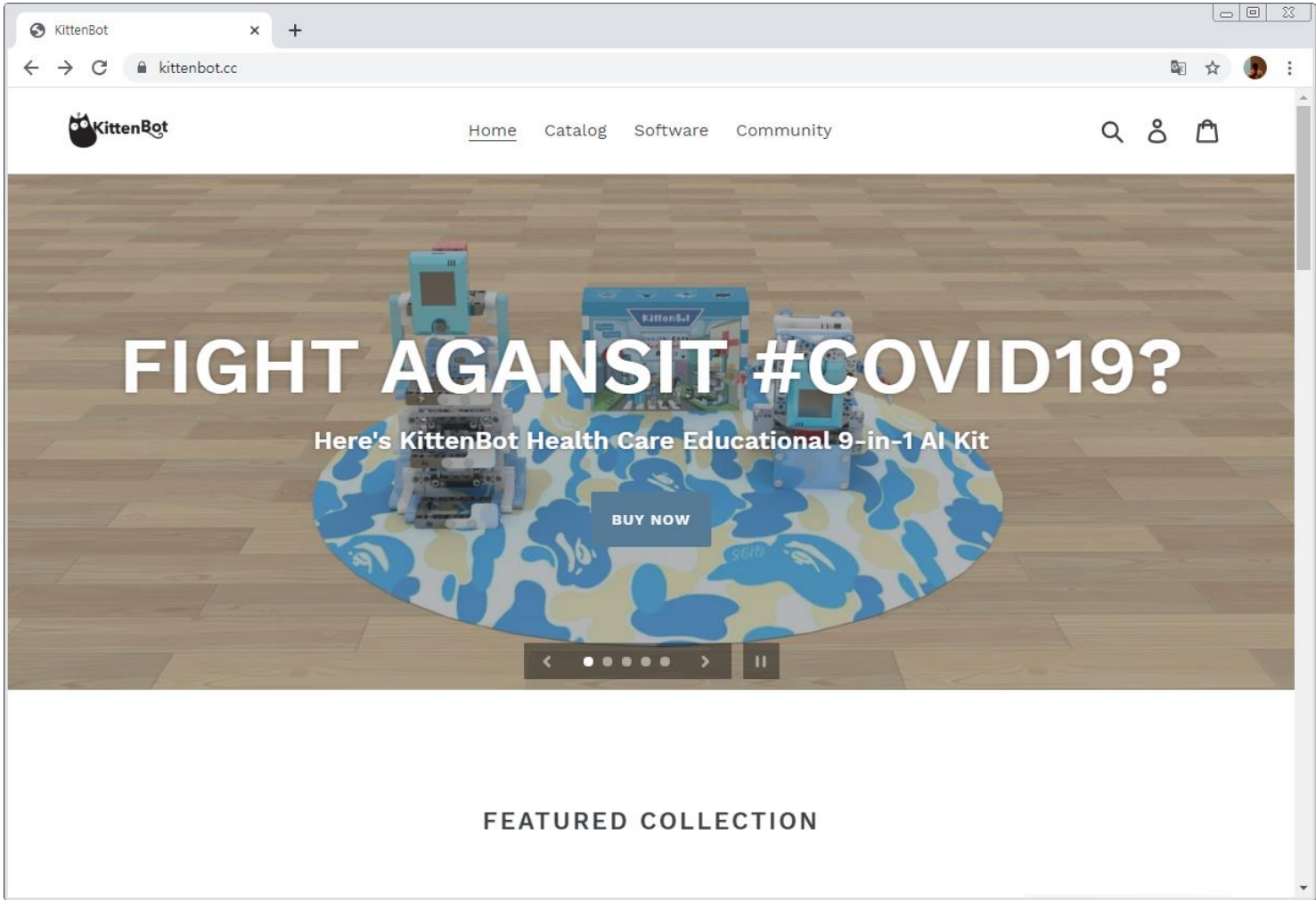
햄스터 · 햄스터S로 배우는 인공 지능

블록 코딩 #3

블록 코딩 추가

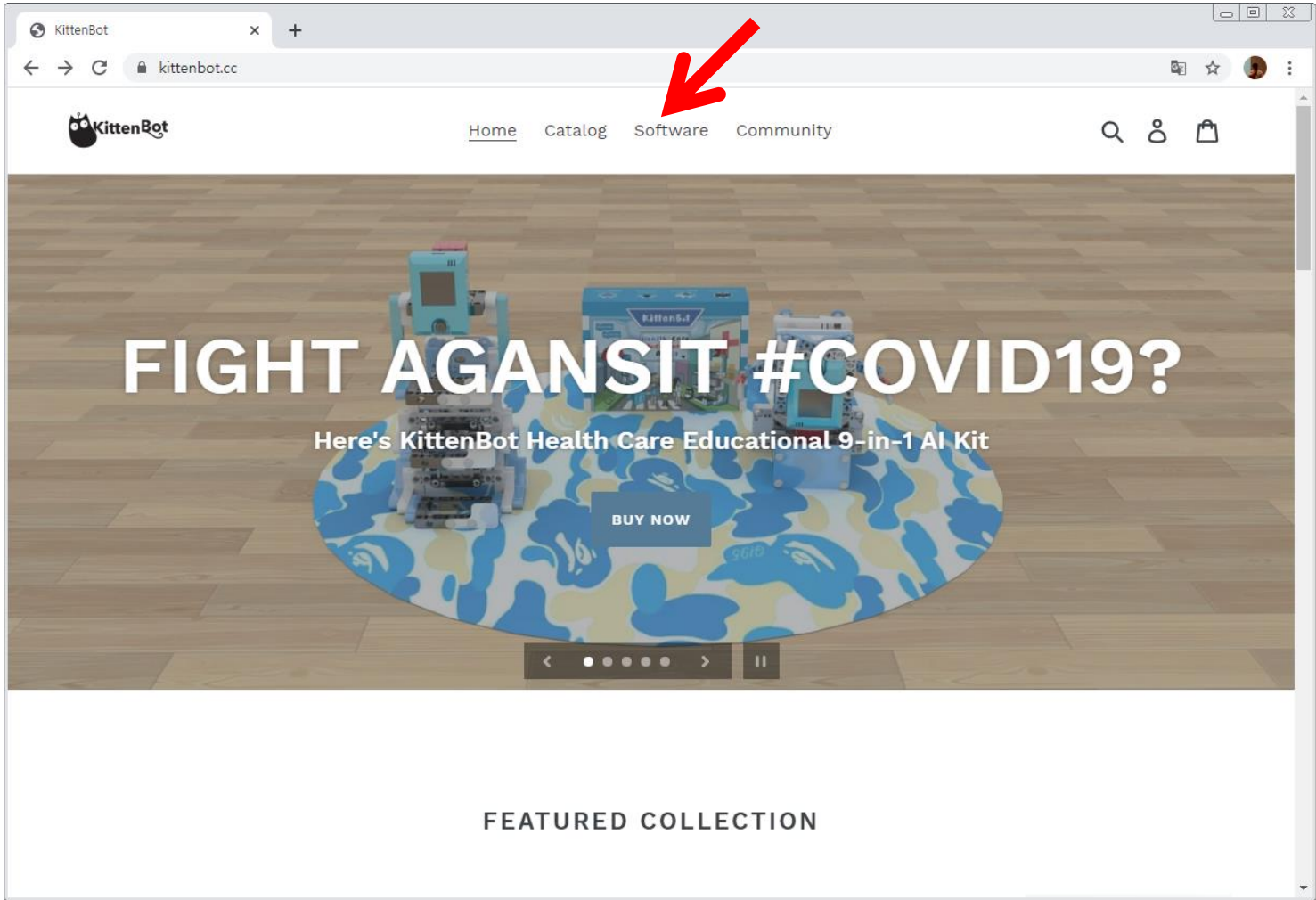
키튼블록 설치

1 https://www.kittenbot.cc



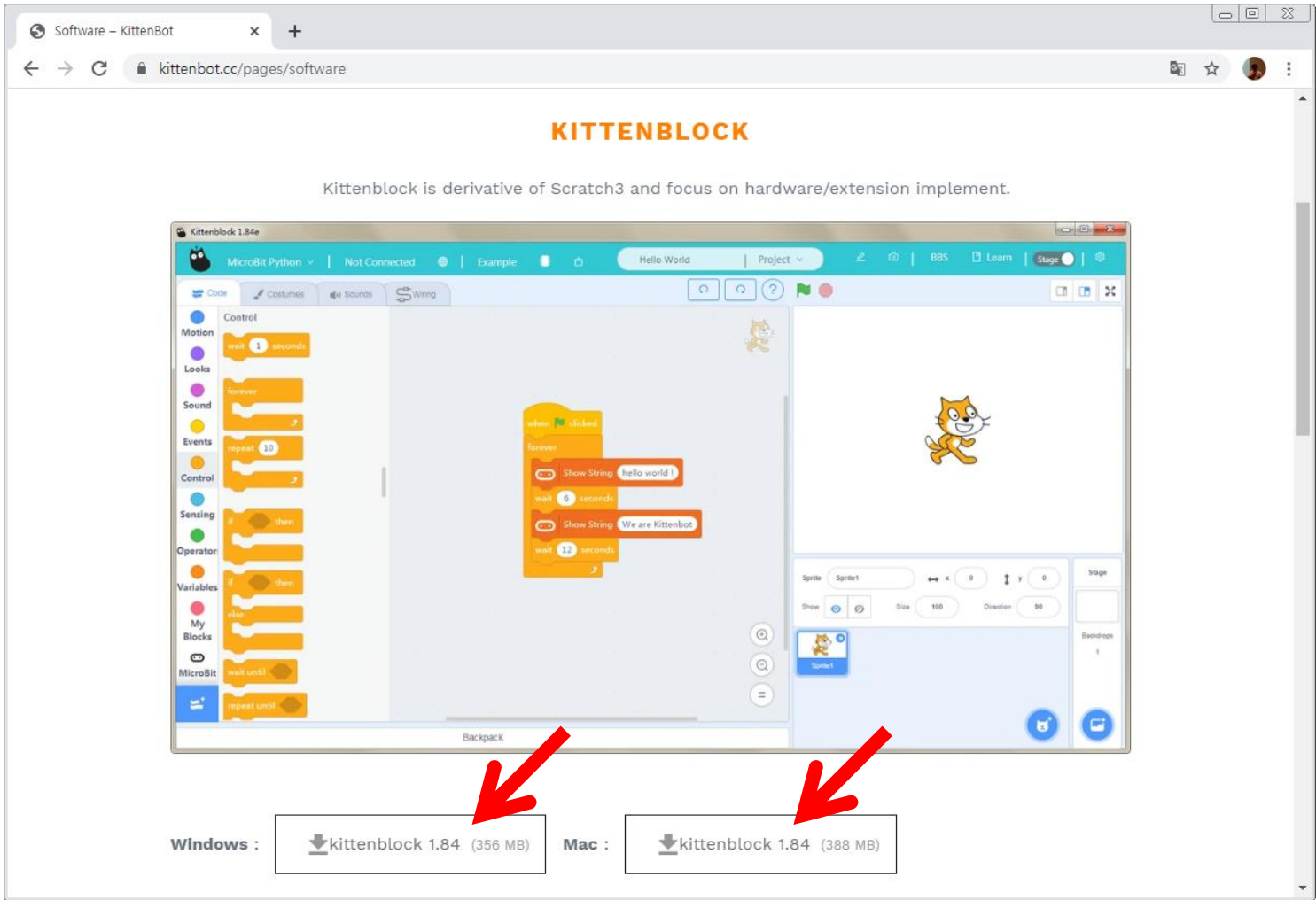
키튼블록 설치

2 Software 클릭



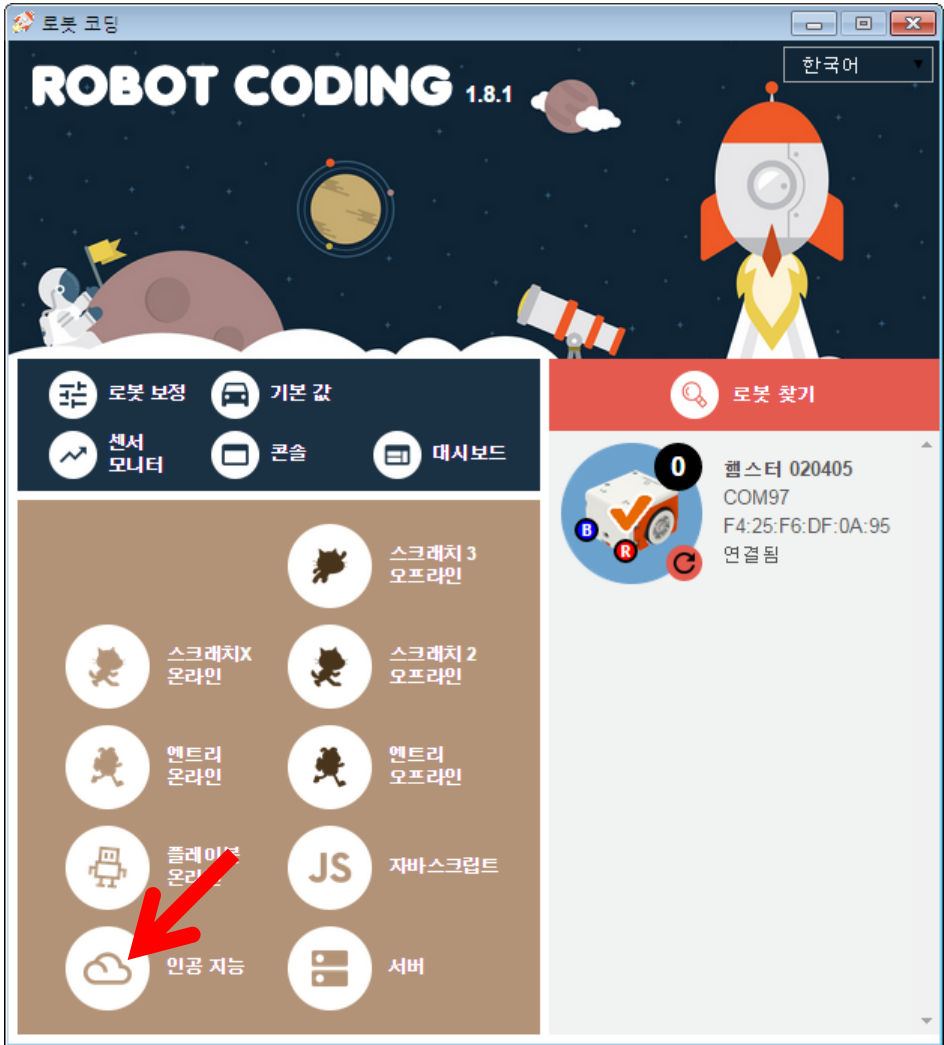
키튼블록 설치

3 아래로 약간 스크롤하여 OS에 맞는 설치 파일 다운로드



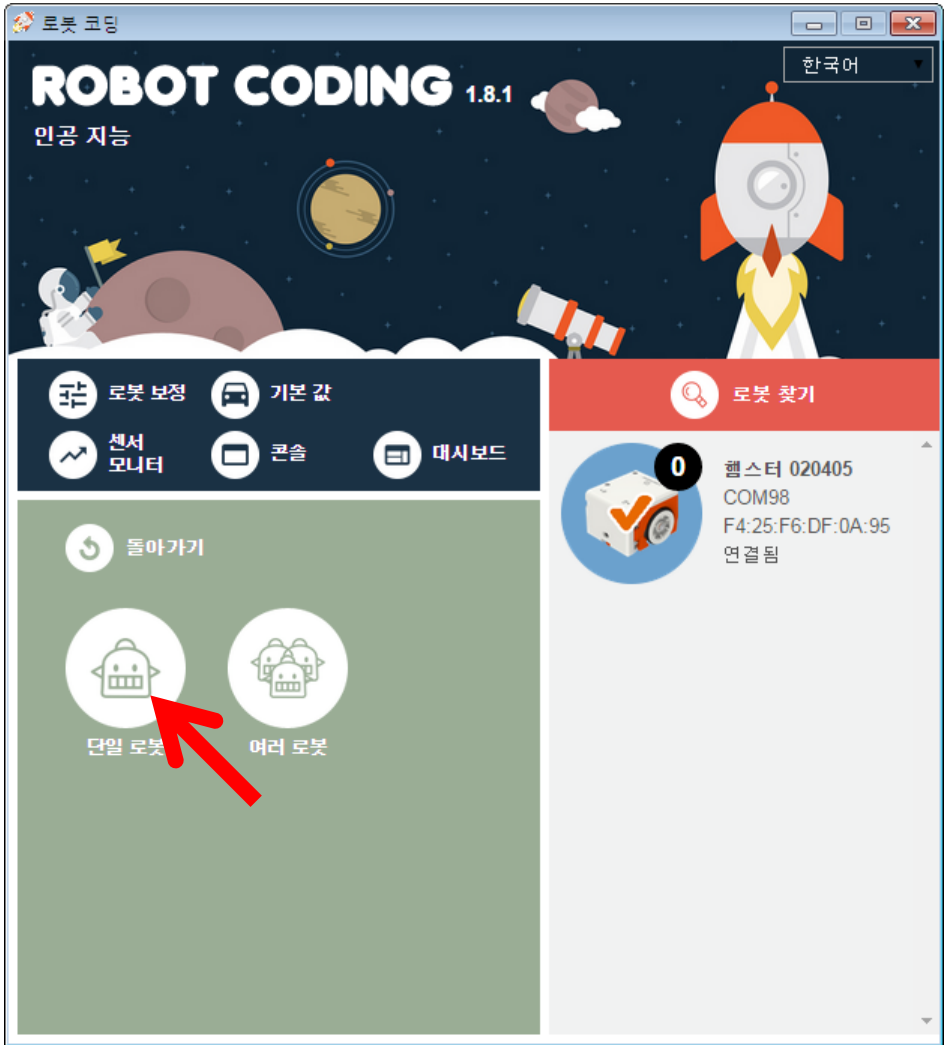
키튼블록 실행

1 인공 지능 클릭



키튼블록 실행

2 단일 로봇 클릭



키튼블록 실행

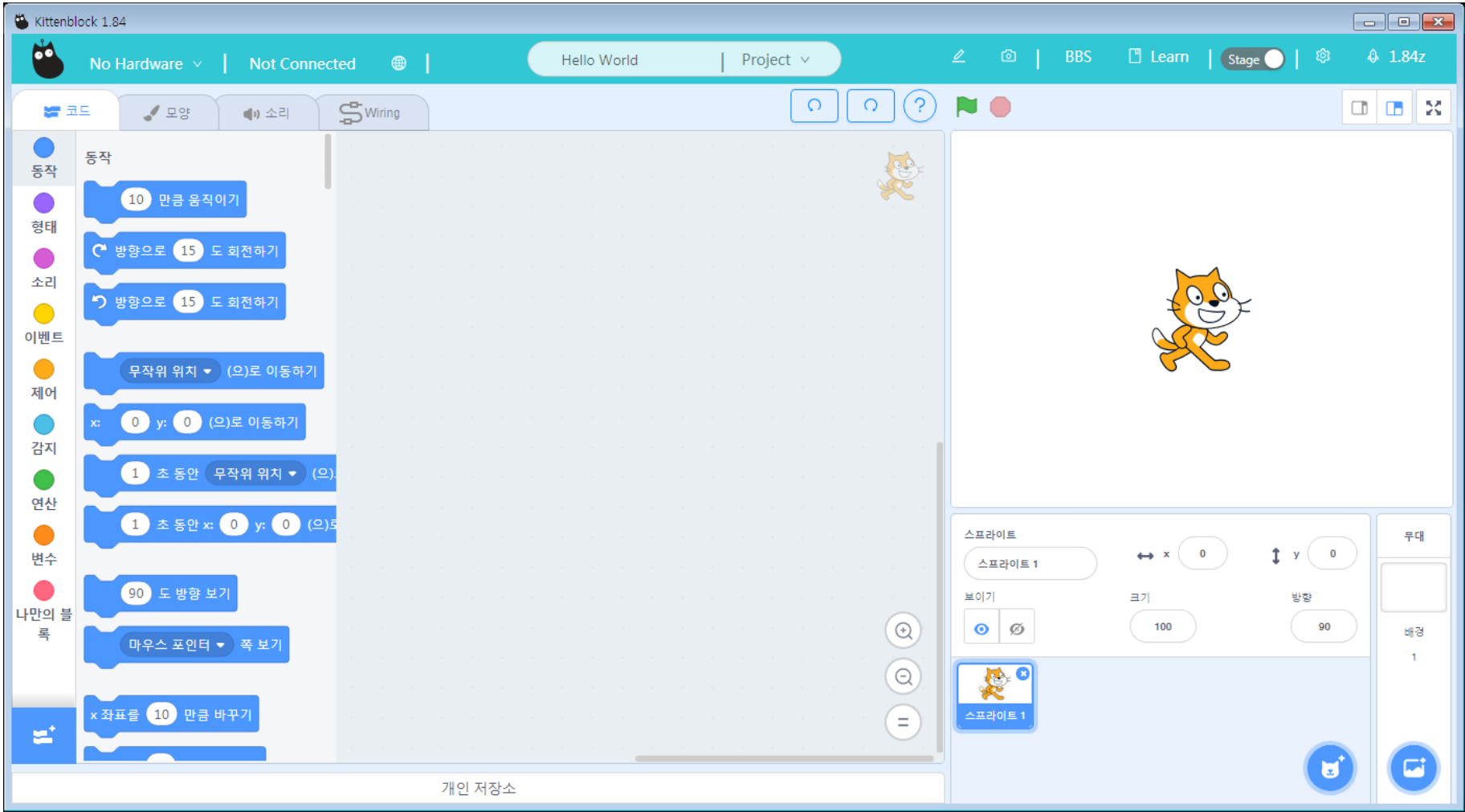
3 키튼블록 클릭

키튼블록 실행

4 키튼블록 연결 프로그램 실행됨

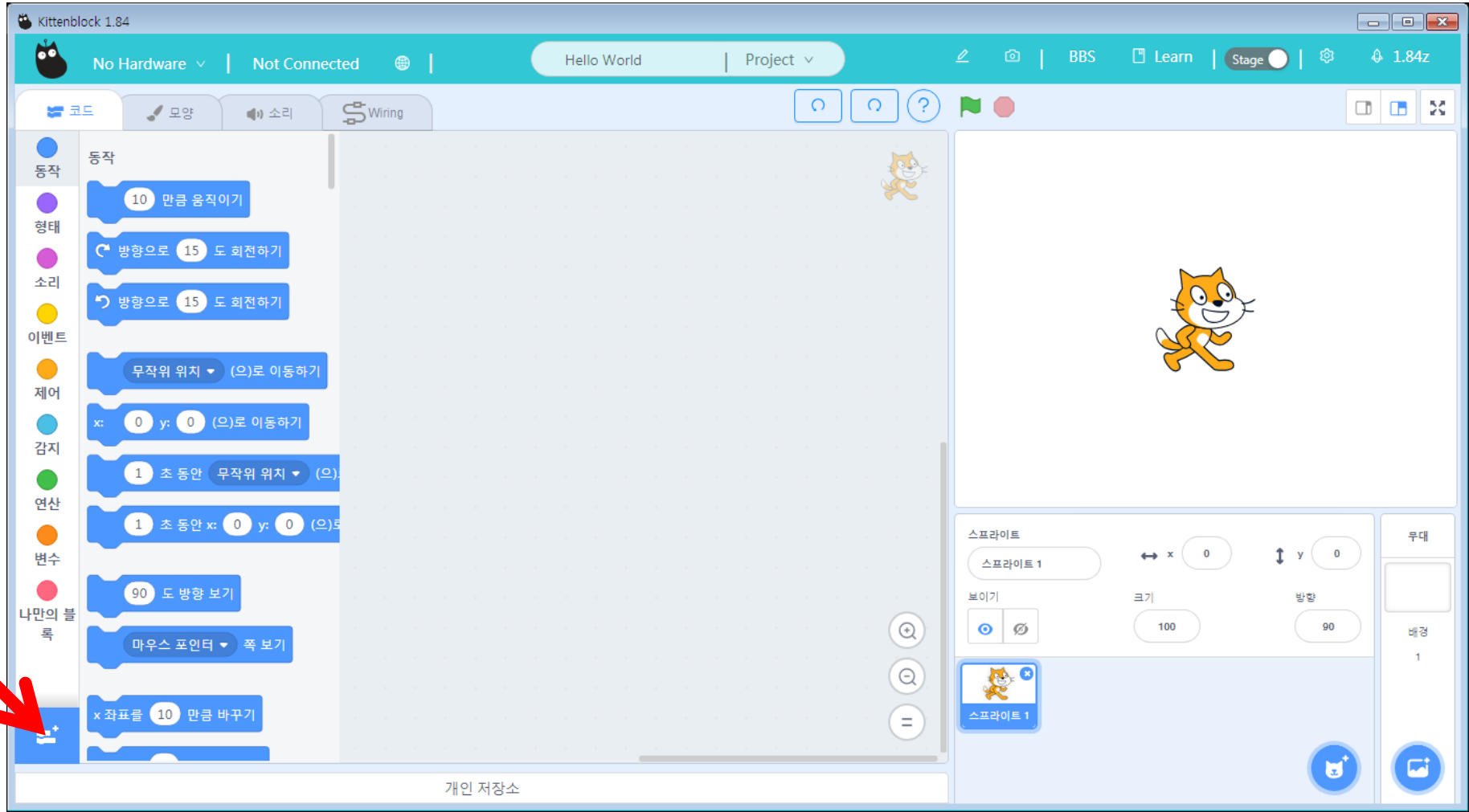
키튼블록 실행

5 설치된 키튼블록 실행



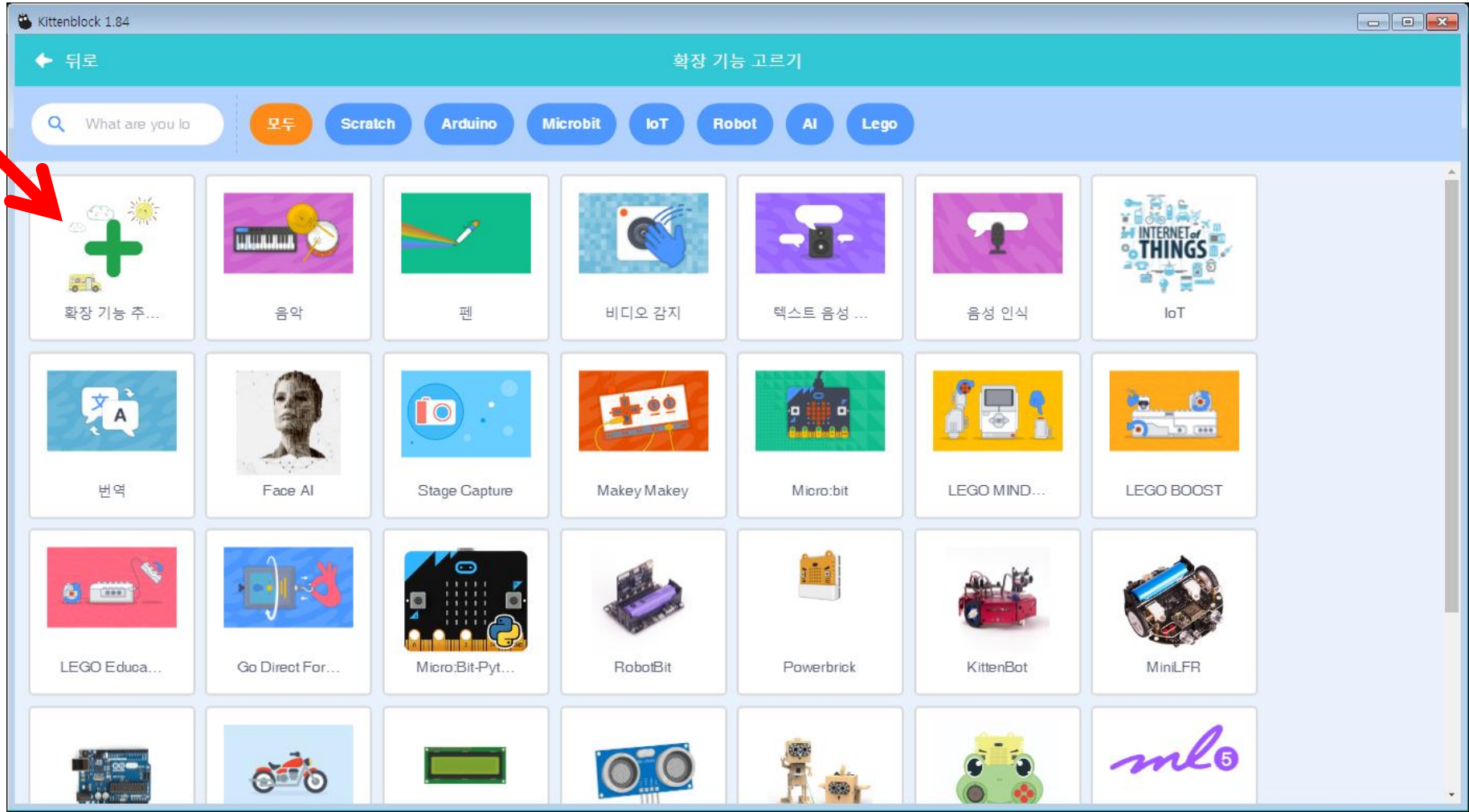
키튼블록 실행

6 확장 클릭



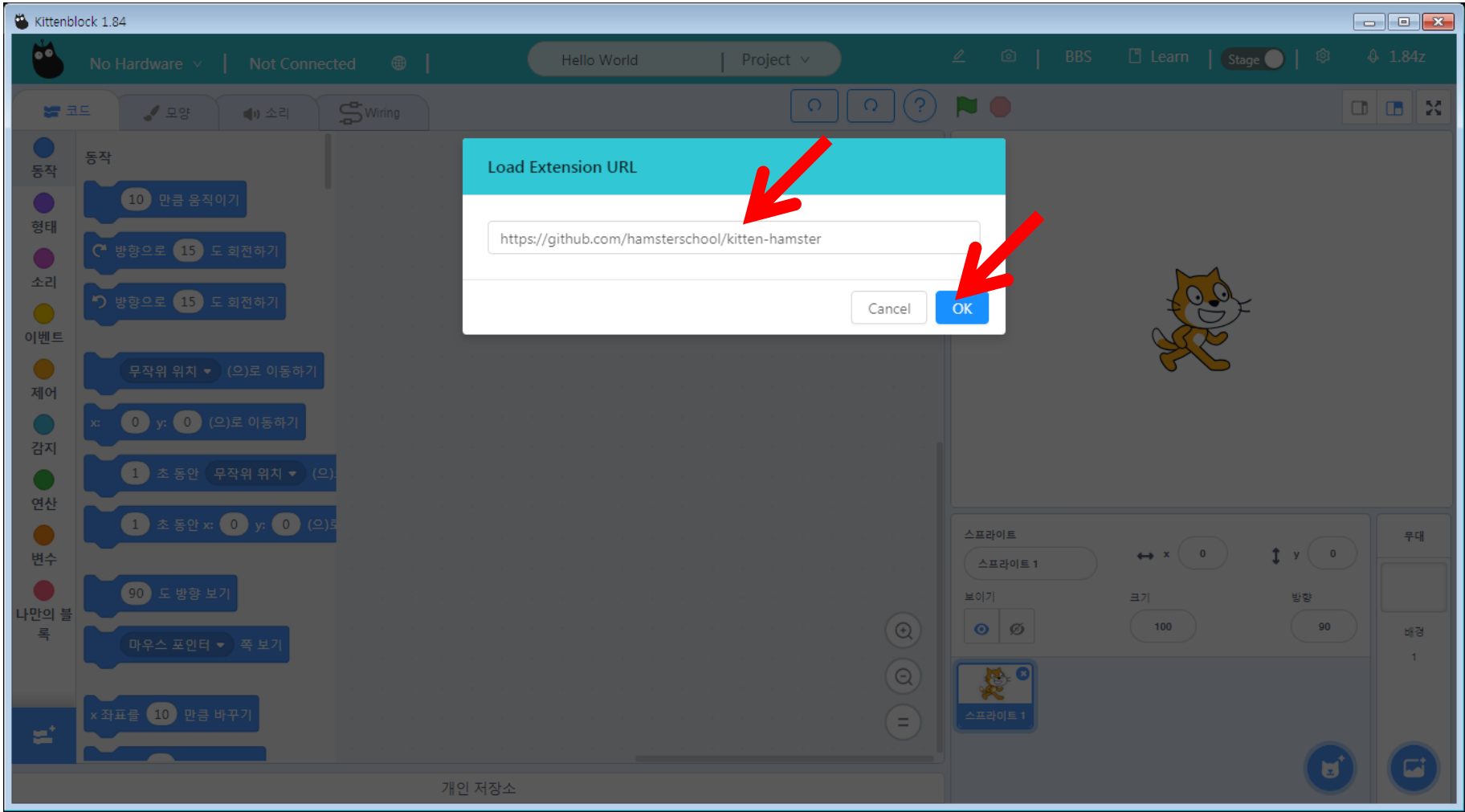
키튼블록 실행

7 확장 기능 추가 클릭



키튼블록 실행

8 URL 입력 → **OK** 클릭



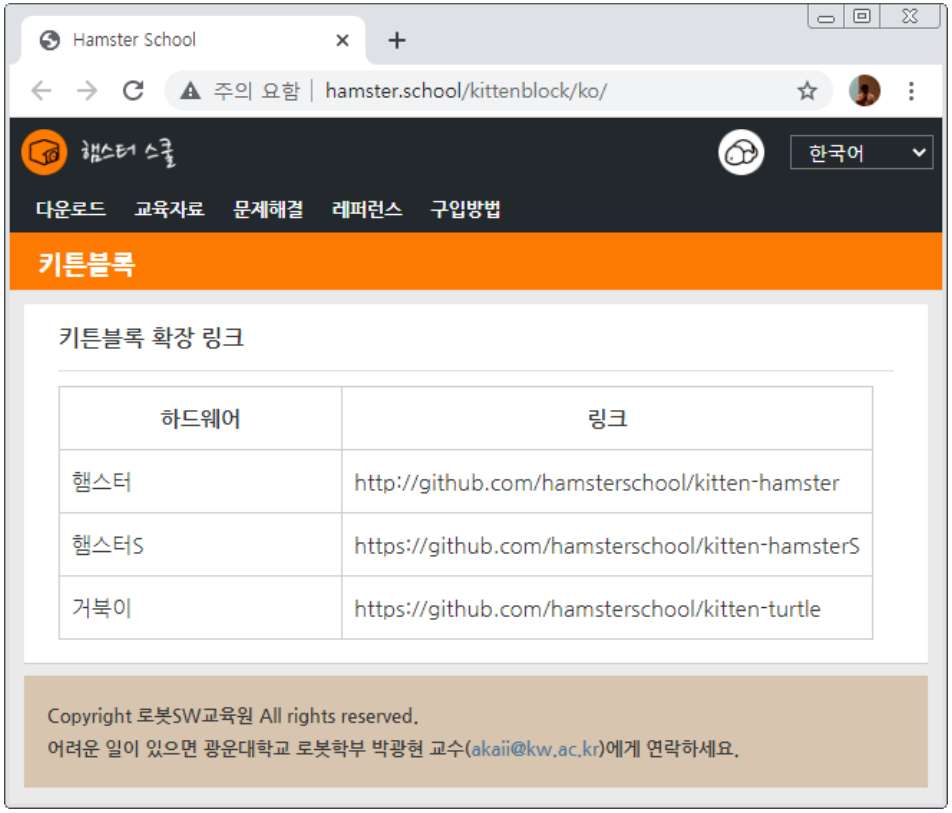
키튼블록 실행

- URL 리스트

하드웨어	링크 주소
햄스터	http://github.com/hamsterschool/kitten-hamster
햄스터S	https://github.com/hamsterschool/kitten-hamsterS
거북이	https://github.com/hamsterschool/kitten-turtle

