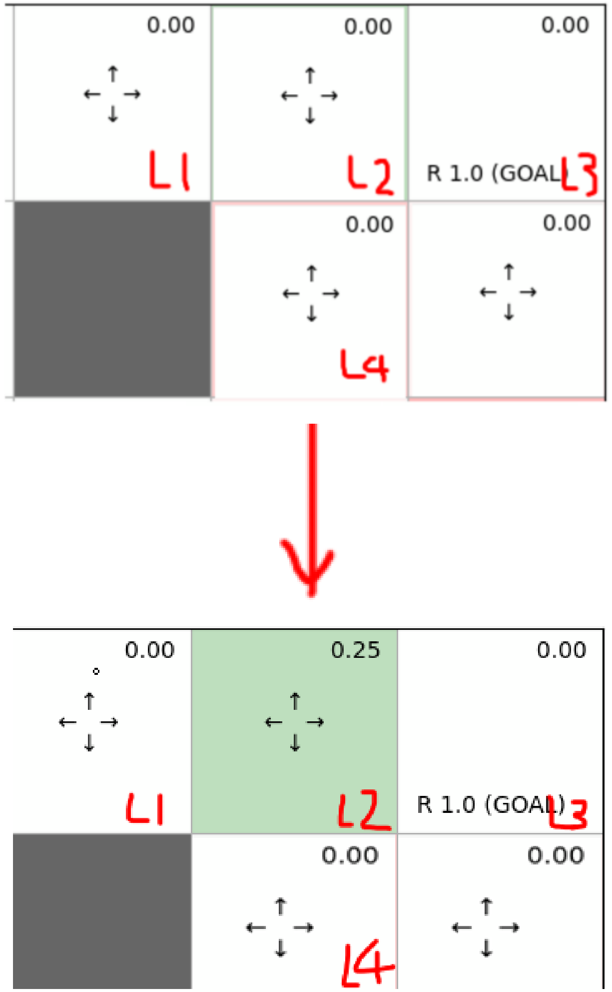


초기 상태

- 랜덤 방향으로 움직인다고 가정했을 때,
  각 칸의 가치를 구할 것임



정책 평가의 작동

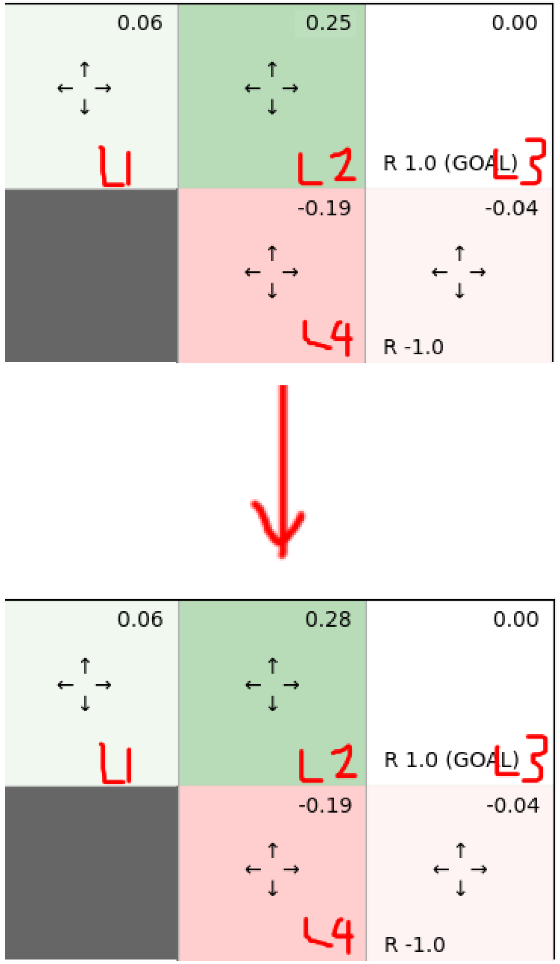


$$V_{k+1}(s) = \Sigma_{a,s'}\,\pi(a|s)\,p(s'|s,a)\,\{r(s,a,s') + \gamma V_k(s')\}$$

- 왼쪽: 0.25 * 1 * (0 + 0.9 * 0) = 0
- 오른쪽: 0.25 * 1 * (1 + 0.9 * 0) = 0.25
- 위쪽: 0.25 * 1 * (0 + 0.9 * 0) = 0
- 아래쪽: 0.25 * 1 * (0 + 0.9 * 0) = 0

