강화학습 스터디 1주차

RL.start() 초보자 B그룹

Introduction of Reinforcement Learning

20.08.25 (화)

발표자: https://github.com/jinmang2

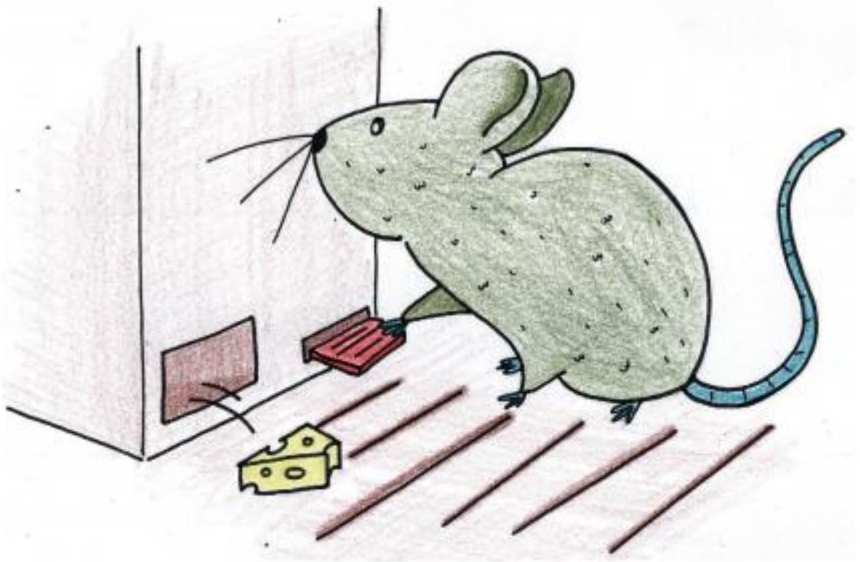
01장 강화학습 개요

Reinforcement Learning = Reinforcement + Machine Learning

1. Reinforcement는 무엇인가?

2. Machine Learning은 무엇인가?

강화(強化): 배우지 않았지만 직접 시도하면서 행동과 그 결과로 나타나는 보상 사이의 상관관계를 학습하는 것



Skinner의 쥐 실험

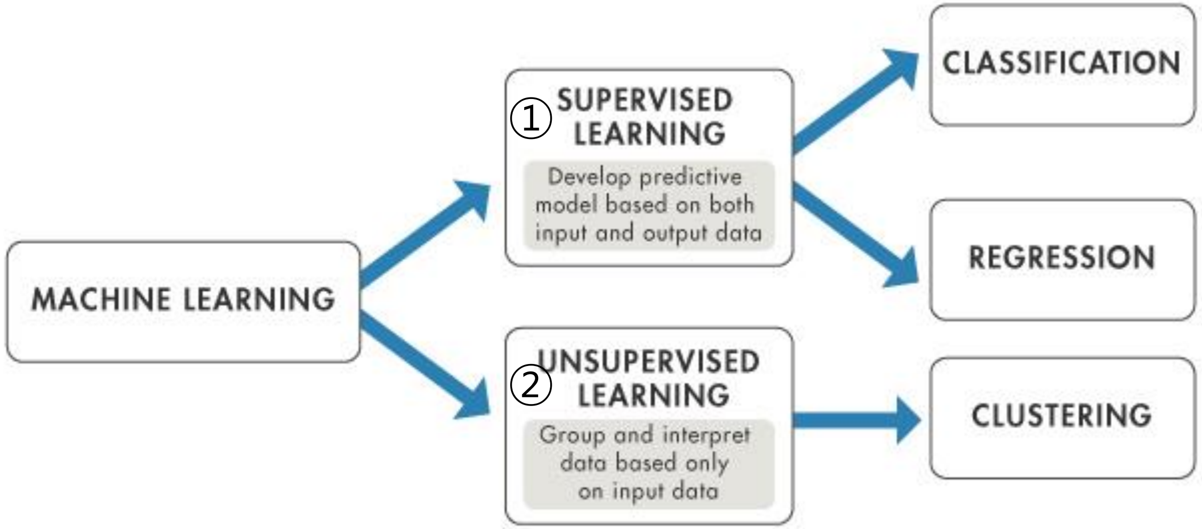
1. 굶은 쥐를 상자에 넣는다 랜덤하게 탐색
2. 쥐는 돌아다니다가 **우연히** 상자 안에 있는 지렛대를 누르게 된다
3. **지렛대를 누르자 먹이가 나온다** 보상
4. 지렛대를 누르는 행동과 먹이와의 상관관계를 모르는 쥐는 **다시 돌아다닌다.** 반복적으로 탐색
5. 그러다가 우연히 쥐가 다시 지렛대를 누르면 쥐는 이제 먹이와 지렛대 사이의 관계를 알게 되고 점점 지렛대를 자주 누르게 된다. 상관관계를 학습
6. 이 과정을 반복하면서 쥐는 **지렛대를 누르면 먹이를 먹을 수 있다는 것을 학습**한다.

01장 강화학습 개요 – 강화학습의 개념

기계학습: 컴퓨터가 스스로 학습하게 하는 알고리즘을 개발하는 분야

스스로 뭘 학습한다는 얘기? → 데이터 기반으로!

어떻게 학습하는 건데?? → 음… ① 답을 알려주거나 ② 기계 스스로 찾게 만들거나!



