Chapter. 03

모델없이 세상 알아가기

도박의 도시 Monte-carlo 그리고 MC 정책추정

FAST CAMPUS ONLINE

강화학습 A-Z I

강사. 박준영

ㅣ지난 이야기…

“강화학습 문제의 풀이기법”

<챕터 01>

“강화학습 문제”

MDP

<챕터 02>

DP

MC

TD

환경에 대해서 알 때

환경에 대해서 모를 때

- 정책 반복 (Policy iteration)
  - 정책 평가
  - 정책 개선
- 가치 반복 (Value iteration)
- 비동기 DP

\+ (상대적으로) 문제를 해결하기 쉬움

\+ 매우 효율적임

\- 현실적이지 않음

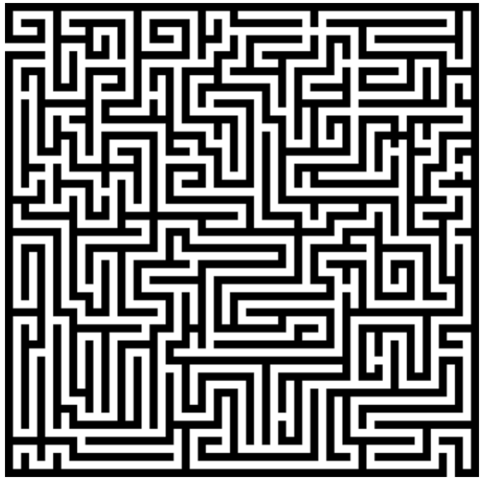
???

ㅣ강화 “학습”

Dynamic programming

$$R^{\pi} = \begin{bmatrix} R_1 \\ R_2 \\ R_3 \end{bmatrix} \quad P^{\pi} = \begin{bmatrix} P_{11} & P_{12} & P_{13} \\ P_{11} & P_{12} & P_{23} \\ P_{11} & P_{12} & P_{33} \end{bmatrix}$$

행동 (action)



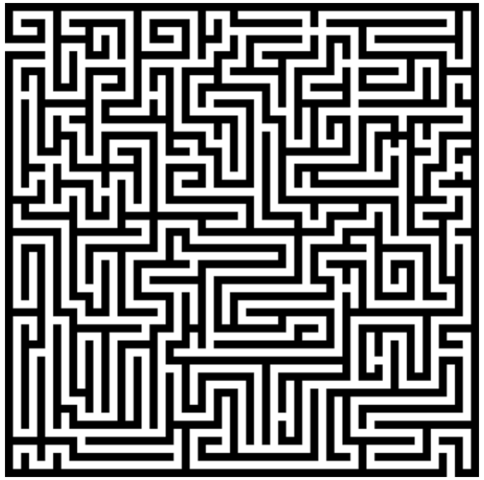
| 강화 “학습”

$$R^{\pi} = \begin{bmatrix} ?? \\ ?? \\ ?? \end{bmatrix} \quad P^{\pi} = \begin{bmatrix} ?? & ?? & ?? \\ ?? & ?? & ?? \\ ?? & ?? & ?? \end{bmatrix}$$

$V^{\pi} = ??\ Q^{\pi} = ??$, ℗=??

행동 (action)

상태 (state)
보상 (reward)



세상에 대한 지식이 없기 때문에, 환경과 상호작용을 통해 <u>가치함수</u> 및 <u>정책</u> 혹은 <u>환경의 모델</u>을 추산한다.