-> L2의 가치 추정치 (1번째) : 0 + 0.25 + 0 + 0 = 0.25

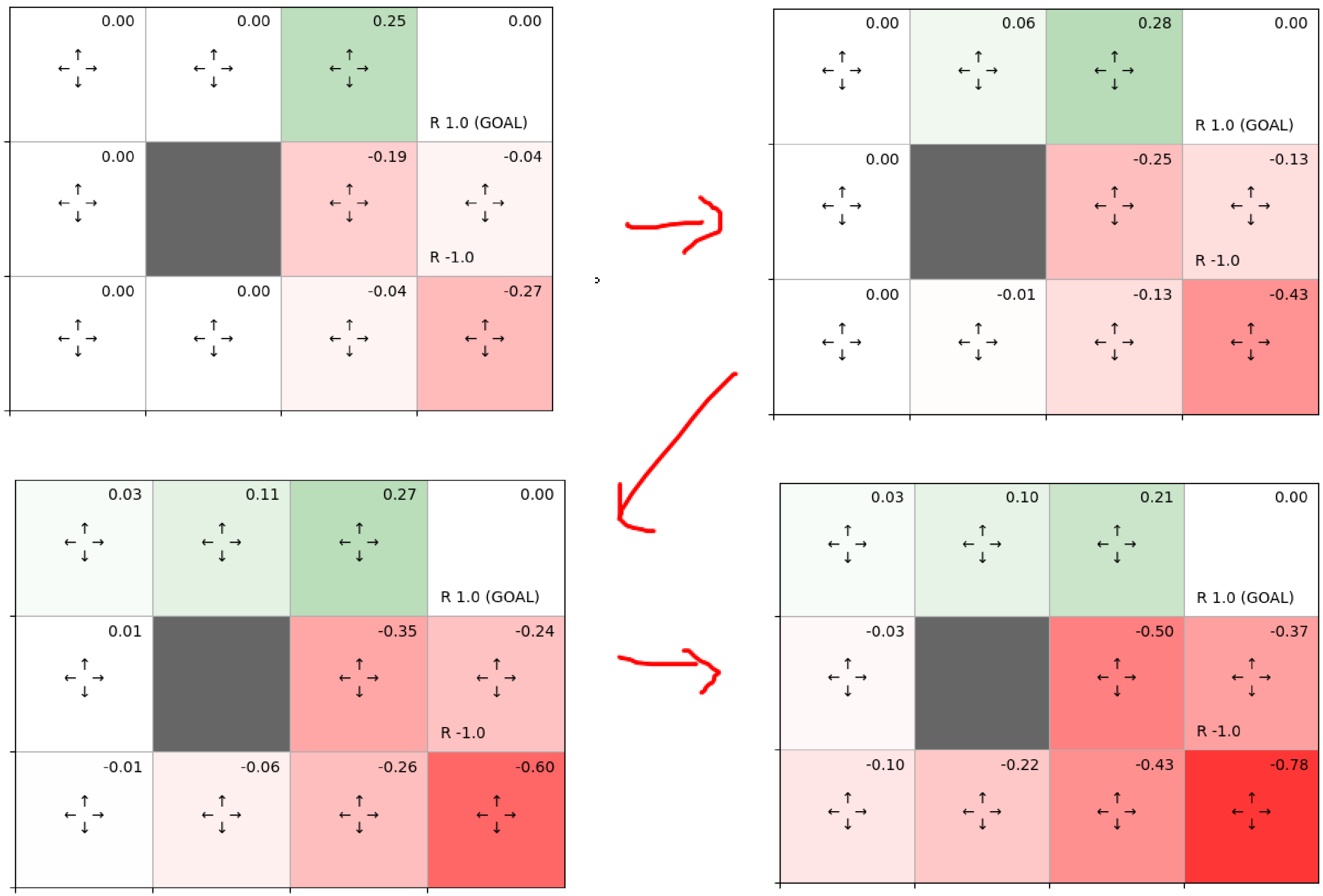
정책 평가의 작동



$$V_{k+1}(s) = \Sigma_{a,s'} \pi(a|s)\, p(s'|s,a) \{r(s,a,s') + \gamma V_k(s')\}$$