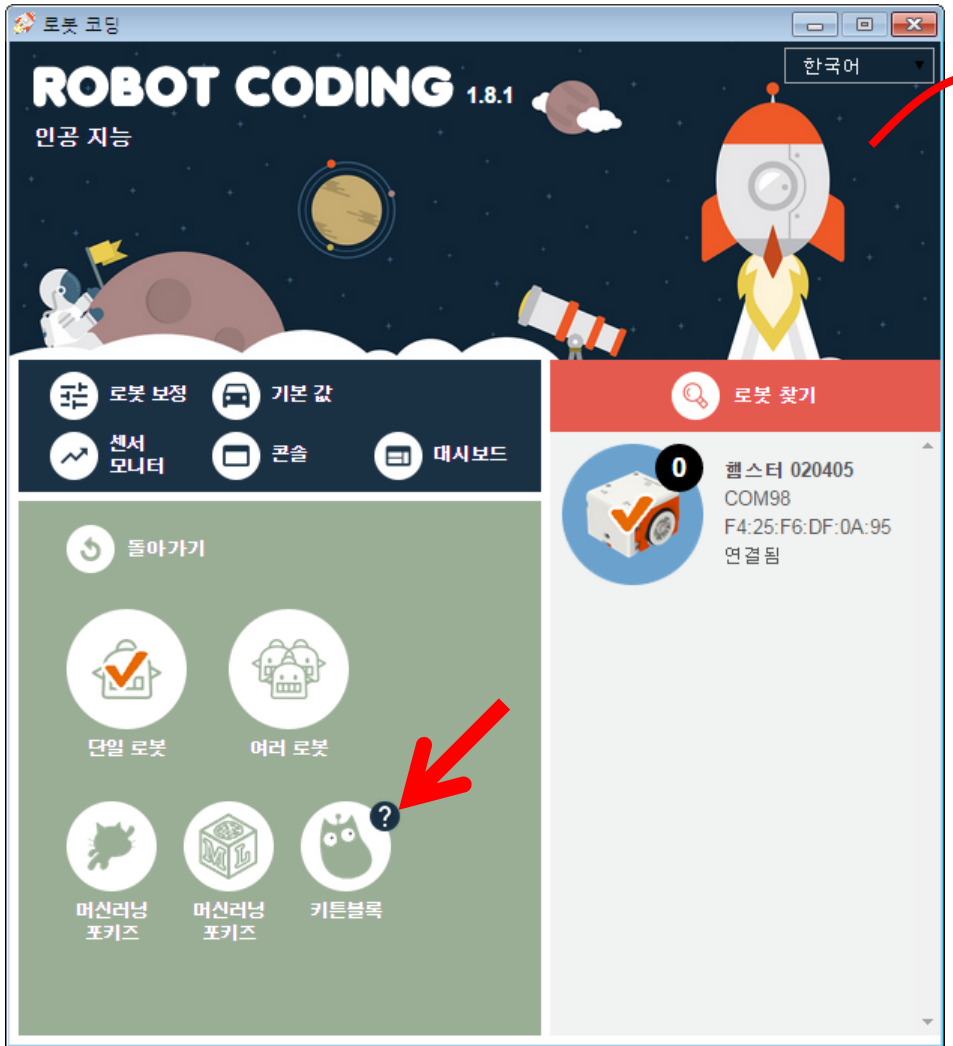
키튼블록 실행

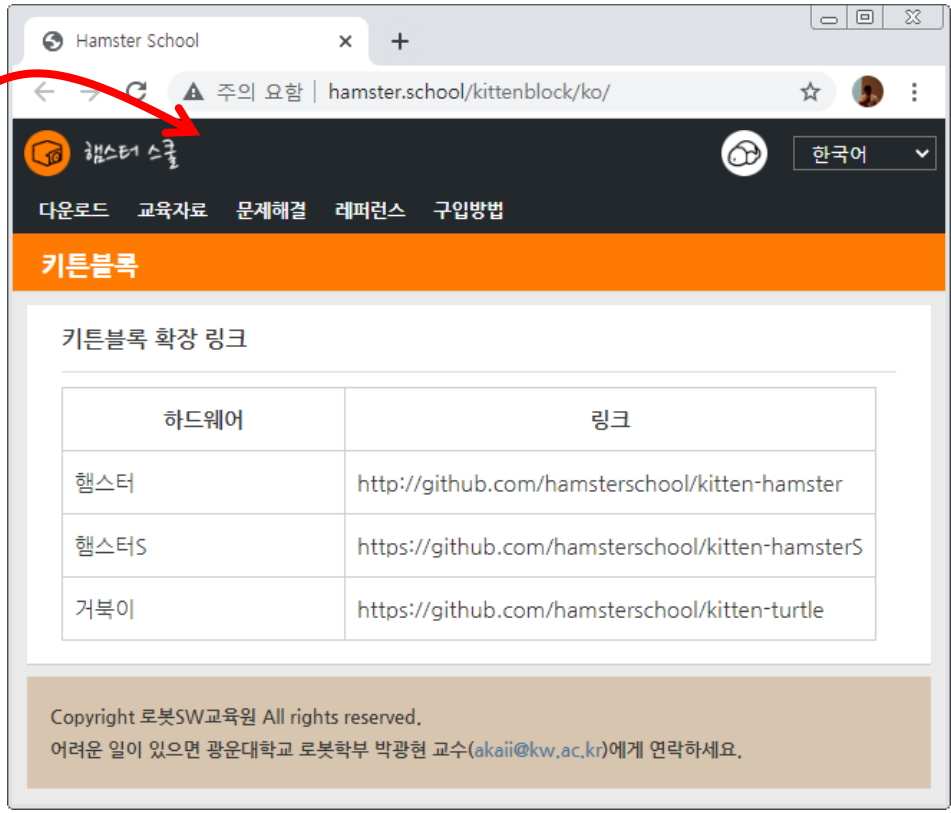
- 또는 http://hamster.school/kittenblock



키튼블록 실행

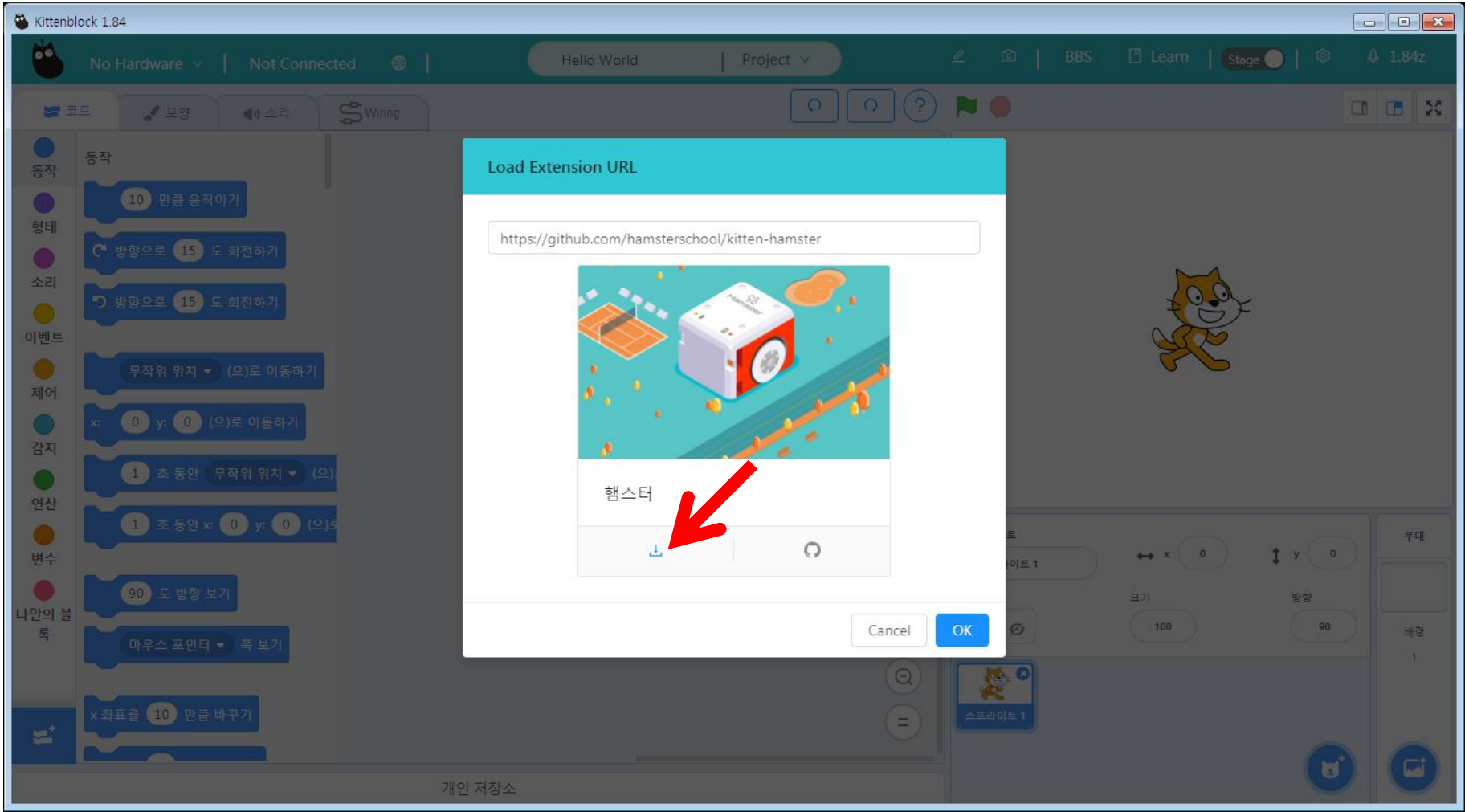
- 또는 물음표 클릭





키튼블록 실행

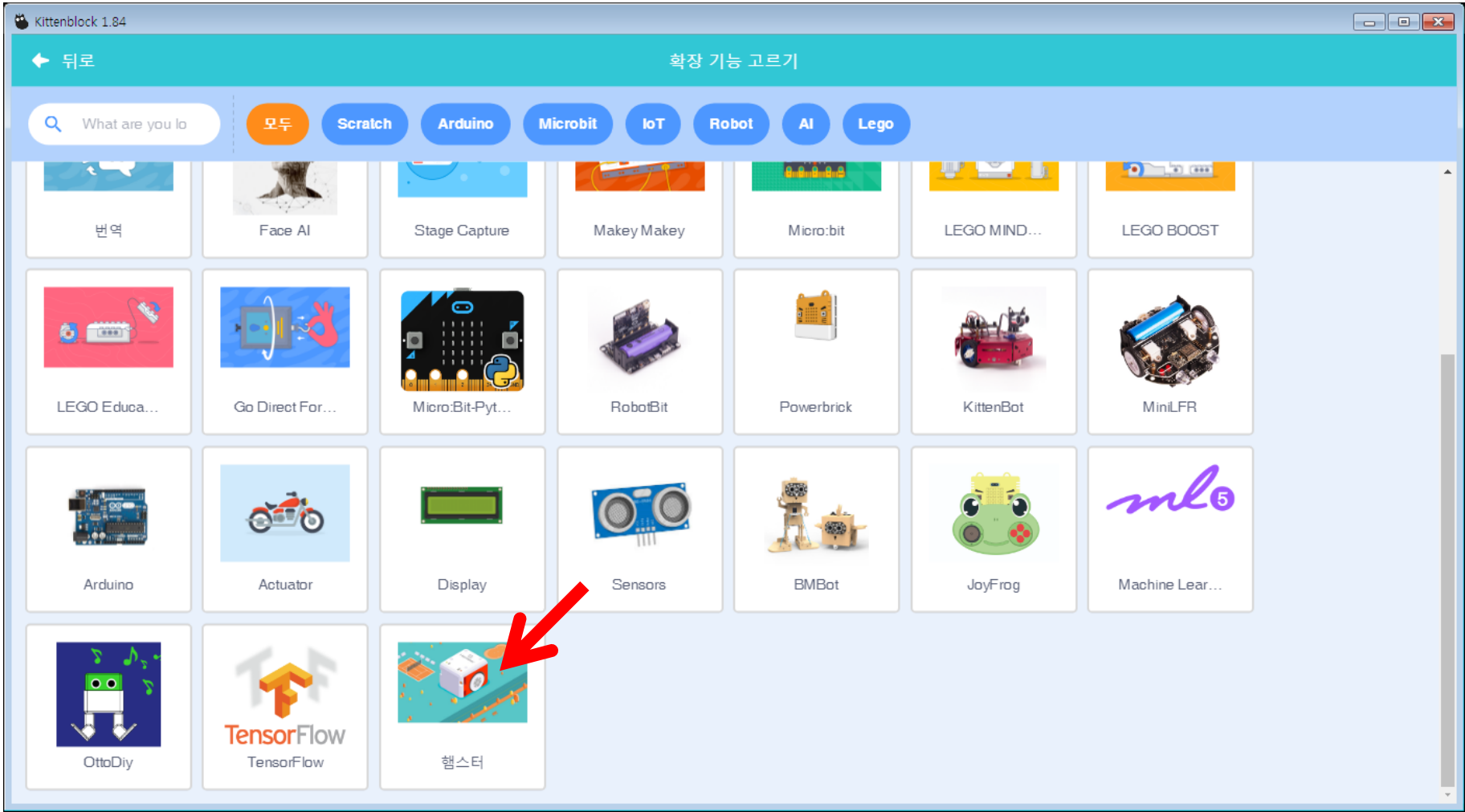
9 다운로드 버튼 클릭



키튼블록 실행

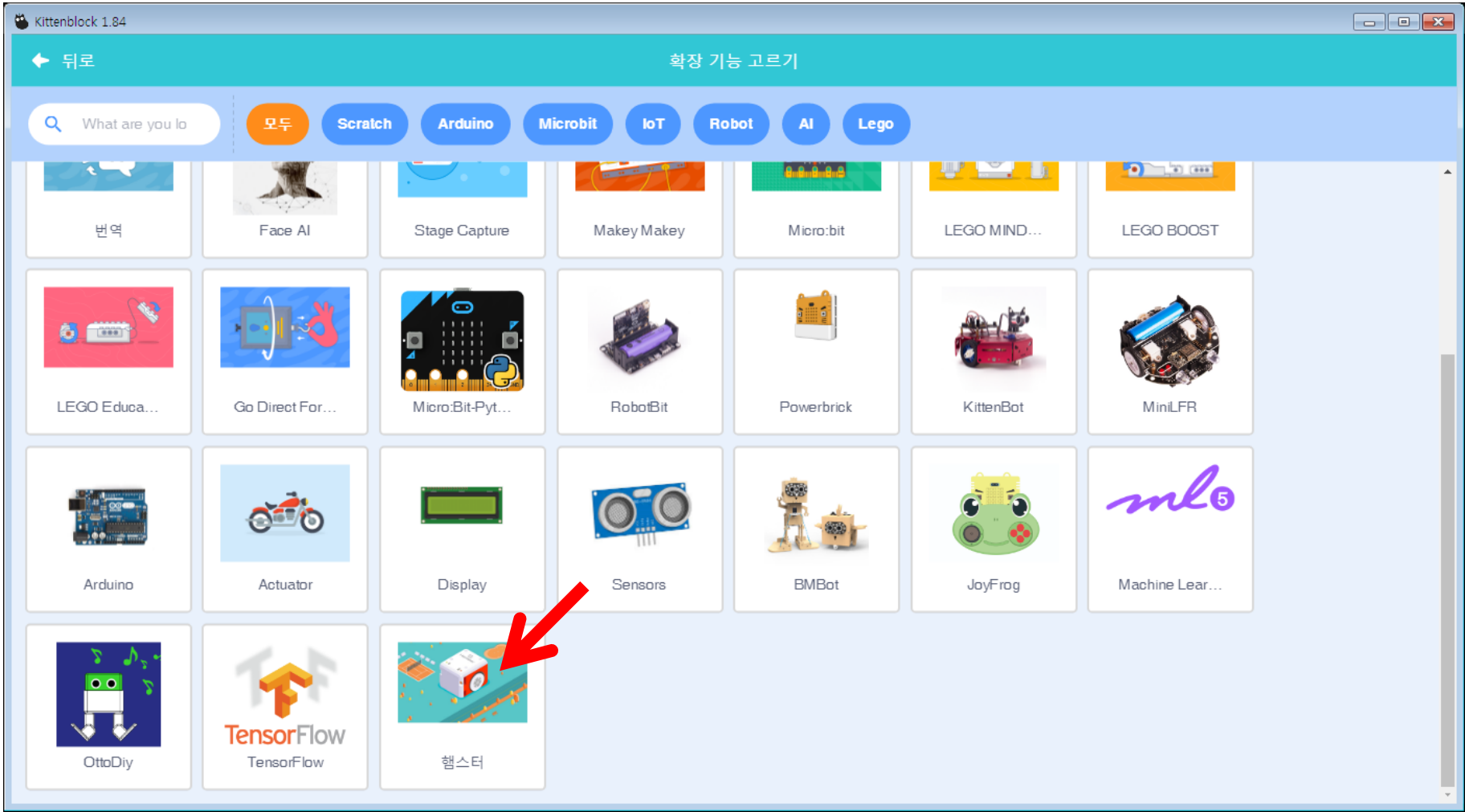
추가된 이후에는 다시 URL 입력하여 다운로드할 필요 없습니다.

10 키튼블록 새로 실행됨 → 확장 기능에 **햄스터** 추가되어 있음



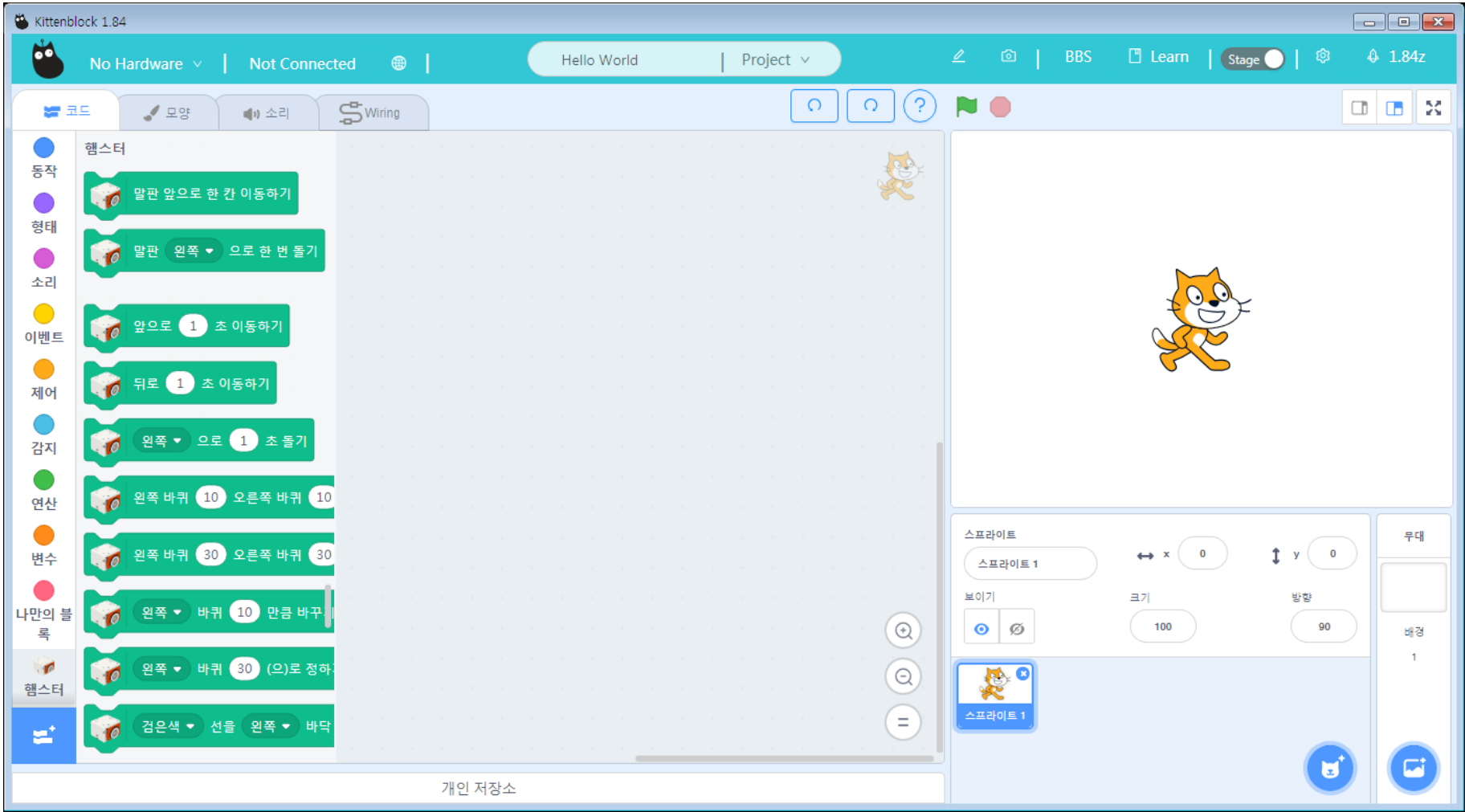
키튼블록 실행

11 햄스터 클릭



키튼블록 실행

12 햄스터 블록 추가됨

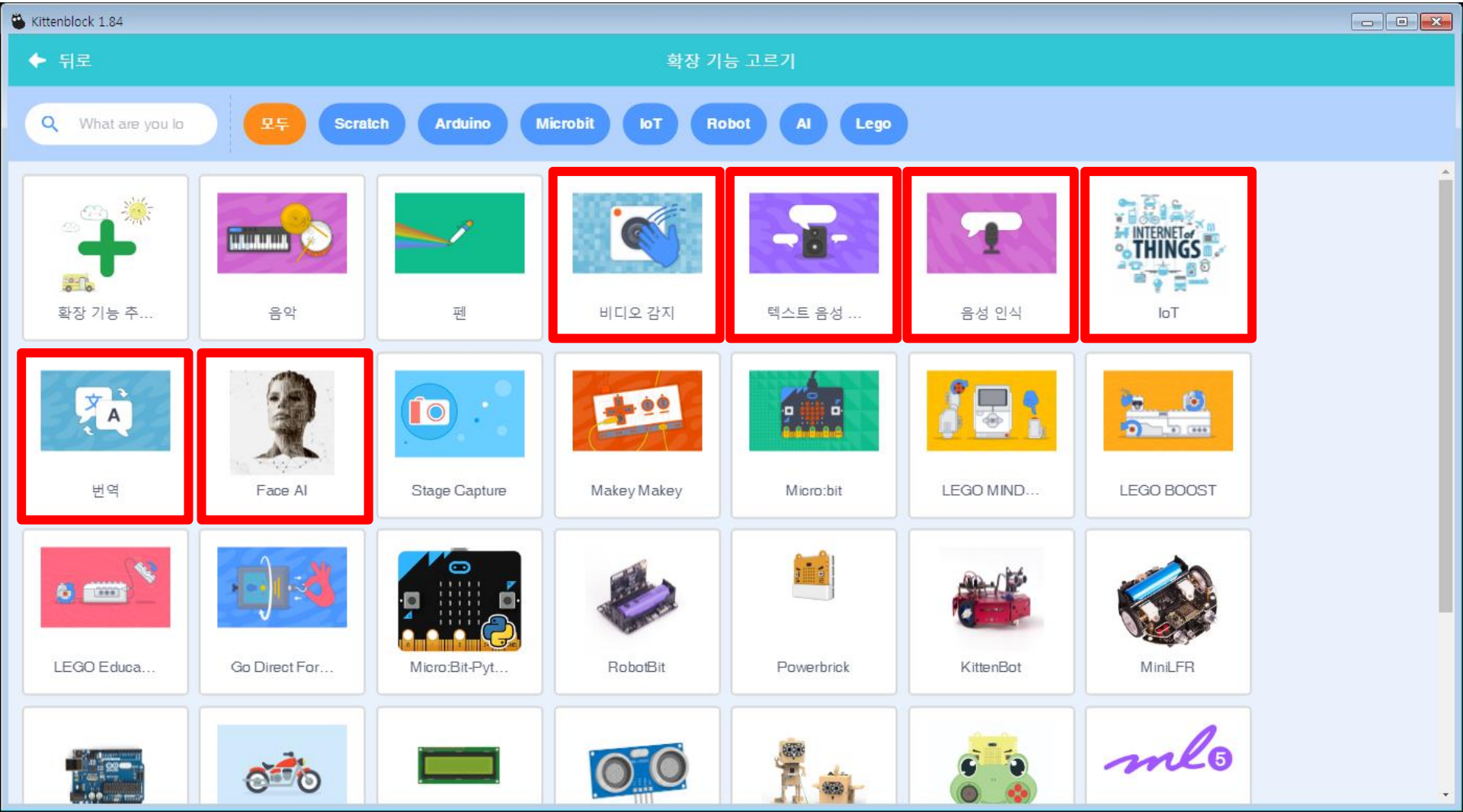


키튼블록 실행

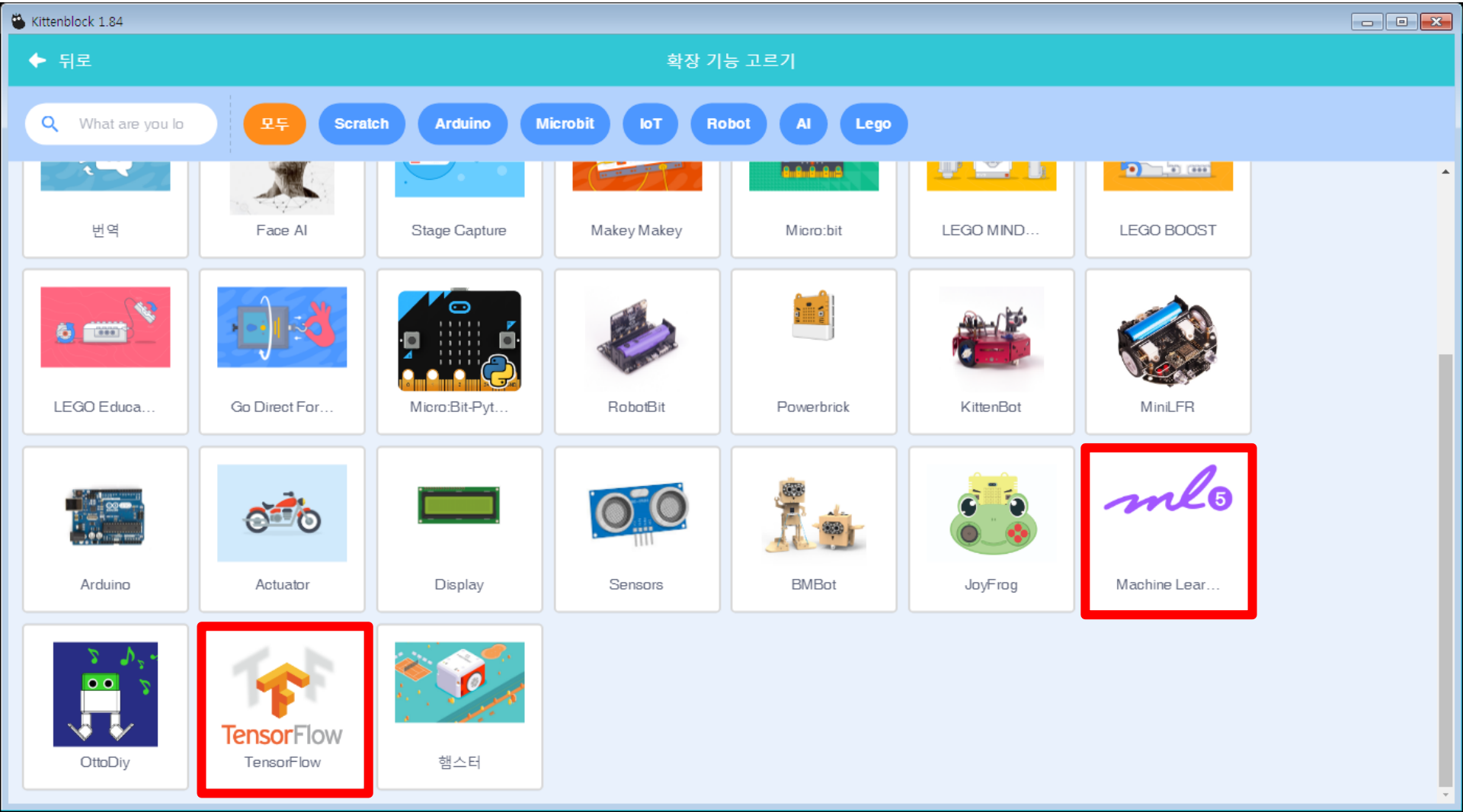
아래 두 가지 중 어느 것을 먼저 해도 상관 없습니다.

- 로봇 코딩에서 키튼블록 연결 프로그램 실행
- 키튼블록에서 확장 기능 추가 → 햄스터 추가

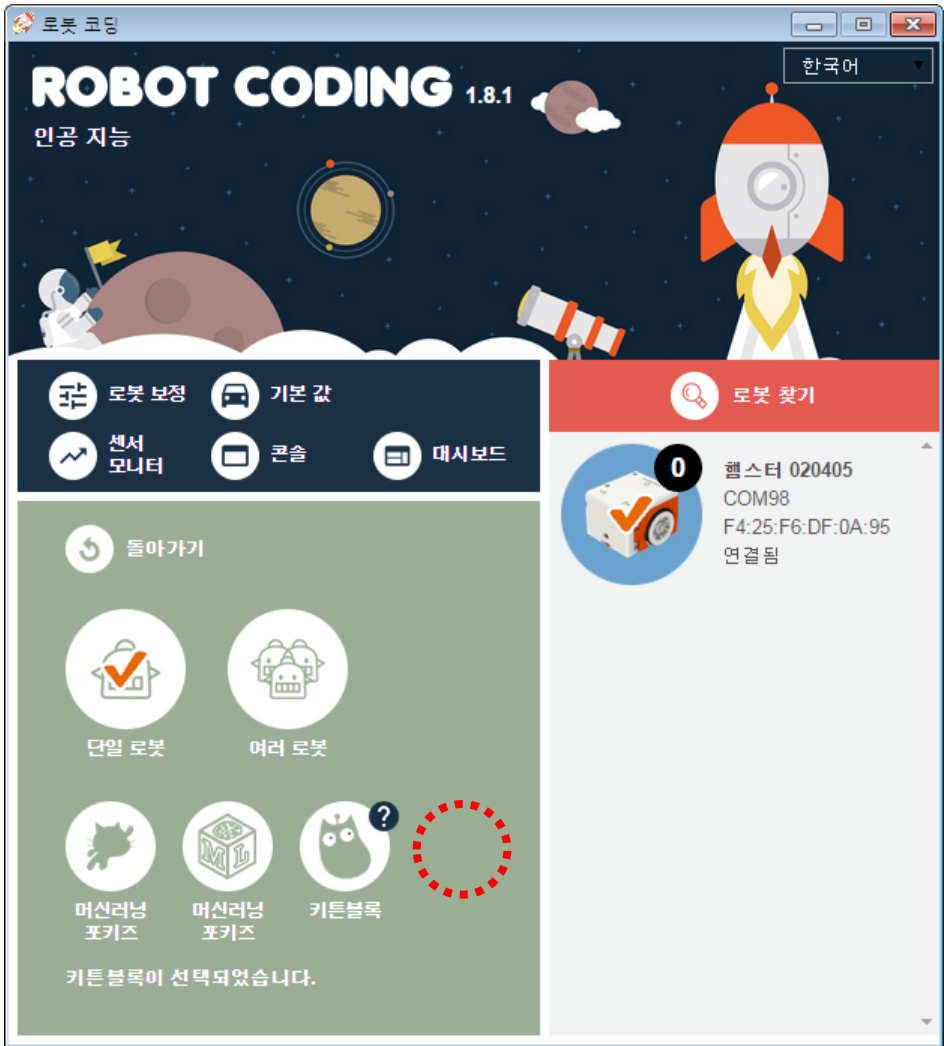
어떤 기능이 있을까요?


