Chapter. 02

동적 계획법

강화학습의 근간: 동적계획법

FAST CAMPUS ONLINE

강화학습 A-Z I

강사. 박준영

I 지난 이야기…

- 마르코프 결정과정 (Markov decision process: MDP)
  - 정책함수 π
  - 상태 가치 함수 V_π
  - 행동 가치 함수 Q_π
  - 최적 가치 함수 V^*, Q^*
  - 최적 정책 함수 π^*
  - 최적 가치 함수 V^*, Q^* $\leftrightarrow$ 최적 정책 함수 π^*
- Bellman 기대값 방정식 (Bellman expectation equation: BEE)
- Bellman 최적 방정식 (Bellman expectation equation: BOE)
  - 직접해가 없다!

ㅣ“강화학습 문제” 와 “강화학습 문제의 풀이기법”

“강화학습 문제”

MDP

Markov **D**ecision **P**rocess
(마르코프 결정과정)

- MDP를 풀었다! = π^* 를 안다.
- π^* 를 어떻게 찾죠? BOE를 풀어보자!

“강화학습 문제의 풀이기법”

Dynamic **P**rogramming
(동적최적화)

DP

환경에 대해서 알 때

MC

TD

환경에 대해서 모를 때

I 동적 계획법 (Dynamic programming: DP)

동적 계획법은 복잡한(큰) 문제를 작은 문제로 나눈 후

작은 문제의 해법을 조합해 큰 문제의 해답을 구하는 기법의 총칭

(대부분의 경우, 반복적인 방식을 통해서 이루어짐)

벨만 저 '태풍의 눈' 에서 밝혀짐

Dynamic "동적" 시간에 대해서 변화, 여러 단계로 나뉘어짐 등을 표현

Programming "계획법" 1950 년대 미 공군에서 사용하던 용어

리처드 어니스트 **벨먼** (1920-1984)

- 1953년에 동적 계획법을 고안

I 동적 계획법 (Dynamic programming: DP)

Fibonacci 수열: $F_0 = 0, F_1 = 1, F_{n+2} = F_{n+1} + F_n$

$$F_6 = ??$$

해법 1) 그냥 계산한다 (Top-down)

$F_6 = \boldsymbol{F_5} + \boldsymbol{F_4}$

$\boldsymbol{F_5} = F_4 + F_3$

$F_4 = F_3 + F_2$

$F_3 = F_2 + F_1$

$F_2 = F_1 + F_0$

$F_2 = F_1 + F_0$

$F_3 = F_2 + F_1$

$F_2 = F_1 + F_0$

$F_4 = F_3 + F_2$

$F_3 = F_2 + F_1$

$F_2 = F_1 + F_0$

$F_2 = F_1 + F_0$

반복적인 구조가 나타난다!

Life Changing Education

FAST CAMPUS ONLINE

박준영 강사.

Fast campus

ㅣ동적 계획법 (Dynamic programming: DP)

Fibonacci 수열: $F_0 = 0, F_1 = 1, F_{n+2} = F_{n+1} + F_n$

$$F_6 = ??$$

해법 1) 그냥 계산한다 (Top-down)

$F_6 = \boldsymbol{F_5} + \boldsymbol{F_4}$

$\boldsymbol{F_5} = F_4 + F_3$
$F_4 = F_3 + F_2$
$F_3 = F_2 + F_1$
$F_2 = F_1 + F_0$
$F_2 = F_1 + F_0$
$F_3 = F_2 + F_1$
$F_2 = F_1 + F_0$

$F_4 = F_3 + F_2$
$F_3 = F_2 + F_1$
$F_2 = F_1 + F_0$
$F_2 = F_1 + F_0$

해법 2) 동적 계획법을 활용 (Bottom-up)

$F_6 = \boldsymbol{F_5} + \boldsymbol{F_4}$

$F_3 = F_2 + F_1$
$\boldsymbol{F_4} = F_3 + F_2$
$\boldsymbol{F_5} = F_4 + F_3$

조그마한 문제의 값을 먼저 알고 그 값을 재활용함으로써
계산량을 효과적으로 줄일 수 있다.