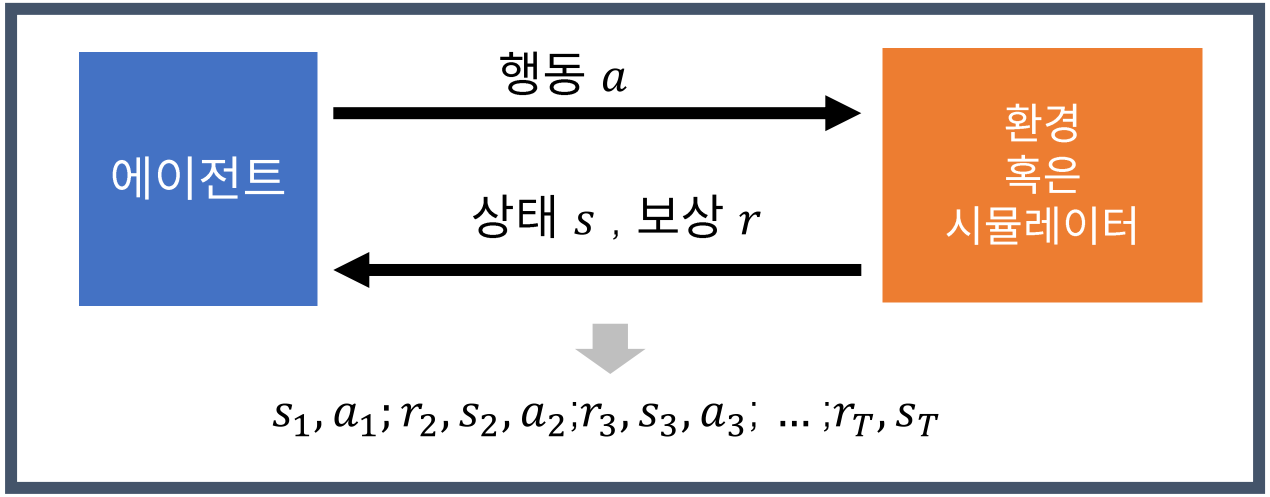
<파트 2. 가치기반 강화학습><파트 6. 모델기반 강화학습>

<파트 4. 정책 최적화>

FAST CAMPUS
ONLINE

박준영 강사.

강화학습의 템플릿



강화학습 템플릿

반복 $t = 1, 2, 3, \ldots$

행동을 결정 $a_t = \pi(s_t)$ (어떻게?)

환경에 a_t 을 가한 후, 보상 r_{t+1} 을 받고 새로운 상태 s_{t+1} 관측

상태 가치 함수 $V^{\pi}(s)$ 및 행동 가치 함수 $Q^{\pi}(s, a)$ 추정 (어떻게?)

Generalized Policy Iteration

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

일반화된 정책 반복 (Generalized Policy iteration)

입력: 임의의 정책 정책 π

출력: 개선된 정책 π'

1. **임의의 방식**을 활용해 적용해 $V^{\pi}(s)$ 계산
2. **임의의 방식**을 활용해 적용해 π' 계산 ($\pi' \geq \pi$ 를 만족)

1. 어떻게 $V^{\pi}(s)$ 을 계산할 것인가? – 가치 추산
2. 어떻게 π' 을 계산할 것인가? – 정책 개선

I Model free 가치 추산 알고리즘의 분류

Non-Bootstrap		*Bootstrap
몬테 카를로 기법 (Monte Carlo methods: MC)	TD(λ)	Temporal-difference 기법 (TD)

*부츠 (Boot) 끈 (strap) 을 묶을 때,
아래의 끈 묶음이 위의 끈 묶음에 영향을 준다!

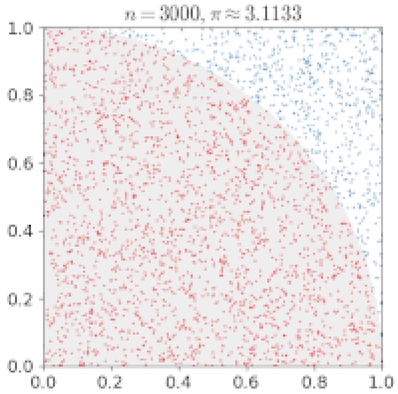
I 몬테 카를로 (Monte Carlo)

도박의 도시 "몬테 카를로"

몬테 카를로 기법:
계산하기 어려운 값을 수 많은 확률 시행을 거쳐 추산하는 기법
스타나스와프 울람이 명명.

1930 년, **엔리코 페르미**가 중성자의 특성을 연구하기 위해 처음으로 사용됨.
1940 년대, '맨하튼 계획'에서 원자폭탄 폭발반경을 계산하기 위해서도 사용.

<몬테카를로 기법을 활용한 원주율 π 계산>



점을 $[0, R] \times [0, R]$ 에서
임의로 n 개를 생성

$$\frac{\pi R^2}{4} = \frac{\text{원의 넓이}}{4} \approx \frac{\text{빨간점의 수}}{\text{점의 수}}$$

I 몬테카를로 기법을 활용한 가치 함수 추산

목적: 주어진 정책 π 에 대해서, $V^{\pi}(s)$ 를 추산

$$V_{\pi}(s) = \mathbb{E}_{\pi}[G_t | S_t = s]$$

$$G_t \stackrel{\text{def}}{=} R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ \ldots + \gamma^{T-1} R_T$$

T 는 확률 변수로 "에피소드" 가 끝나는 시점을 의미

<몬테카를로 기법>

G_t 를 여러 번 시뮬레이션해서 그 시뮬레이션 값의 평균을 계산하면 $V_{\pi}(s)$ 과 비슷해진다!