Kittenblock 1.84
뒤로
확장 기능 고르기
What are you lo
모두
Scratch
Arduino
Microbit
IoT
Robot
AI
Lego
확장 기능 추...
음악
펜
비디오 감지
텍스트 음성 ...
음성 인식
IoT
번역
Face AI
Stage Capture
Makey Makey
Micro:bit
LEGO MIND...
LEGO BOOST
LEGO Educa...
Go Direct For...
Micro:Bit-Pyt...
RobotBit
Powerbrick
KittenBot
MiniLFR

어떤 기능이 있을까요?


Kittenblock 1.84
뒤로
확장 기능 고르기
What are you lo
모두
Scratch
Arduino
Microbit
IoT
Robot
AI
Lego
번역
Face AI
Stage Capture
Makey Makey
Micro:bit
LEGO MIND...
LEGO BOOST
LEGO Educa...
Go Direct For...
Micro:Bit-Pyt...
RobotBit
Powerbrick
KittenBot
MiniLFR
Arduino
Actuator
Display
Sensors
BMBot
JoyFrog
Machine Lear...
OttoDiy
TensorFlow
TensorFlow
햄스터

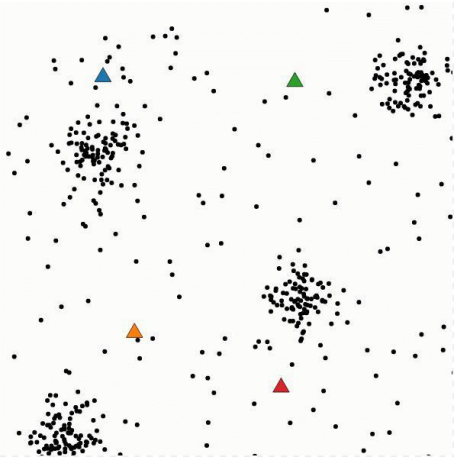
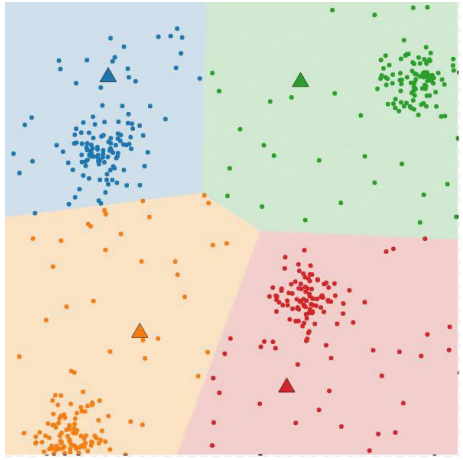
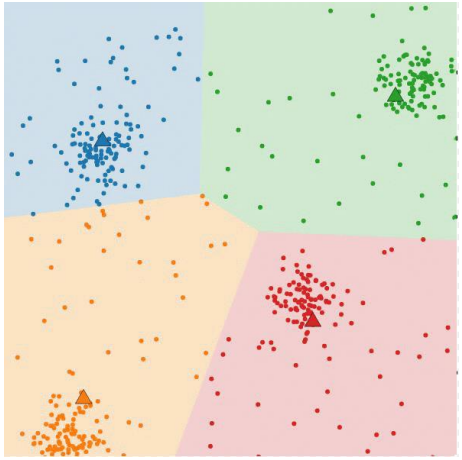
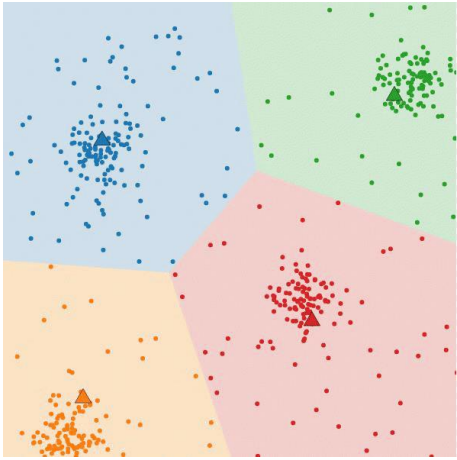
앞으로의 계획?


로봇 코딩
ROBOT CODING 1.8.1
인공 지능
한국어
로봇 보정
기본 값
센서 모니터
콘솔
대시보드
로봇 찾기
0
햄스터 020405
COM98
F4:25:F6:DF:0A:95
연결됨
돌아가기
단일 로봇
여러 로봇
머신러닝 포키즈
머신러닝 포키즈
키튼블록
키튼블록이 선택되었습니다.

선생님을 위한 보충 설명

k-Means 알고리즘

1. 초기화 : k개의 중심 설정
2. 각 데이터를 가장 가까운 중심점의 클러스터에 할당
3. 각 클러스터에 속한 데이터들의 평균을 새로운 중심점으로 갱신

k-Means 알고리즘

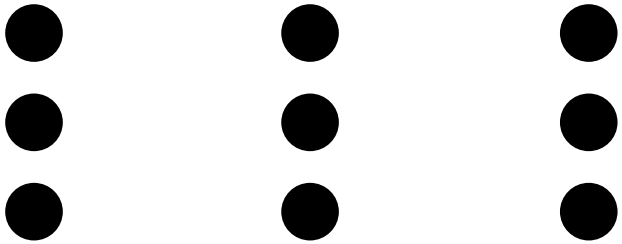
1. 초기화 : k개의 중심 설정
2. 각 데이터를 가장 가까운 중심점의 클러스터에 할당
3. 각 클러스터에 속한 데이터들의 평균을 새로운 중심점으로 갱신

종료 조건

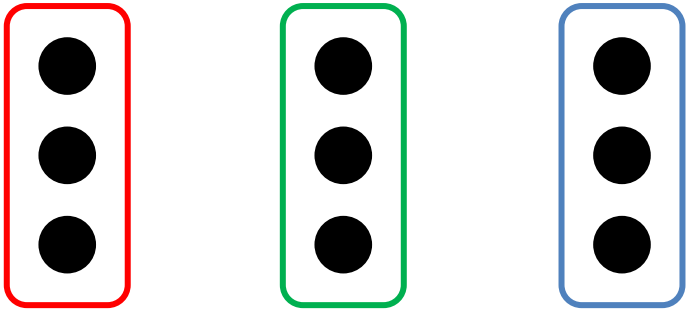
- 중심점에 변화가 없을 때까지
- 각 데이터의 소속 클러스터가 바뀌지 않을 때까지
- 지정된 횟수만큼

초기화 방법

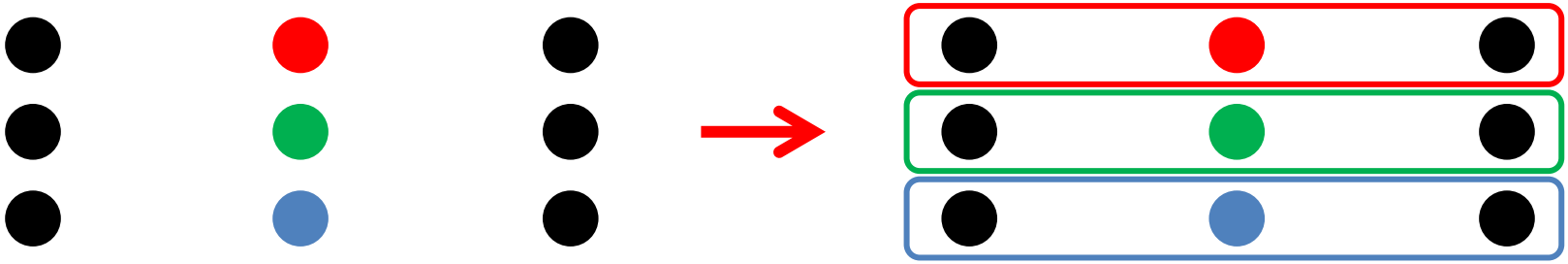
k=3



이렇게 되어야 할 것 같은데

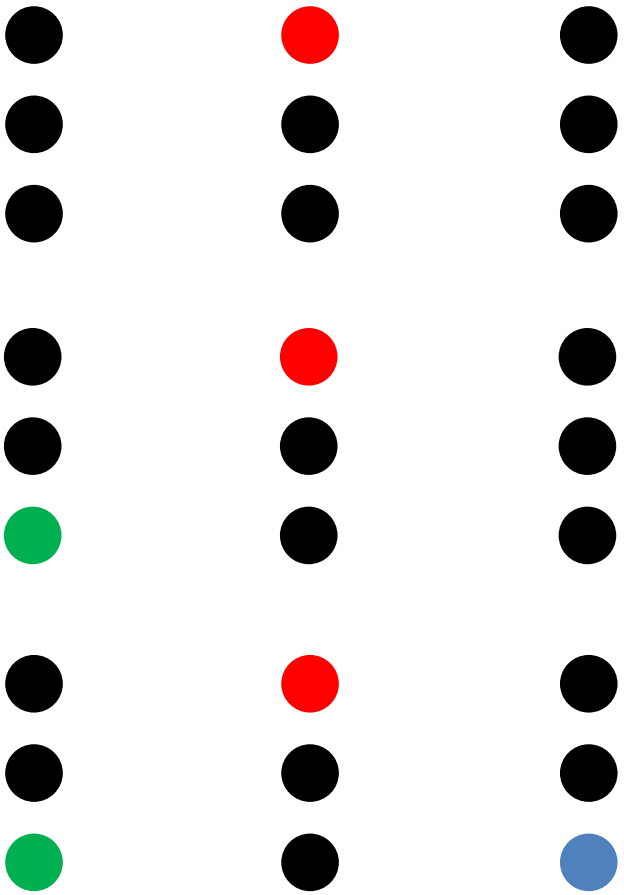


초기 중심을 이렇게 하면



개선된 초기화 방법 #1

k=3

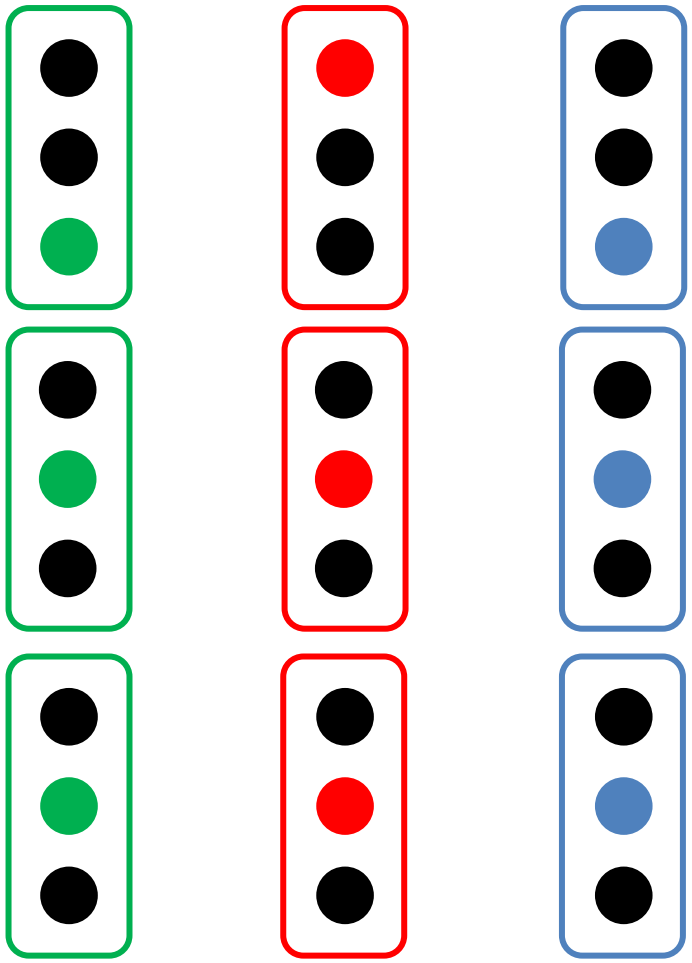


첫 번째 중심은 랜덤으로

두 번째 중심은 첫 번째 중심에서 가장 먼 것

세 번째 중심은 첫 번째와 두 번째 중심에서 가장 먼 것

개선된 초기화 방법 #1



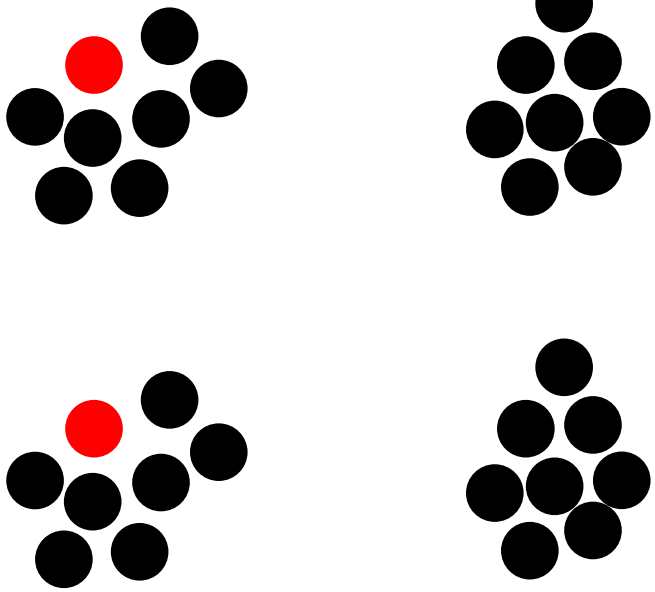
가장 가까운 중심의 클러스터에 할당

중심 위치 갱신

가장 가까운 중심의 클러스터에 할당

개선된 초기화 방법 #1

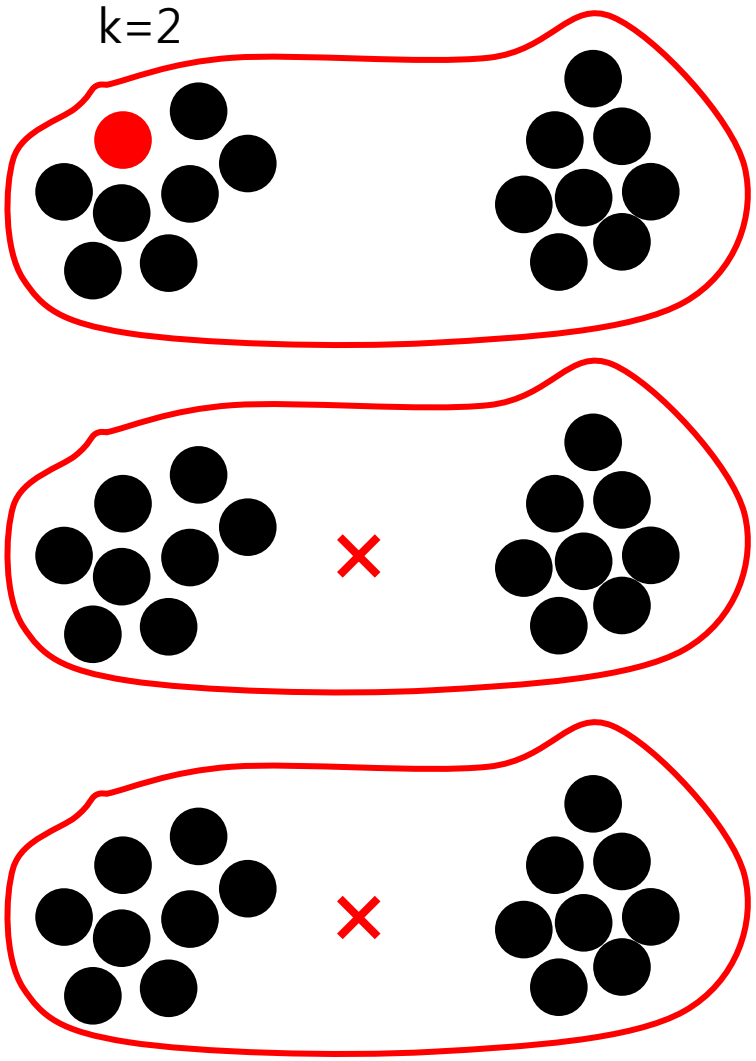
k=2



첫 번째 중심은 랜덤으로

두 번째 중심은 첫 번째 중심에서 가장 먼 것

개선된 초기화 방법 #1



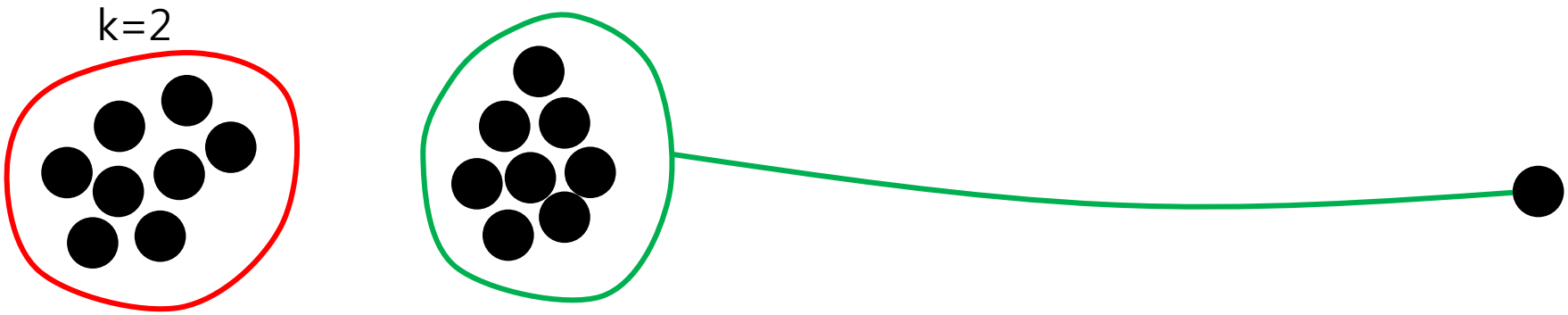
가장 가까운 중심의 클러스터에 할당

중심 위치 갱신

가장 가까운 중심의 클러스터에 할당

개선된 초기화 방법 #1

이렇게 되어야 할 것 같은데

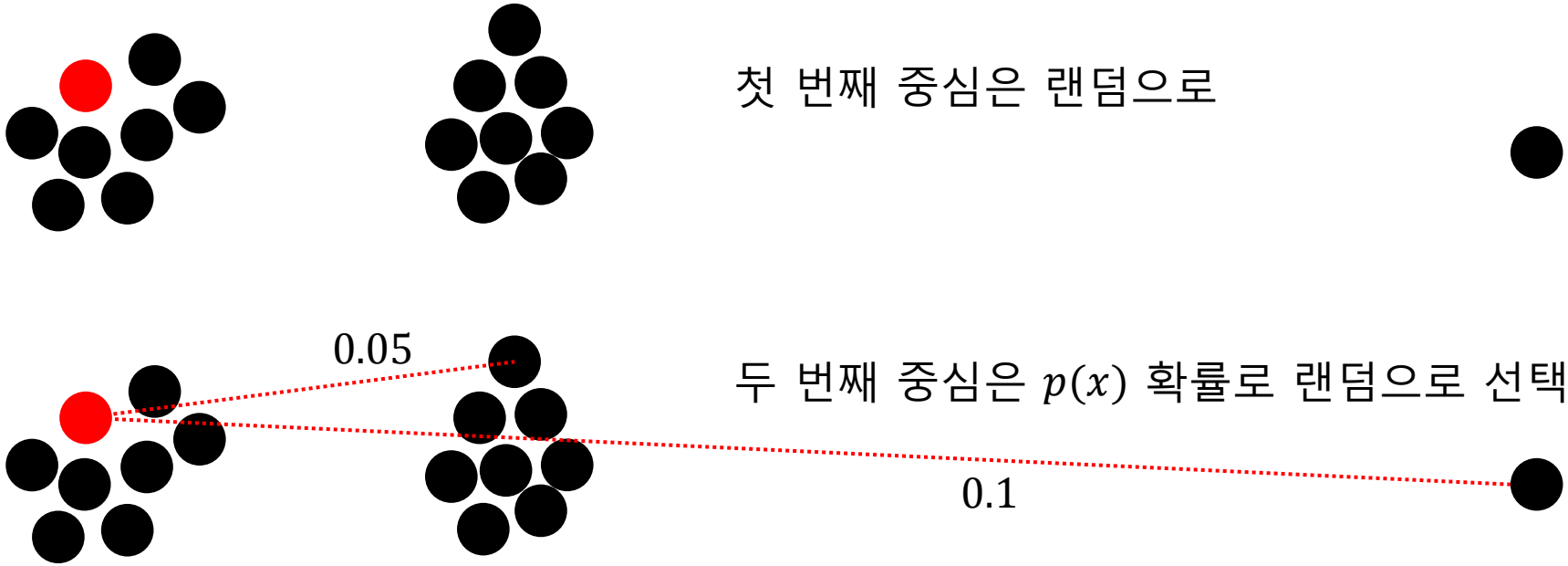


개선된 초기화 방법 #2

$D(x)$: 데이터 x에 대해 가장 가까운 중심까지의 거리

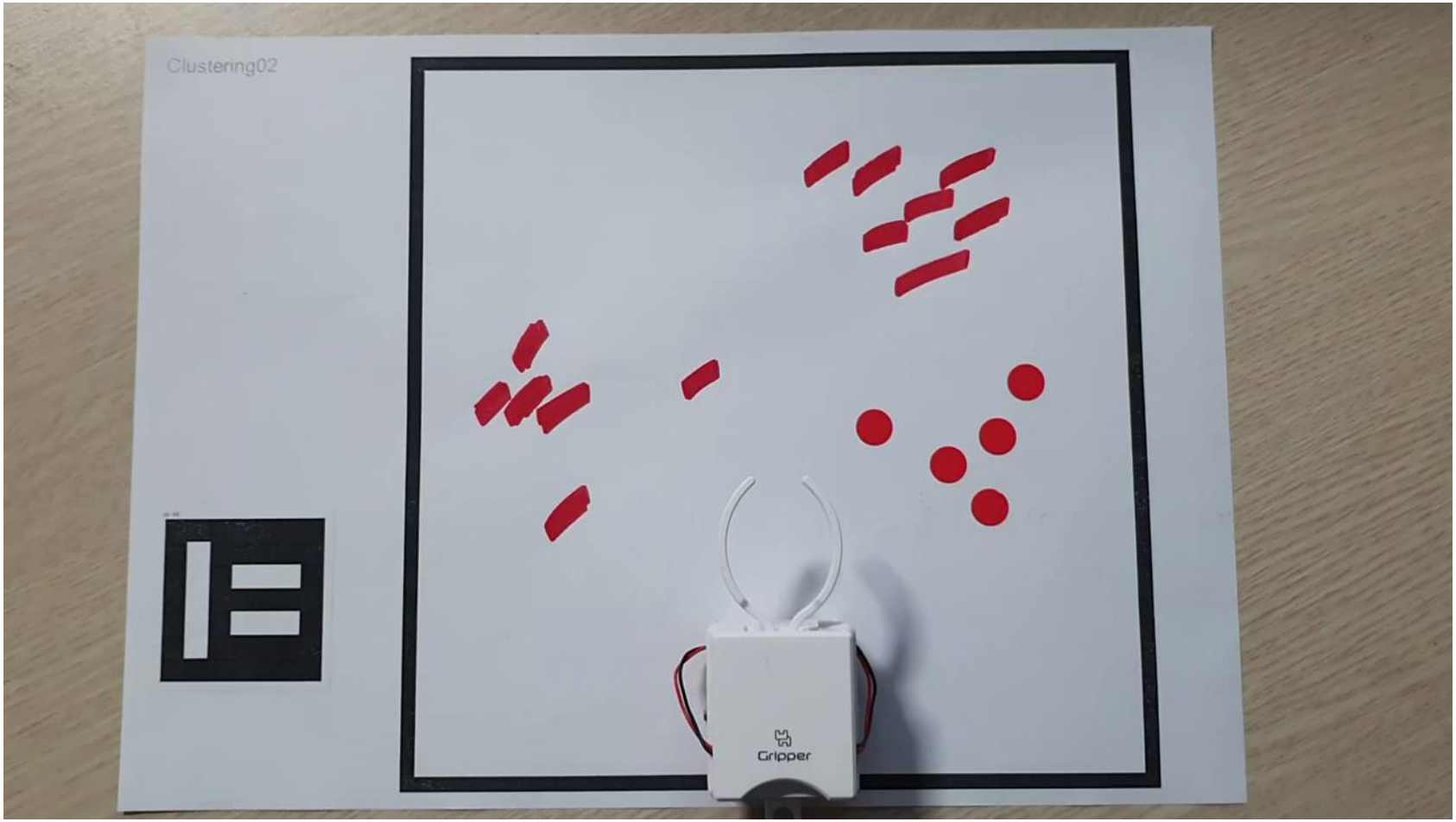
$p(x)$: $\frac{D(x)^2}{\sum_{x'} D(x')^2}$ → $D(x)^2$에 비례