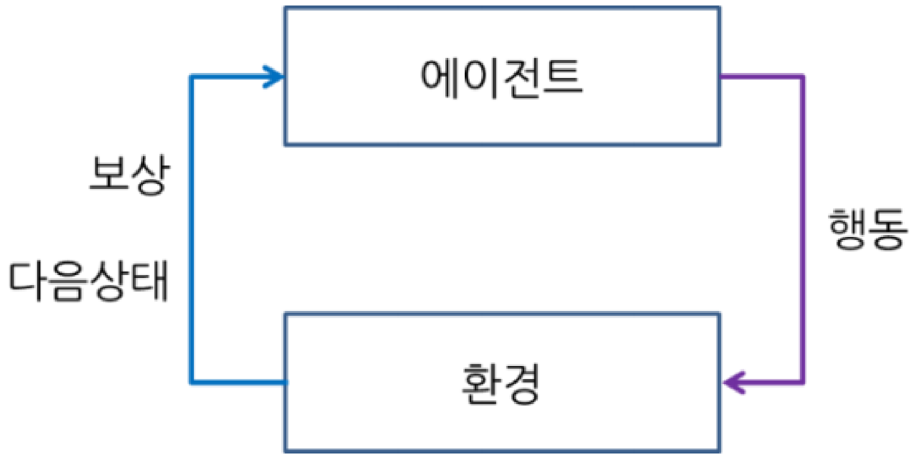
https://www.mathworks.com/help/stats/machine-learning-in-matlab.html

01장 강화학습 개요 – 강화학습의 개념

강화학습

- “보상(Reward)”을 통해 학습
- 정답이 주어진 적은 아니지만 그저 주어진 데이터에 의해 학습 하는 것도 아님
- 강화학습을 수행하는 컴퓨터는 행동심리학에서 살펴본 "강화" 처럼 보상을 얻게 하는 행동을 점점 많이 하도록 학습

01장 강화학습 개요 – 강화학습의 개념



- Agent: 강화학습을 통해 스스로 학습하는 컴퓨터
- 환경은 말 그대로의 환경! Ex) 스타크래프트, 단어 공간
- 보상은 +, – 설정 가능 (강화학습을 문제에 적용할 때 고려해야 하는 점)
- 강화학습의 장점: 환경에 대한 사전지식이 없어도 학습 가능

01장 강화학습 개요 – 강화학습 문제

- 모든 문제를 강화학습으로 풀 수 있는 것이 아님!
- 문제 자체에 대해 잘 이해하지 않으면 엉뚱한 결과를 낳음!
- 강화학습은 **결정을 순차적으로 내려야 하는 문제**에 적용
- 순차적 행동 결정 문제
  - Dynamic Programming
  - Evolutionary Algorithm
  - Reinforcement Learning
- 문제에 맞는 Tool을 사용!

01장 강화학습 개요 – 강화학습 문제

1. **강화학습이 풀고자 하는 문제: Sequential Decision Problem** **순차적 의사 결정 문제**
2. 문제에 대한 수학적 정의: Markov Decision Process
3. MDP를 계산으로 푸는 방법: Dynamic Programming
4. MDP를 학습으로 푸는 방법: Reinforcement Learning
5. 상태공간이 크고 차원이 높을 때 쓰는 방법: Function Approximation
6. 바둑과 같은 복잡하고 어려운 문제를 푸는 방법: Deep Reinforcement Learning

https://tykimos.github.io/warehouse/2018-2-7-ISS_Near_and_Far_DeepRL_4.pdf

02장 강화학습 기초1:

MDP와 벨만 방정식

02장 강화학습 기초1: MDP와 벨만 방정식

1. 강화학습이 풀고자 하는 문제: Sequential Decision Problem
2. **문제에 대한 수학적 정의: Markov Decision Process** → 이번 절에서 다룰 내용들!
3. MDP를 계산으로 푸는 방법: Dynamic Programming
4. MDP를 학습으로 푸는 방법: Reinforcement Learning
5. 상태공간이 크고 차원이 높을 때 쓰는 방법: Function Approximation
6. 바둑과 같은 복잡하고 어려운 문제를 푸는 방법: Deep Reinforcement Learning

https://tykimos.github.io/warehouse/2018-2-7-ISS_Near_and_Far_DeepRL_4.pdf

02장 강화학습 기초1: MDP와 벨만 방정식

책에 내용이 굉장히 많음!!

