강화학습 소개

What is RL?

- 인공지능 에이전트를 학습시키는 과정 중 하나
- 에이전트 보고 어떤 행동을 하게 한 다음에 그 행동이 잘 했으면 보상(reward)을 주고, 잘못 했으면 벌(punishment)을 주는 방식으로 학습한다.
- 룰을 일일이 가르쳐주는게 매우 어려운 환경(게임)에서 유리함

Basic RL Model

- Goal : 에이전트를 제어하는 정책을 찾는다. (최대의 보상을 얻을 수 있는)
- 에이전트 : 어떤 상태, 그 상태에서 취할 수 있는 액션Set. State가 변화하면서 reward가 주어짐. 이를 반복하면서 '어떤 상태에서는 어떤 액션을 취해야 되겠구나.'를 스스로 학습
- Reward를 어떻게 정의하고 어떻게 주느냐?

Major Components of an RL Agent

- Policy : 에이전트가 어떤 상태에서 어떤 액션을 할지 정해주는 함수
  - Deterministic : 어떤 상태에서 어떤 액션을 할지 하나하나 정의
  - Stochastic : 어떤 state에서 어떤 액션을 취할지 확률분포로 주어지는 것(확률이 높긴 하지만 항상 그 액션을 취하는 것은 아님)

  - ► Deterministic policy: $a = \pi(s)$
  - ► Stochastic policy: $\pi(a|s) = \mathbb{E}[a_t = a|S_t = s]$

Major Components of an RL Agent

- Value function : 어떤 state에서 어떤 state로 갈 때 그 state가 얼마나 좋은지 나타내는 함수, 미래의 보상에 대한 예측 값
  - Policy가 주어졌을 때 S라는 state의 value가 얼마냐?
  - V : 어떤 policy를 따를 때 어떤 state의 value
    - 감마 : 먼 미래에 대한 reward를 discount하기 위함(보통 1보다 작은 값)

$$V_\pi(s) = \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots | S_t = s]$$

Major Components of an RL Agent

- Model : 에이전트가 환경을 어떻게 표현하고 있는지 그것에 대한 모든 것
  - P : S라는 state에 있을 때 a라는 액션을 취하면 S'로 갈 확률
  - R : S라는 state에 있을 때 a라는 액션을 취하면 얻는 Reward

$$P_{ss'}^{a} = \mathbb{P}[S_{t+1} = s' | s_t = s, A_t = a]$$

$$R_{s}^{a} = \mathbb{P}[R_{t+1} | s_t = s, A_t = a]$$

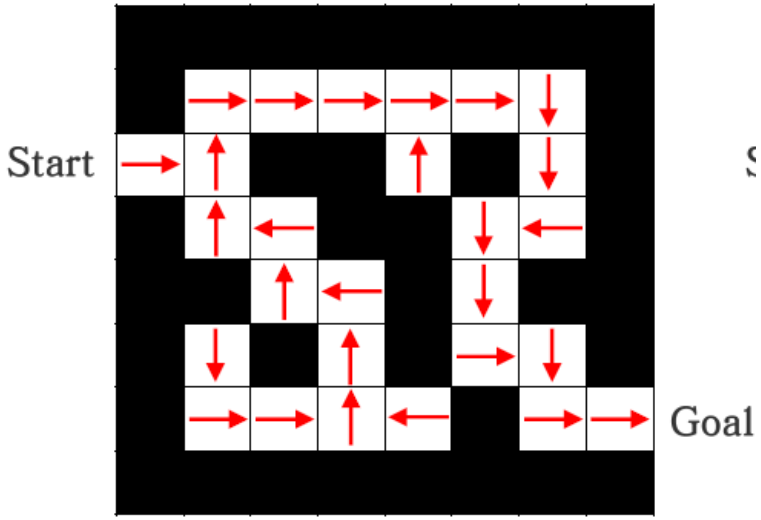
Maze Example

- Goal : 최단거리로 start부터 goal까지 가는 것
- Rewards : -1 per time-step
- Action : N, E, S, W
- States : Agent’s location