k=2

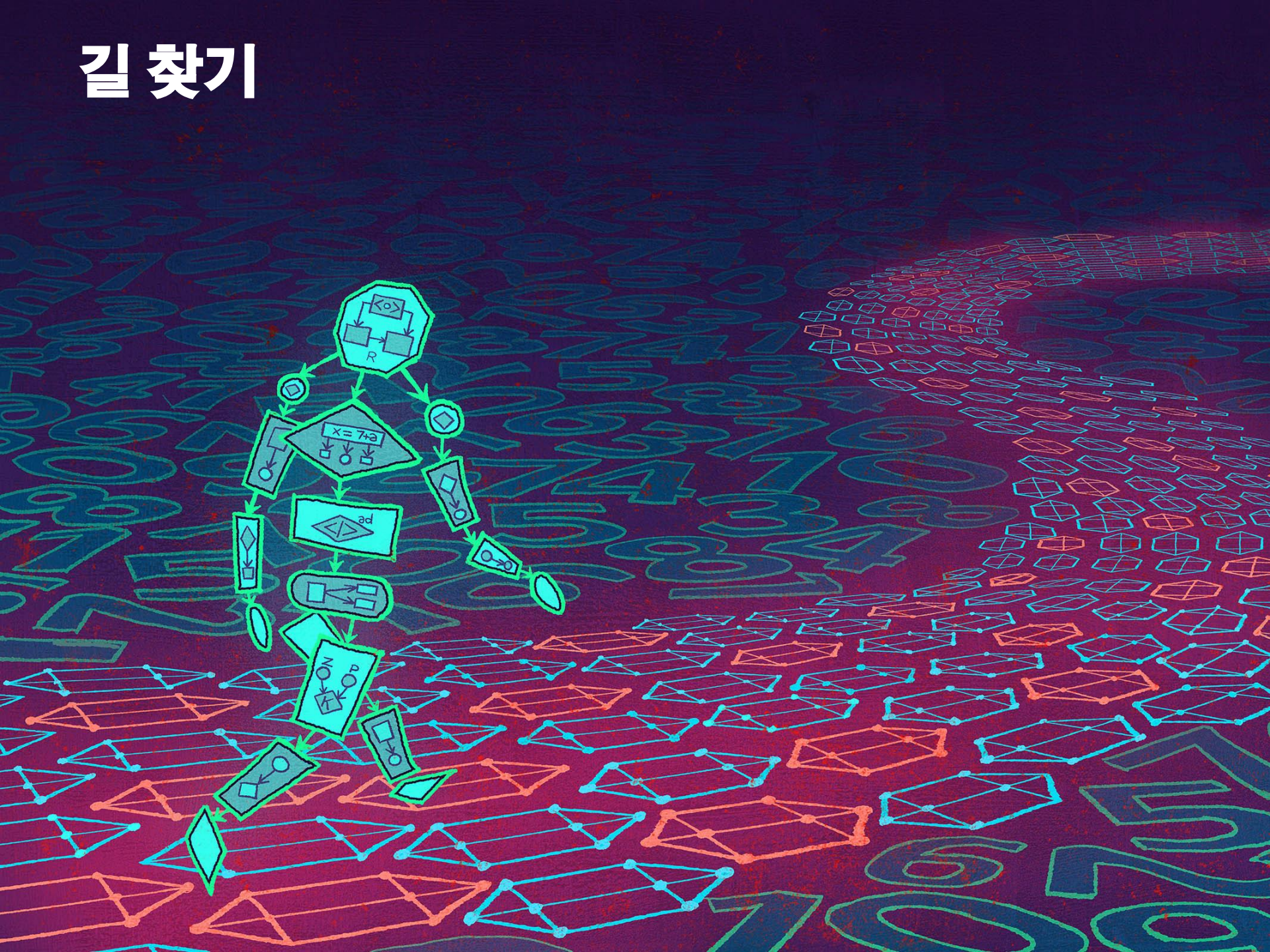


클러스터링 데모

https://youtu.be/QyqctNf3q_o



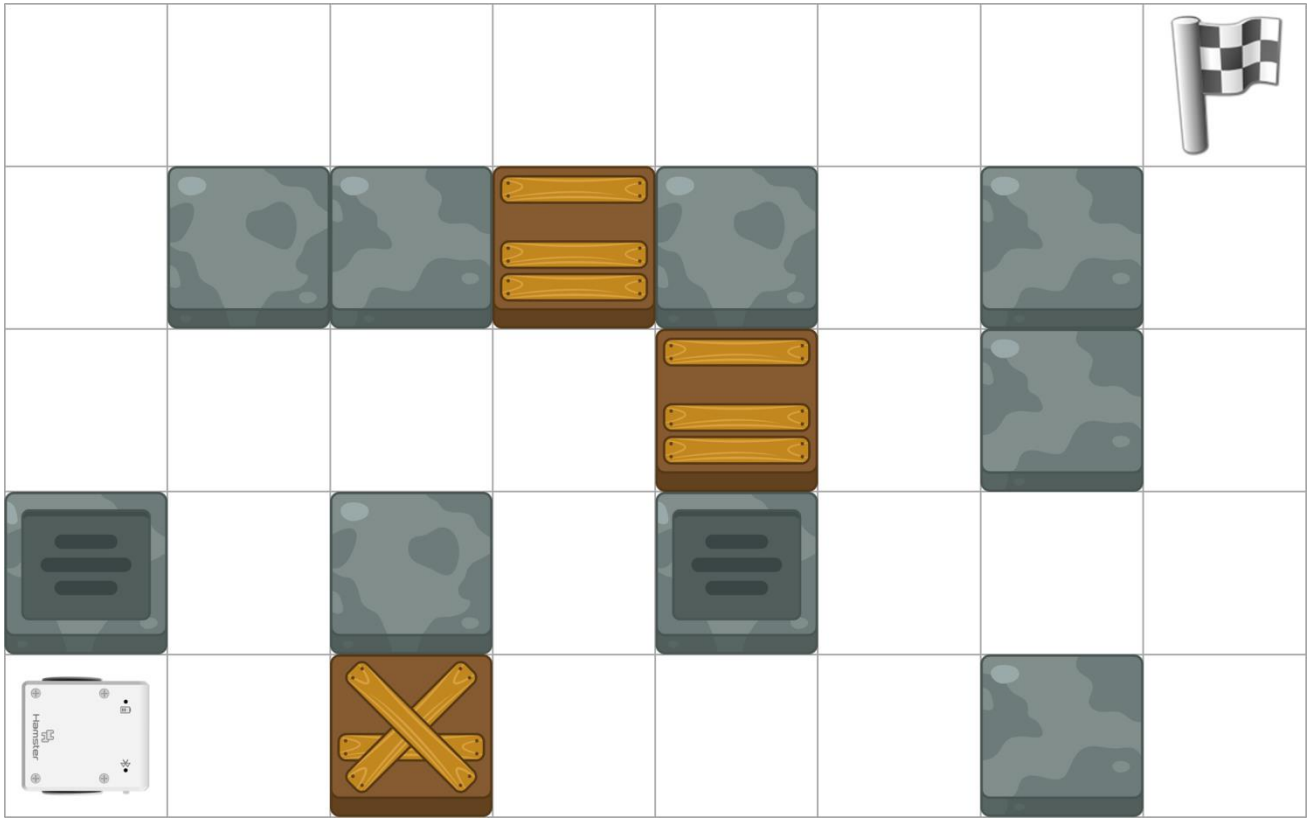
길 찾기



규칙을 직접 코딩에서 기계 학습으로

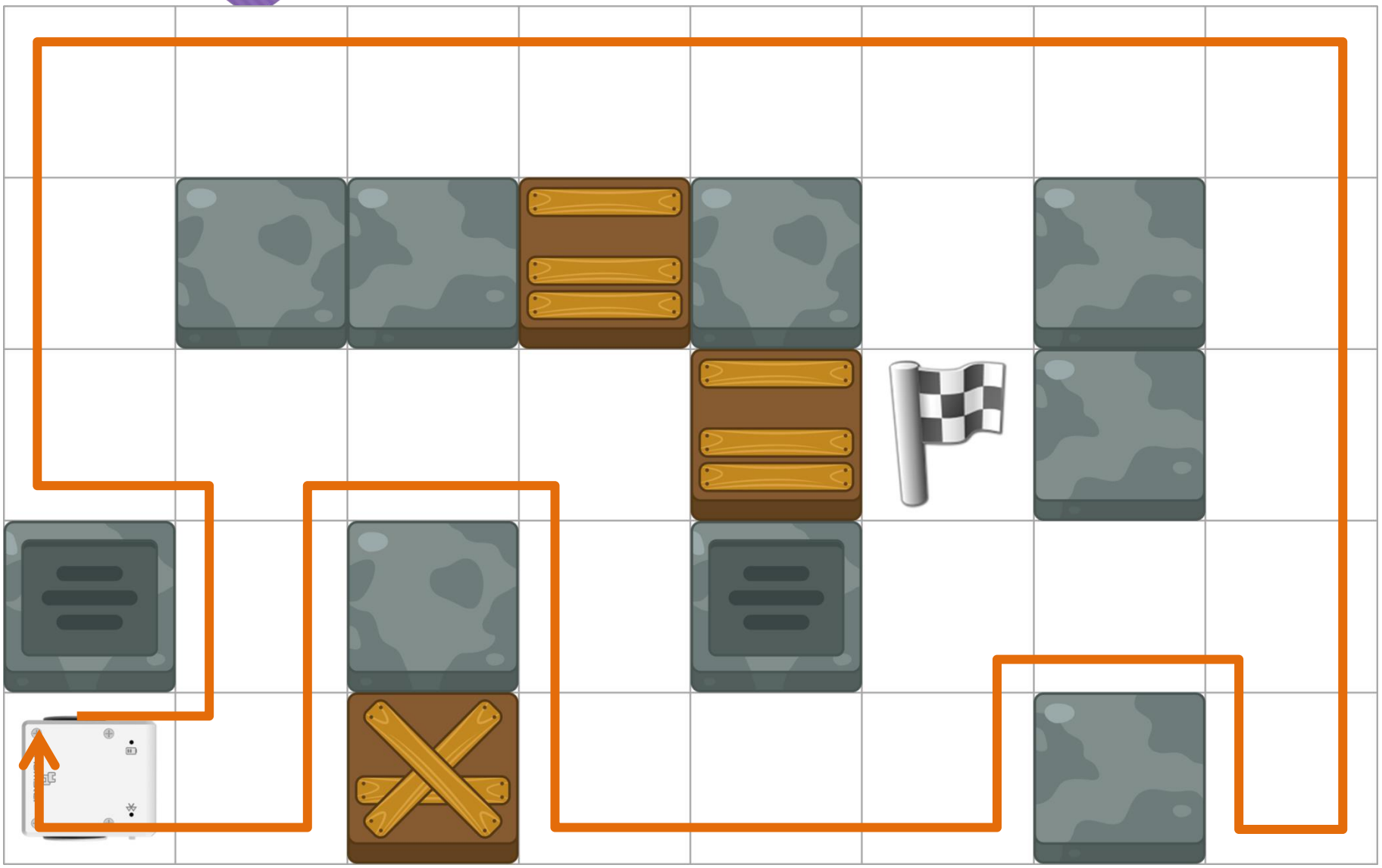
규칙을 직접 코딩에서 기계 학습으로

규칙을 직접 코딩에서 기계 학습으로

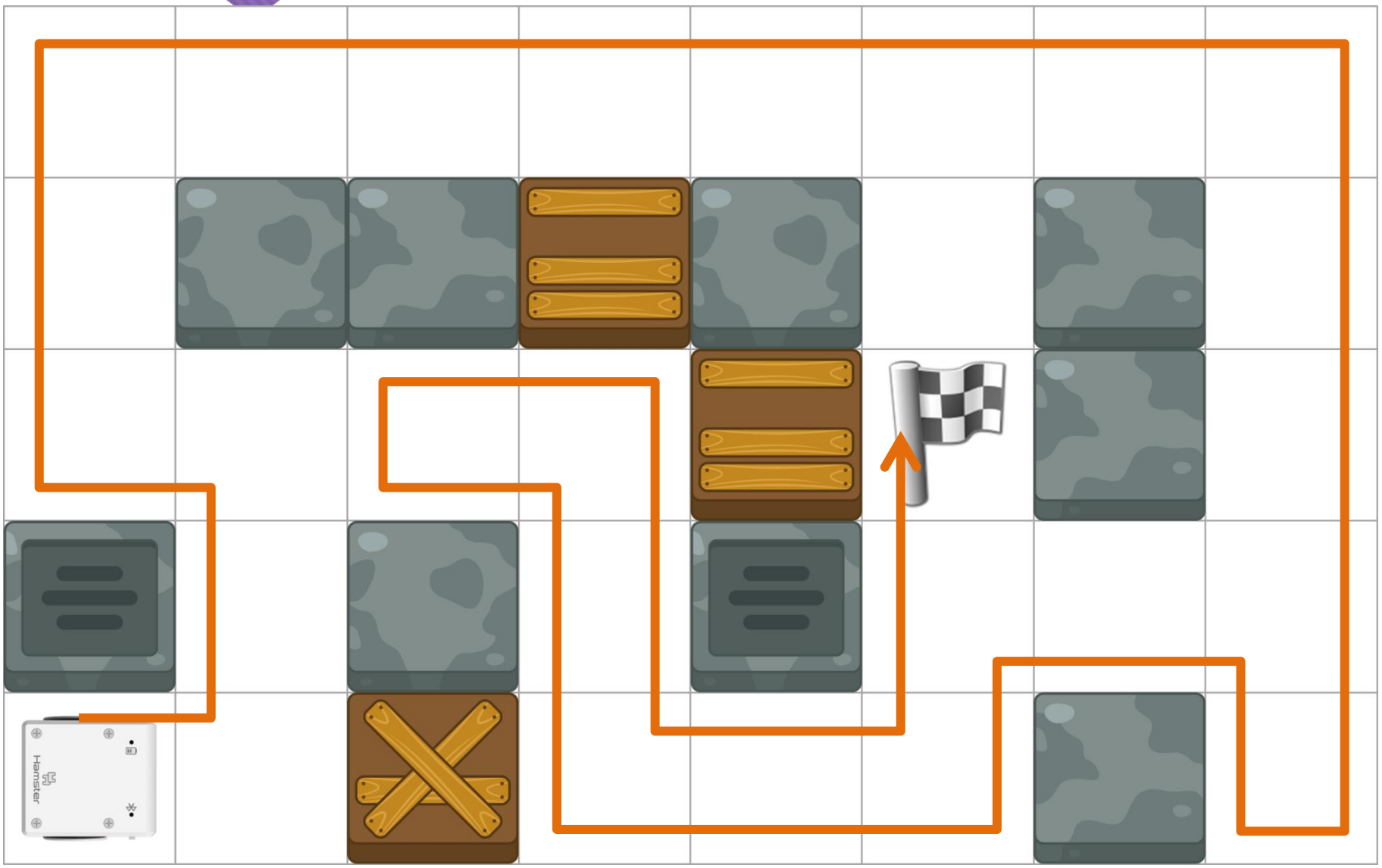


좌수법

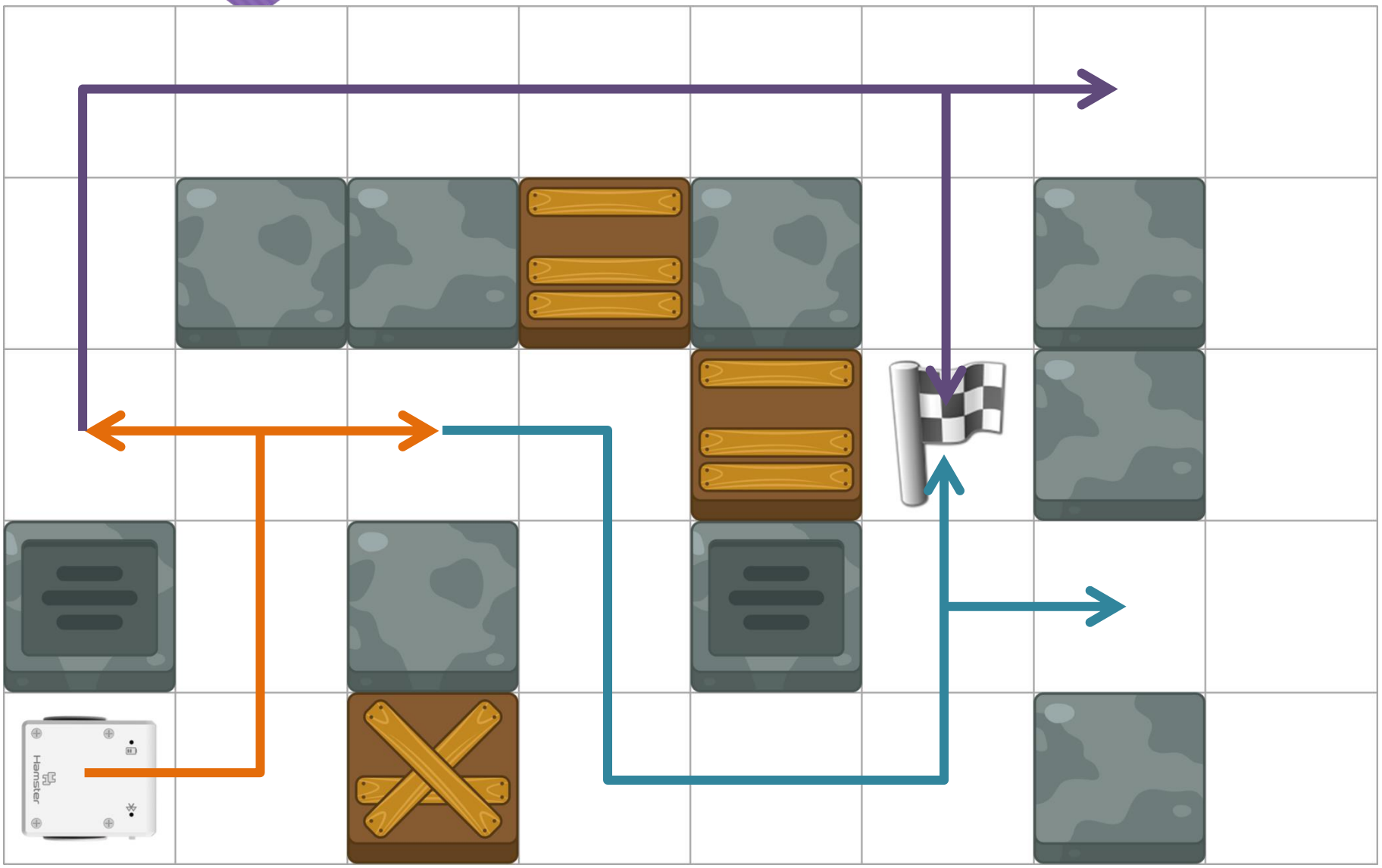
좌수법



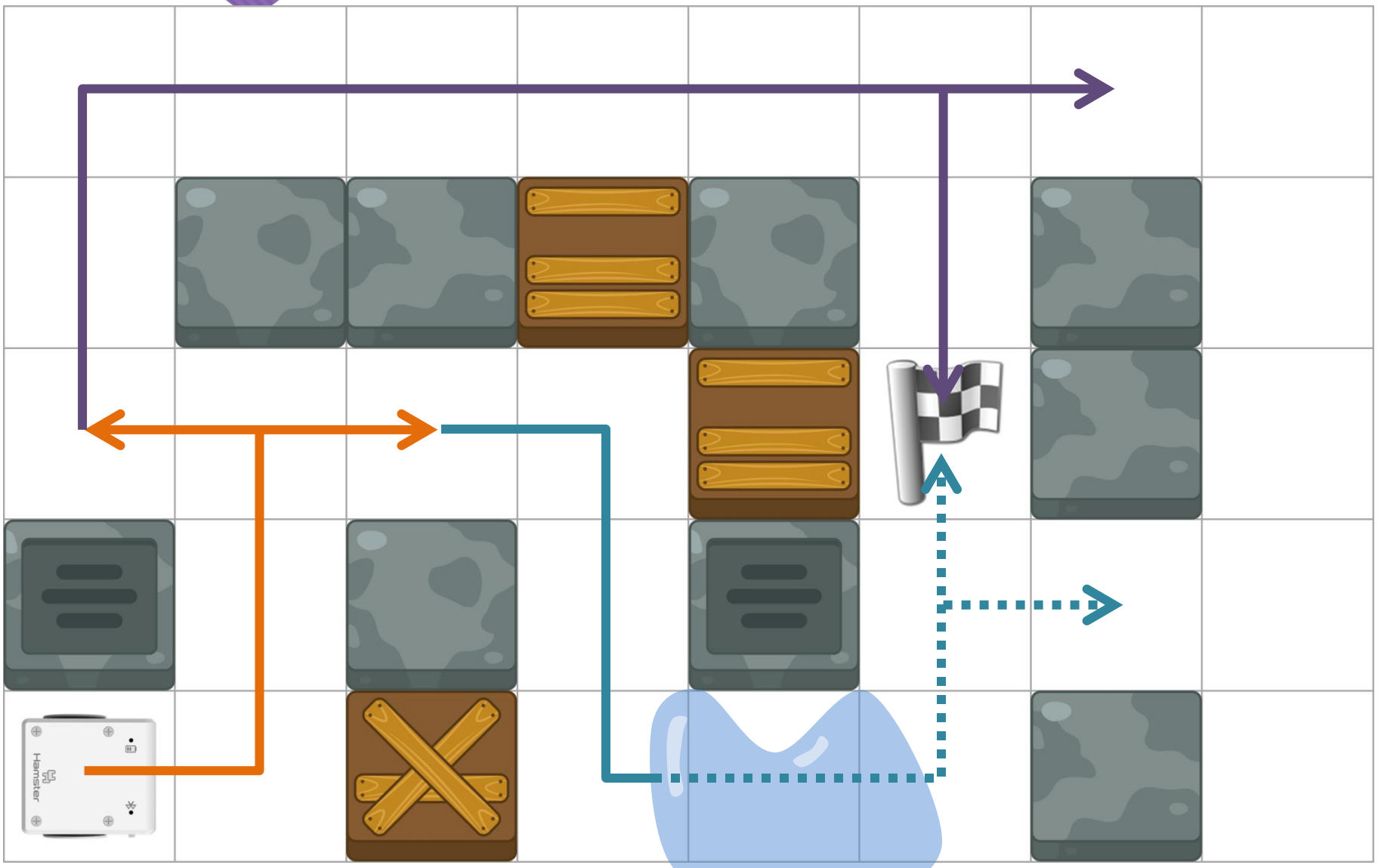
깊이 우선 탐색



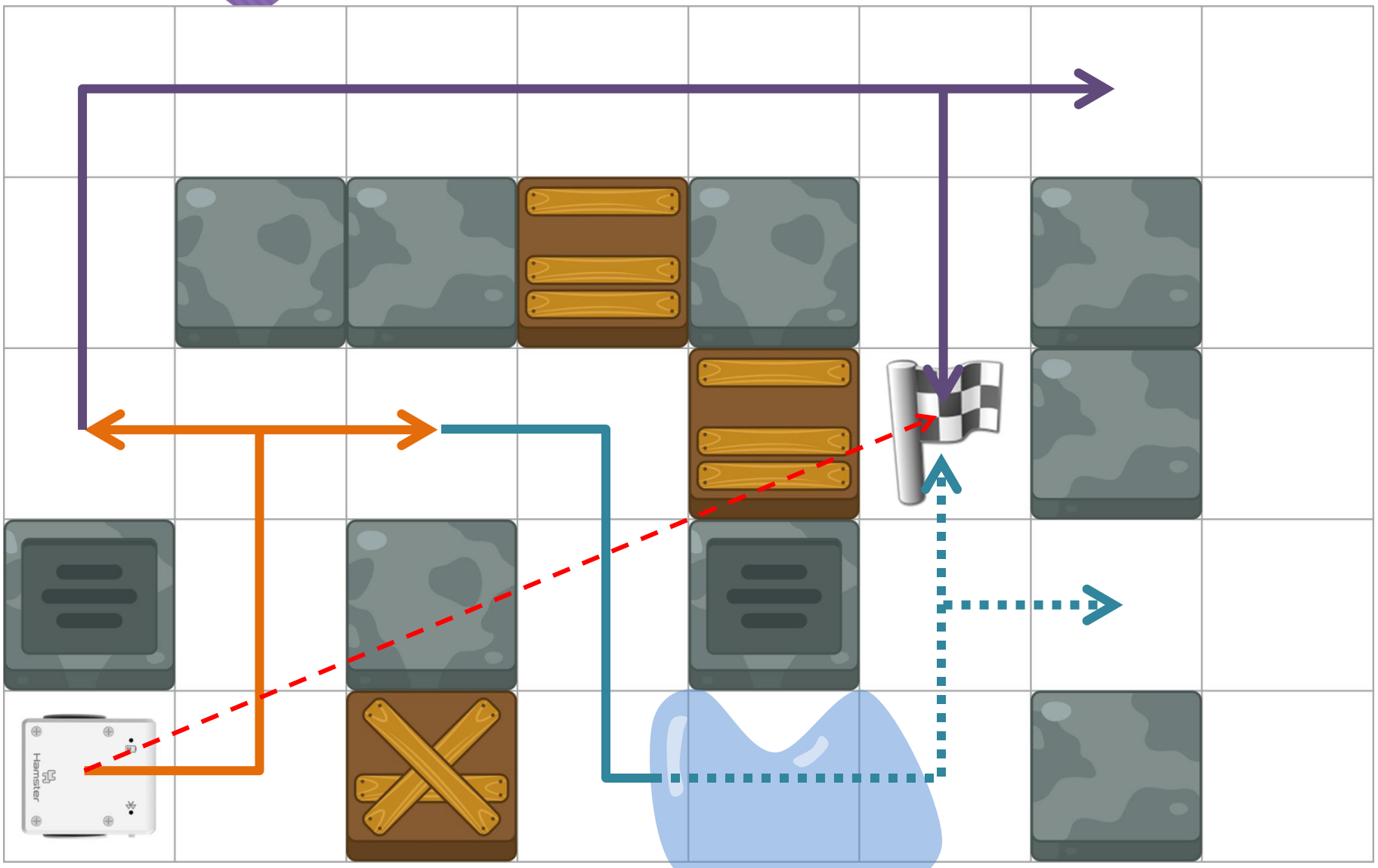
너비 우선 탐색



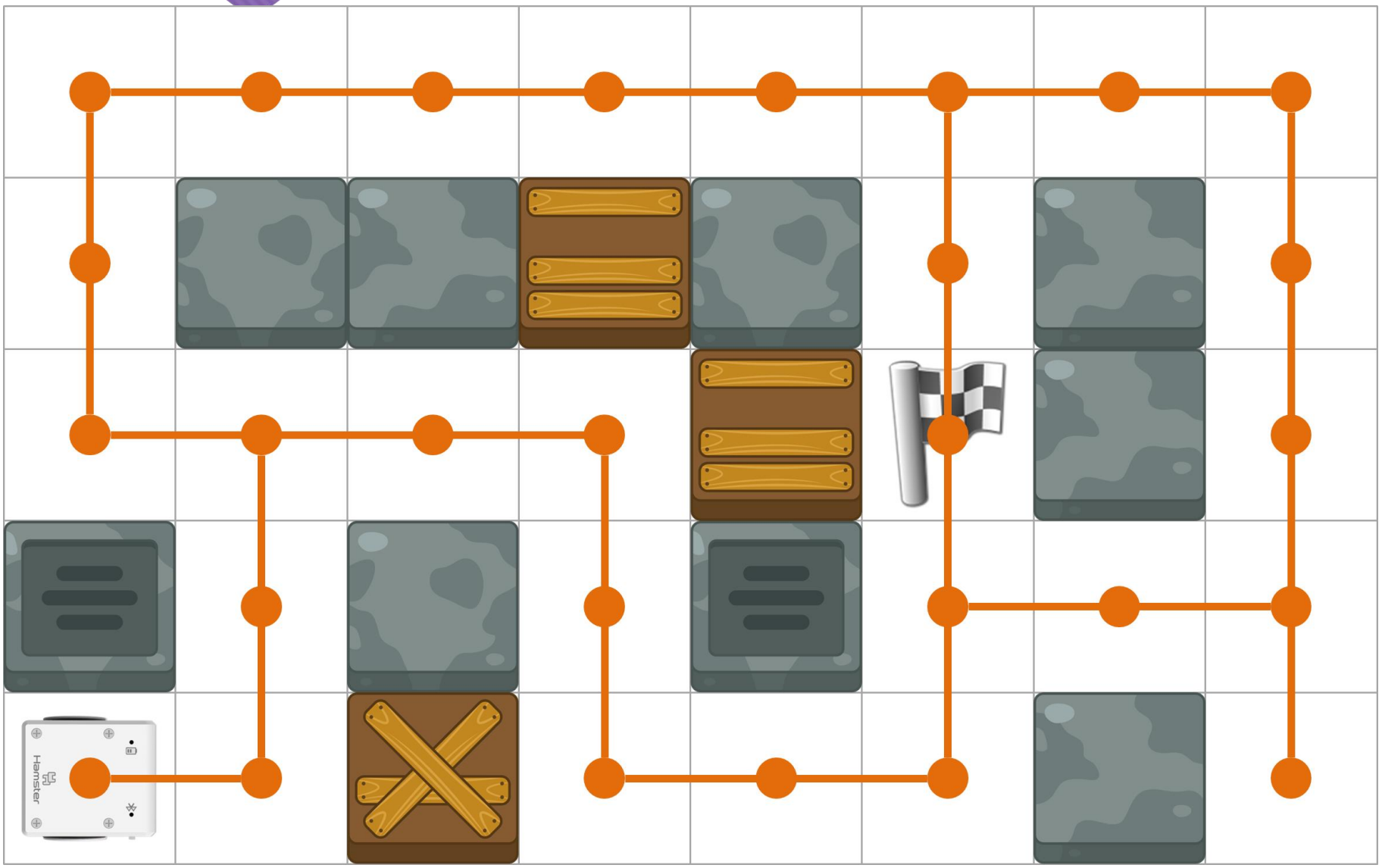
다익스트라 알고리즘



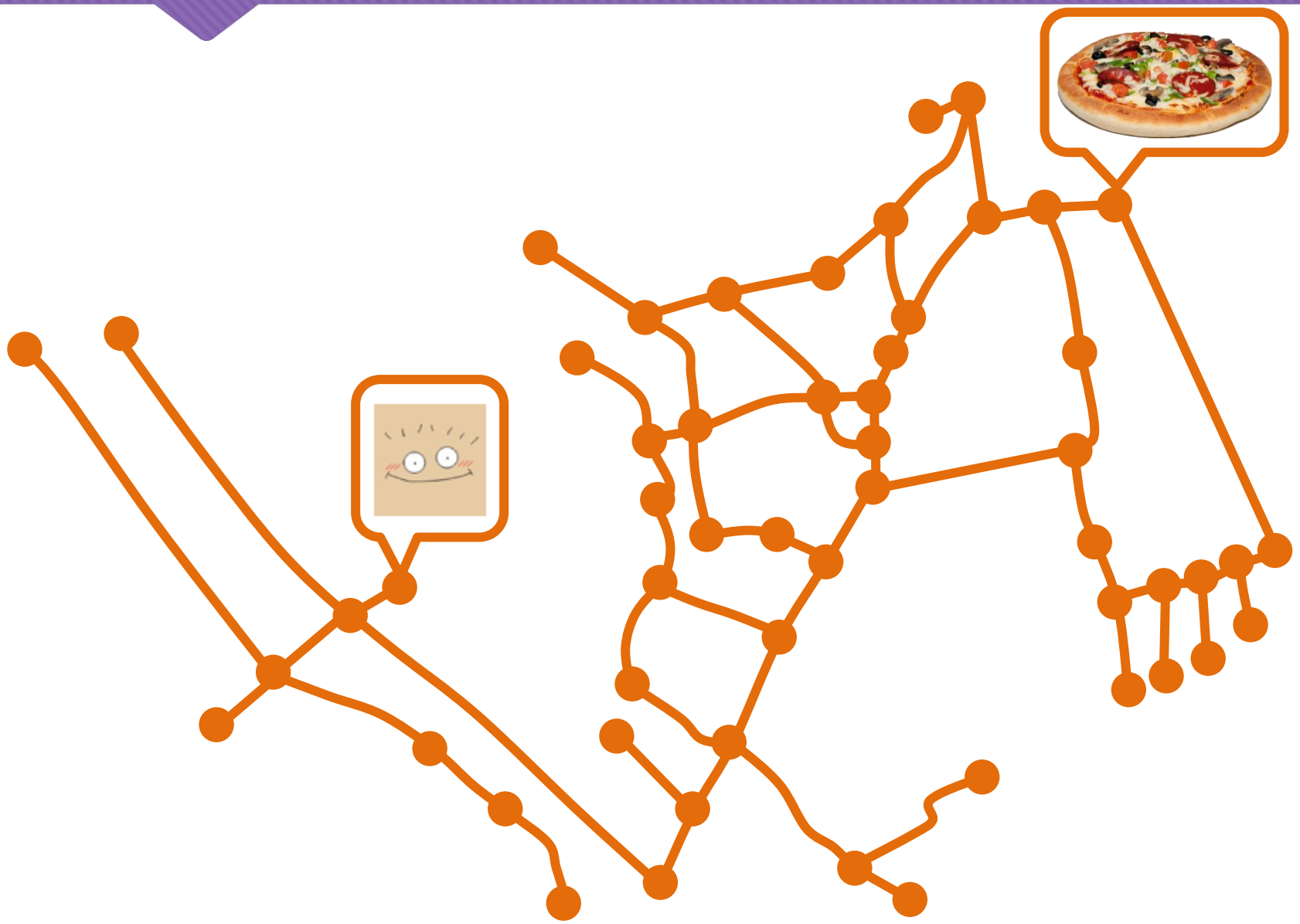
A* 알고리즘



규칙을 직접 코딩에서 기계 학습으로

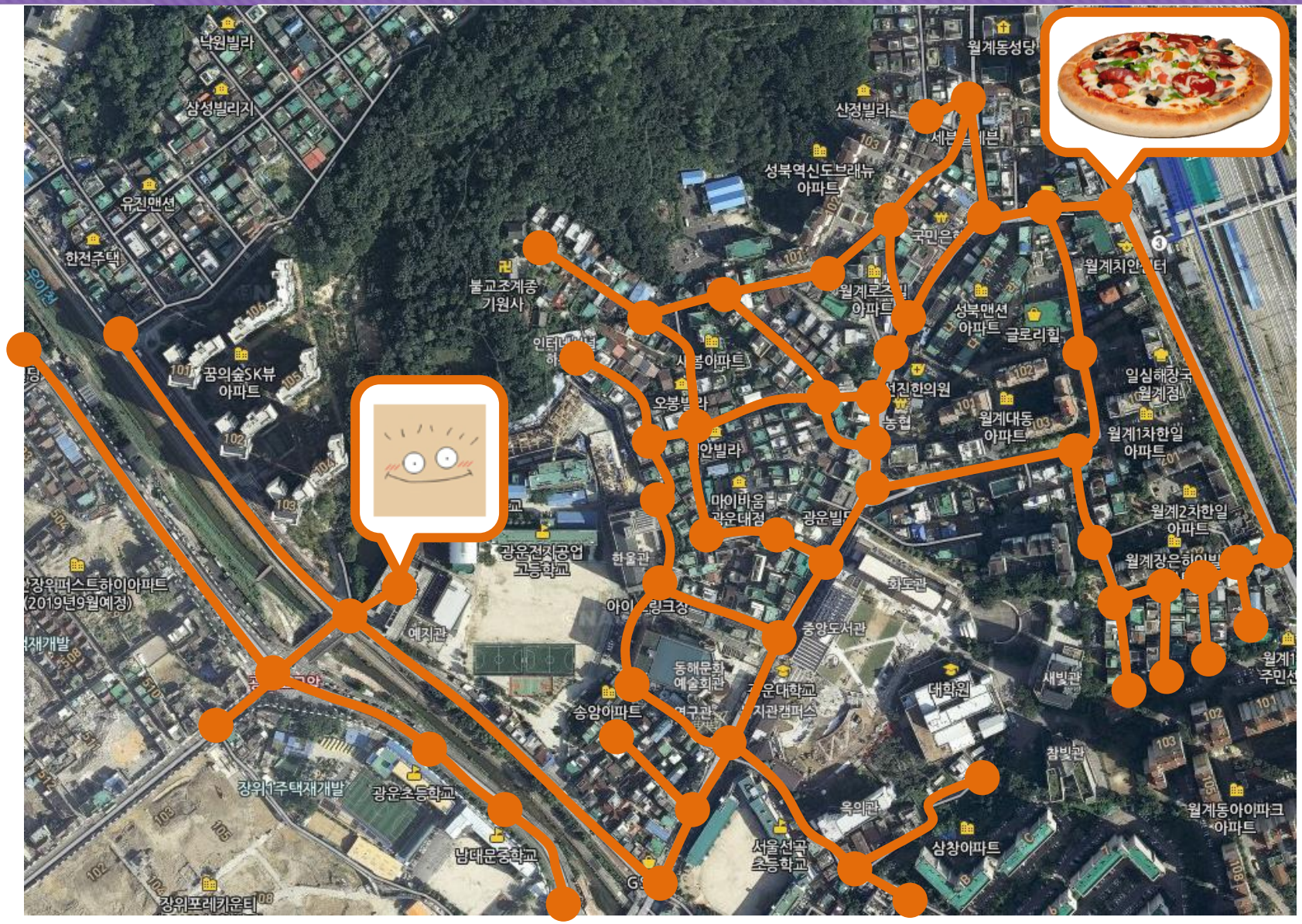


규칙을 직접 코딩에서 기계 학습으로

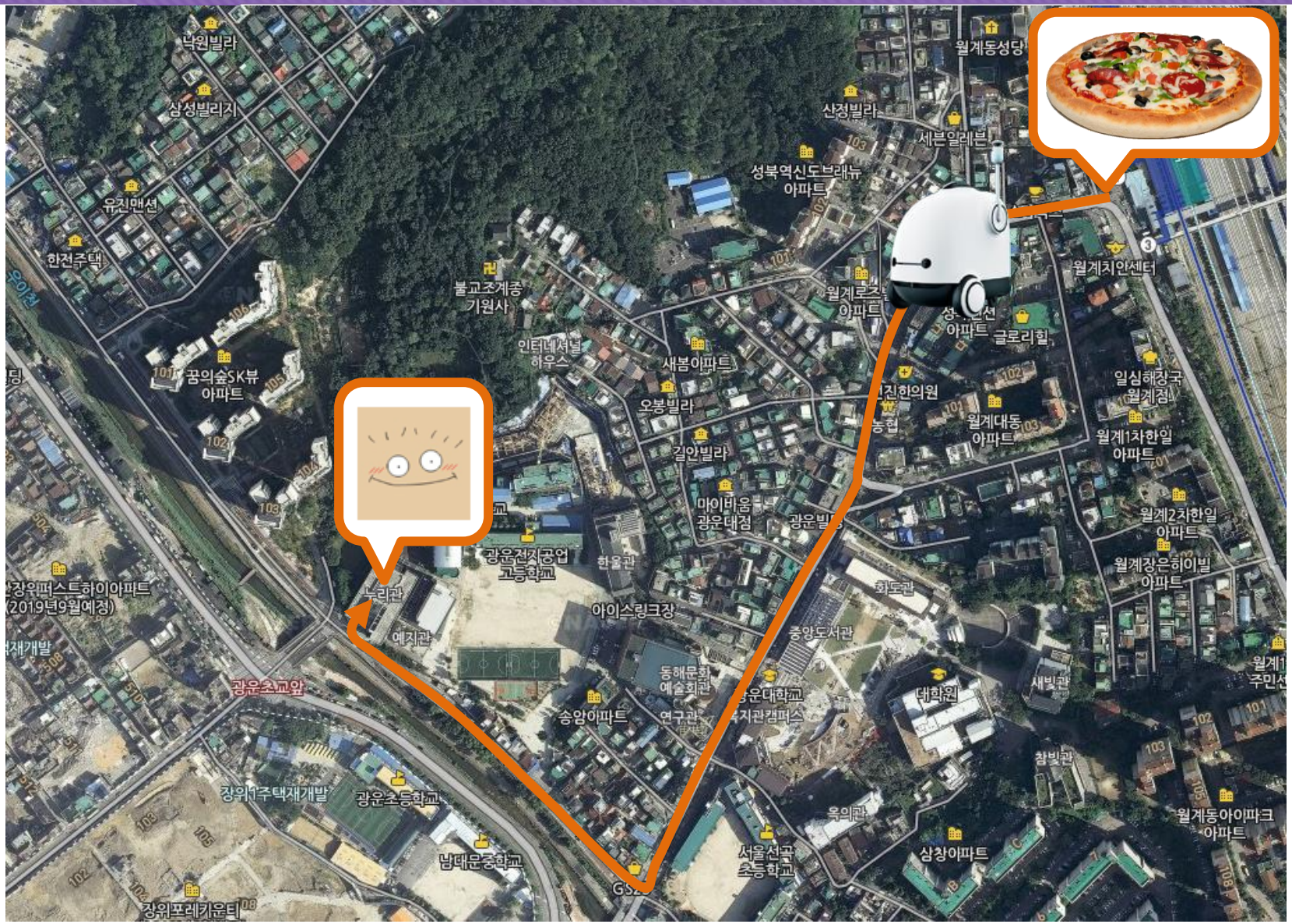


규칙을 직접 코딩에서 기계 학습으로

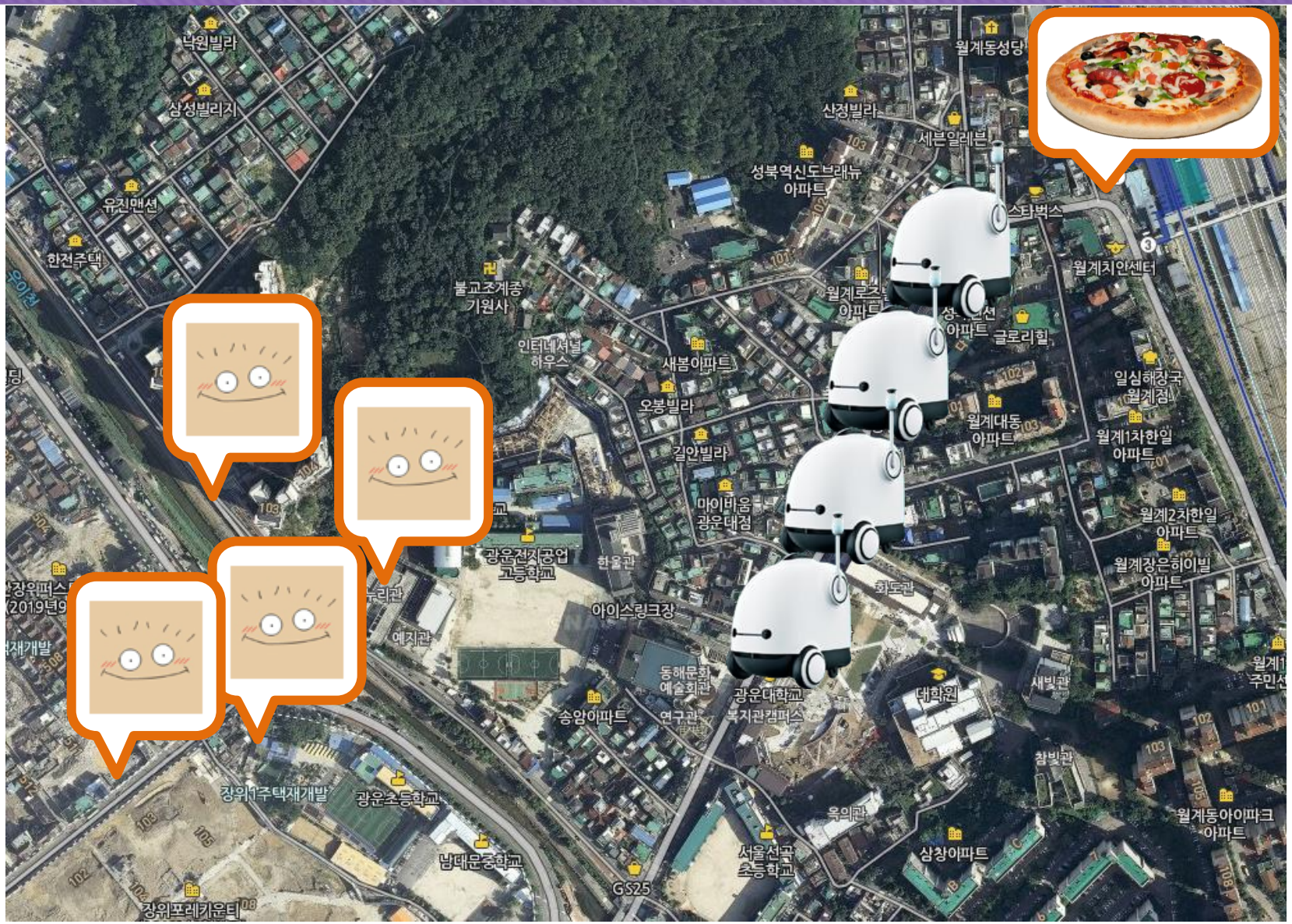
규칙을 직접 코딩에서 기계 학습으로



규칙을 직접 코딩에서 기계 학습으로

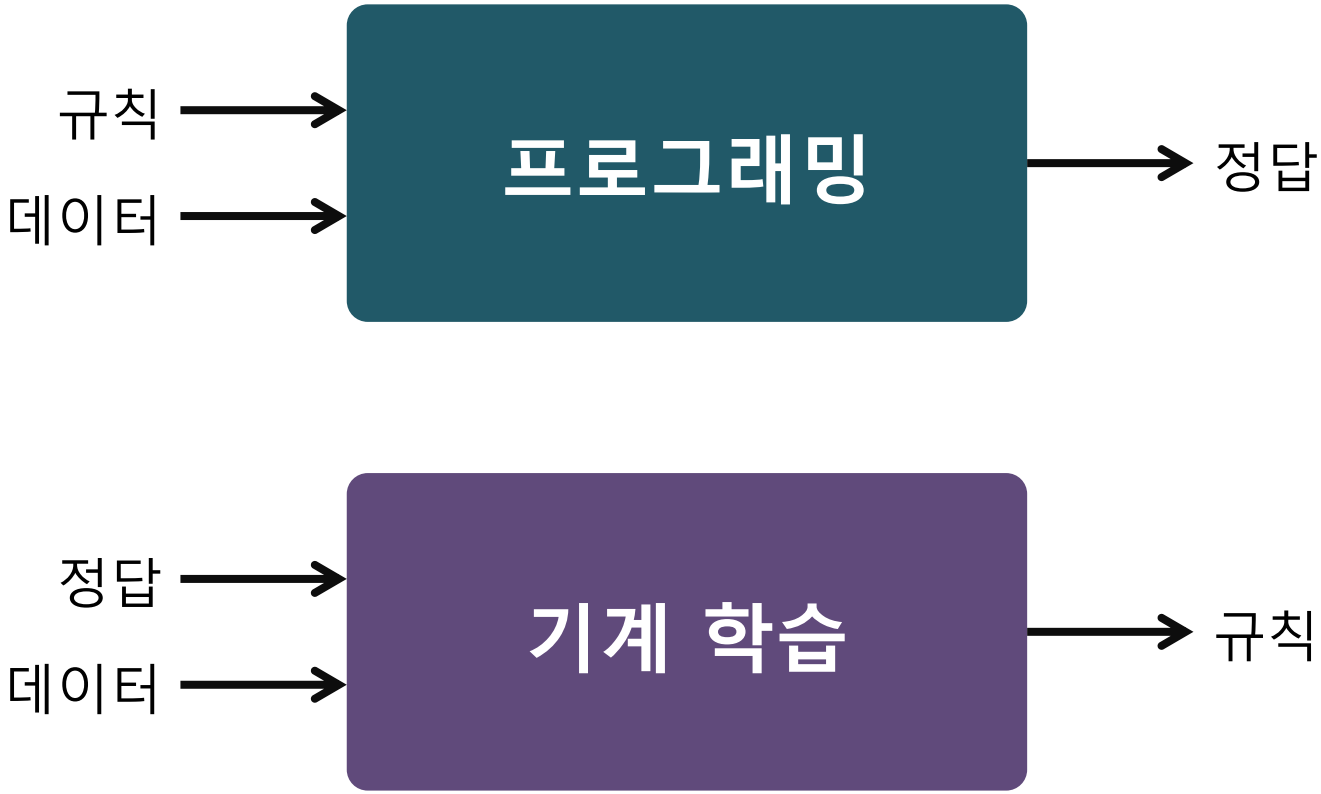


규칙을 직접 코딩에서 기계 학습으로

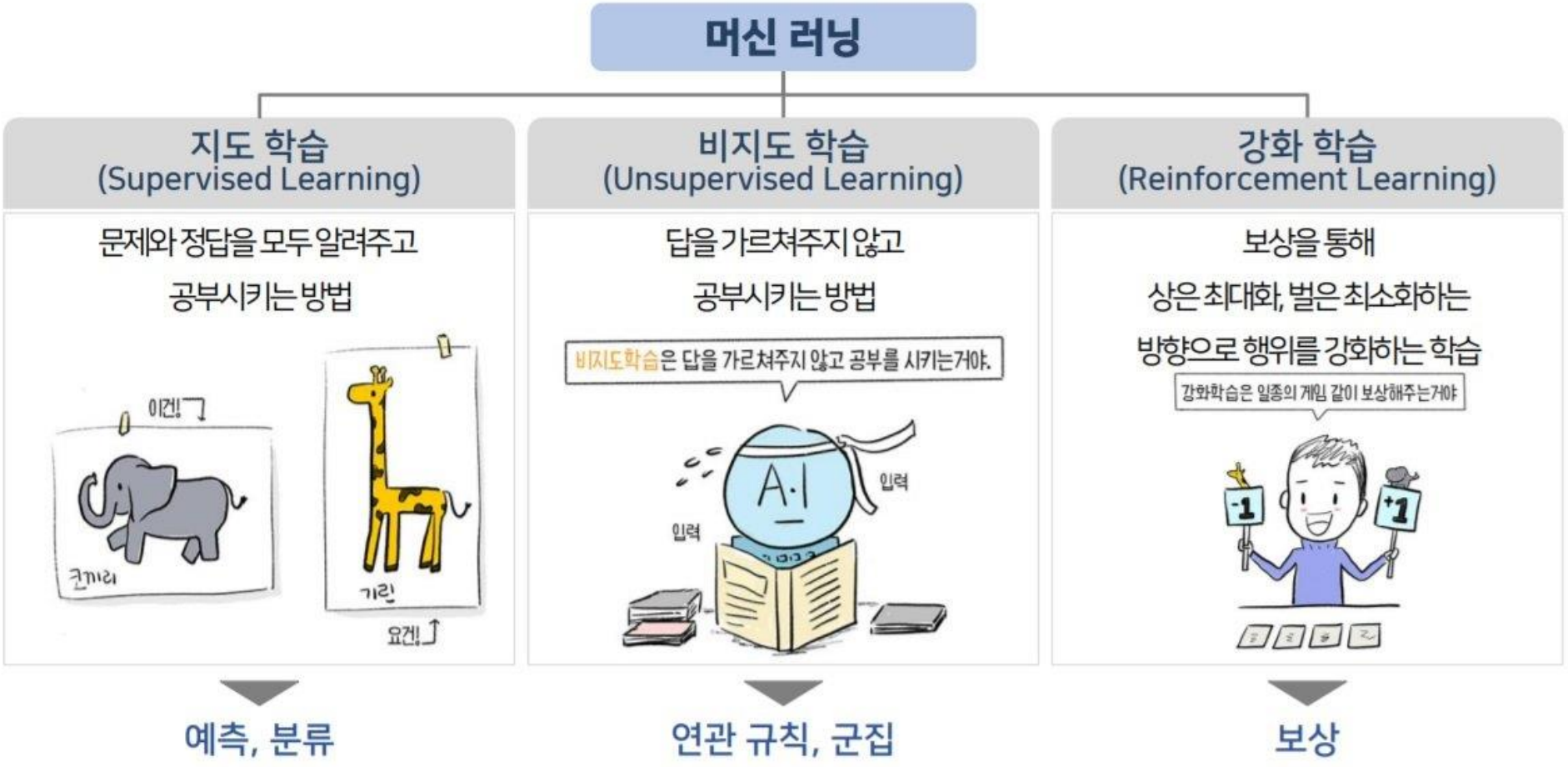


규칙을 직접 코딩에서 기계 학습으로

문제가 복잡해 질수록 규칙을 넣어 주기가 어려움



기계 학습



https://steemitimages.com/0x0/https://cdn.steemitimages.com/DQmXQfK5MVgU63WQYRcBaZFXhsLK5xxYZ73fNpz51UrZmmF/%EB%A8%B8%EC%8B%A0%EB%9F%AC%EB%8B%9D.jpg