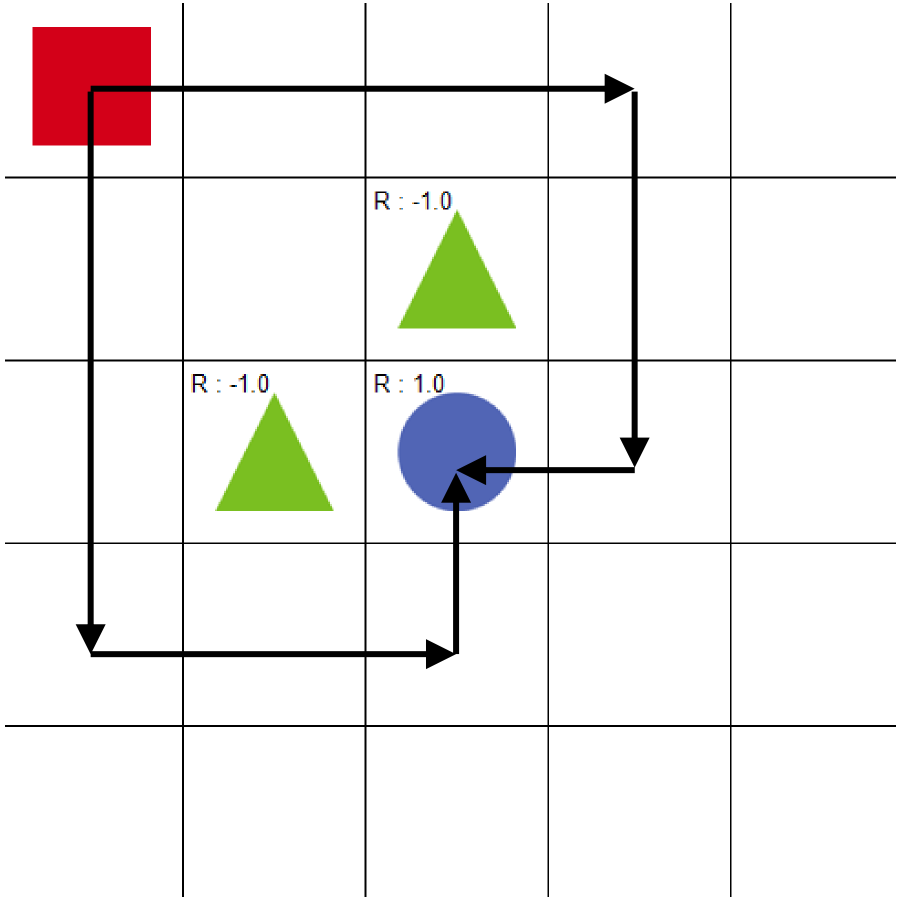
떨지 말고, 중요한 부분은 아래 개념들을 이해하면 2장 완료

- 강화학습은 Sequential Decision Problem을 푸는 방법 중 하나!
- 문제를 풀려면 문제를 정의해야지! → Markov Decision Problem
- MDP의 구성 요소들: **상태, 행동, 보상 함수, 상태 변환 확률, 할인률**
- MDP를 풀어서 구하고자 하는 게 뭔데? → **최적의 정책** π_*
- 뭘 풀면 최적의 정책을 구할 수 있는데? → Bellman Equation
- 어떻게 푸는데? → 그건 3장~ 에서... DP, EA, RL!

최소한의 수식과 알찬 개념으로 감을 잡자!

02장 강화학습 기초1: MDP와 벨만 방정식

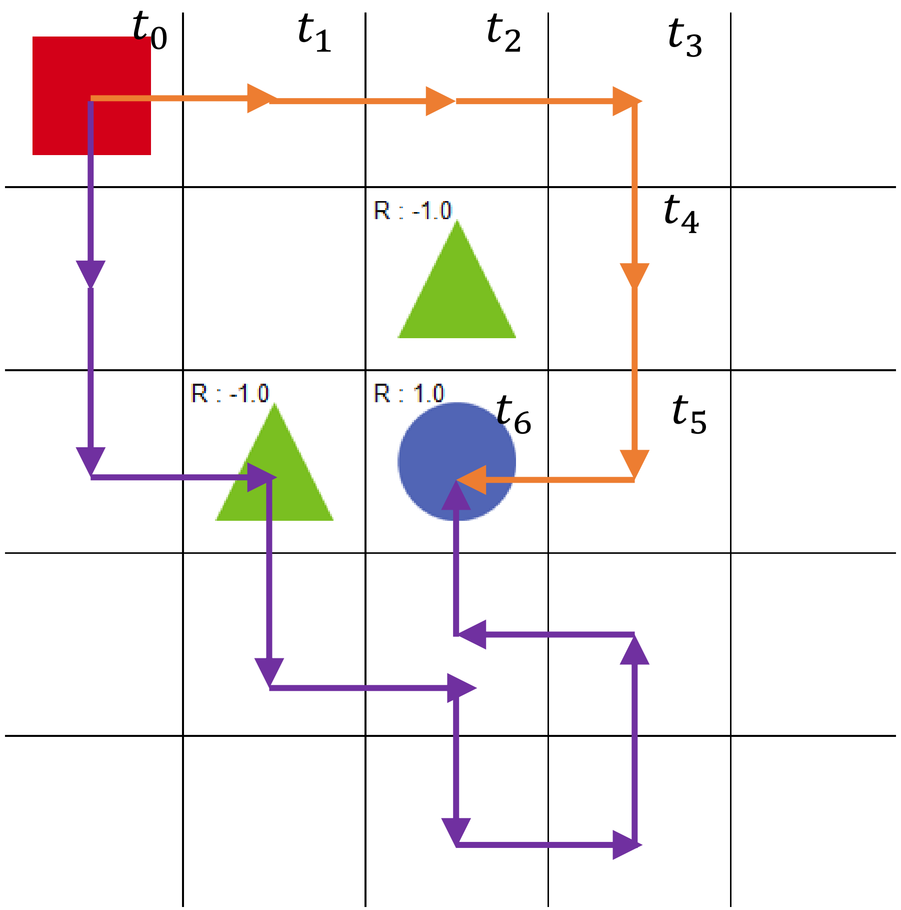
- 강화학습은 **Sequential Decision Problem**을 푸는 방법 중 하나!