- 왼쪽: 0.25 * 1 * (0 + 0.9 * 0.06) = 0.013
- 오른쪽: 0.25 * 1 * (1 + 0.9 * 0) = 0.250
- 위쪽: 0.25 * 1 * (0 + 0.9 * 0.25) = 0.056
- 아래쪽: 0.25 * 1 * (0 + 0.9 * -0.19) = -0.042

-> L2의 가치 추정치 (2번째) : 0.013 + 0.250 + 0.056 – 0.042 = 0.28


0.00
0.00
0.25
0.00
R 1.0 (GOAL)
0.00
-0.19
-0.04
R -1.0
0.00
0.00
-0.04
-0.27
0.00
0.06
0.28
0.00
R 1.0 (GOAL)
0.00
-0.25
-0.13
R -1.0
0.00
-0.01
-0.13
-0.43
0.03
0.11
0.27
0.00
R 1.0 (GOAL)
0.01
-0.35
-0.24
R -1.0
-0.01
-0.06
-0.26
-0.60
0.03
0.10
0.21
0.00
R 1.0 (GOAL)
-0.03
-0.50
-0.37
R -1.0
-0.10
-0.22
-0.43
-0.78

정책 반복법

- 평가와 개선을 반복하는 알고리즘

최적 정책

최적 정책

$$\mu_*(s)$$
$$= argmax_a q_*(s, a)$$

> 상태 s가 주어졌을 때 최대 가치를 내는 행동 a만을 하는 정책 [탐욕 정책]

임의의 결정적 정책

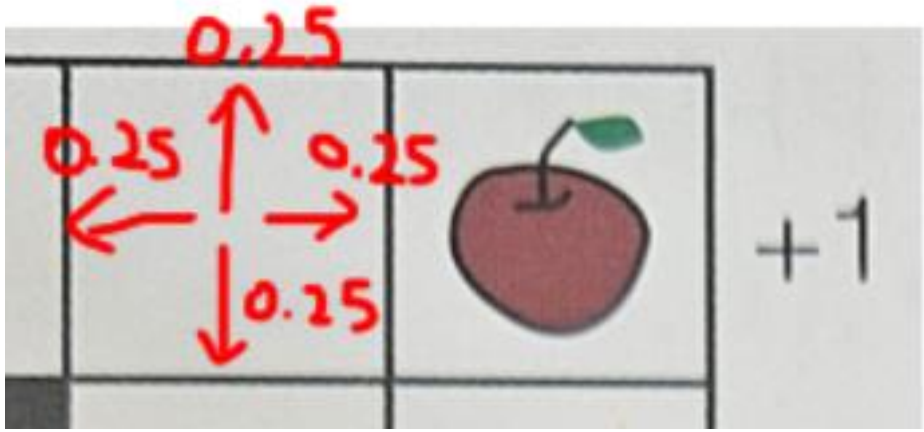
$$\mu'(s)$$
$$= argmax_a q_\mu(s,a)$$

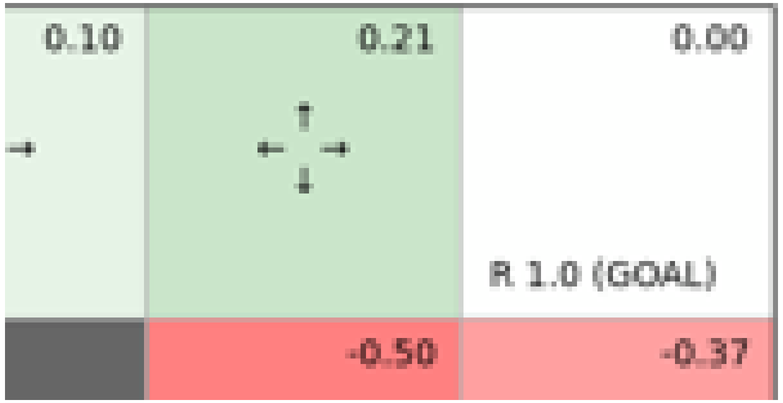
모든 상태 s에서 $\mu'(s)$가 갱신되지 않는다면, $\mu(s)$는 이미 최적 정책이라는 뜻