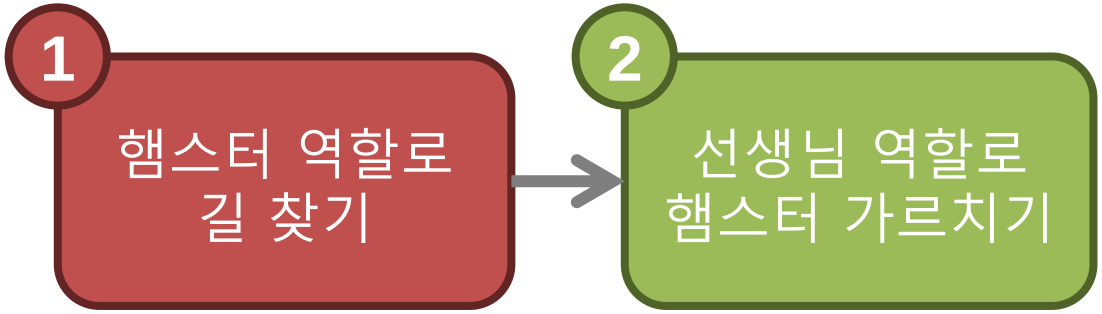
초등학교 체험 수준

활동 12

강화 학습 역할 놀이

학습 목표

- 로봇이 시행착오를 통해 목표 지점으로 이동하는 학습 과정을 이해할 수 있다.
- 학습을 수행하는 입장(햄스터)과 가르치는 입장의 두 가지 역할을 수행할 수 있다.



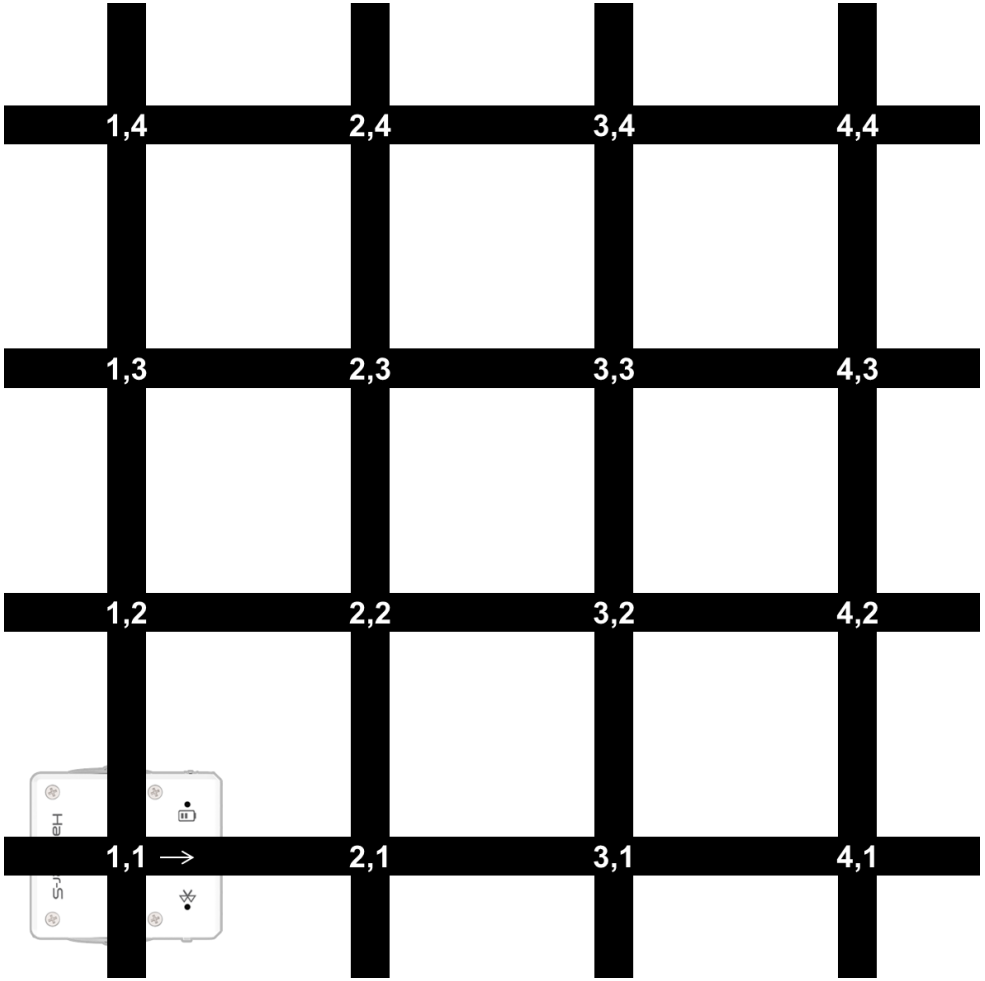
1. 햄스터의 입장

시행 착오를 통해

함정을 피해 목표 지점으로 갑니다.

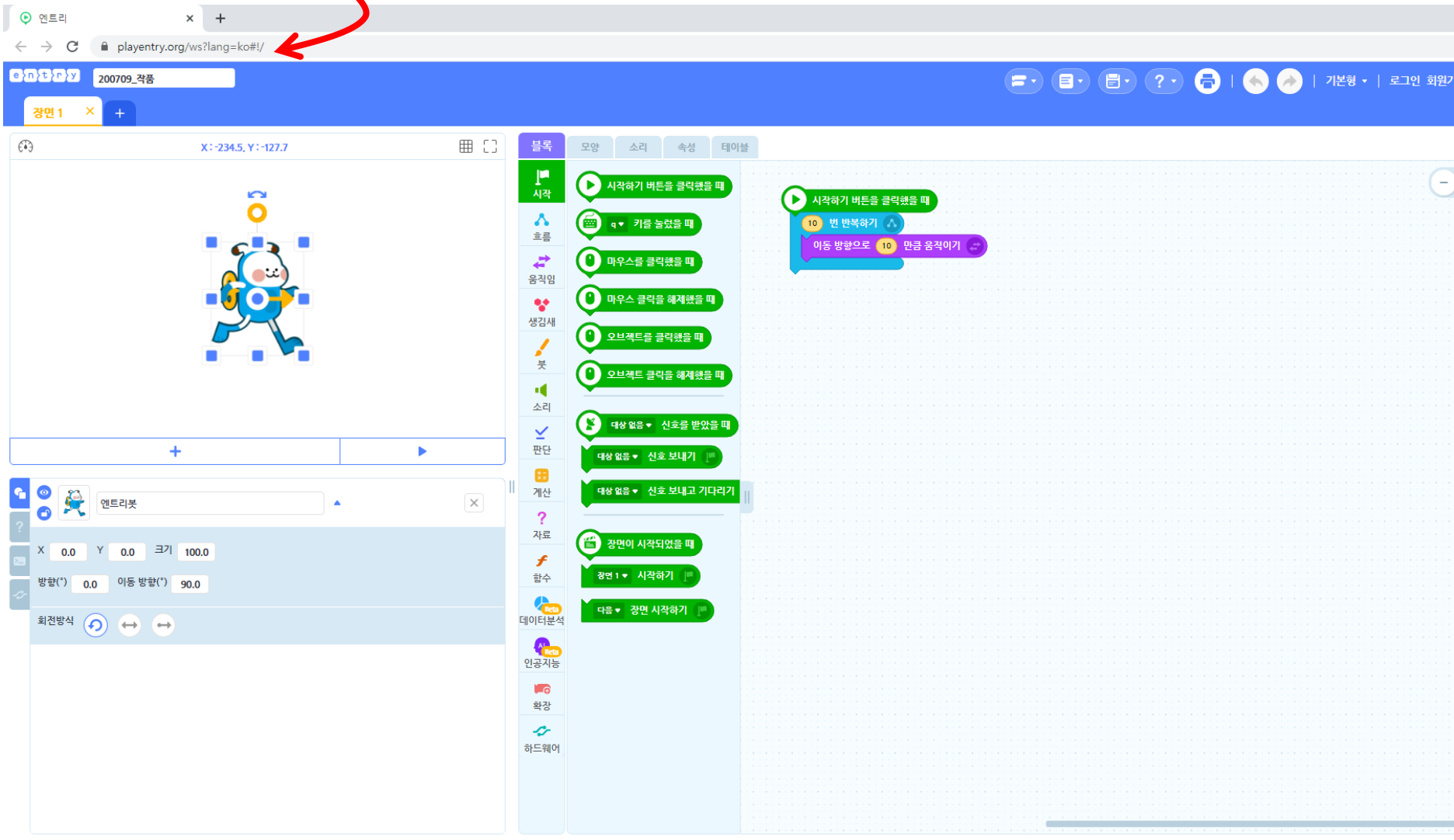
준비물

출발 위치에 로봇의 **방향을 맞추어** 올려 놓습니다.



주소 입력

http://naver.me/IGAe3rR9



게임 규칙

- 함정이 몇 개인지, 함정이 어디에 있는지, 목표 위치가 어디인지 알 수 없습니다.

- 방향키를 눌러 로봇을 조종합니다.
  - 왼쪽 방향키 : 왼쪽으로 90도 회전
  - 오른쪽 방향키: 오른쪽으로 90도 회전
  - 위쪽 방향키: 앞으로 한 칸 이동

- 함정에 빠지면 “삐삐” → 제자리로 돌아갑니다.
- 목표 위치에 도착하면 “도미솔” → 제자리로 돌아갑니다.
- 중간에 로봇의 위치가 이상하면 로봇을 손으로 들어 출발 위치에 놓고 “r”키를 눌러 위치 정보를 리셋합니다.
- 함정에 빠지지 않고 목표 위치에 도착하는 것을 연속 3번하면 됩니다.

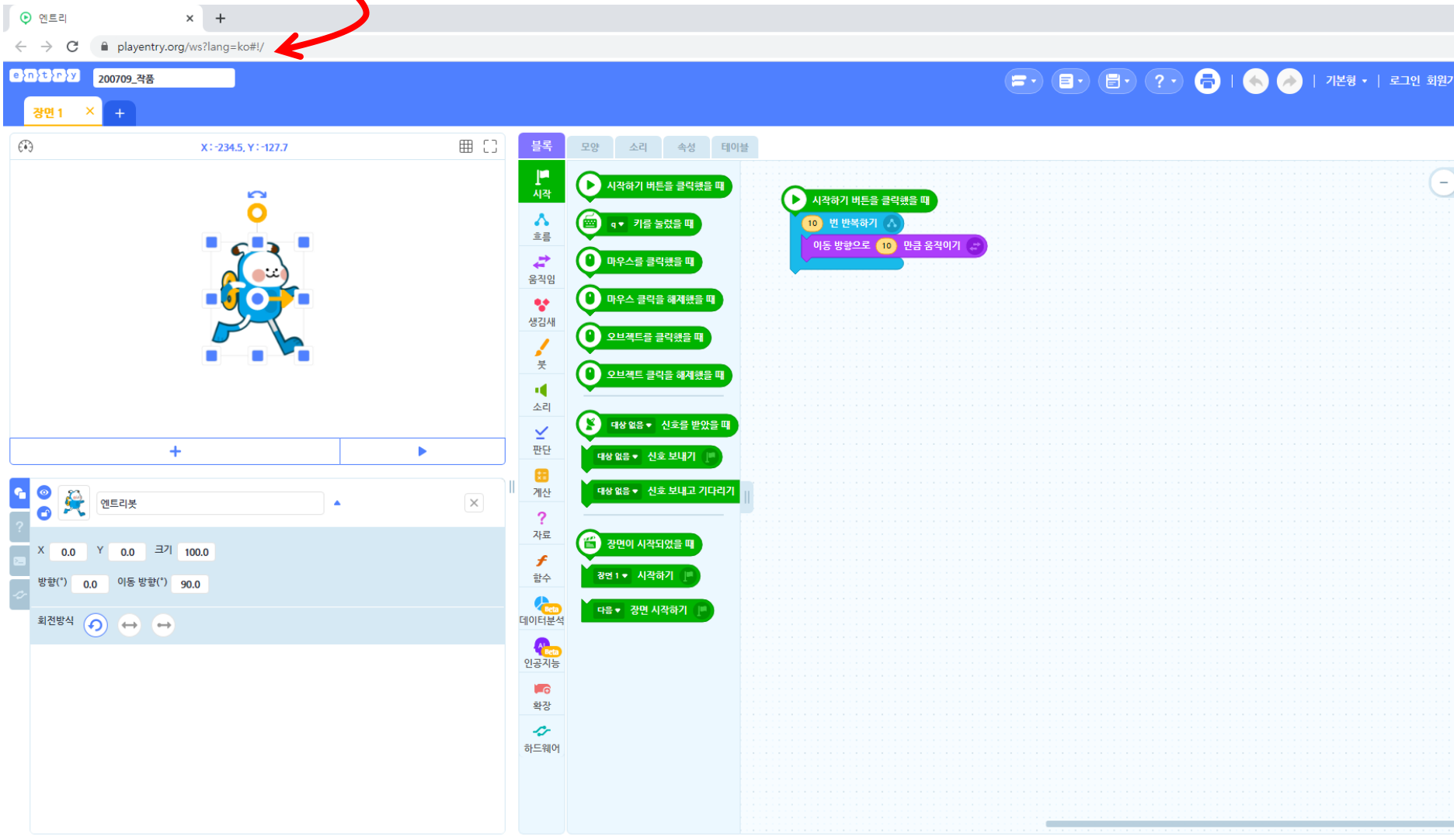
문제 5개

2. 선생님의 입장

채찍과 당근으로
햄스터를 훈련시킵니다.

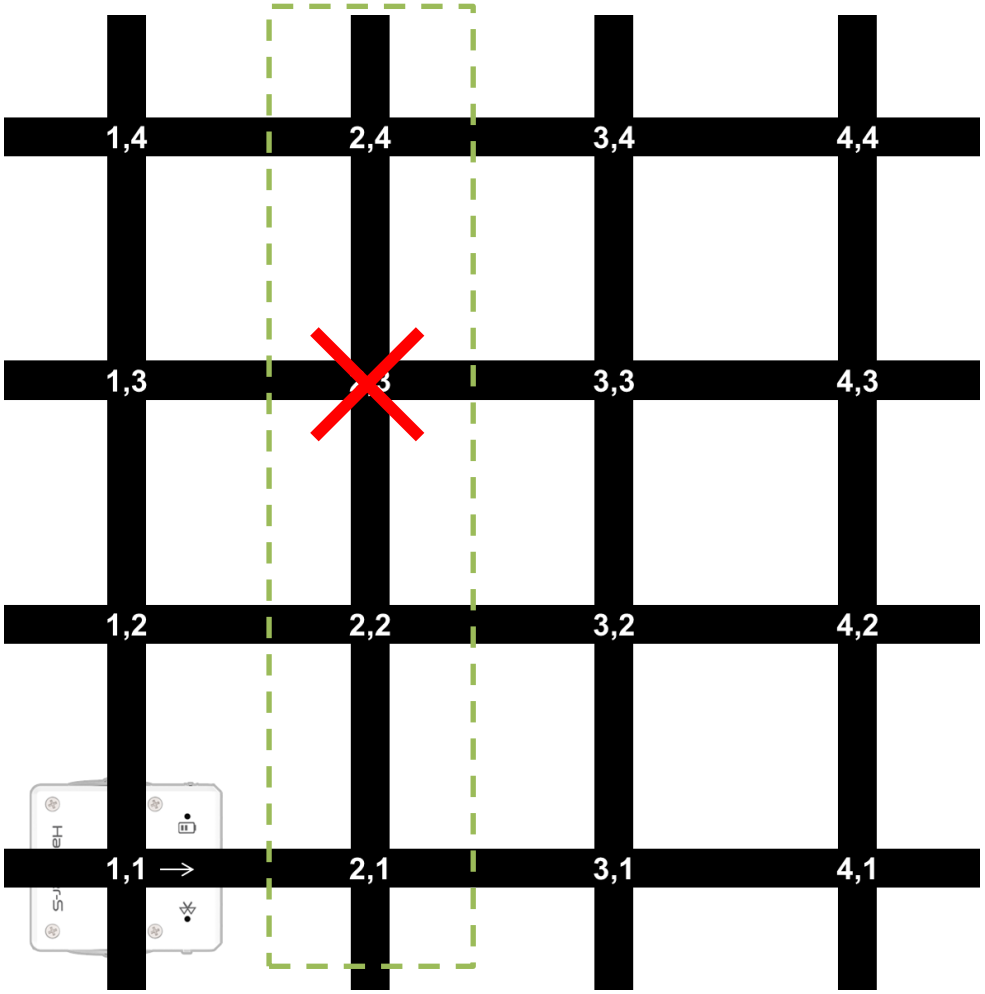
주소 입력

http://naver.me/5EfiMCTz



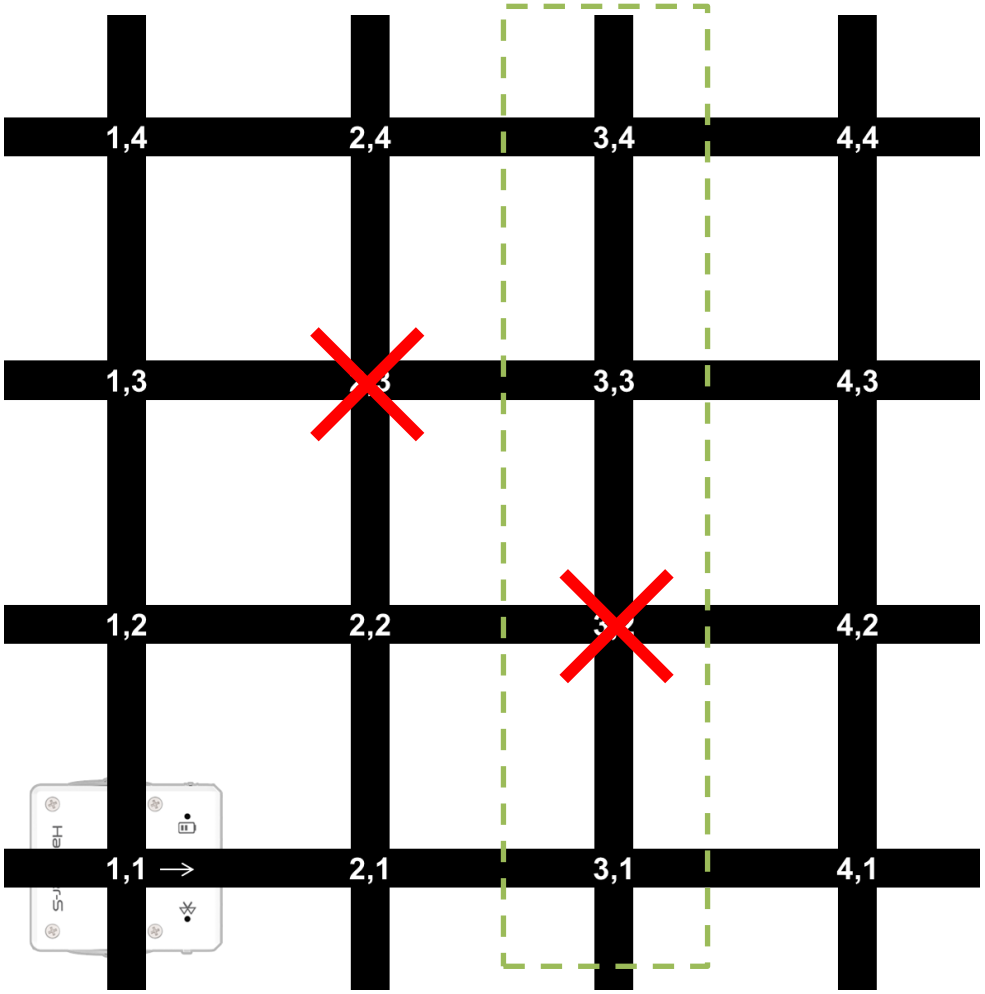
게임 규칙

두 번째 열의 4개 칸 중 하나에 함정을 표시합니다.



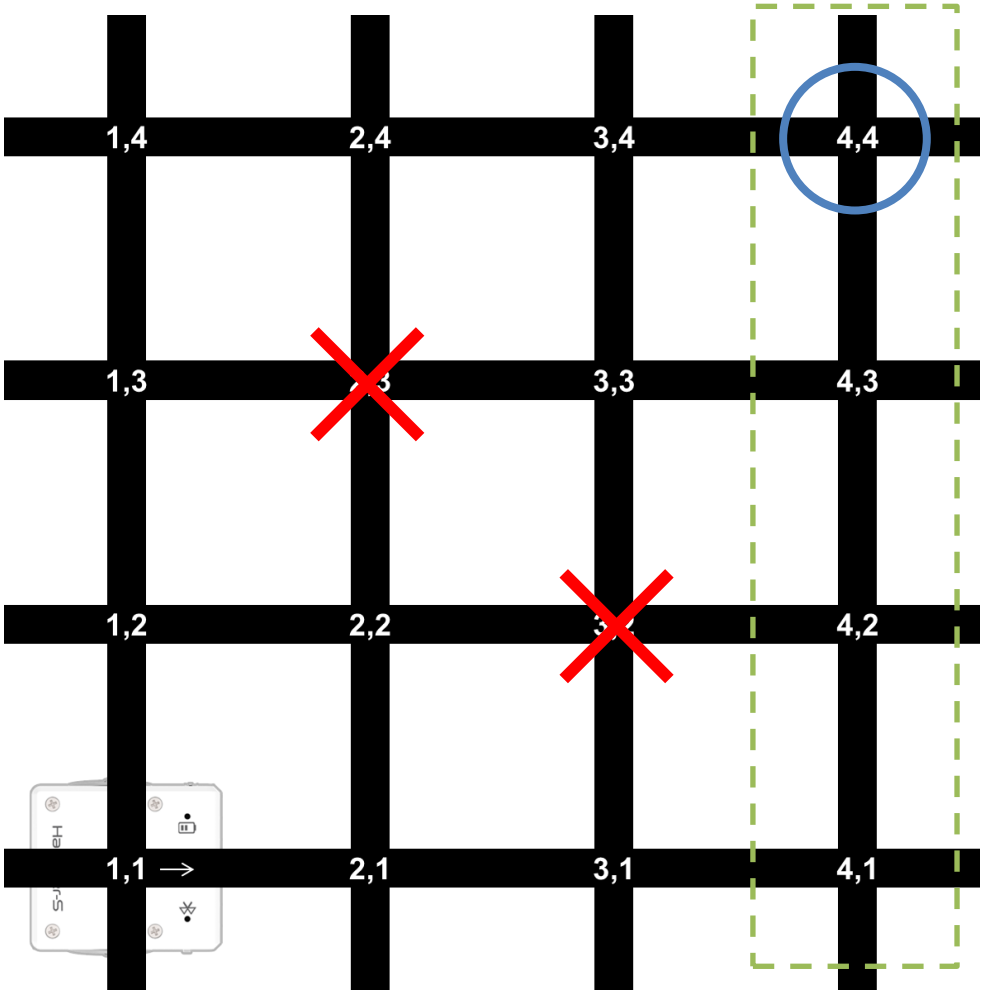
게임 규칙

세 번째 열의 4개 칸 중 하나에 함정을 표시합니다.