I 동적 계획법 (Dynamic programming: DP)

동적 계획법으로 해결할 수 문제는 다음과 같은 특징을 가진다

1. 최적 하위구조 (Optimal substructure)
   - 큰 문제를 분할한 작은 문제의 최적 값이 큰 문제에서도 최적 값임
   - Principle of optimality 라고도 불림
2. 중복 하위문제 (Overlapping problems)
   - 큰 문제의 해를 구하기 위해서 , 작은 문제의 최적 해를 재사용.
   - 여러 번의 재사용을 하기 때문에 일반적으로 테이블에 저장해 둠.

MDP 에서 정의한 Bellman 기대/최적 방정식은 두 가지 특성을 만족시킨다!

즉, 우리는 DP 를 활용해 Bellman 기대/최적 방정식의 해를 효율적으로 계산할 수 있다.

I 정책 평가 (policy evaluation : PE)

정책 평가 (Policy evaluation) 는 반복적인 과정을 통해 'Bellman 기대값 방정식'의 해를 구하는 방법 중 하나.

정책 평가 알고리즘

꼭 0 이 아니라 어떤 값으로 시작해도 하나의 값으로 알고리즘의 결과는 수렴함.

초기화 $V_0^\pi(s) \leftarrow 0$ 모든 $s \in \mathcal{S}$

반복 $(t = 0, \dots,)$:

각 상태에 s 대하여 :

"행렬표현: $V_{t+1}^\pi = R^\pi + \gamma P^\pi V_t^\pi$"

$$V_{t+1}^\pi(s) \leftarrow \sum_{a \in \mathcal{A}} \pi(a|s) \left(R_s^a + \gamma \sum_{s' \in \mathcal{S}} P_{ss'}^a V_t^\pi(s') \right)$$

까지 $\max_{s \in \mathcal{S}} |V_{t+1}^\pi(s) - V_t^\pi(s)| \le \epsilon$

MDP 강의록에 있던 그 식?

Q) 이 알고리즘이 과연 유일한 하나의 값으로 수렴하나요?
A) 네, 수렴합니다. 이번 강의의 마지막에 다시 설명하겠습니다.

I 정책 개선 (Policy Improvement: PI)

정책 개선 알고리즘

입력: 현재 정책 π, 가치함수 $V^{\pi}(s)$

출력: 개선된 정책 π'

1. $Q^{\pi}(s,a) = R_s^a + \gamma \sum_{s' \in \mathcal{S}} P_{ss'}^a V^{\pi}(s')$ 의 관계식을 활용해 $Q^{\pi}(s,a)$ 계산.

2. $\pi'(a|s) = \begin{cases} 1, & if\ a = \underset{a \in \mathcal{A}}{\operatorname{argmax}}\, Q^{\pi}(s,a) \\ 0, & otherwise \end{cases}$

과연 이렇게 만들어진 π' 가 π 보다 좋을까? ($\pi' \geq \pi$ **??**)

(정책함수 π 간의 대소관계를 다음과 같이 정의 한다. $\pi' \geq \pi$ 만약 $V_{\pi'}(s) \geq V_{\pi}(s), \forall\, s \in \mathcal{S}$)

I 정책 개선 정리 (Policy improvement theorem)

개선 후 정책: $\pi'(s) \overset{\text{def}}{=} \underset{a\in\mathcal{A}}{\operatorname{argmax}}\, Q^\pi(s,a)$

$$V^\pi(s) = Q^\pi(s,\pi(s)) \le \max_{a\in\mathcal{A}} Q^\pi(s,a) = Q^\pi(s,\pi'(s))$$

$$V^\pi(s) \le Q^\pi(s,\pi'(s)) \quad \boxed{1}$$

$$= \mathbb{E}_{\pi'}\left[R_{t+1}^{\pi'(s_t)} + \gamma V^\pi(s_{t+1}) | s_t = s\right] \quad \boxed{2}$$

$$\le \mathbb{E}_{\pi'}\left[R_{t+1}^{\pi'(s_t)} + \gamma Q^\pi(s_{t+1}, \pi'(s_{t+1})) | s_t = s\right] \quad \boxed{3}$$

$$= \mathbb{E}_{\pi'}\left[R_{t+1}^{\pi'(s_t)} + \gamma \mathbb{E}_{\pi'}\left[R_{t+2}^{\pi'(s_{t+1})} + \gamma V^\pi(s_{t+2})\right] | s_t = s\right] \quad \boxed{4}$$