정책 반복법의 작동

1. π_0 정책에서 시작, π_0는 확률적일 수 있으므로 $\mu_0(s)$가 아닌 $\pi_0(s|a)$로 표기함
2. π_0의 가치 함수를 평가하여 V_0를 얻음 (반복적 정책 평가 알고리즘 이용)
3. 가치 함수 V_0를 이용하여 탐욕화 수행, μ_1 정책을 획득
4. 1~3 반복

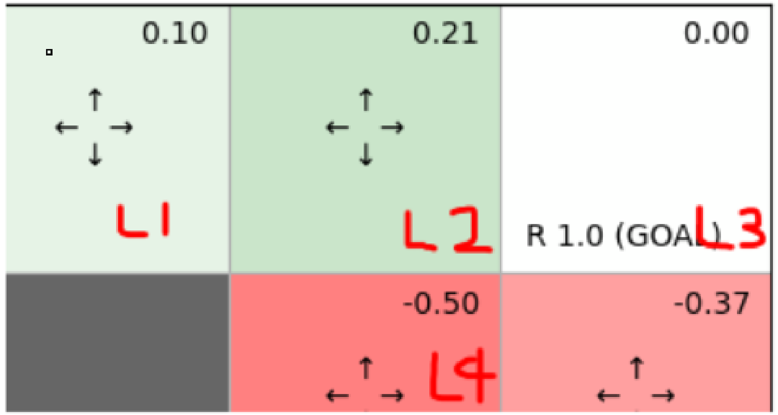
정책 반복법의 작동





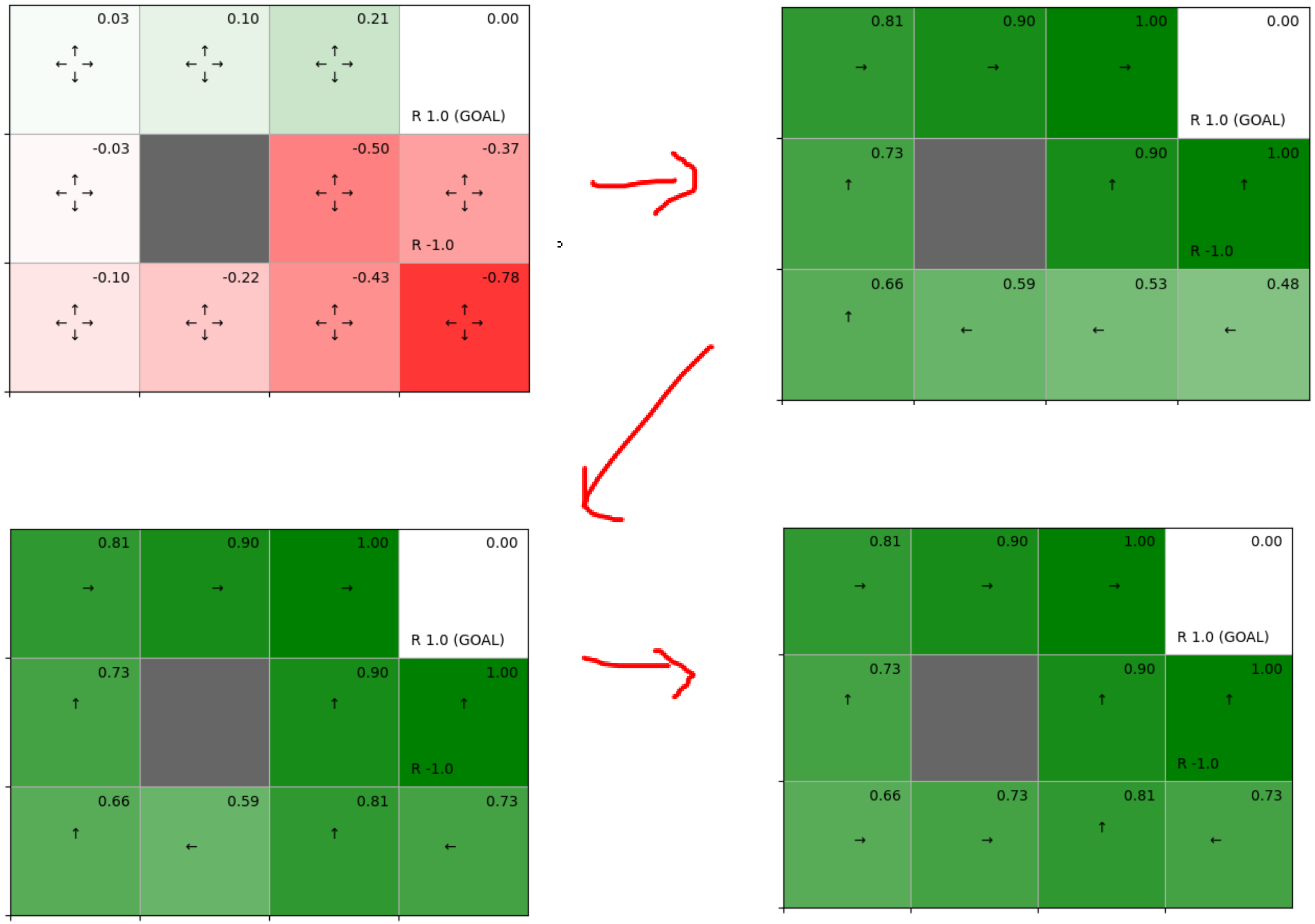
정책 반복법의 작동

$$\mu'(s) = argmax_a q_\mu(s,a) = argmax_a \Sigma_{s'} p(s'|s,a)\{r(s,a,s') + \gamma v_\mu(s')\}$$



- A = 왼쪽 -> 1 * (0 + 0.9 * 0.10) = 0.09
- A = **오른쪽** -> 1 * (1 + 0.9 * 0) = 1