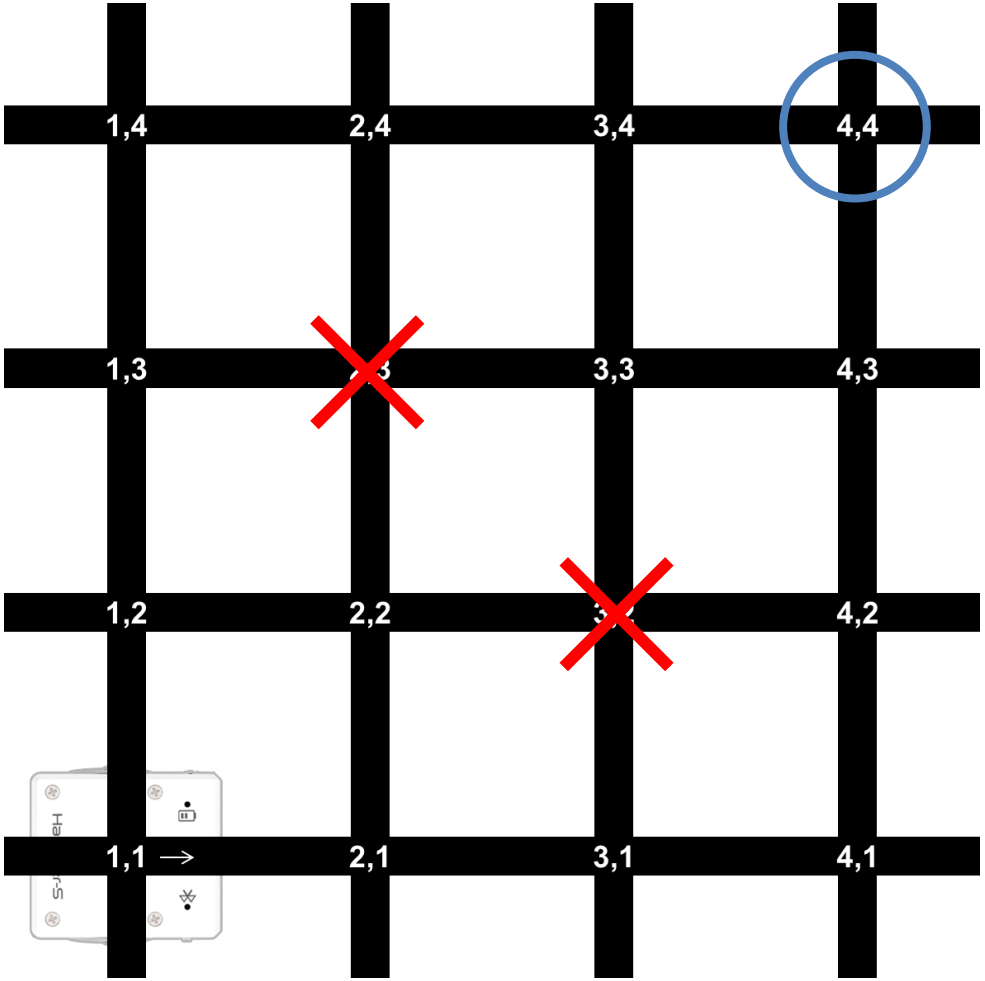
게임 규칙

네 번째 열의 4개 칸 중 하나에 목표 지점을 표시합니다.



게임 규칙

출발 위치에 로봇의 **방향을 맞추어** 올려 놓습니다.



게임 규칙

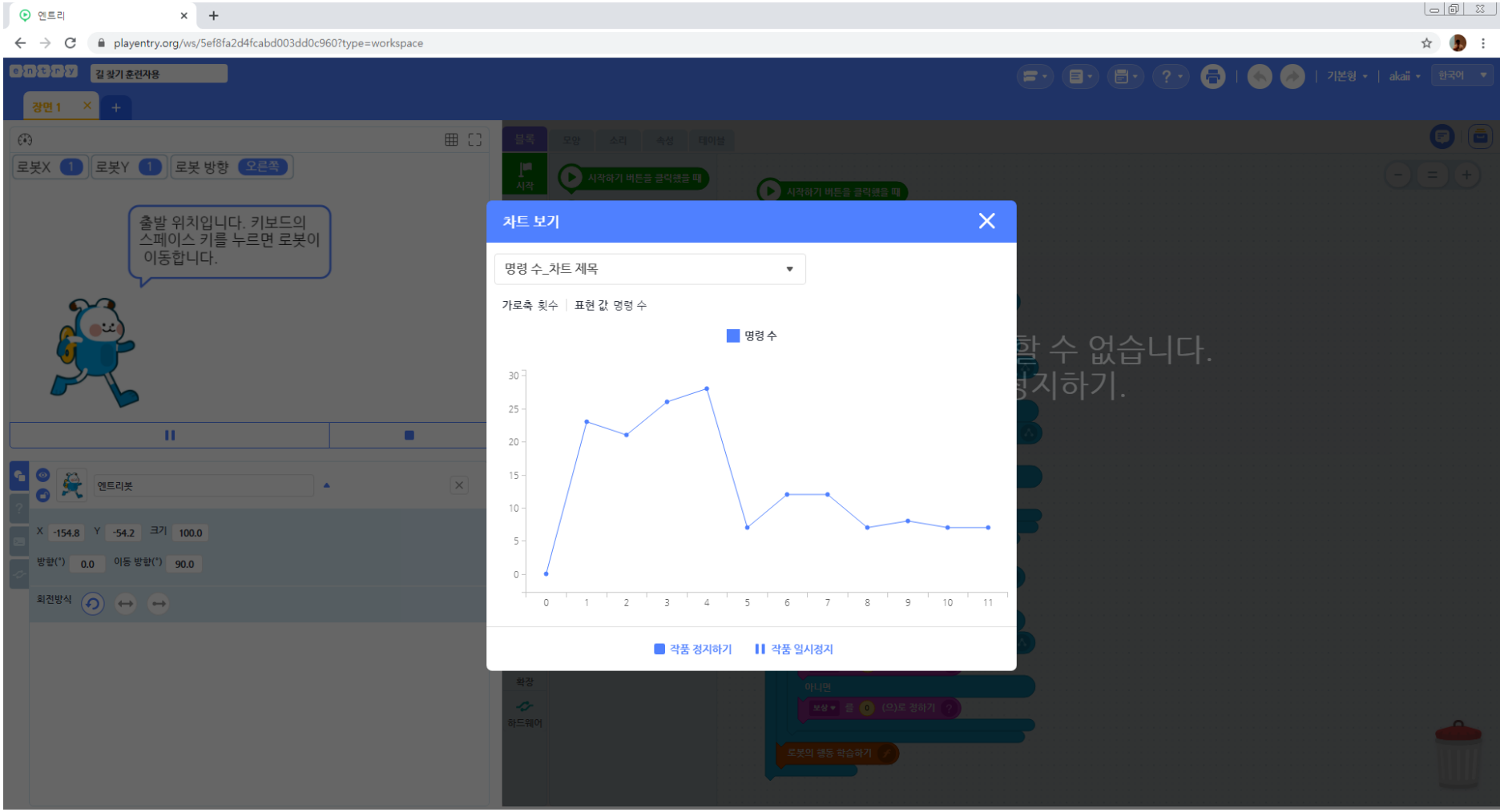
- 스페이스 키를 누를 때마다 로봇이 한 번씩 움직입니다.

- 함정에 빠지면 키보드의 x 키를 누릅니다.
- 목표 위치에 도착하면 키보드의 o 키를 누릅니다.

- 로봇이 출발 위치로 이동하면 다시 반복합니다.
- 로봇이 목표 지점으로 더 잘 주행함을 차트를 통해 관찰합니다.
  차트에는 목표 위치에 도착할 때까지 내린 명령 횟수가 표시됩니다.
  명령 횟수가 점점 줄어듦을 관찰합니다.

- 중간에 로봇의 위치가 이상하면 로봇을 손으로 들어 출발 위치에 놓고 “r”키를 눌러 위치 정보를 리셋합니다.

성공할 때까지의 누적 명령 수 변화



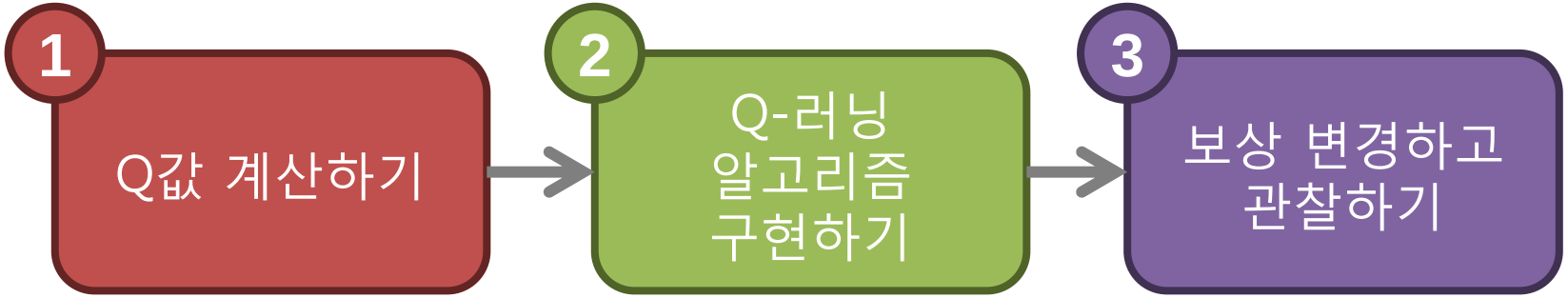
초등학교 동아리 ~ 중학교

활동 13

강화 학습을 활용한 자율 주행 자동차

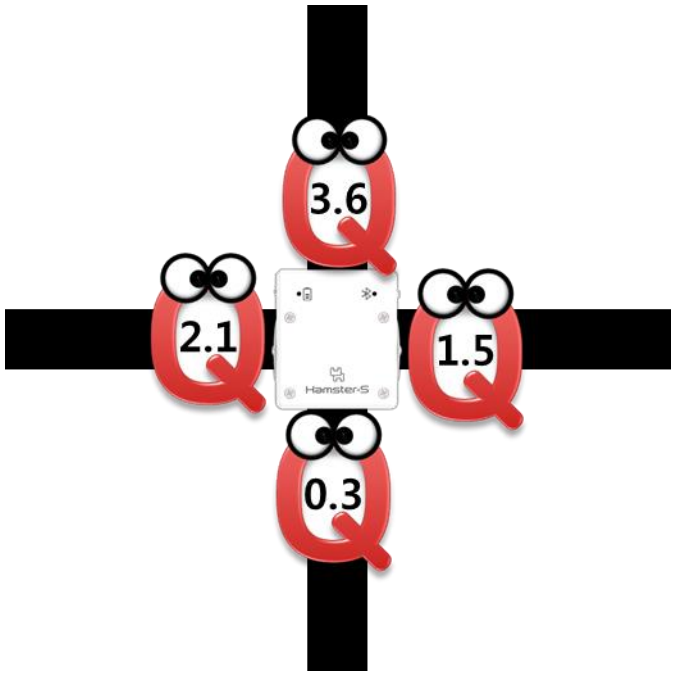
학습 목표

- 경험과 보상을 통해 로봇이 목표 지점으로 이동하는 학습 과정을 이해할 수 있다.
- 강화 학습 기법 중 하나인 Q-러닝 알고리즘을 코드로 구현할 수 있다.



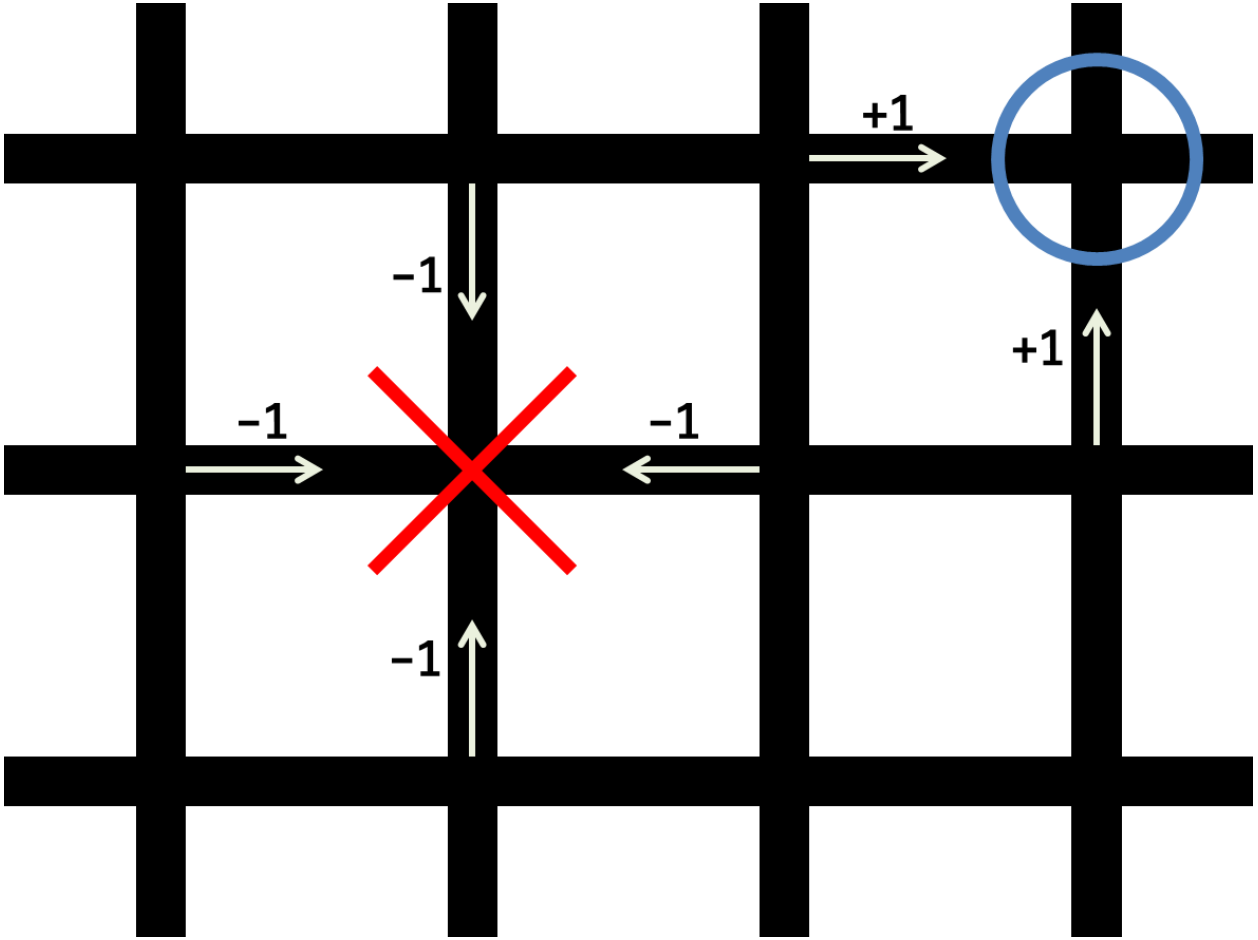
Q친구들

어느 방향으로 가면 좋을지를 네 명의 Q친구들에게 물어 봅니다.
가장 큰 숫자를 얘기하는 방향으로 로봇이 이동하면 됩니다.



보상 (점수)

Q 친구를 지나갈 때 각 이동(행동)에 대한 보상(점수)

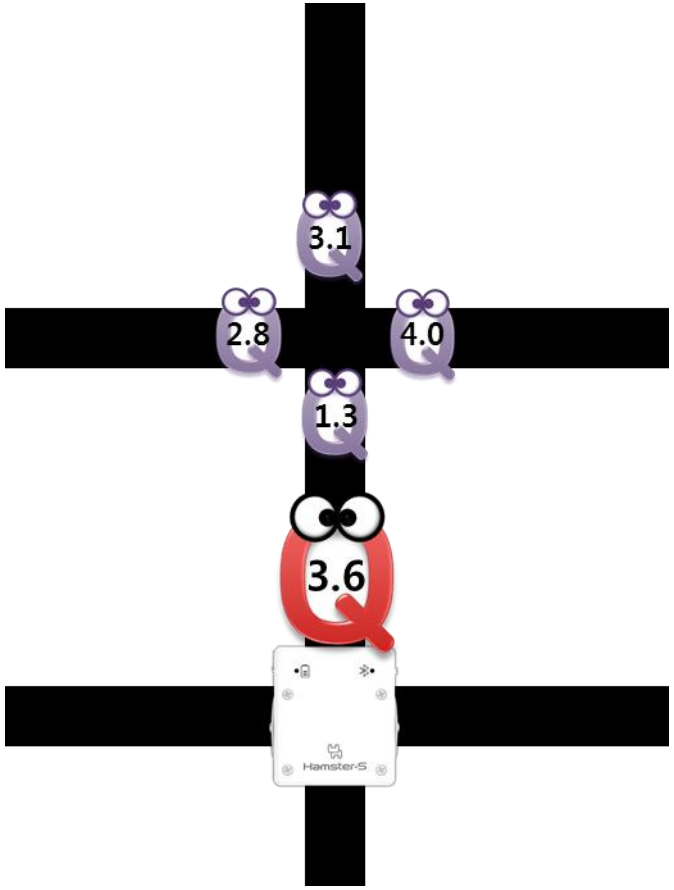


Q 친구들이 말하는 숫자

다음 교차로의 Q친구들에게 물어 보고 가장 큰 숫자

x 0.9

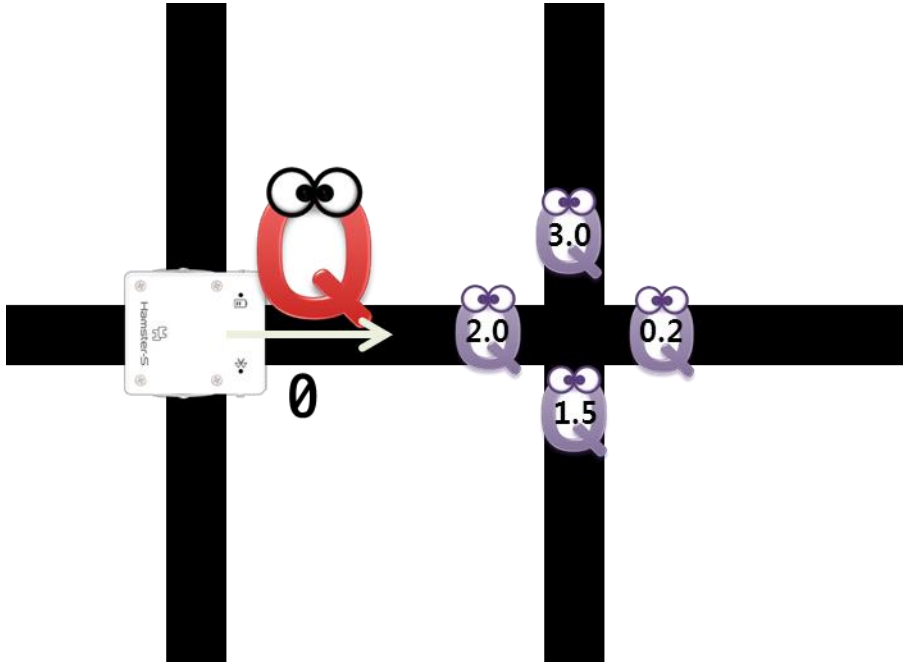
+ 자기 점수



Q값 계산해 보기

[현재 교차로에서 Qa 값] ← [점수 r] + 0.9 x [다음 교차로에서 Qa 값들 중 최댓값]

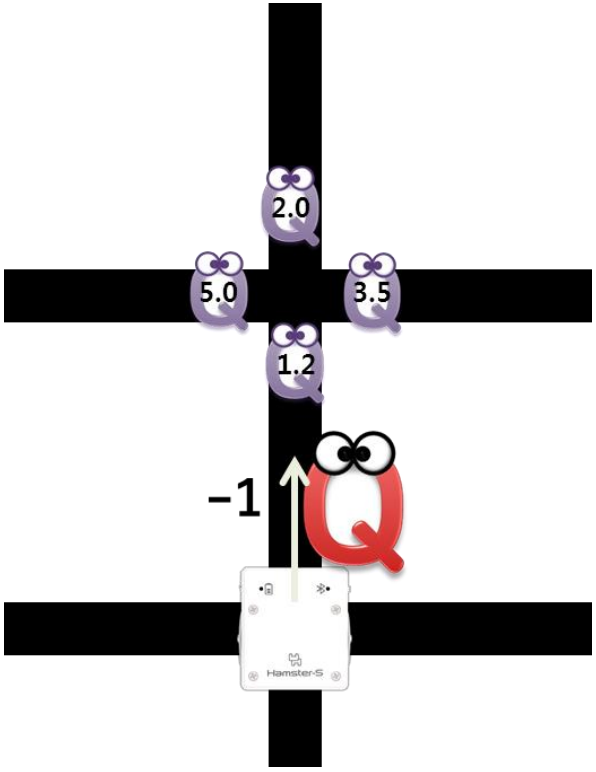
점수 r	다음 교차로의 보라색 Q값					현재 교차로의 오른쪽 방향 빨간색 Q값
0	왼쪽	오른쪽	위쪽	아래쪽	최댓값	



Q값 계산해 보기

[현재 교차로에서 Qa 값] ← [점수 r] + 0.9 x [다음 교차로에서 Qa 값들 중 최댓값]

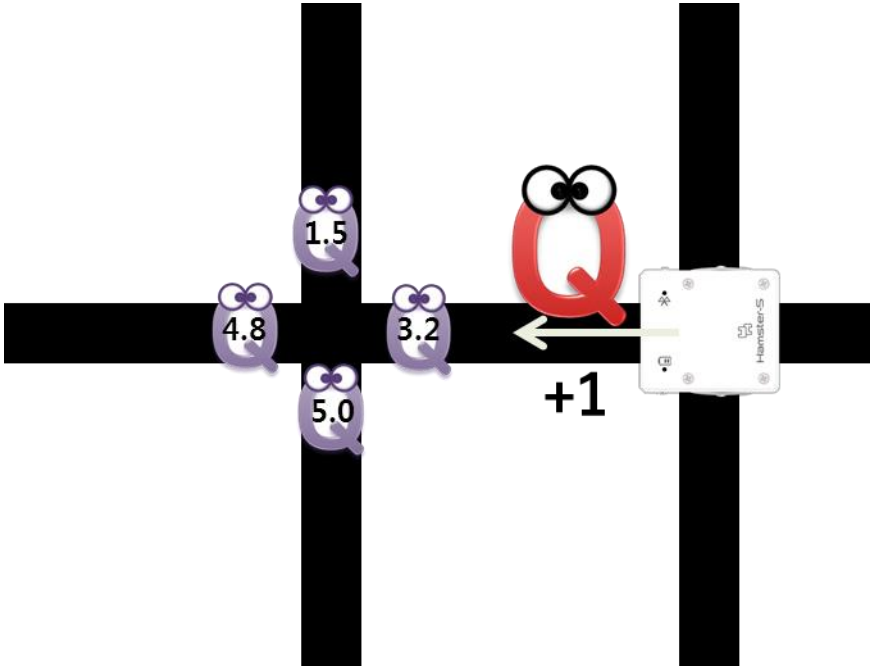
점수 r	다음 교차로의 보라색 Q값					현재 교차로의 오른쪽 방향 빨간색 Q값
	왼쪽	오른쪽	위쪽	아래쪽	최댓값	
-1						



Q값 계산해 보기

[현재 교차로에서 Qa 값] ← [점수 r] + 0.9 x [다음 교차로에서 Qa 값들 중 최댓값]

점수 r	다음 교차로의 보라색 Q값					현재 교차로의 오른쪽 방향 빨간색 Q값
1	왼쪽	오른쪽	위쪽	아래쪽	최댓값	

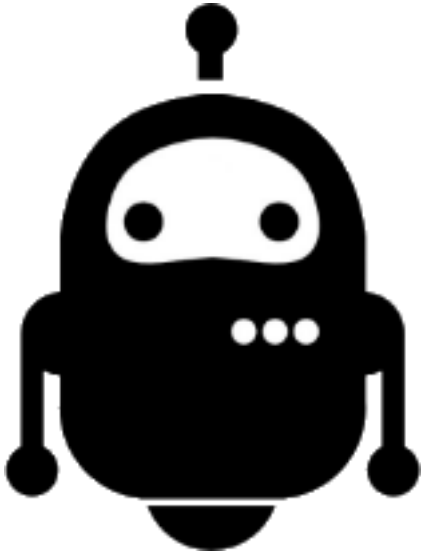


선생님을 위한 보충 설명

강화 학습

강화 학습

에이전트 (agent)

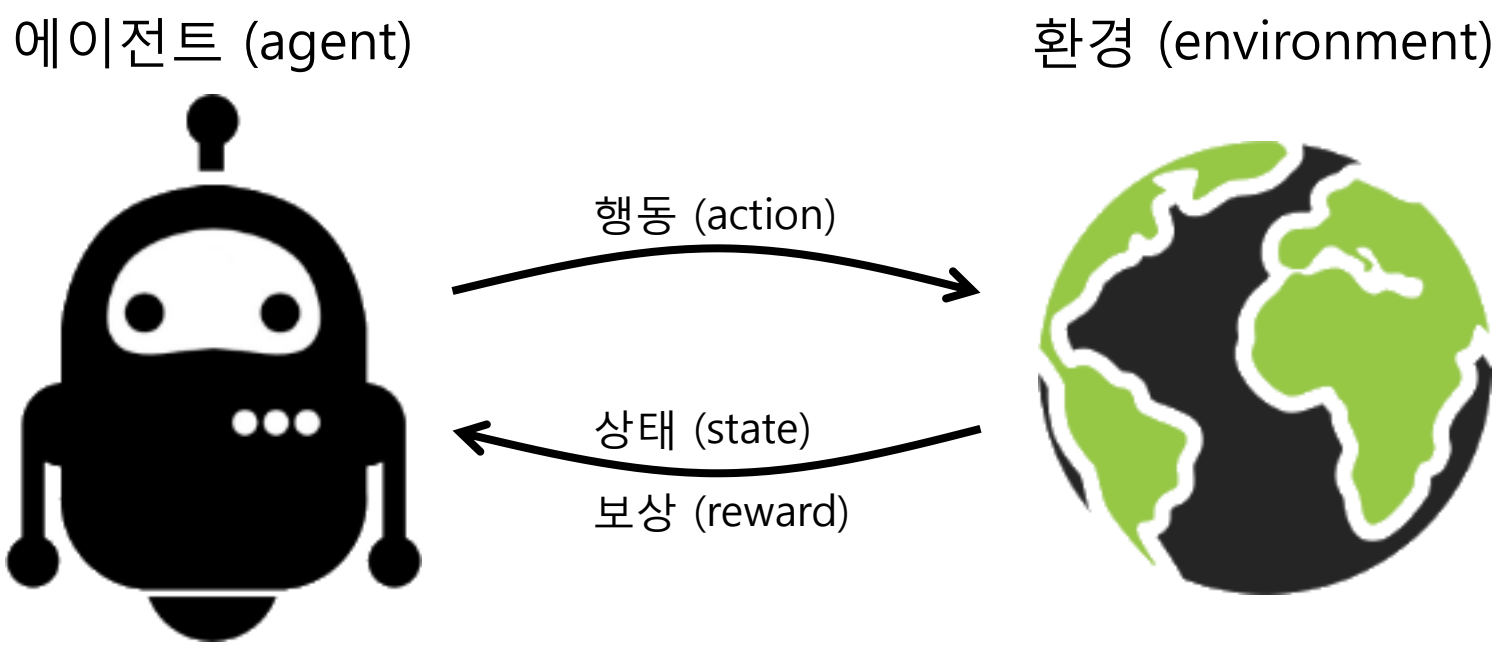


환경 (environment)

환경을 인지하고 그에 대응하는 행동을 하는 컴퓨터 프로그램

- 인터넷 봇 (웹 크롤러)
- 게임 봇 (인공지능 플레이어)
- 챗봇

강화 학습



환경을 인지하고 그에 대응하는 행동을 하는
컴퓨터 프로그램

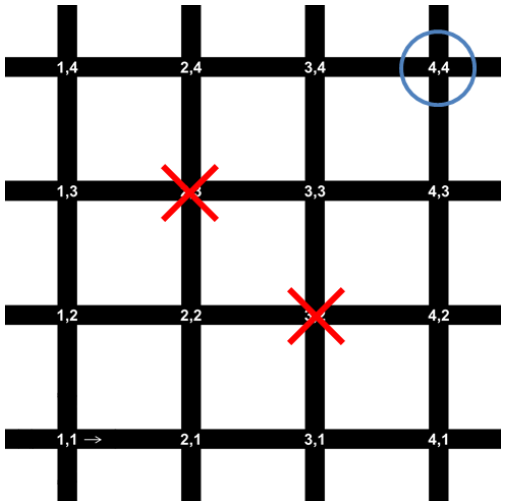
- 인터넷 봇 (웹 크롤러)
- 게임 봇 (인공지능 플레이어)
- 챗봇

강화 학습

왼쪽으로 이동, 오른쪽으로 이동
위로 이동, 아래로 이동

에이전트 (agent)

환경 (environment)



행동 (action)

상태 (state)

보상 (reward)

로봇의 위치(x, y)

o키 클릭
x키 클릭
내버려 두기

강화 학습

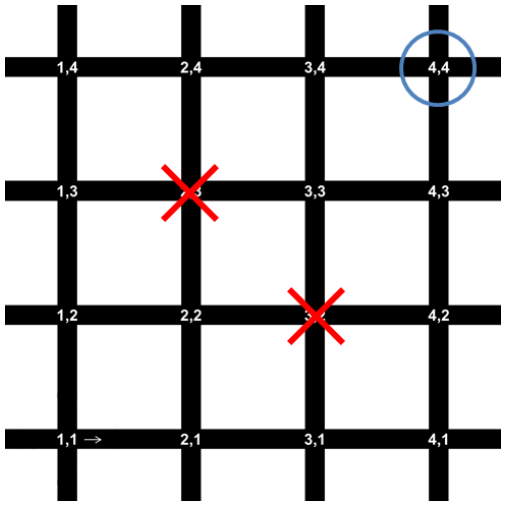
에이전트 (agent)

행동 (action)

상태 (state)

보상 (reward)

환경 (environment)



목적 : 현재부터 미래까지의 모든 보상의 합을 최대화

학습의 결과 : 각 상태에서 취해야 할 행동 → 정책 (policy)

로봇의 위치

왼쪽으로, 오른쪽으로
위로, 아래로

강화 학습

현재: t

다음: $t+1$

에이전트 (agent)

a_t a_{t+1}

행동 (action)

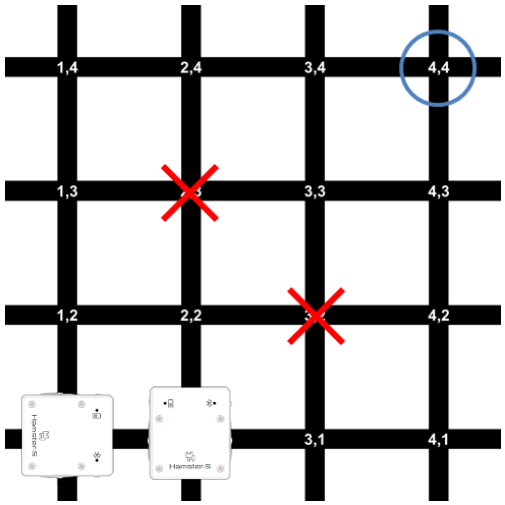
s_t s_{t+1}

상태 (state)

보상 (reward)

r_t r_{t+1}

환경 (environment)