$$= \mathbb{E}_{\pi'}\left[R_{t+1}^{\pi'(s_t)} + \gamma R_{t+2}^{\pi'(s_{t+1})} + \gamma^2 V^\pi(s_{t+2}) | s_t = s\right] \quad \boxed{5}$$

$$\le \mathbb{E}_{\pi'}\left[R_{t+1}^{\pi'(s_t)} + \gamma R_{t+2}^{\pi'(s_{t+1})} + \gamma^2 Q^\pi(s_{t+2}, \pi'(s_{t+2})) | s_t = s\right] \quad \boxed{6}$$

$$= \mathbb{E}_{\pi'}\left[R_{t+1}^{\pi'(s_t)} + \gamma R_{t+2}^{\pi'(s_{t+1})} + \gamma^2 \mathbb{E}_{\pi'}[r_{t+3} + \gamma V^\pi(s_{t+3})] | s_t = s\right]$$

$$= \mathbb{E}_{\pi'}\left[R_{t+1}^{\pi'(s_t)} + \gamma R_{t+2}^{\pi'(s_{t+1})} + \gamma^2 R_{t+3}^{\pi'(s_{t+2})} + \gamma^3 V^\pi(s_{t+3}) | s_t = s\right]$$

$$= \mathbb{E}_{\pi'}\left[R_{t+1}^{\pi'(s_t)} + \gamma R_{t+2}^{\pi'(s_{t+1})} + \gamma^2 R_{t+3}^{\pi'(s_{t+2})} + \gamma^3 R_{t+4}^{\pi'(s_{t+3})} + \cdots | s_t = s\right]$$

$$= V^{\pi'}(s)$$

$(1) \rightarrow (2)$: Q^π 와 $V^\pi(s)$ 의 관계식 활용: MDP 강의 10 Page

$(2) \rightarrow (3)$: 파란색 수식 활용

$(3) \rightarrow (4)$: Q^π 와 $V^\pi(s)$ 의 관계식 활용: MDP 강의 10 Page

$(4) \rightarrow (5)$: 안쪽 $\mathbb{E}_{\pi'}[\cdot]$ 밖으로 항들을 끌어냄. 밖 $\mathbb{E}_{\pi'}[\cdot\,|s_t]$ 때문에 가능

$(5) \rightarrow (6)$: 파란색 수식 활용

PI 알고리즘에 의해 구해진 π' 는 $\pi' \ge \pi$ 을 만족한다!

정책 반복 (Policy iteration)

정책 반복 (Policy iteration) 은 (1) 정책 평가와 (2) 정책 개선을 적용해 Bellman 방정식을 푸는 알고리즘.

정책 반복 (Policy iteration)

입력: 임의의 정책 정책 π

출력: 개선된 정책 π'