- A = 위쪽 -> 1 * (0 + 0.9 * 0.21) = 0.189
- A = 아래쪽 -> 1 * (0 + 0.9 * -0.5) = -0.4

- -> L2에서 무조건 **오른쪽**으로 가게 정책 변경



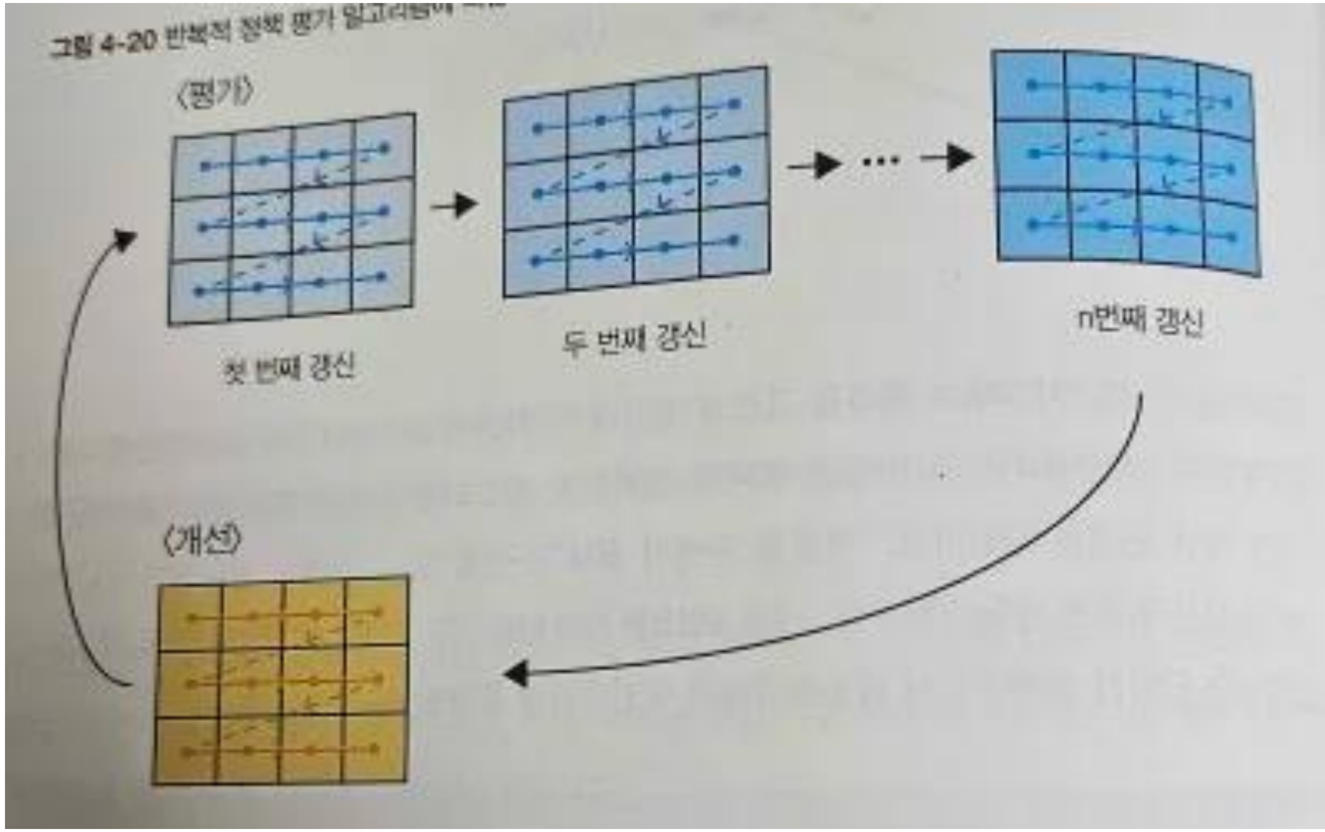
0.03
0.10
0.21
0.00
R 1.0 (GOAL)
-0.03
-0.50
-0.37
R -1.0
-0.10
-0.22
-0.43
-0.78
0.81
0.90
1.00
0.00
R 1.0 (GOAL)
0.73
0.90
1.00
R -1.0
0.66
0.59
0.53
0.48
0.81
0.90
1.00
0.00
R 1.0 (GOAL)
0.73
0.90
1.00
R -1.0
0.66
0.59
0.81
0.73
0.81
0.90
1.00
0.00
R 1.0 (GOAL)
0.73
0.90
1.00
R -1.0
0.66
0.73
0.81
0.73