$r_t + r_{t+1} + r_{t+2} + \cdots$

목적 : 현재부터 미래까지의 모든 보상의 합을 최대화

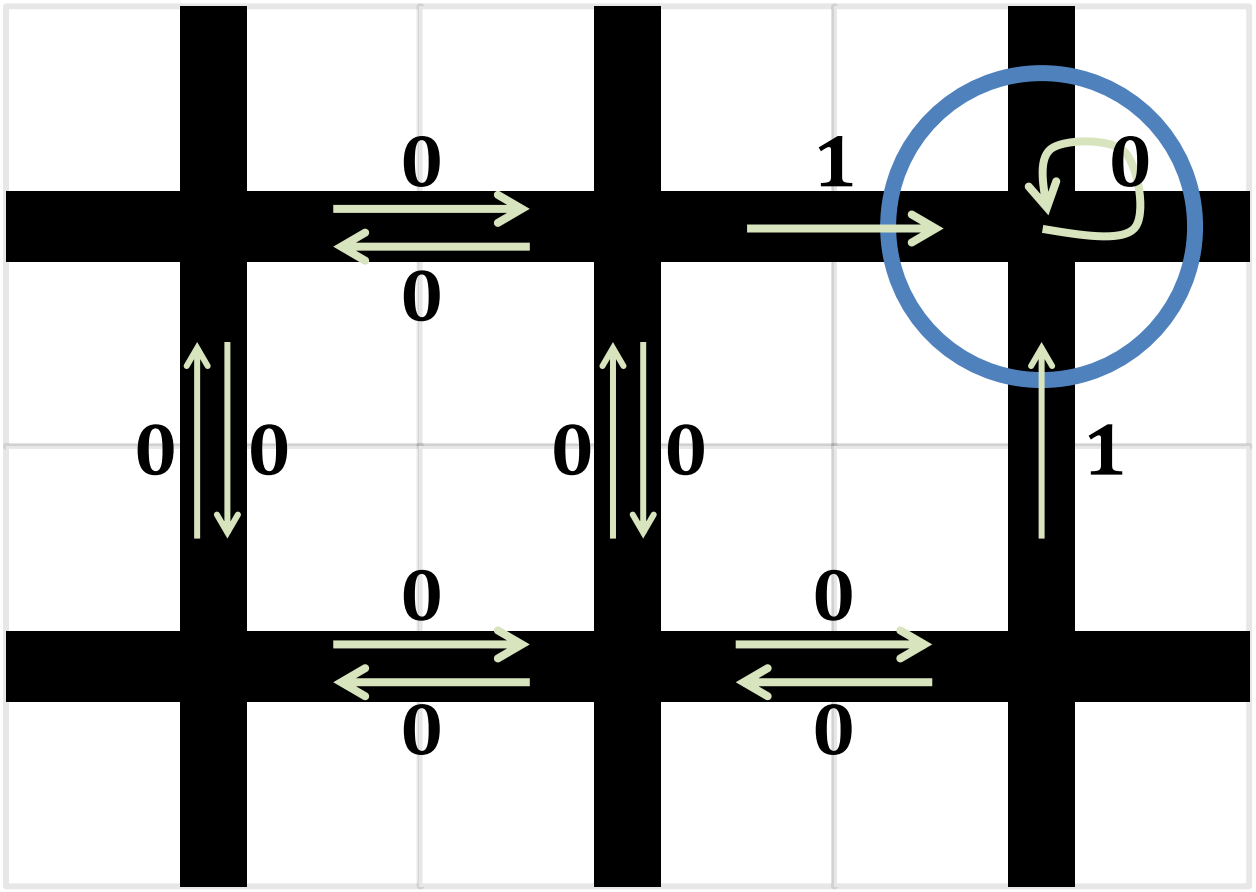
학습의 결과 : 각 상태에서 취해야 할 행동 → 정책 (policy)

로봇의 위치

왼쪽으로, 오른쪽으로
위로, 아래로

각 상태에서의 행동에 대한 보상

상태 (state) : 로봇의 위치 → 회색 테두리의 사각형 칸
행동 (action) : 연두색 화살표
보상 (reward) : 화살표 옆의 숫자 (0 또는 1)

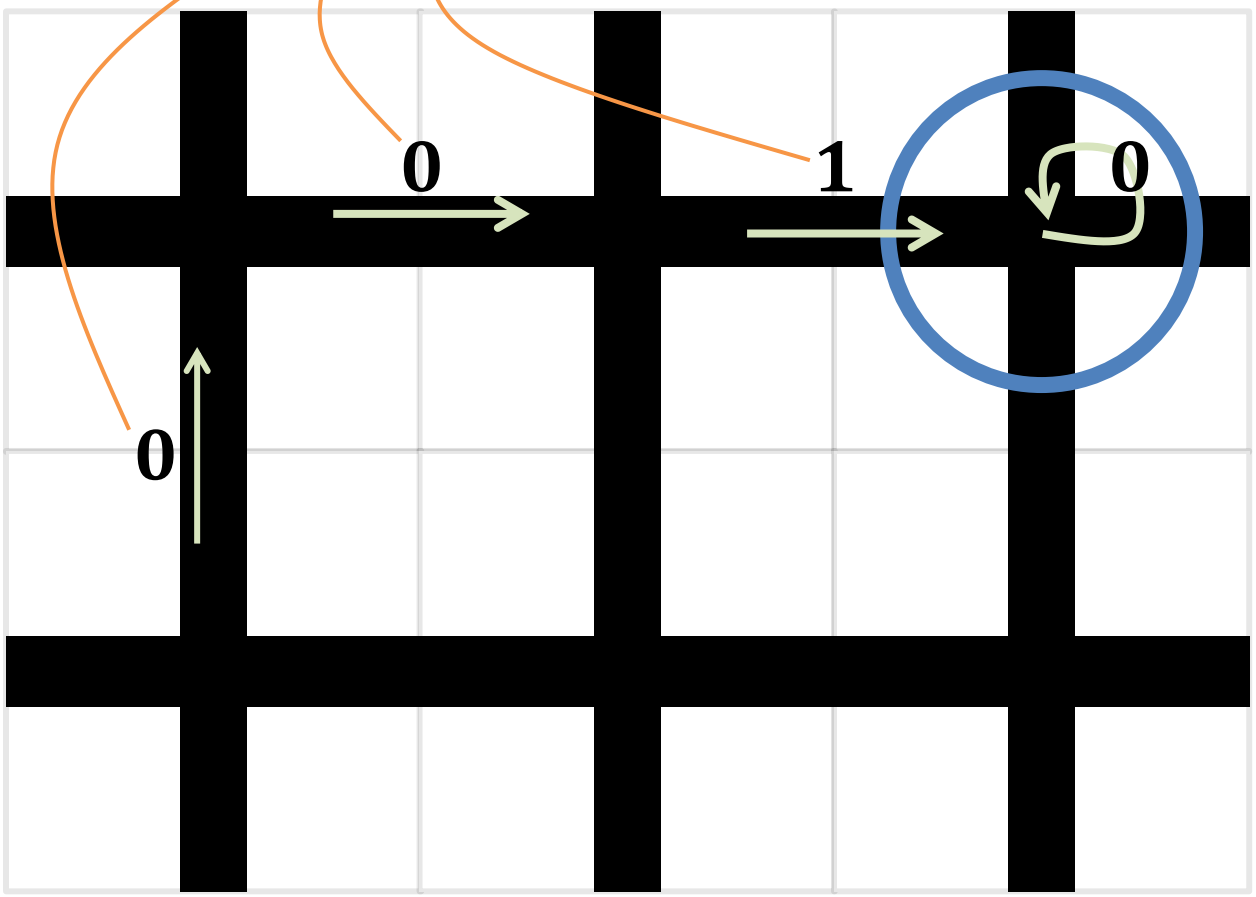


누적 보상

현재부터 미래까지의 모든 보상의 합

$r_t + r_{t+1} + r_{t+2} + \cdots$

누적 보상 : $0 + 0 + 1 = 1$

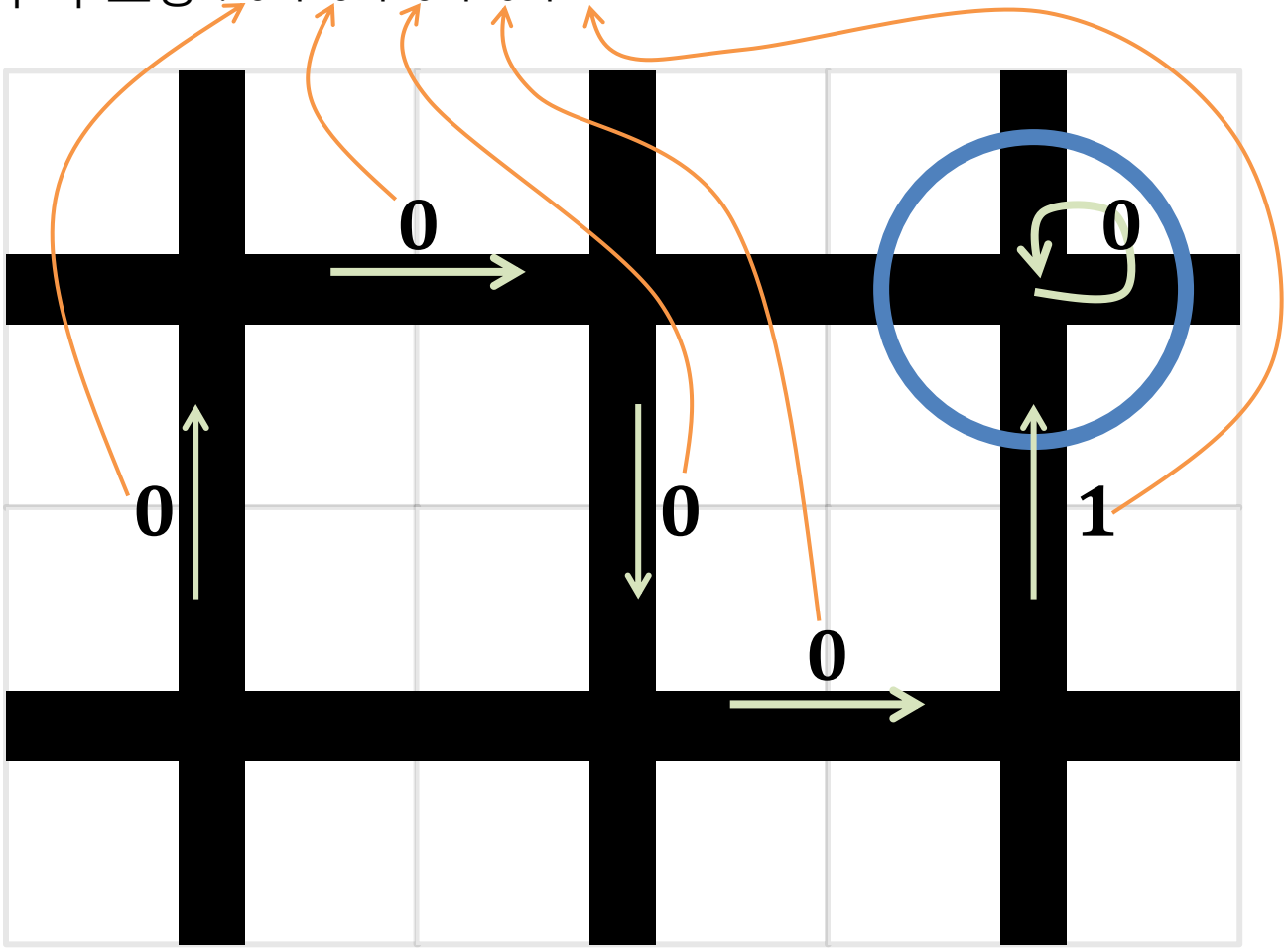


누적 보상

현재부터 미래까지의 모든 보상의 합

$r_t + r_{t+1} + r_{t+2} + \cdots$

누적 보상 : $0 + 0 + 0 + 0 + 1 = 1$



누적 보상

$$r_t + r_{t+1} + r_{t+2} + \cdots$$

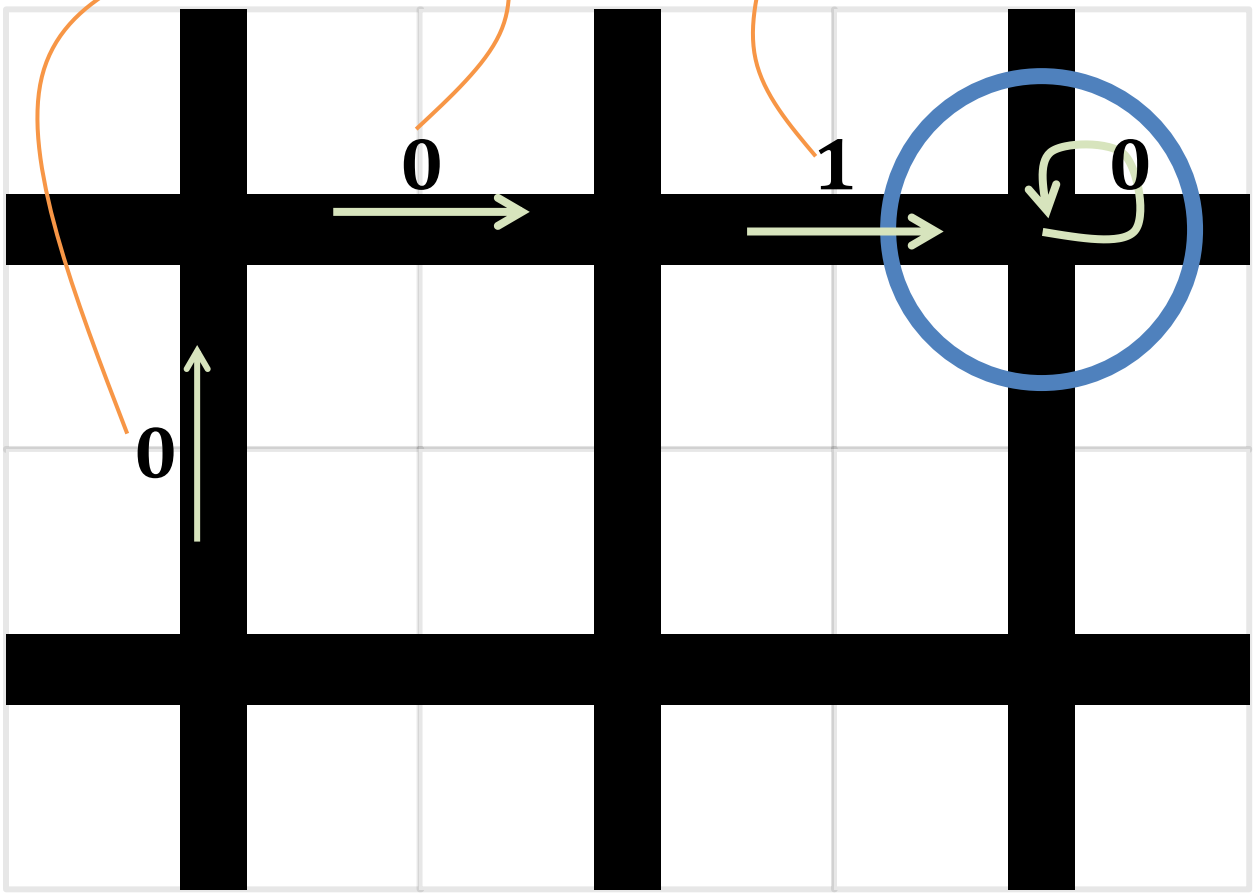
$$r_t + (0.9)r_{t+1} + (0.9)^2 r_{t+2} + (0.9)^3 r_{t+3} + \cdots$$

누적 보상

현재부터 미래까지의 모든 보상의 합

$r_t + (0.9)r_{t+1} + (0.9)^2 r_{t+2} + (0.9)^3 r_{t+3} + \cdots$

누적 보상 : $0 + (0.9) \times 0 + (0.9)^2 \times 1 = 0.81$

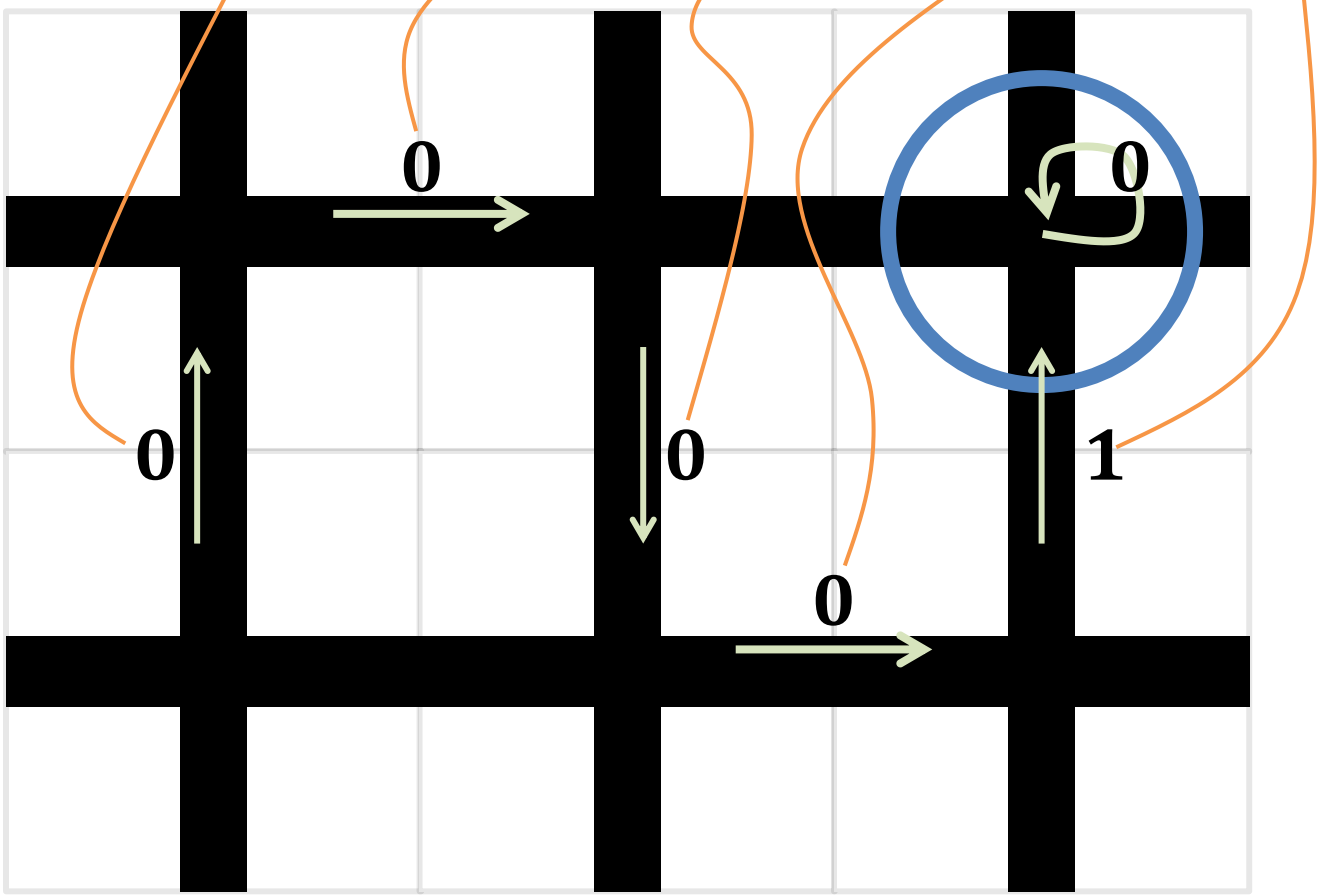


누적 보상

현재부터 미래까지의 모든 보상의 합

$r_t + (0.9)r_{t+1} + (0.9)^2 r_{t+2} + (0.9)^3 r_{t+3} + \cdots$

누적 보상 : $0 + (0.9) \times 0 + (0.9)^2 \times 0 + (0.9)^3 \times 0 + (0.9)^4 \times 1 = 0.6561$



개미는 정말 짧은 길을 알아낼 수 있을까?

https://youtu.be/5E32W45TuMk

누적 보상

$$r_t + r_{t+1} + r_{t+2} + \cdots$$

$$r_t + (0.9)r_{t+1} + (0.9)^2 r_{t+2} + (0.9)^3 r_{t+3} + \cdots$$

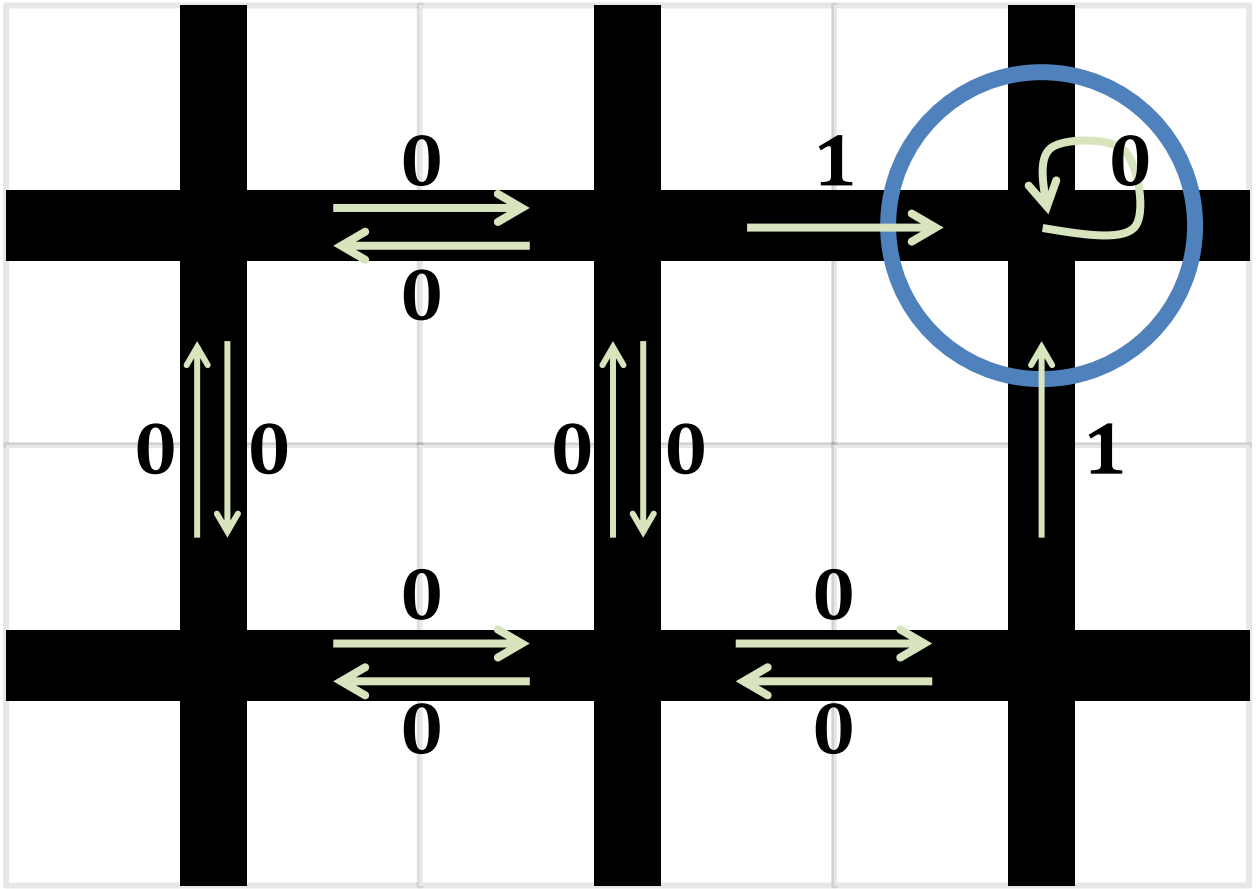
γ : 할인율(discount factor), $0 \leq \gamma < 1$

누적 보상

각 상태에서 취해야 할 행동

누적 보상이 최대가 되도록 하는 정책을 구하라?

→ 미래의 상태 변화, 보상을 모두 알고 있어야 함



Q-러닝

$$\hat{Q}(s,a) \leftarrow r + \gamma \max_{a'} \hat{Q}(s',a')$$

0.9

모든 상태에서의 모든 행동을 계속 경험하면 $\hat{Q}(s,a) \rightarrow Q(s,a)$로 수렴

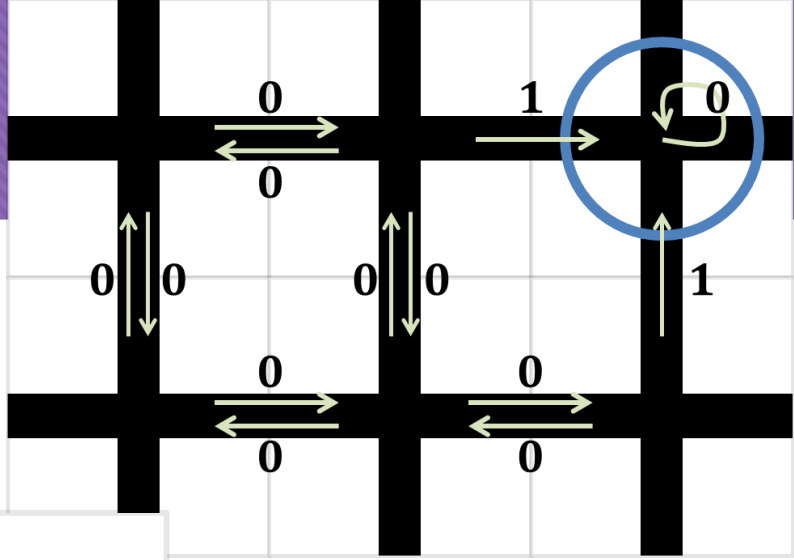
$$\pi^*(s) = \operatorname{argmax}_a Q(s,a)$$