- 순차적 의사결정 문제
- 책의 예시, Grid World를 생각해보자. (격자 세계)
- 문제를 우선 개념적으로 정의!
  - 사각형에서 시작하여 삼각형을 피해 원으로 가는 최단 경로는?
- 이 정도 문제는 너무나도 쉬워서 우리가 눈으로 보고 풀 수 있음
- 그러나, 로봇이 탁구를 치는 게임을 하는 문제를 푼다고 생각하면?
- 위와 같이 복잡한 문제를 풀기 위해 우리는 이러한 SDP를
  - 수학적으로 정의해야 하고
  - 문제를 풀 알고리즘을 선정해야 한다.
- 이번 절에서는 순차적 의사결정 문제(SDP)를 수학적으로 어떻게 정의할 지 학습
  - Markov Decision Process
  - 인간이 정의 문제를 컴퓨터가 이해하고 학습할 수 있게 정의

02장 강화학습 기초1: MDP와 벨만 방정식

- 문제를 풀려면 문제를 정의해야지! → Markov Decision Problem

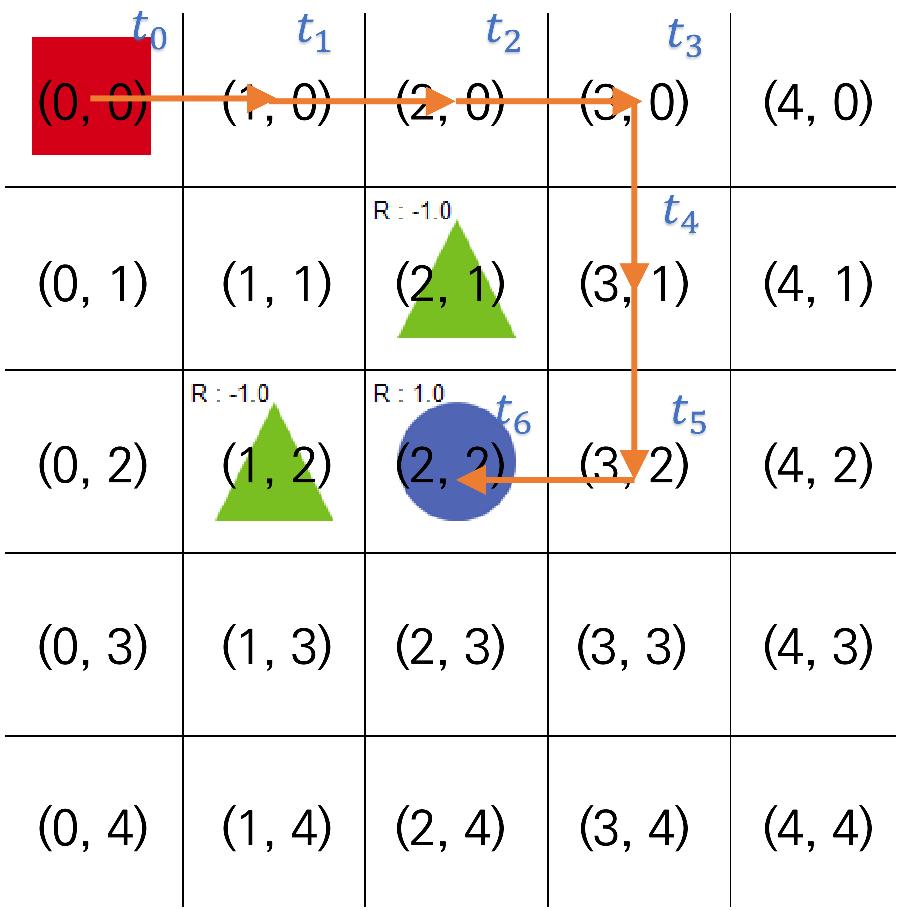


MDP(Markov Decision Problem)

- 상태 $S_t = s$
- 행동 $A_t = a$
- 보상 함수 $r_{(s,a)} = E[R_{t+1} | S_t = s \,, A_t = a \,]$
- 상태 변환 확률 $P_{ss'}^{a} = P[S_{t+1} = s' | S_t = s \,, A_t = a]$
- 할인률 $\gamma \in [0, 1]$