<수학적 특성>

- MC 는 불편추정기법이기 때문에 시뮬레이션의 횟수가 늘어나면 그 추정치가 참 값과 같아진다.
- MC 시뮬레이션의 횟수가 늘어나면 추정치의 분산이 줄어든다 (= 시뮬레이션 횟수가 적으면 추정치의 불확실성이 커짐.)

I 최초 방문 몬테 카를로 정책 추정 (First-visit Monte Carlo policy evaluation)

데이터 (정책 π 을 따라서 생성): $s_t \in \mathcal{S} = \{s^1, s^2, s^3\}, a_t \in \mathcal{A} = \{a^1, a^2, a^3\}$

Episode 1: $s_1, a_1, r_1; s_2, a_2, r_2;\ s_3, a_3, r_3;\ \dots;\ s_T, a_T, r_T$
Episode 2: $s_1, a_1, r_1; s_2, a_2, r_2;\ s_3, a_3, r_3;\ \dots;\ s_T, a_T, r_T$
Episode 3: $s_1, a_1, r_1; s_2, a_2, r_2;\ s_3, a_3, r_3;\ \dots;\ s_T, a_T, r_T$

(상태, 행동, 보상) 정책 π 을 따라서 생성

Episode 1 : $(s^1, a^2, 1); (s^3, a^1, 5);\ (s^2, a^3, 3), (s^1, a^3, 10),\ (s^2, a^2, 2)$
Episode 2 : $(s^3, a^1, 5);\ (s^2, a^2, 2); (s^1, a^2, 1); (s^2, a^3, 3), (s^1, a^3, 10)$
Episode 3 : $(s^2, a^3, 3); (s^1, a^2, 1); (s^3, a^1, 5); (s^1, a^3, 10),\ (s^2, a^2, 2)$

($\gamma = 1$ 을 가정)

Episode 1: s^1 에 해당하는 $G_t = 1 + 5 + 3 + 10 + 2 = 21$

Episode 2: s^1 에 해당하는 $G_t = 1 + 3 + 10 = 14$

Episode 3: s^1 에 해당하는 $G_t = 1 + 5 + 10 + 2 = 19$

에피소드내에서 **최초로 방문한** s^1 에 대해서만 리턴을 계산.

$V^\pi(s^1)$ = 모든 에피소드에 대한 리턴의 산술 평균

$$= \frac{21+14+19}{3} = 14.6$$

$$V_\pi(s^1) = \mathbb{E}_\pi[G_t |\, s_t = s^1] \approx \frac{1}{N}\sum_{n=1}^{N} G_n^\pi(s^1)$$

$G_n^\pi(s^1)$: 정책 π를 따르때, n 번째 상태 s^1의리턴 추정치.
(좋은 노테이션은 아니니, 의미로만 받아들여주세요)

I 모든 방문 몬테 카를로 정책 추정 (Every-visit Monte Carlo policy evaluation)

데이터 (정책 π 을 따라서 생성): $s_t \in \mathcal{S} = \{s^1, s^2, s^3\}, a_t \in \mathcal{A} = \{a^1, a^2, a^3\}$

Episode 1: $s_1, a_1, r_1; s_2, a_2, r_2; \; s_3, a_3, r_3; \; \dots; \; s_T, a_T, r_T$
Episode 2: $s_1, a_1, r_1; s_2, a_2, r_2; \; s_3, a_3, r_3; \; \dots; \; s_T, a_T, r_T$
Episode 3: $s_1, a_1, r_1; s_2, a_2, r_2; \; s_3, a_3, r_3; \; \dots; \; s_T, a_T, r_T$

(상태, 행동, 보상) 정책 π 을 따라서 생성

Episode 1 : $(s^1, a^2, 1); (s^3, a^1, 5); \; (s^2, a^3, 3), (s^1, a^3, 10), \; (s^2, a^2, 2)$
Episode 2 : $(s^3, a^1, 5); \; (s^2, a^2, 2); (s^1, a^2, 1); (s^2, a^3, 3), (s^1, a^3, 10)$
Episode 3 : $(s^2, a^3, 3); (s^1, a^2, 1); (s^3, a^1, 5); (s^1, a^3, 10), \; (s^2, a^2, 2)$

($\gamma = 1$ 을 가정) 에피소드내에서 **방문한 모든** s^1 에 대해서만 리턴을 계산.