Maze Example - policy

Policy

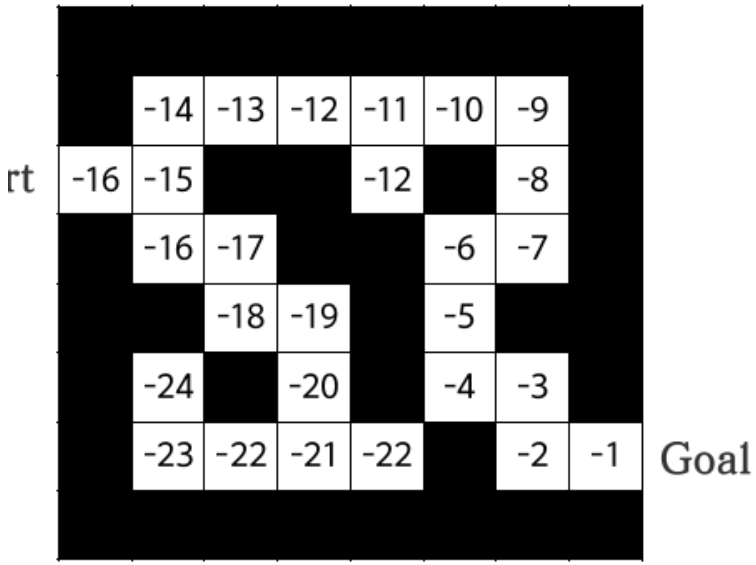
- Arrows represent policy $\pi(s)$ for each state s



모든 state에서 어떤 action을 취해야 하는지 알려준다.

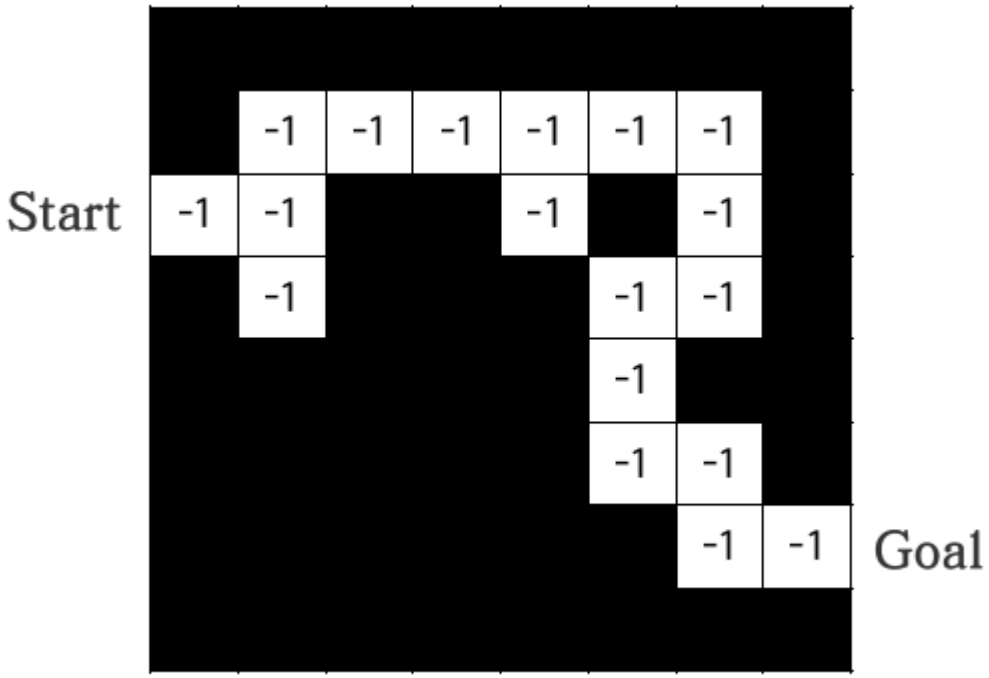
Maze Example – value function

- **Value function**
  - Numbers represent value $V_\pi(s)$ of each state s



어떤 State의 값을 의미
Value Function이 계속 큰 방향으로 이동
(Policy와 관련이 깊다.)Value Function과 Policy 둘 중 하나만 알아도 다른 하나 유도 가능

Maze Example - Model



P는 S라는 sate에 있을 때 a라는 액션을 취하면 S′라는 state로 갈 확률
R : S라는 state에서 a라는 액션을 취했을 때 reward
흰색 : 특정 지점에서 갈 수 있는 경우의 수
검은색 : 가지 못하는 경우
-1 : reward function