02장 강화학습 기초1: MDP와 벨만 방정식

- MDP의 구성 요소들: **상태, 행동, 보상 함수, 상태 변환 확률, 할인률**

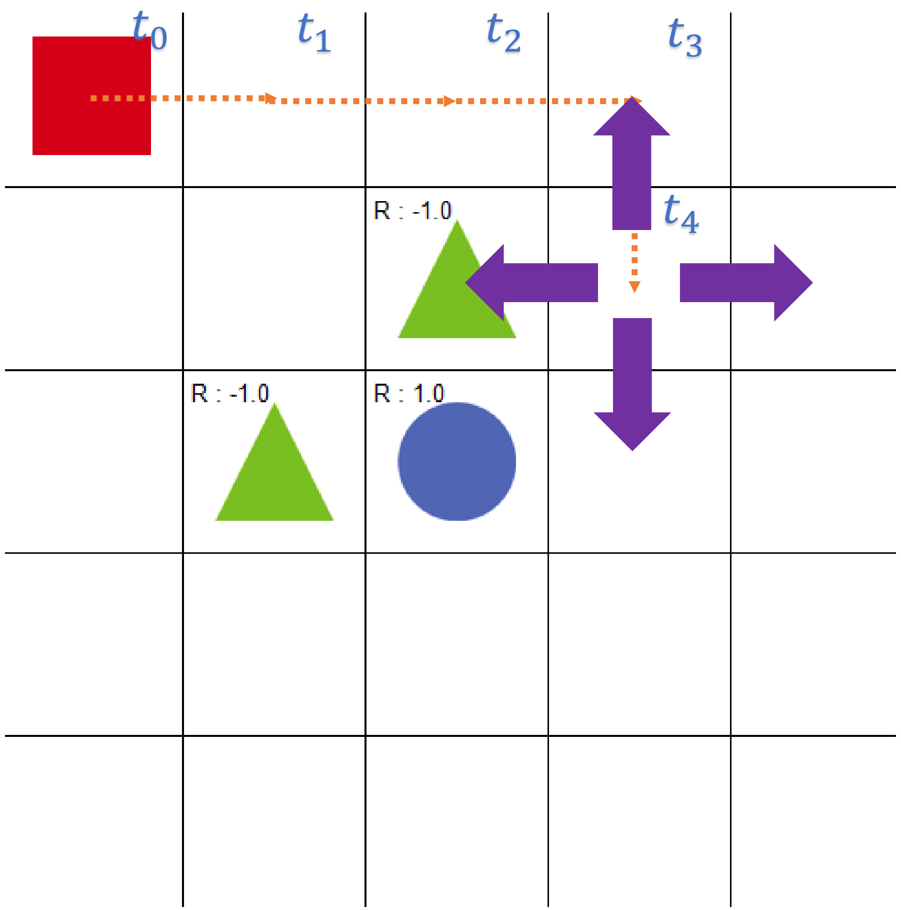


상태, State $S_t = s$

- S는 에이전트가 관찰 가능한 상태의 집합
- $S = \{(0,0).(0,1),\dots,(4,4)\},\ len(S) = 25$
- 상태란? 자신의 상황에 대한 관찰
- 로봇 등 real world에서 상태는 센서 값이 될 것
- 탁구치는 봇을 학습하기 위해선 탁구공의 위치, 속도, 가속도 등과 같은 정보 (동적인 요소도 포함)
- 본 예제에서 상태는 단순히 좌표지만 실제로 우리가 RL을 적용할 상태는 관찰값이라는 것을 기억!
- 주황색 시나리오에서 S_5는 (3,2)가 될 것임

02장 강화학습 기초1: MDP와 벨만 방정식

- MDP의 구성 요소들: **상태, 행동, 보상 함수, 상태 변환 확률, 할인률**



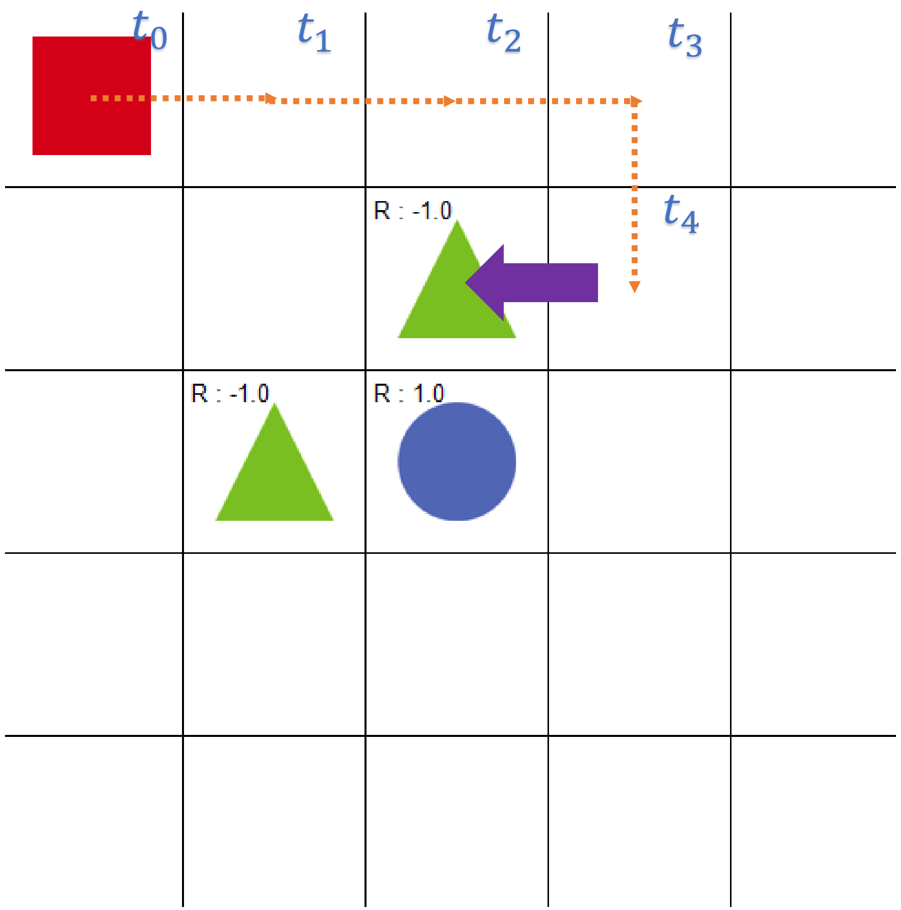
행동, action $A_t = a$

- A는 Agent가 상태 S_t에서 할 수 있는 가능한 행동의 집합
  - 보통 모든 Agent 가 할 수 있는 행동은 모든 상태에서 같음
  - 바꿔말하면, 어떤 문제에서는 다른 상태에서 취할 수 있는 행동이 다를 수 있다.
- Grid World에서 모든 상태에 대한 행동은 같다고 가정하면