가치 반복법

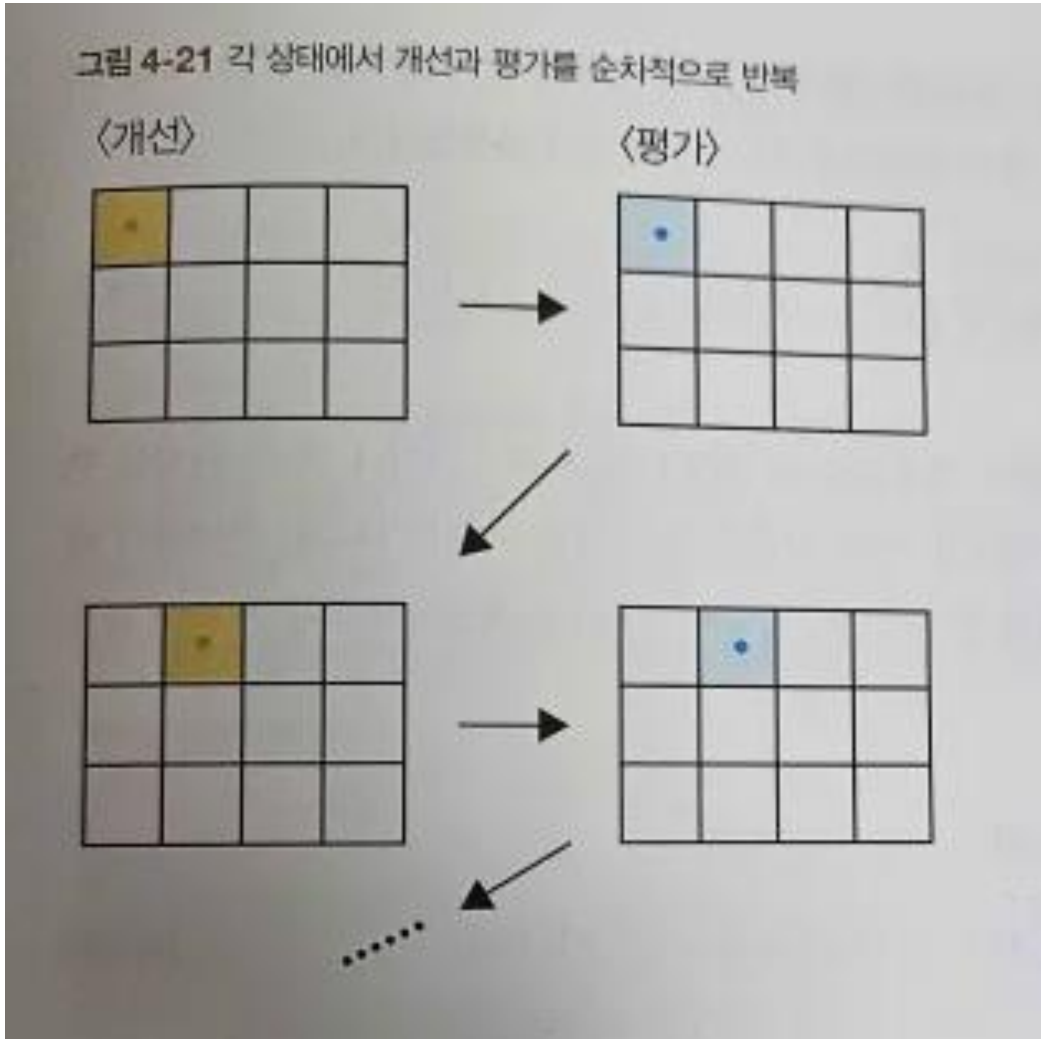
정책 반복법: 평가와 개선을 할 때, **모든 state와 action**에 대해 가치 함수와 정책을 업데이트함.
이때 가치 함수는 앞선 DP 알고리즘에 의해 **여러 번** 갱신됨