Exploration and Exploitation

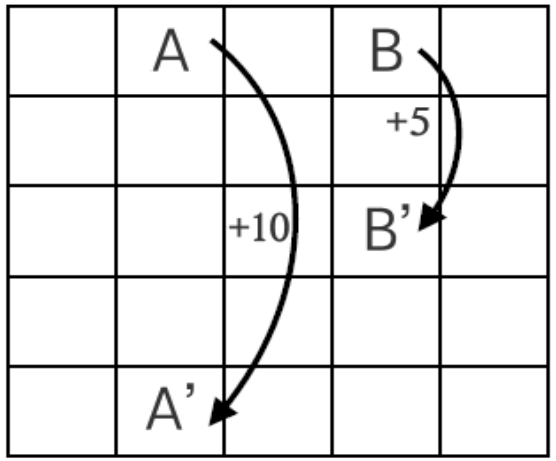
- Exploitation : ‘이런 action을 취했을 때 reward가 많으니까 이런 reward를 취해야지 ' 즉, 알고 있는 지식을 활용하는 경우
  - 처음에 두 번째로 좋은 action을 취해서 reward 1을 받았는데 원래 처음에 제일 좋은 action을 취했으면 reward를 10을 받을 수 있었다.
- Exploration : 일종의 random action을 취해서 ‘내가 이런 상황에서 어떤 액션을 취하면 이런 reward를 받는구나.’ 경험을 쌓는 것도 중요함.
- 식당을 결정할 때 지금까지 가본 가장 맛있는 식당을 가는 경우도 있지만(Exploitation), 가보지는 않았지만 한번 새로운 레스토랑을 시도(Exploration)

Prediction and Control

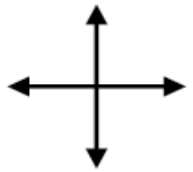
- Prediction : 한번 예측을 해본다.
  - Policy가 주어졌을 때 value function을 구하는 것
  - 어떤 policy π이라는 게 주어졌을 때 가치 함수, 그러니까 Vπ(S)를 구하는 것. π가 주어졌을 때 Vπ(S)를 구하는 과정
- Control : 제어한다.
  - 문제가 주어졌을 때 최적의 policy, reward를 최대한으로 할 수 있는 policy를 찾는 과정

Gridworld Example - Prediction

- A지점에 reward 10받고 A'로 이동, B지점에 reward 5받고 B'로 이동
- Policy : uniform random policy (동,서,남,북 가는 확률이 0.25로 동일)



Gridworld



Actions

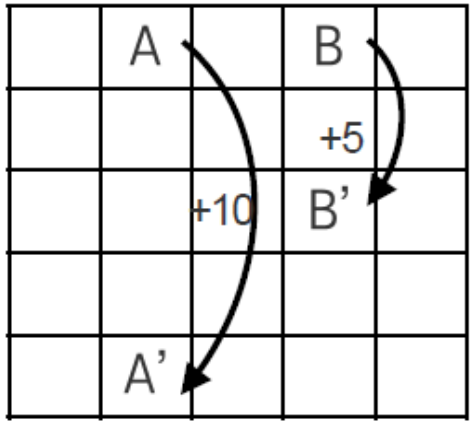
3.3	8.8	4.4	5.3	1.5
1.5	3.0	2.3	1.9	0.5
0.1	0.7	0.7	0.4	-0.4
-1.0	-0.4	-0.4	-0.6	-1.2
-1.9	-1.3	-1.2	-1.4	-2.0

Value function

Policy가 주어지면 value function을 구할 수 있다.

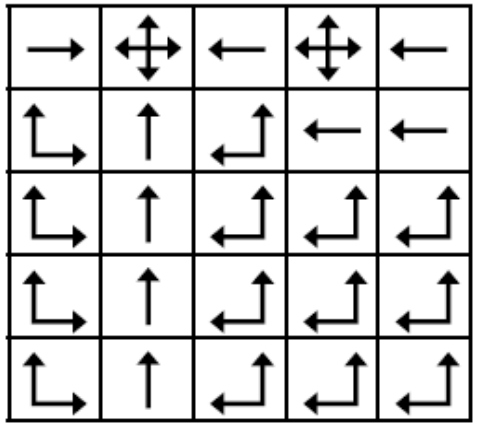
Gridworld Example – Control

- A지점에 reward 10받고 A'로 이동, B지점에 reward 5받고 B'로 이동
- Policy : uniform random policy (동,서,남,북 가는 확률이 0.25로 동일)



22.0	24.4	22.0	19.4	17.5
19.8	22.0	19.8	17.8	16.0
17.8	19.8	17.8	16.0	14.4
16.0	17.8	16.0	14.4	13.0
14.4	16.0	14.4	13.0	11.7

V_*



π_*

V : 수많은 value function 중 optimal, 최적의 value function
Π : 내가 취할 수 있는 수 많은 policy 중에서 reward를 가장 maximizing하는 policy(특정 지점에서 reward를 최대한 하는 방향)