$$A = \{\mathrm{up}, \mathrm{down}, \mathrm{left}, \mathrm{right}\}$$

02장 강화학습 기초1: MDP와 벨만 방정식

- MDP의 구성 요소들: **상태, 행동, 보상 함수, 상태 변환 확률, 할인률**



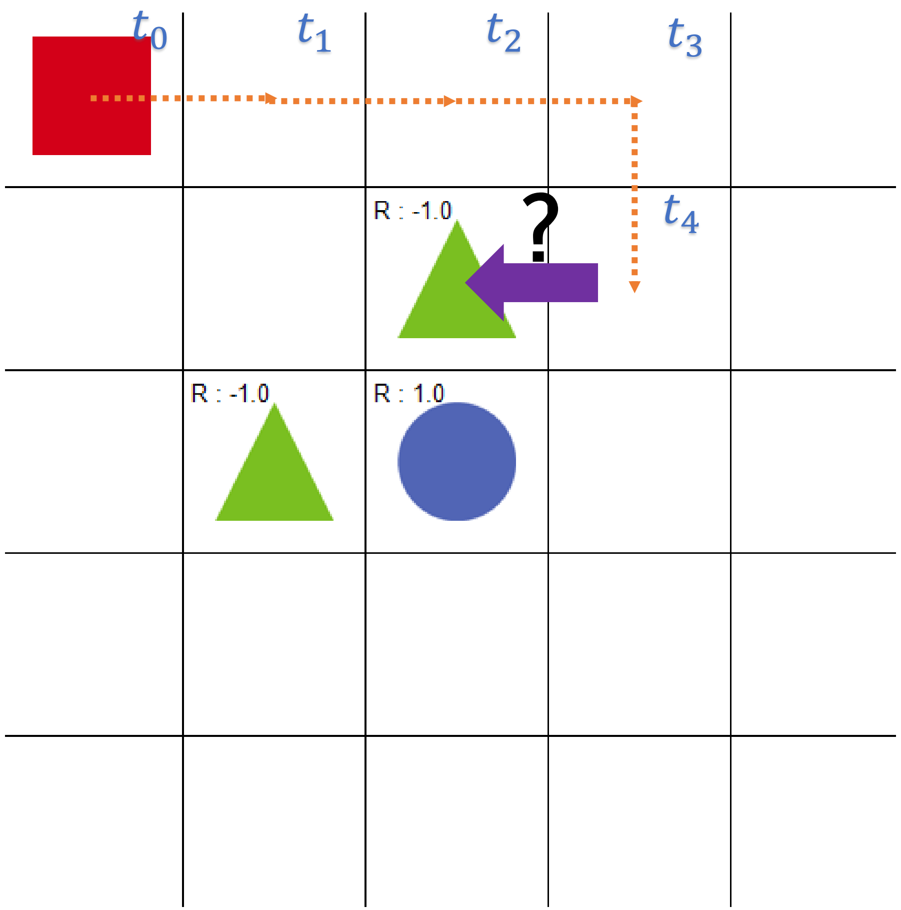
보상 함수, Reward function

$$r_{(s,a)} = E[R_{t+1} | S_t = s, A_t = a]$$

- $r_{(s,a)} = E[R_{t+1} | S_t = s, A_t = a]$, 기댓값 표현
- $r_{(s,a)} = E[R_{t+1} | S_t = s, A_t = a]$, 상태, 행동 전제 하
- $r_{(s,a)} = E[R_{t+1} | S_t = s, A_t = a]$, 보상을 받는 것은 다음 시점
- 왜 기댓값을 취할까?
  - 환경에 따라서 같은 상태에서 같은 행동을 취하더라도 다른 보상을 줄 수도 있기 때문
  - 본 Grid World에선 세모로 가면 −1, 원으로 가면 +1