Episode 1: s^1 에 해당하는 $G_t = 1 + 5 + 3 + 10 + 2 = 21$; s^1 에 해당하는 $G_t = 10 + 2 = 12$

Episode 2: s^1 에 해당하는 $G_t = 1 + 3 + 10 = 14$; s^1 에 해당하는 $G_t = 10 = 10$

Episode 3: s^1 에 해당하는 $G_t = 1 + 5 + 10 + 2 = 19$; s^1 에 해당하는 $G_t = 10 + 2 = 12$

$V^\pi(s^1)$ = 모든 에피소드에 대한 리턴의 산술 평균

$$= \frac{21+12+14+10+19+12}{6} = 15$$

First-visit MC vs. Every-visit MC?
(현실에서는 Every-visit MC 가 선호되는 편입니다.)

만약, 계속적으로 새로운 데이터가 생긴다면?

데이터 배치 1 (정책 π 을 따라서 생성):

Episode 1: $s_1, a_1, r_1; s_2, a_2, r_2;\ s_3, a_3, r_3;\ \dots;\ s_T, a_T, r_T$
Episode 2: $s_1, a_1, r_1; s_2, a_2, r_2;\ s_3, a_3, r_3;\ \dots;\ s_T, a_T, r_T$
Episode 3: $s_1, a_1, r_1; s_2, a_2, r_2;\ s_3, a_3, r_3;\ \dots;\ s_T, a_T, r_T$
…
Episode n: $s_1, a_1, r_1; s_2, a_2, r_2;\ s_3, a_3, r_3;\ \dots;\ s_T, a_T, r_T$

MC 가치평가 $\rightarrow V_\pi^1(s)$

몬테카를로 기법은 더욱 많은 시뮬레이션을 통해 더욱 정교한 가치 추산이 가능.

"더 많은 양의 데이터를 얻을 수 있다면, 더 정교한 가치 추산에 사용하는 것이 더욱 유리"

데이터 배치 2 (정책 π 을 따라서 생성):

Episode n+1: $s_1, a_1, r_1; s_2, a_2, r_2;\ s_3, a_3, r_3;\ \dots;\ s_T, a_T, r_T$
Episode n+2: $s_1, a_1, r_1; s_2, a_2, r_2;\ s_3, a_3, r_3;\ \dots;\ s_T, a_T, r_T$
Episode n+3: $s_1, a_1, r_1; s_2, a_2, r_2;\ s_3, a_3, r_3;\ \dots;\ s_T, a_T, r_T$
…
Episode 2n: $s_1, a_1, r_1; s_2, a_2, r_2;\ s_3, a_3, r_3;\ \dots;\ s_T, a_T, r_T$

MC 가치평가 $\rightarrow V_\pi^2(s)$

두번째 MC 가치평가를 수행하기 위해서, <u>기존의 모든 G_t</u> 를 저장해야 함.

- 매우 메모리 비효율적
- RL agent가 계속적으로 학습하는데 한계점으로 작용

| 배치 산술평균을 온라인 평균기법으로 변환

이동평균, 단계적 평균등과 비슷

μ_k : k시점까지의 평균 x_k : k시점의 데이터

(데이터 x_k 는 순차적으로 관측된다고 가정)

$$
\begin{aligned}
\mu_k &= \frac{1}{k}\sum_{j=1}^{k} x_j \\
&= \frac{1}{k}\left(x_k + \sum_{j=1}^{k-1} x_j\right) \\
&= \frac{1}{k}\left(x_k + (k-1)\mu_{k-1}\right) \\
&= \mu_{k-1} + \frac{1}{k}\left(x_k - \mu_{k-1}\right)
\end{aligned}
$$

"기존의 알고있던 지식" + "새로운 관측으로 바뀐 지식"