1. 정책 평가 (PE) 를 적용해 $V^{\pi}(s)$ 계산
2. 정책 개선 (PI) 를 적용해 π' 계산

I 정책 반복 알고리즘을 활용한 최적 정책계산

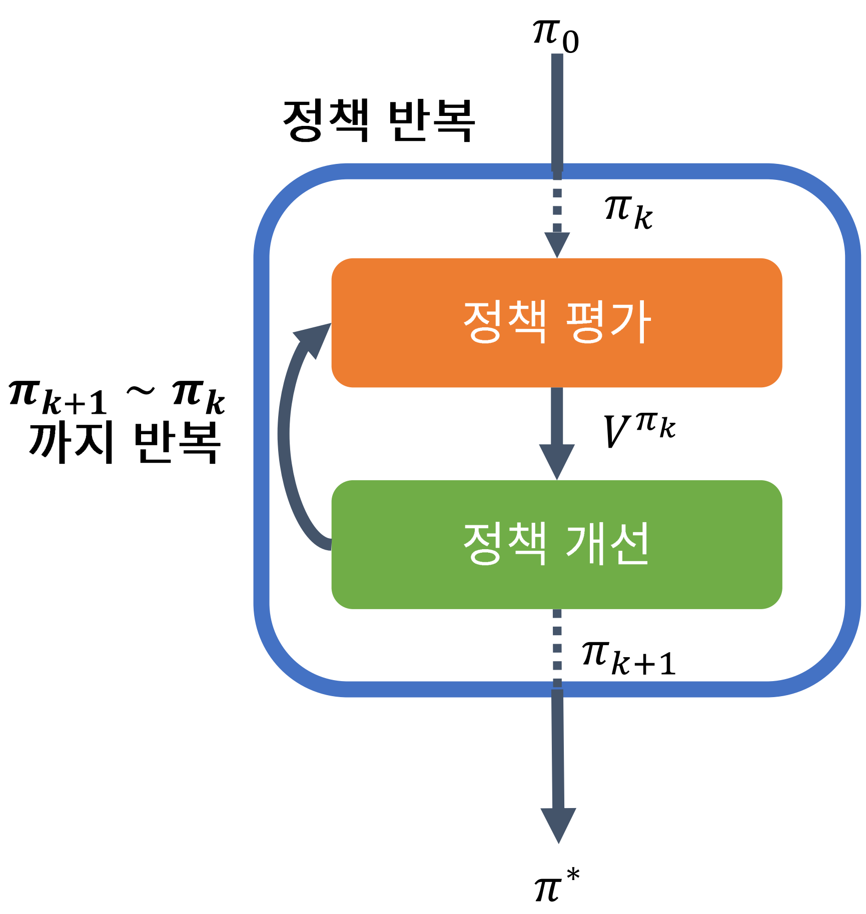
입력: 임의의 정책 π_0

출력: 최적 정책 π^*

반복: $(k = 0, \ldots,)$

- $\pi_{k+1} \leftarrow$ 정책 반복 (π_k)
- 만약 $\pi_{k+1} \sim \pi_k$, 반복문 탈출

최적 정책 $\pi^* \leftarrow \pi_{k+1}$



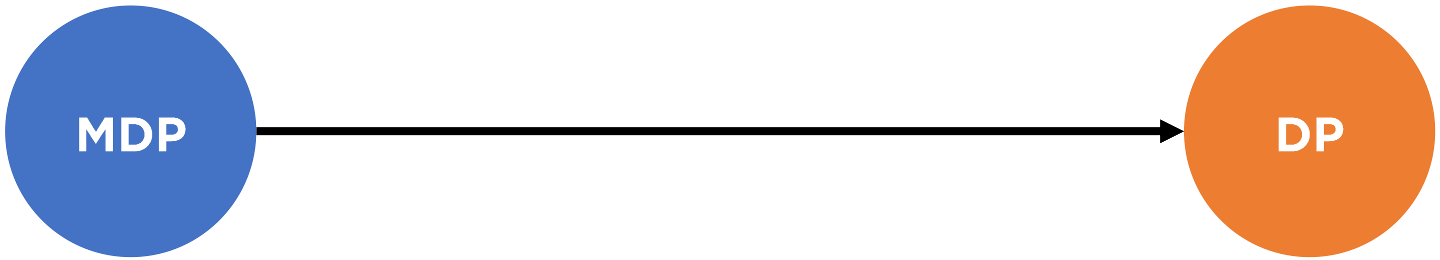
| 큰 그림 돌아보기

<MDP 강의>

<DP 강의>

"강화학습 문제"

"강화학습 문제의 풀이기법"



목적: (1) 최적 상태 $V^*(s)$, 행동 가치함수 $Q^*(s,a)$

(2) 최적 정책 π^*

"최적 가치함수 ↔ 최적 정책"
즉, (1), (2) 중 하나만 알면 나머지는 쉽게 알 수 있음

Bellman 기댓값/최적 방정식 및 그 구조를 활용해
최적 가치함수들이 동적계획법 (DP) 를 활용해 풀 수 있는 형태임을 보임.

정책 반복 (Policy iteration): DP를 활용해 효율적으로 최적 정책을 계산하는 방법

- 정책 평가 (Policy evaluation) / 정책 개선 (Policy improvement) 가 사용됨.

정책 반복을 통해 $V^*(s), Q^*(s,a), \pi^*$ 을 계산

즉, MDP 를 DP를 활용해서 효율적으로 풀 수 있다.