가치 반복법: 하나의 상태만 1번 갱신하고 바로 개선

- 정책 반복법



- 가치 반복법



가치 반복법

개선 단계

$$\mu(s) = argmax_a \Sigma_{s'} p(s'|s,a)\{r(s,a,s') + \gamma V(s')\}$$

평가 단계 결정적 정책

$$V'(s) = \Sigma_{s'}\, p(s'|s,a)\,\{r(s,a,s') + \gamma V(s')\}$$

가치 반복법

개선 단계

$$\mu(s) = argmax_a \Sigma_{s'} p(s'|s,a)\{r(s,a,s') + \gamma V(s')\}$$

평가 단계 결정적 정책

$$V'(s) = \Sigma_{s'}\, p(s'|s,a)\,\{r(s,a,s') + \gamma V(s')\}$$

이때, $p(s'|s,a)\,\{r(s,a,s') + \gamma V(s')\}$ 부분이 중복됨, 그러므로 다음과 같이 줄일 수 있음:

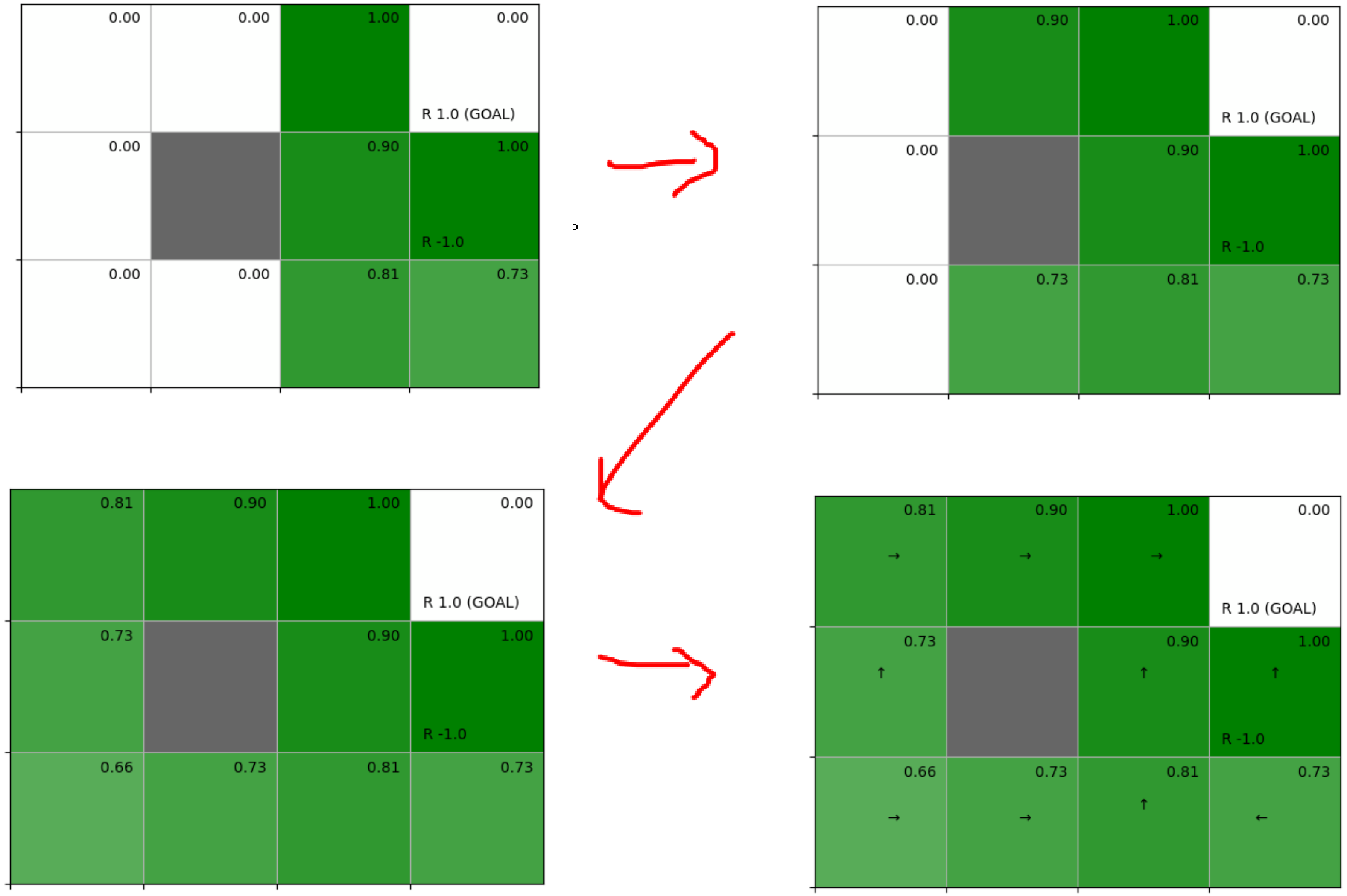
$V'(s) = max_a \Sigma_{s'}\, p(s'|s,a)\,\{r(s,a,s') + \gamma V(s')\}$

가치 반복법

$$V'(s) = max_a \Sigma_{s'}\ p(s'|s,a)\,\{r(s,a,s') + \gamma V(s')\}$$

- 어떤 상태의 가치:

그 상태에서 할 수 있는 모든 행동 중 최대 수익을 뽑아내는 것의 가치



0.00 0.00 1.00 0.00