I Incremental MC policy evaluation

MC policy evaluation $V(s) \leftarrow \frac{S(s)}{N(s)}$

$S(s)$: 상태 s에 대한 G_t 들의 합
$N(s)$: 상태 s를 (처음) 방문한 횟수

Incremental
MC policy evaluation

상태 s (처음) 방문때마다,

$$N(s) \leftarrow N(s) + 1$$

$$V(s) \leftarrow V(s) + \frac{1}{N(s)}\left(G_t - V(s)\right)$$

현실에서는,
$N(s)$ 을 세는 것조차 어려움

$$V(s) \leftarrow V(s) + \alpha\left(G_t - V(s)\right)$$

"적당히 작은 α" 에 대해서 참값으로 수렴함이 증명되어 있음 [1]
α 는 학습 비율 (learning rate) 라고도 불림."

s가 실수인 경우/ s의 종류가 알려지지 않은 경우/ s가 이미지인경우 / $|\mathcal{S}|$ 가 매우 큰 경우 등등..

[1] A stochastic approximation method. Herbert Robbins and Sutton Monro (1951)

MC를 활용한 행동 가치함수 추산

상태 가치 함수 $V_\pi(s)$만으로는 (탐욕적) 정책 개선을 수행할 수 없다.

따라서, 행동 가치 함수 $Q^\pi(s,a)$ 를 추산하는 것이 필요하다.

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

(DP 에서도: $V^\pi(s) \xrightarrow{P,R} Q^\pi(s,a) \rightarrow \pi'$)

I MC를 활용한 행동 가치함수 추산

데이터 (정책 π 을 따라서 생성): $s_t \in \mathcal{S} = \{s^1, s^2, s^3\}, a_t \in \mathcal{A} = \{a^1, a^2, a^3\}$

Episode 1: $s_1, a_1, r_1; s_2, a_2, r_2; s_3, a_3, r_3; \dots; s_T, a_T, r_T$
Episode 2: $s_1, a_1, r_1; s_2, a_2, r_2; s_3, a_3, r_3; \dots; s_T, a_T, r_T$
Episode 3: $s_1, a_1, r_1; s_2, a_2, r_2; s_3, a_3, r_3; \dots; s_T, a_T, r_T$

(상태, 행동, 보상) 정책 π 을 따라서 생성

Episode 1 : $(s^1, a^2, 1); (s^3, a^1, 5); (s^2, a^3, 3), (s^1, a^3, 10), (s^2, a^2, 2)$
Episode 2 : $(s^3, a^1, 5); (s^2, a^2, 2); (s^1, a^2, 1); (s^2, a^3, 3), (s^1, a^3, 10)$
Episode 3 : $(s^2, a^3, 3); (s^1, a^2, 1); (s^3, a^1, 5); (s^1, a^3, 10), (s^2, a^2, 2)$

($\gamma = 1$ 을 가정)

Episode 1: (s^1, a^2)의 리턴 $= 1 + 5 + 3 + 10 + 2 = 21$

Episode 2: (s^1, a^2)의 리턴 $= 1 + 3 + 10 = 14$

Episode 3: (s^1, a^2)의 리턴 $= 1 + 5 + 10 + 2 = 19$

$Q^\pi(s^1, a^2)$ = 모든 에피소드에 대한 리턴의 산술 평균

$$= \frac{21+14+19}{3} = 14.6$$

에피소드내에서 **최초로 방문한** (s^1, a^2) 에 대해서만 리턴을 계산.

비슷한 방식으로
- "방문한 모든" (s^1, a^2) 에 대해 리턴을 계산하는 것도 가능
- Incremental 한 방식의 추산도 가능

I Monte Carlo Policy Evaluation

장점:

- 환경에 대한 선험적 지식이 필요 없음
- 직관적이고 구현하기 쉬움
- 항상 **정확한** 가치 함수 값을 계산함 (MC PE는 Value function 의 unbiased estimator)

단점:

- (엄밀하게는) **Episode가 끝나야만 적용 가능** 무한한 길이의 episode에 대해서도, 리턴 계산에서 충분히 작은 미래의 보상들을 무시하면 사용가능. 하지만 **불편추정치**의 특성을 잃게 됨.
- DP와는 다르게 각 상태와 행동의 관계에 대해서 전혀 활용하지 않음 DP 가 각 상태/행동이 다음 상태에 영향을 미친다는 점을 활용해 계산량을 줄인 것을 생각해보자!
- 정확한 값을 얻기 위해 많은 시뮬레이션을 필요로 함. (일반적으로 수렴속도가 느림)