그리고 남은 질문, (Q) 정책 평가 알고리즘은 유일한 정답으로 수렴하나요?

ㅣ벡터의 크기를 재는 방법 Norm

벡터의 p-norm $\|v\|_p = \sqrt[p]{\sum_{i=1}^{n}(v(i))^p}$

$v \in \mathbb{R}^n$

벡터의 ∞-norm $\|v\|_\infty = \max_{i=1,\ldots,n} v(i)$

예시) $v = (3, 4)$

$\|v\|_2 = \sqrt{3^2 + 4^2} = 5$: (Euclidean 거리)

$\|v\|_\infty = \max(\{3,4\}) = 4$: 가장 큰 원소

Bellman expectation backup 오퍼레이터

$$T^{\pi}(V) \stackrel{\text{def}}{=} R^{\pi} + \gamma P^{\pi} V$$

$T^{\pi}(V)$ 는 Bellman expectation backup 오퍼레이터 (연산자) 라고 불림.

< γ – 수축 사상>

$T^{\pi}(\cdot)$ 일종의 함수라고 고려했을 때 ($T^{\pi}(\cdot)$ 에 V를 넣으면 $R^{\pi} + \gamma P^{\pi} V$를 반환),

$T^{\pi}(\cdot)$ 는 'γ – 수축 사상' 이다. 즉, $T^{\pi}(u)$ 와 $T^{\pi}(v)$ 사이의 거리는 u 와 v 사이의 거리보다 가깝다.

<증명>

$$\begin{aligned}\|T^{\pi}(u) - T^{\pi}(v)\|_p &= \|(R^{\pi} + \gamma P^{\pi} u) - (R^{\pi} + \gamma P^{\pi} v)\|_p \\ &= \|\gamma P^{\pi}(u - v)\|_p \\ &= \gamma \|P^{\pi}(u - v)\|_p \\ &\le \gamma \|u - v\|_p\end{aligned}$$

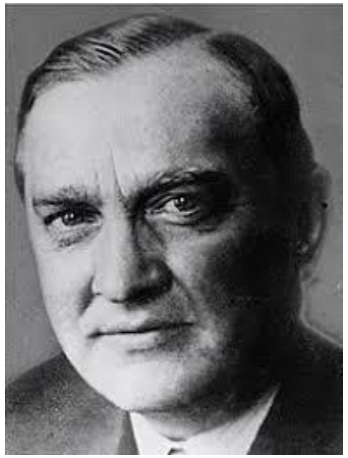
즉, 서로 다른 벡터 u, v 의 거리보다 $T^{\pi}(u), T^{\pi}(v)$ 의 거리가 최소 γ 배 작다.

바나흐 고정점 정리

바나흐 고정점 정리 (Banach fixed-point theorem)

완비 거리 공간 $\mathcal{V}$ 에서 정의된 오퍼레이터 $\mathrm{T}(v)$가 $\gamma-$ 축약사상이면 다음이 성립한다.

- $\mathrm{T}(v)$ 를 계속 적용하면, v는 유일한 고정점 v^*로 수렴한다.
- 이때 수렴속도는 γ이다.



스테판 바나흐 (1892-1945)

아...... 그런데 뭐라고요?

정책 평가 알고리즘과 바나흐 고정점 정리

알려진 사실

1. 정책 평가 알고리즘의 하나의 반복은 $T^{\pi}(V) \stackrel{\text{def}}{=} R^{\pi} + \gamma P^{\pi} V$ 이다.
2. $T^{\pi}(\cdot)$ 은 $\gamma -$ 축약사상 이다
3. V 들은 완비 거리공간 안에 있다.
4. V^{π}은 Bellman 기대 값 방정식의 유일한 해다

(1) V 가 "완비 거리 공간" 조건을 만족 (2) $T^{\pi}(V)$ "$\gamma -$ 축약사상" 조건을 만족

"바나흐 고정점 정리"에 의해 유일해가 존재함을 보일 수 있음. 그때 유일해는 $\boldsymbol{V^{\pi}}$.우리가 원하는 것!

즉, 임의의 V에 $T^{\pi}(\cdot)$을 연속적으로 적용하면, $\boldsymbol{V^{\pi}}$가 됨!

정책 평가 알고리즘과 바나흐 고정점 정리