02장 강화학습 기초1: MDP와 벨만 방정식

- MDP의 구성 요소들: **상태, 행동, 보상 함수, 상태 변환 확률, 할인률**

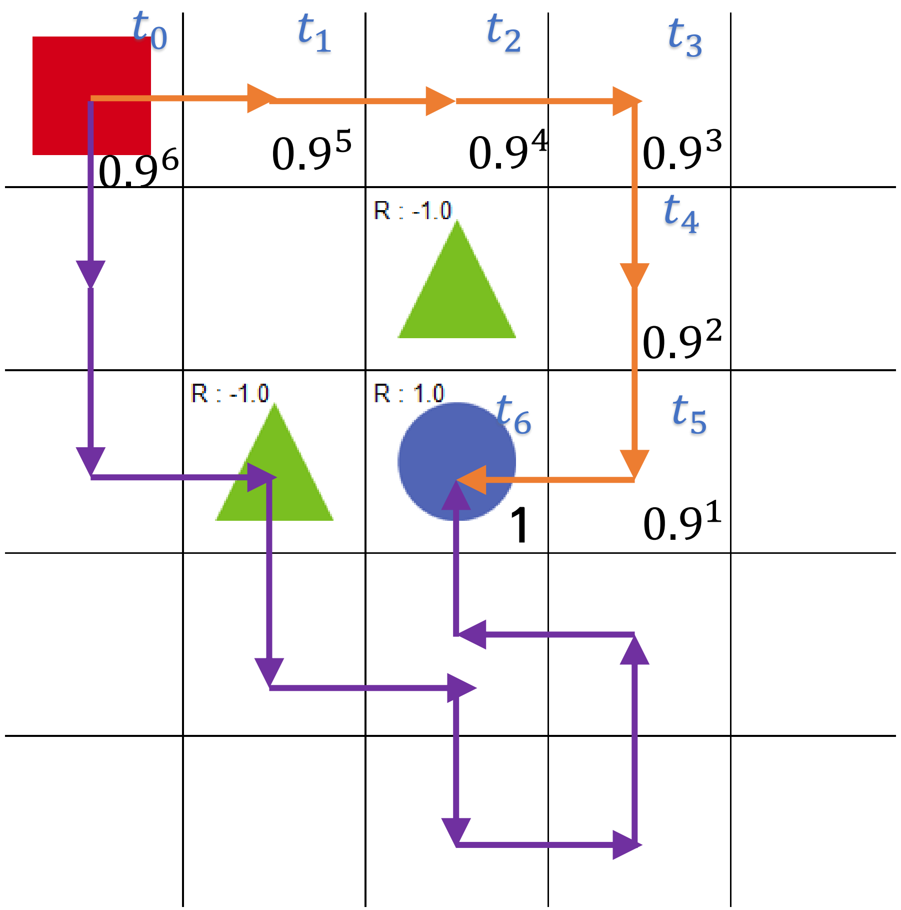


상태 변환 확률

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

- 에이전트가 상태 s에서 행동 a를 취해 다음 타임스텝에 상태 s'로 갈 확률
- 행동을 취한다고 항상 다음 상태로 넘어갈 수 있는 것이 아님. 때문에 이를 모델링
- 상태 변환 확률을 환경의 모델이라고 부름
- Grid World에선 모든 $s, s' \in S$에서 취할 수 있는 행동 a에 대하여
- $P_{ss'}^{a} = 1$이라 가정

- MDP의 구성 요소들: **상태, 행동, 보상 함수, 상태 변환 확률, 할인률**



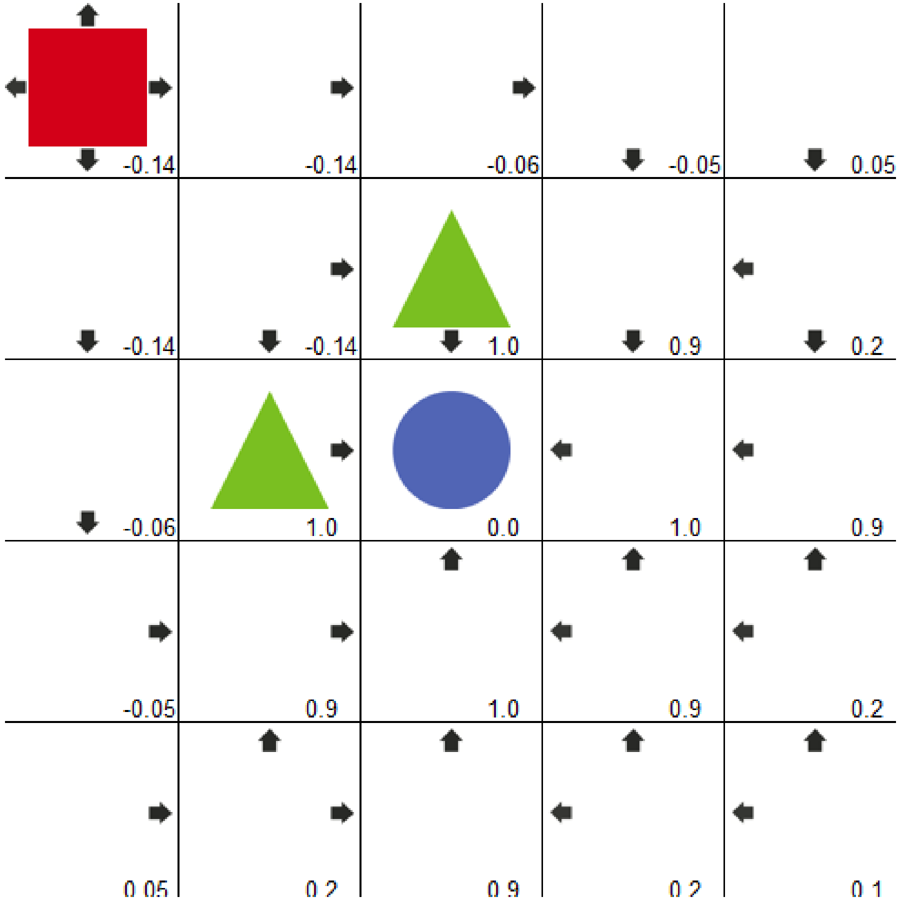
할인률, Discount factor $\gamma \in [0, 1]$

- MDP는 순차적 의사 결정 문제를 수학적으로 모델링
- 순차적이기 때문에 각 시점이 존재
- 서로 다른 시점의 보상을 현재 가치로 환산하여 비교해야 함
- 더 먼 미래에 받은 보상일수록 현재의 Agent는 더 작은 값으로 받아들임

$$\gamma^{k-1} R_{t+k}$$

- 주황색 경로가 받을 보상은 $0.9^6 = 0.531441$
- 보라색 경로가 받을 보상은 $0.9^{14} = 0.228768$

- MDP를 풀어서 구하고자 하는 게 뭔데? → **최적의 정책** π_*



정책, Policy

- 정책은 모든 상태에서 Agent가 할 행동
- 상태가 입력으로 들어오면 행동을 출력으로 내보내는 일종의 함수
- 각 상태에서 단 하나의 행동만을 나타내거나 확률적으로 나타낼 수 있음
- Agent가 강화학습을 통해 학습해야 할 것은 수많은 정책 중에서 최적 정책