R 1.0 (GOAL)
0.00 0.90 1.00
R -1.0
0.00 0.00 0.81 0.73
0.00 0.90 1.00 0.00
R 1.0 (GOAL)
0.00 0.90 1.00
R -1.0
0.00 0.73 0.81 0.73
0.81 0.90 1.00 0.00
R 1.0 (GOAL)
0.73 0.90 1.00
R -1.0
0.66 0.73 0.81 0.73
0.81 → 0.90 → 1.00 → 0.00
R 1.0 (GOAL)
0.73 ↑ 0.90 ↑ 1.00 ↑
R -1.0
0.66 → 0.73 → 0.81 ↑ 0.73 ←

정리

- 상태 전이 확률 $p(s'|s,a)$
- 보상 함수 $r(s,a,s')$
- 정책 $\pi(a|s)$

이걸로 벨만 방정식을 만들고, 연립방정식을 계산하여 $v_\pi(s)$를 얻음

- 연립방정식의 계산 방법:

1. v의 초기값을 아무거나 설정한다
2. 벨만 방정식 비스무리한 걸로 v를 갱신한다
3. 이렇게 DP를 활용해서 v를 정답과 더 가깝게 만들 수 있다

예고편

- 상태 전이 확률 $\cancel{p(s'|s,a)}$
- 보상 함수 $\cancel{r(s,a,s')}$
- 정책 $\cancel{\pi(a|s)}$ 몰?루

이걸로 벨만 방정식을 만들고, 연립방정식을 계산하여 $v_\pi(s)$를 얻음

- 연립방정식의 계산 방법:

1. v의 초기값을 아무거나 설정한다
2. 벨만 방정식 비스무리한 걸로 v를 갱신한다 (못함)
3. uhhhhh

끝

- 감사합니다