$$\pi(a|s) = P[A_t = a \,|S_t = s]$$

02장 강화학습 기초1: MDP와 벨만 방정식

- 뭘 풀면 최적의 정책을 구할 수 있는데? → **Bellman Equation**

✓ 각 상태별로 앞으로 받을 보상을 계산하자 **가치함수**

$$G_t = \sum_{k=1}^{\infty} \gamma^{k-1} R_{t+k}$$

$$v(s) = E[G_t \mid S_t = s]$$

$$v(s) = E[R_{t+1} + \gamma G_{t+1} \mid S_t = s]$$

$$v(s) = E[R_{t+1} + \gamma v(S_{t+1}) \mid S_t = s]$$

$$v_\pi(s) = E_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s]$$

Belllman Expectation Equation

기댓값을 계산하기 위해서는 환경의 모델을 알아야 함

- DP는 가치함수를 계산
- 강화학습은 가치함수를 계산하지 않고 sampling을 통한 approximation

- 뭘 풀면 최적의 정책을 구할 수 있는데? → **Bellman Equation**

✓ 가치함수에 더하여 각 행동까지 고려한 가치를 알려주는 함수! **행동가치함수, Q function**

1. 각 행동을 했을 때 앞으로 받을 보상인 큐함수 $q_\pi(s,a)$를 $\pi(a|s)$에 곱한다
2. 모든 행동에 대해 큐함수와 $\pi(a|s)$를 곱한 값을 더하면 가치함수가 된다

$$v_\pi(s) = \sum_{a \in A} \pi(a|s) q_\pi(s,a)$$

가치함수는 큐함수에 대한 기댓값

$$v_\pi(s) = \boldsymbol{E}_{a\sim\pi}[q_\pi(s,a)|S_t = s]$$

$$q_\pi(s,a) = E_\pi[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1})|\, S_t = s, A_t = a]$$

가치함수와 동일, 단지 행동이 조건으로 붙은 것

02장 강화학습 기초1: MDP와 벨만 방정식

- 뭘 풀면 최적의 정책을 구할 수 있는데? → **Bellman Equation**

✓ 벨만 기대 방정식을 컴퓨터가 계산 가능하게 식 변환!

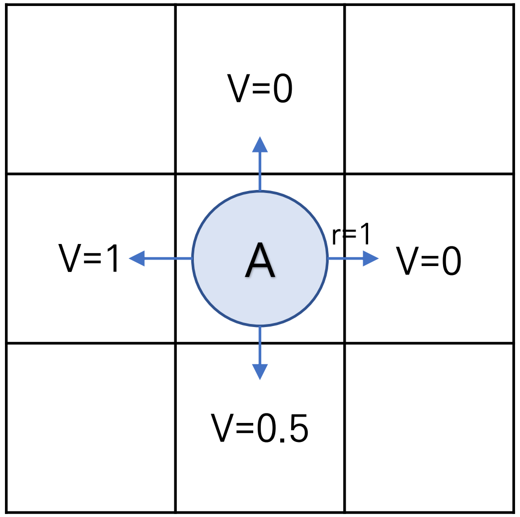
$$v_\pi(s) = \underline{E_\pi[R_{t+1} + \gamma v_\pi(S_{t+1})|}\, S_t = s]$$

상태 s에서 행동 a를 취했을 시 상태 s'로 갈 확률

$$v_\pi(s) = \sum_{a\in A} \boxed{\pi(a|s)} \left(\boxed{\mathrm{r}_{(s,a)} + \gamma \sum_{s'\in S} \boxed{P_{ss'}^{a}} v_\pi(s')} \right)$$

현재 상태 s에서 행동 a를 할 확률

$$v_\pi(s) = \sum_{a\in A} \pi(a|s)\left(\mathrm{r}_{(s,a)} + \gamma v_\pi(s')\right)$$



1	a=상 $0.25 \times (0 + 0.9 \times 0) = 0$
2	a=하 $0.25 \times (0 + 0.9 \times 0.5) = 0.1125$
3	a=좌 $0.25 \times (0 + 0.9 \times 1) = 0.225$
4	a=우 $0.25 \times (1 + 0.9 \times 0) = 0.25$
총합	기댓값$=0 + 0.1125 + 0.225 + 0.25 = 0.5875$

- 어떻게 푸는데? → 벨만 최적 방정식, 푸는 방법은 DP, EA, RL 등

✓ 가치함수를 계속 업데이트하여 현재 정책에 대한 참 가치함수를 구함

$$v_{k+1}(s) \leftarrow \sum_{a \in A} \pi(a|s)\left(\mathrm{r}_{(s,a)} + \gamma v_k(s')\right)$$

✓ 최적의 가치함수, 큐함수 이를 구하는 것이 순차적 행동 결정 문제를 푸는 것!

$$v_*(s) = \max_\pi[v_\pi(s)] \qquad q_*(s,a) = \max_\pi[q_\pi(s,a)]$$

```
# 벨만 기대 방정식
for action in self.env.possible_actions:
    next_state = self.env.state_after_action(state, action)
    reward = self.env.get_reward(state, action)
    next_value = self.get_value(next_state)
    value += (self.get_policy(state)[action] *
              (reward + self.discount_factor * next_value))
```

02장 강화학습 기초1: MDP와 벨만 방정식

- 어떻게 푸는데? → 벨만 최적 방정식, 푸는 방법은 DP, EA, RL 등

✓ optimal 가치함수, 큐함수로 구할 수 있는 최적 정책

$$\pi_*(s,a) = \begin{cases} 1 & if\ a = \underset{a \in A}{\operatorname{argmax}}\, q_*(s,a) \\ 0 & otherwise \end{cases}$$

✓ 가치, 행동에 대한 벨만 최적 방정식

$$v_*(s) = \max_a E[R_{t+1} + \gamma v_*(S_{t+1}) \mid S_t = s, A_t = a]$$

$$q_*(s,a) = E\left[R_{t+1} + \gamma \max_{a'} q_*(S_{t+1}, a') \mid S_t = s, A_t = a\right]$$

벨만 기대 방정식 및 벨만 최적 방정식을 이용해 MDP로 정의되는 문제를

- "계산"으로 푸는 것 DP
- "학습"으로 푸는 것 RL

Any Question?