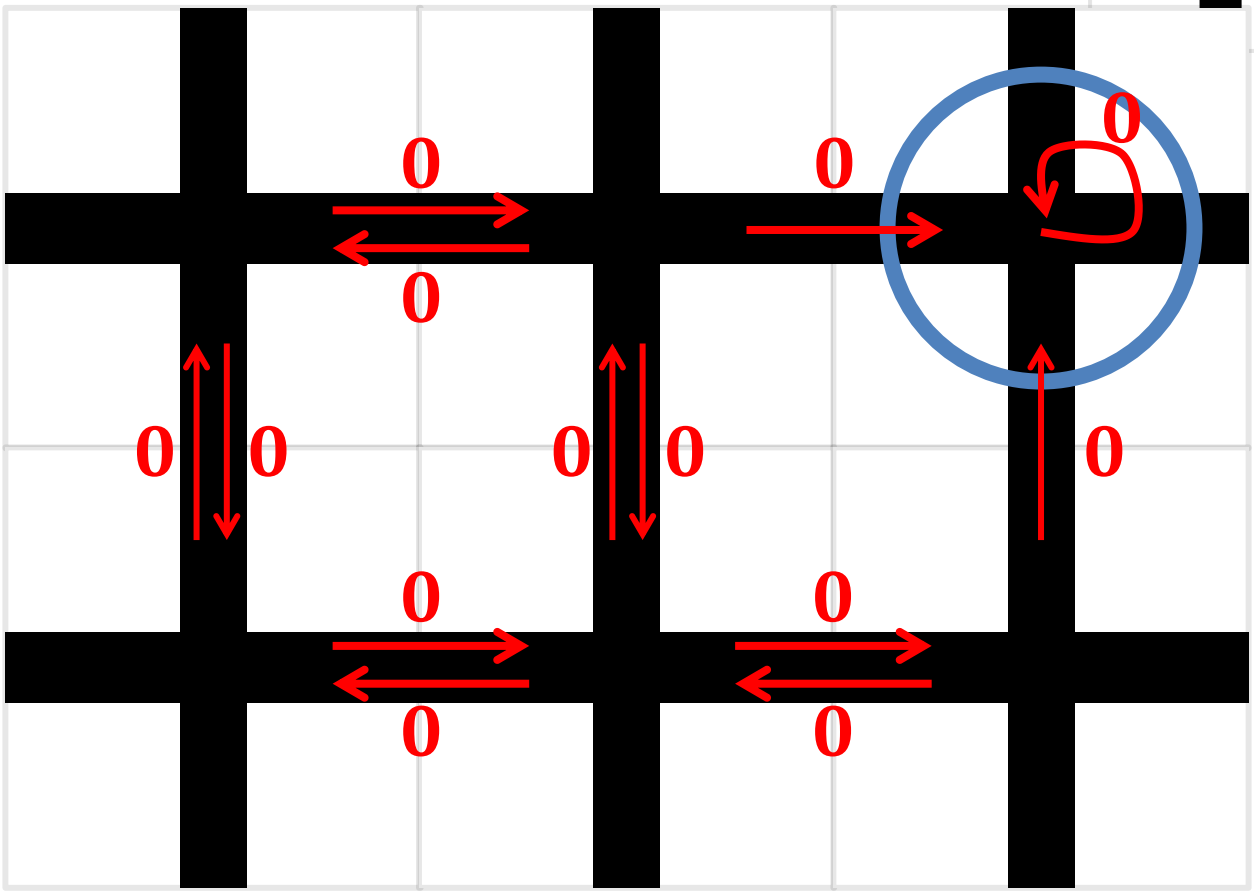
각 상태에서 $Q(s,a)$가 최대인 행동(a)을 취하면 됨

Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

$\hat{Q}$ 초기 값 : 모두 0

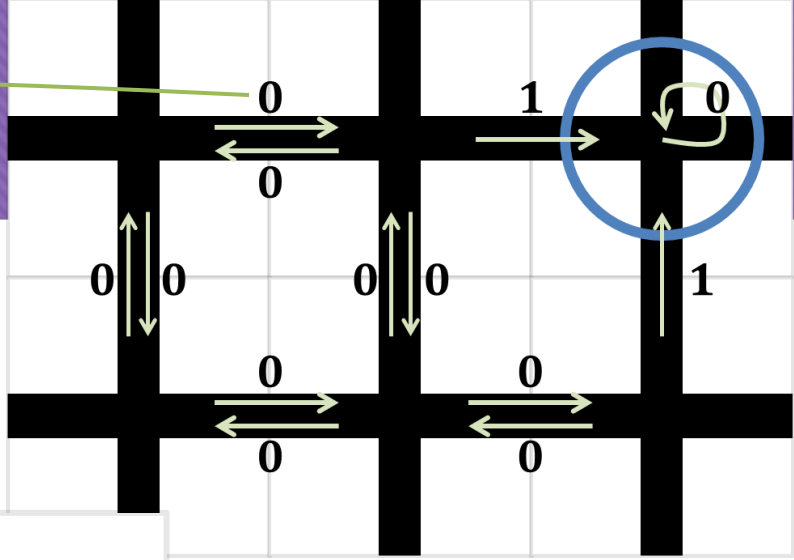


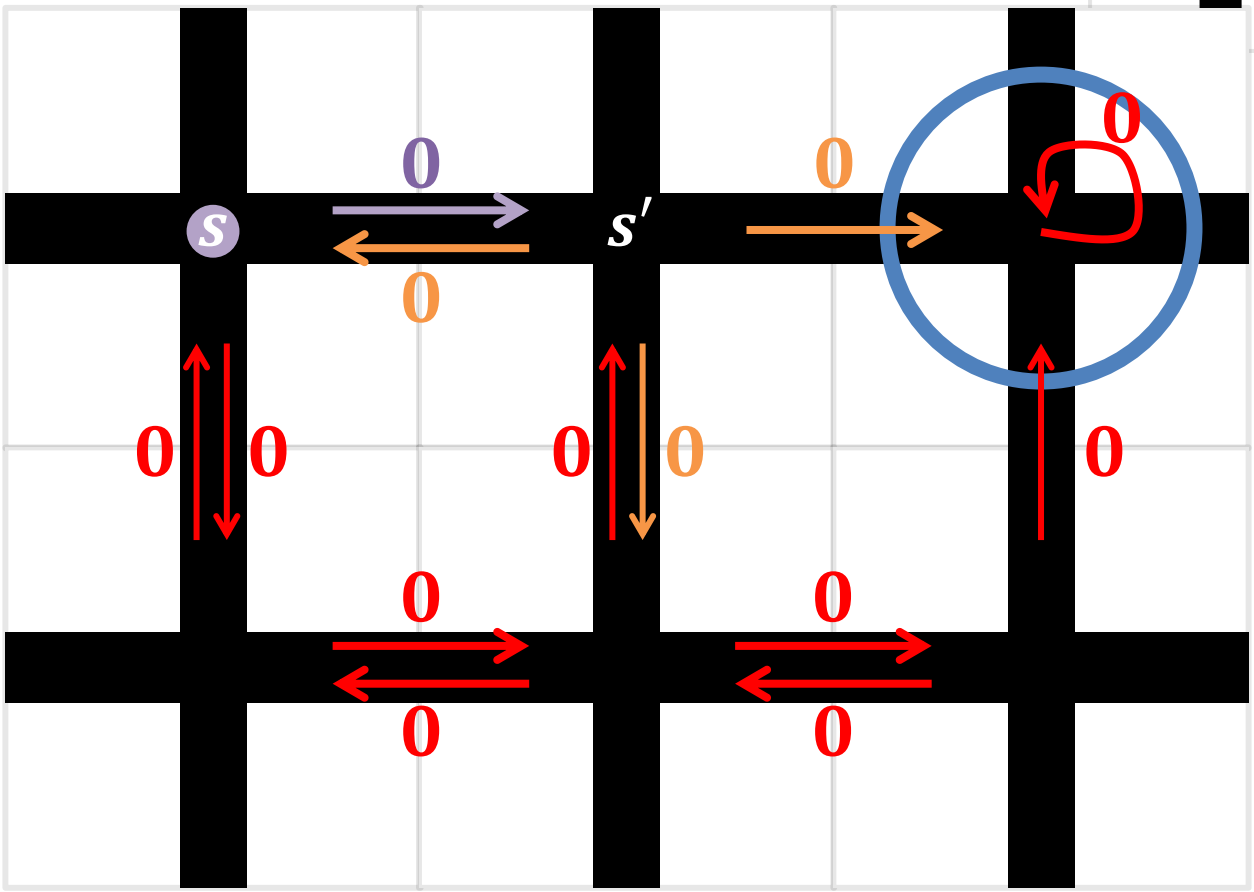


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 0

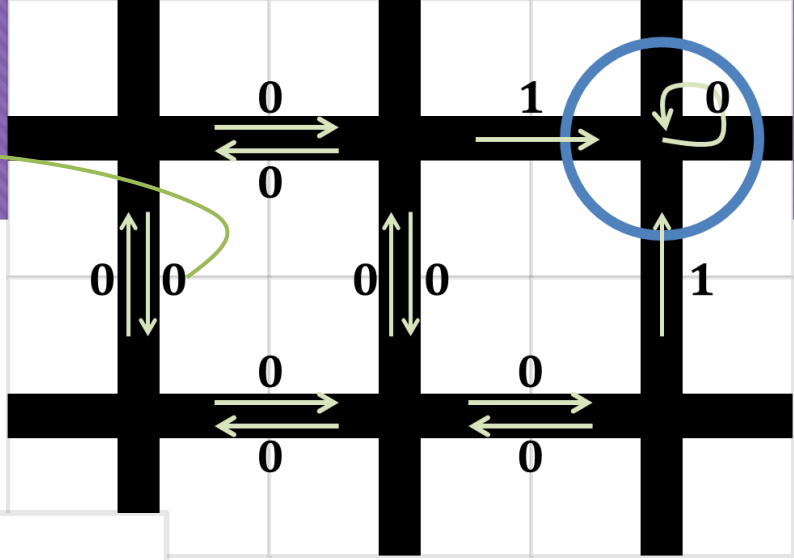


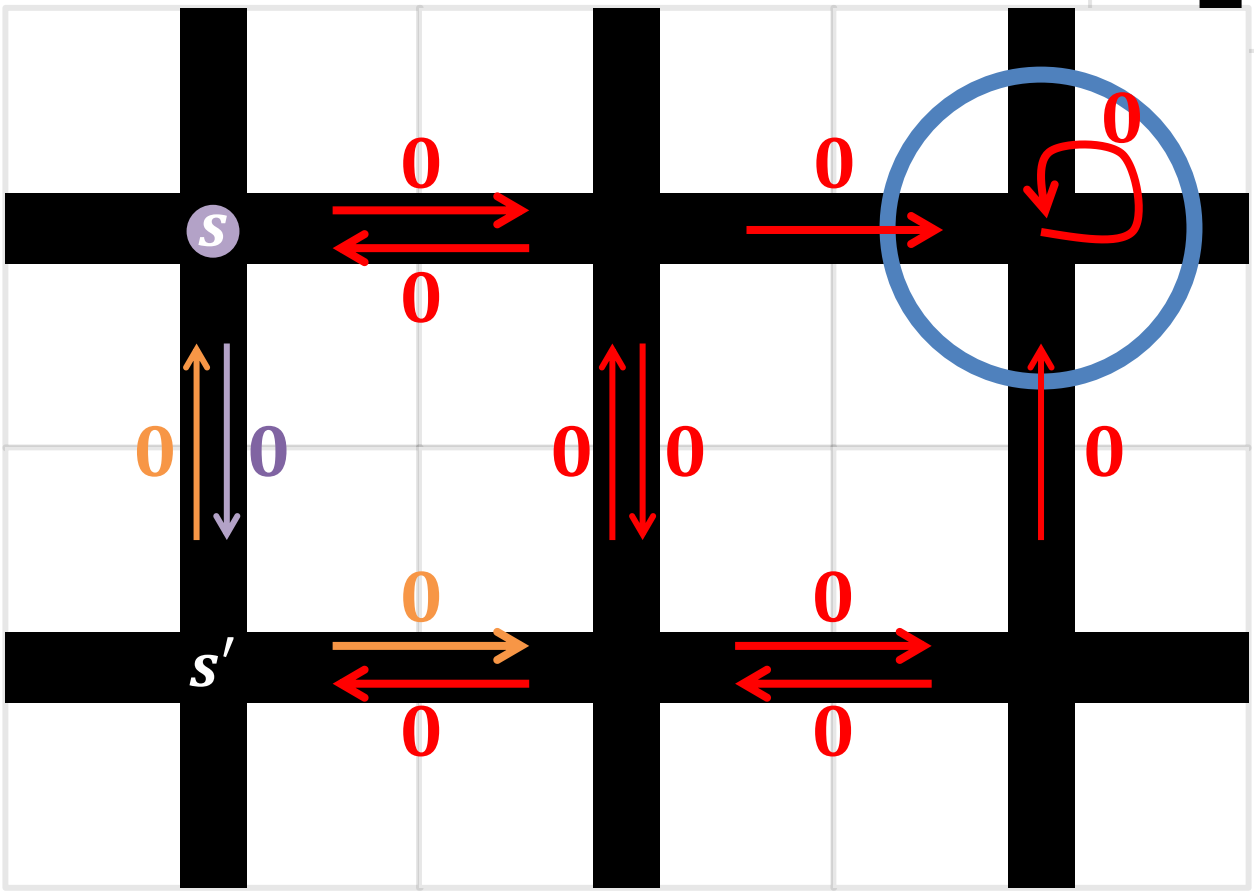


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 0

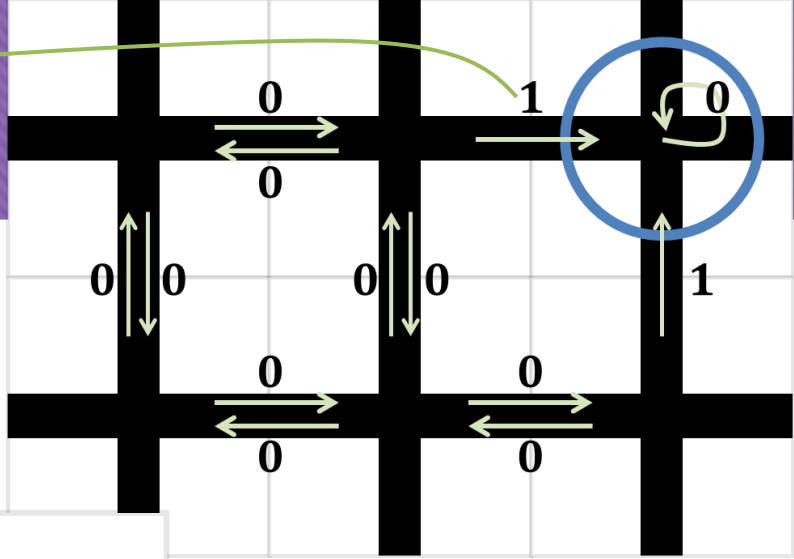


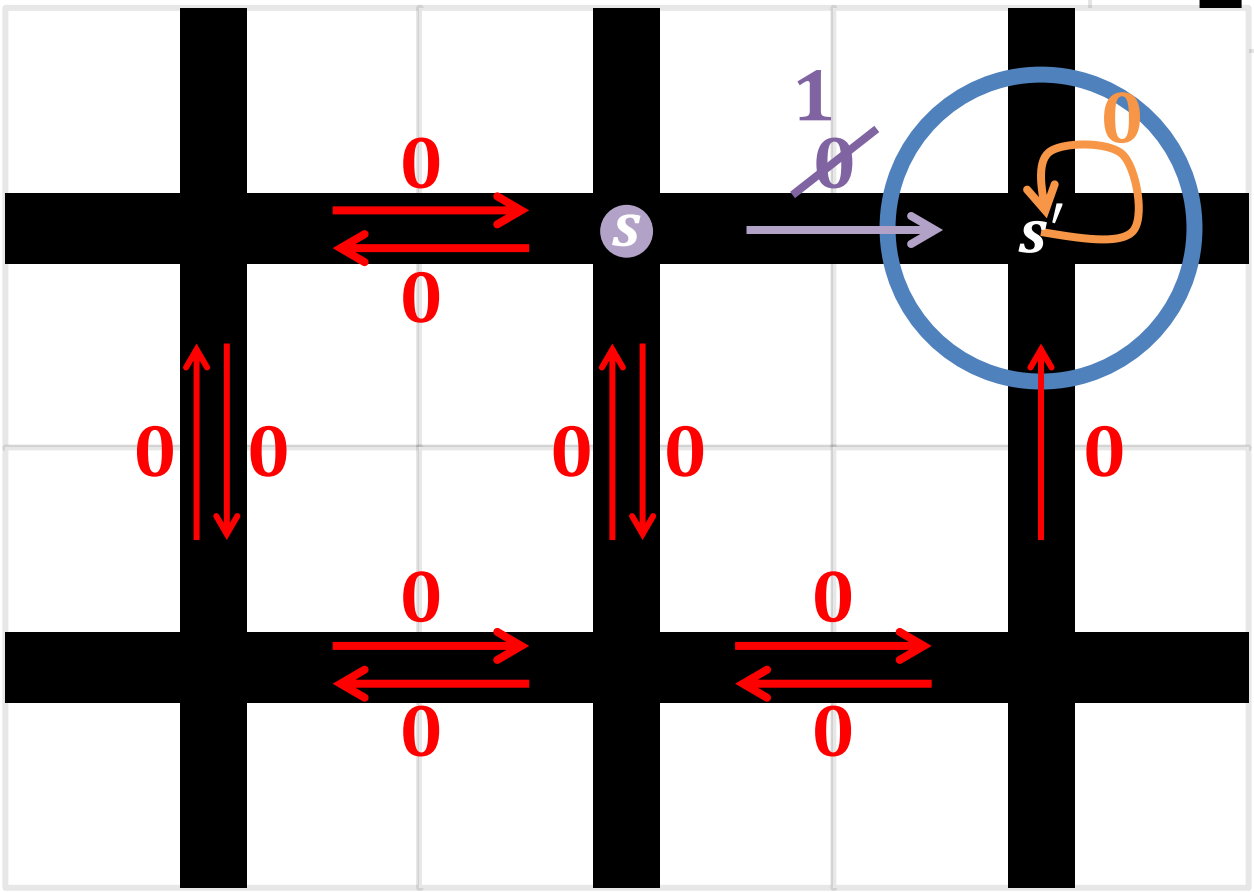


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

1 0

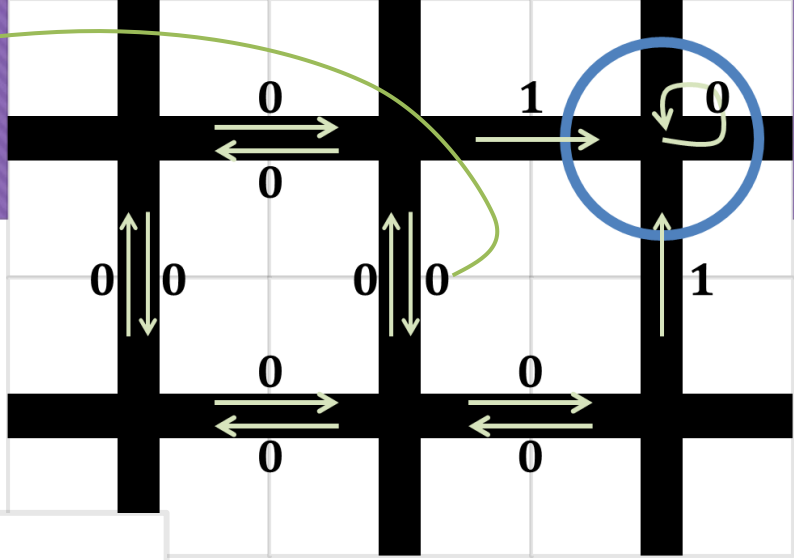


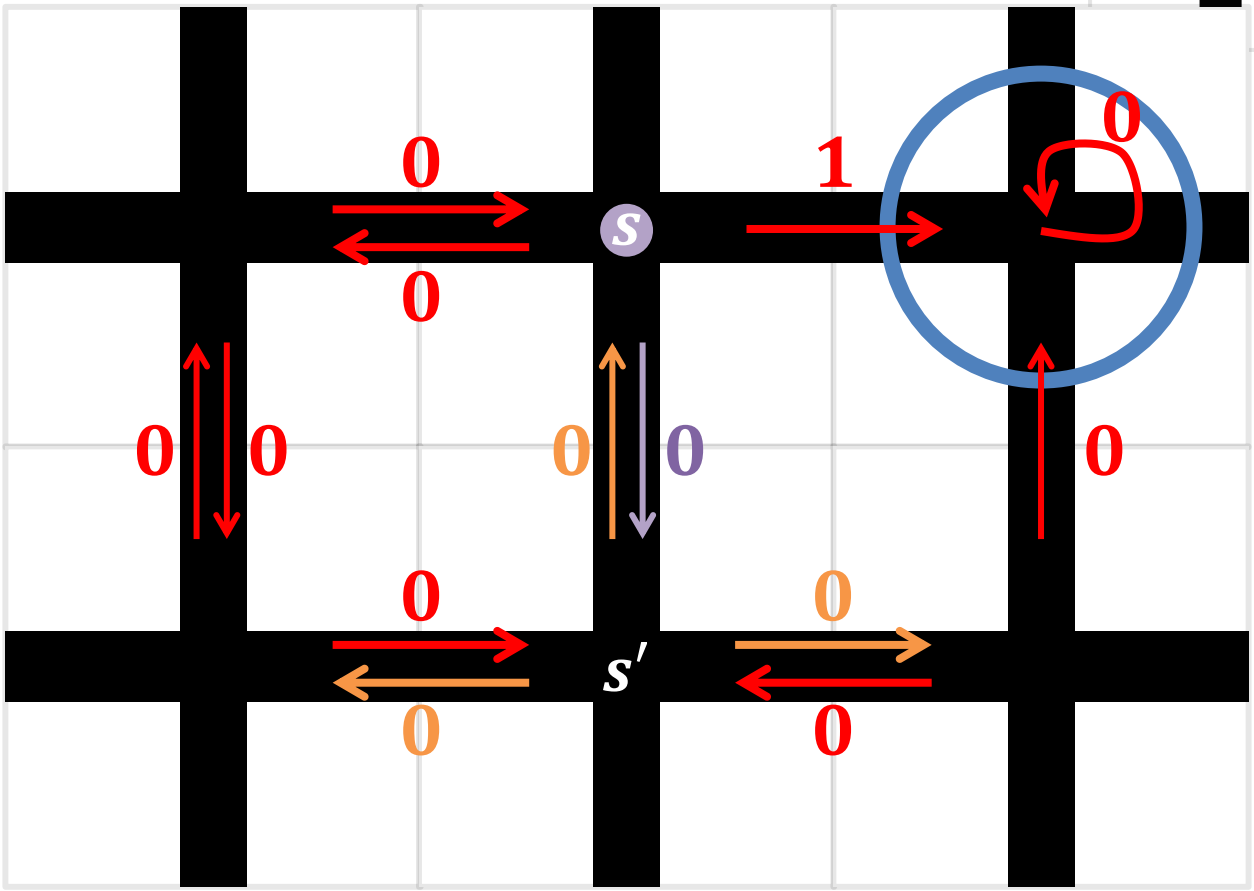


Q-러닝

$$\widehat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \widehat{Q}(s',a')$$

0 0

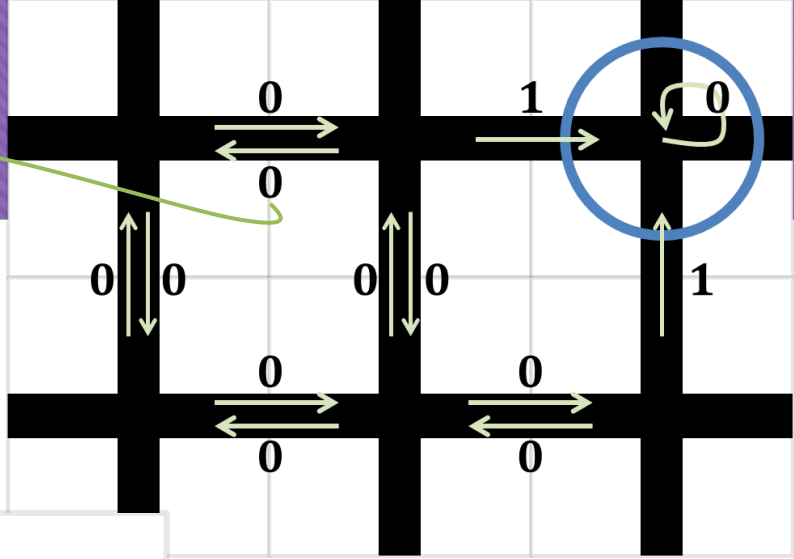


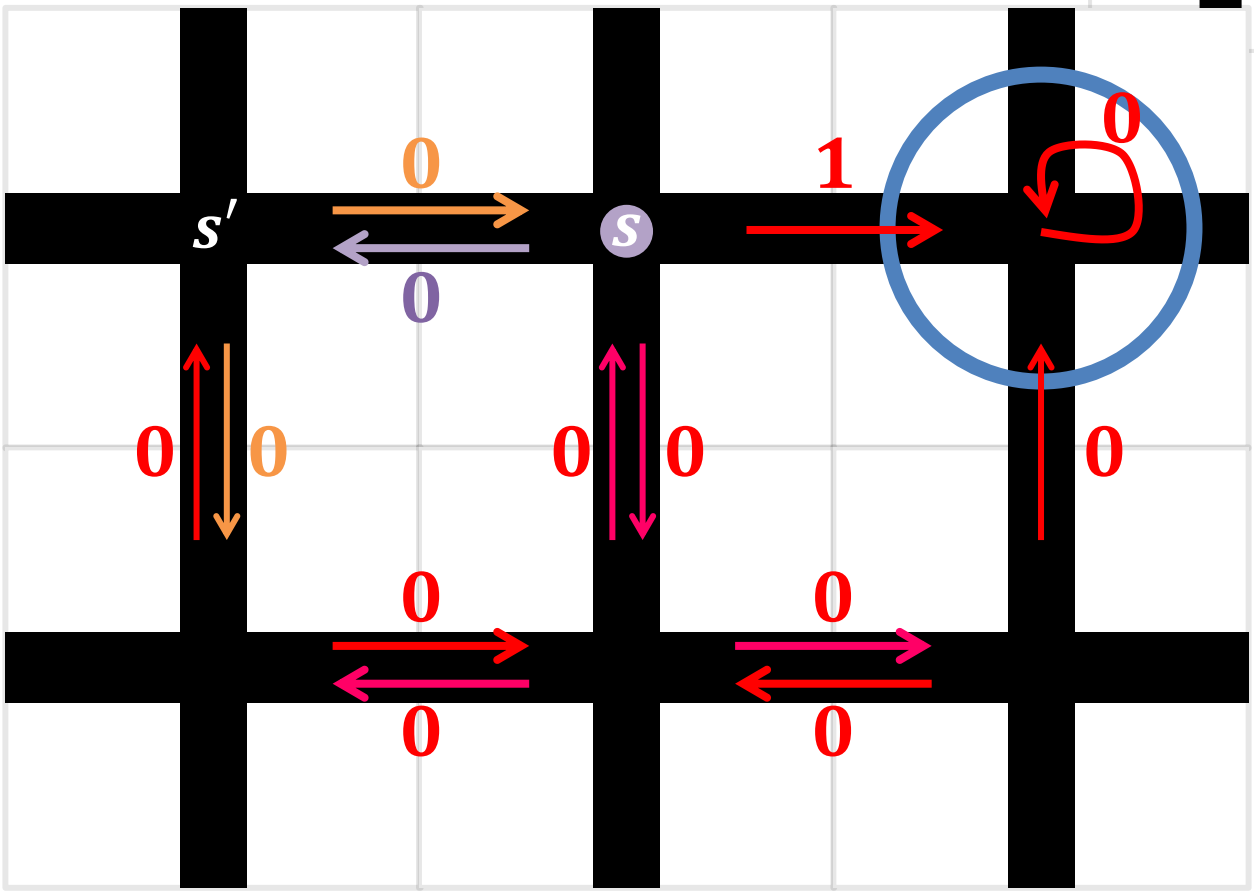


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 0

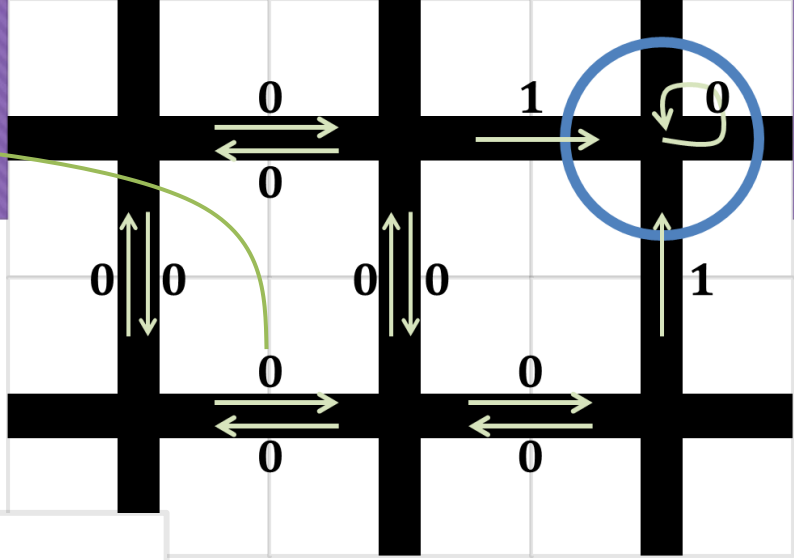


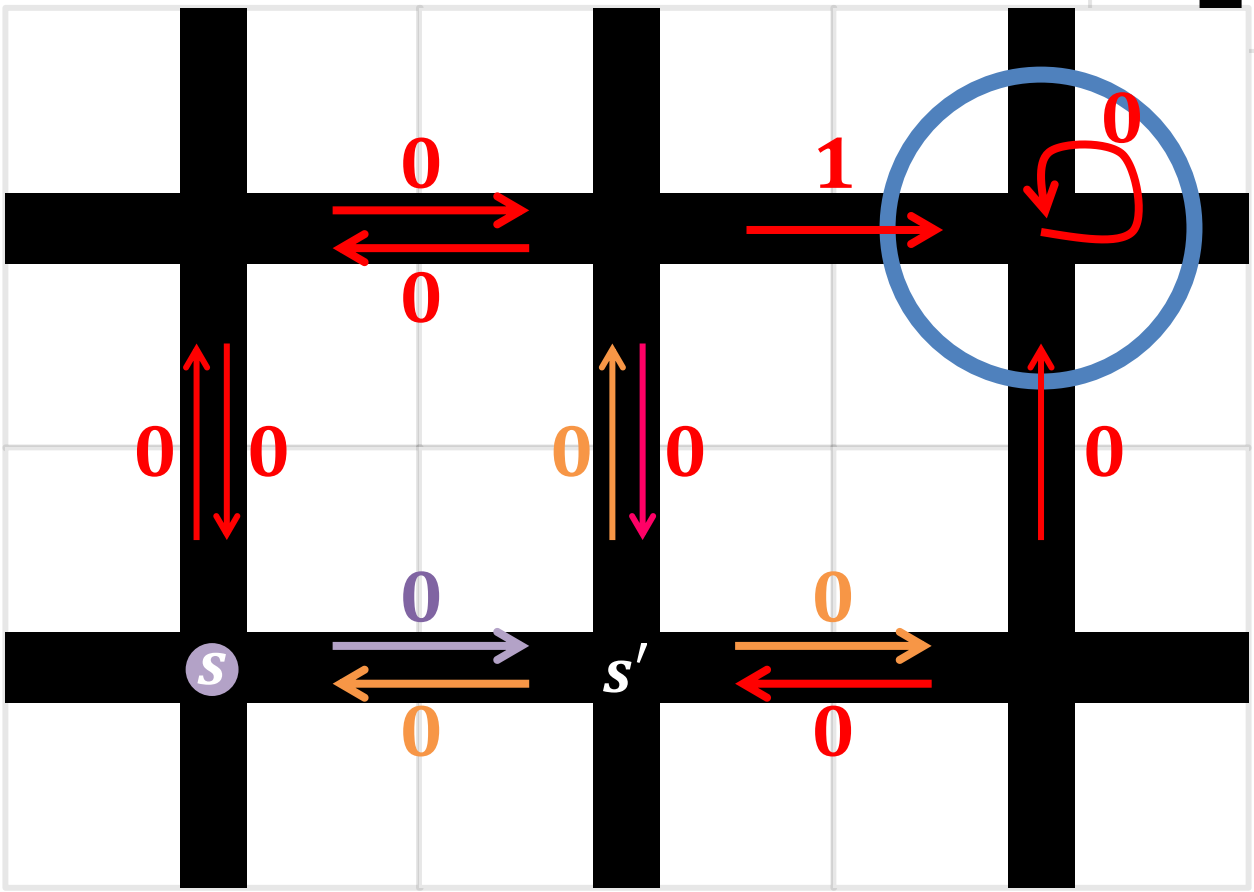


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 0

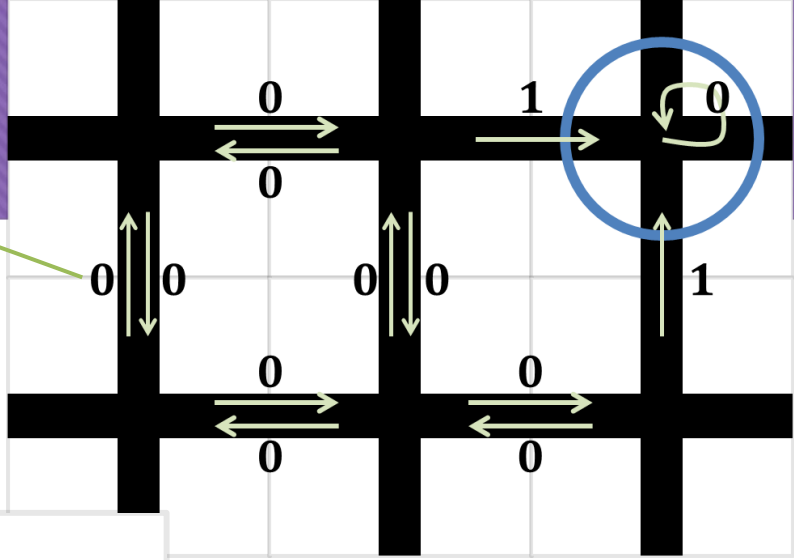


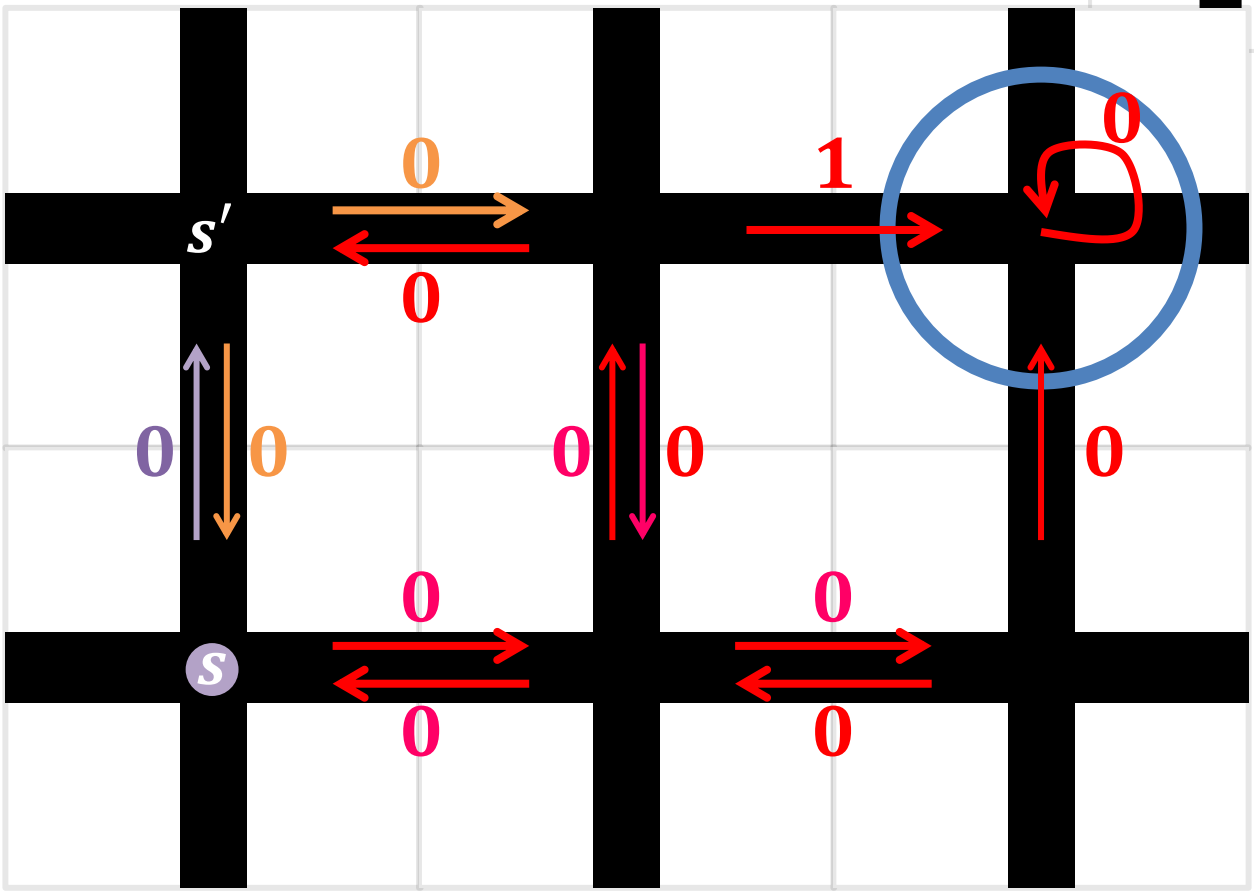


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 0

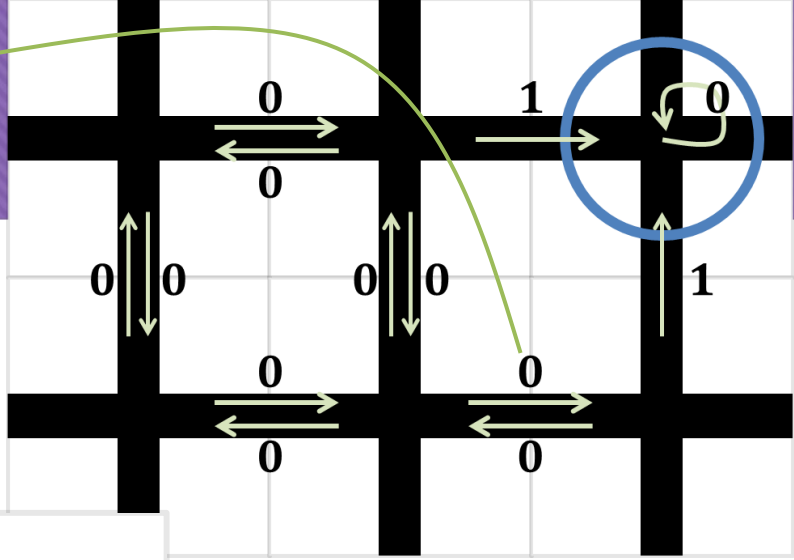


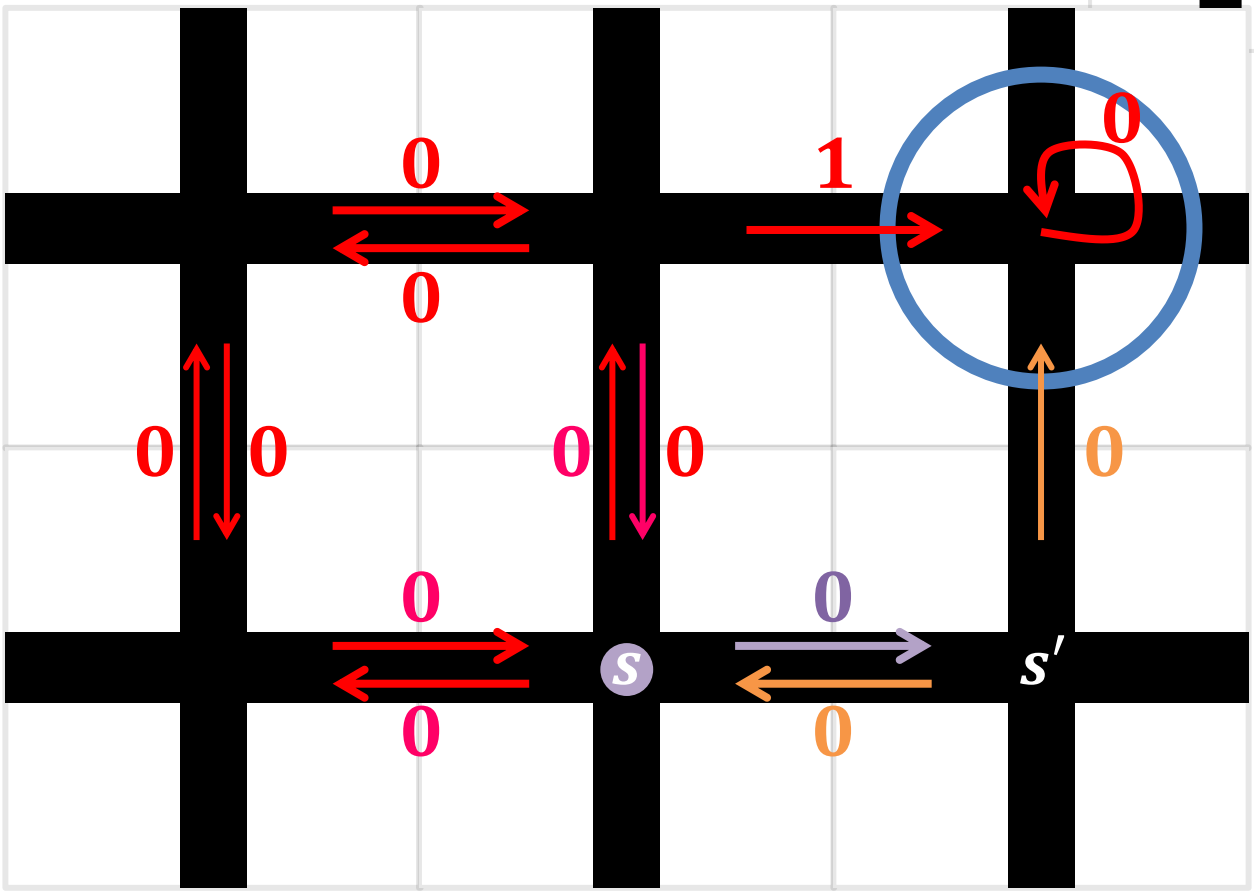


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 0

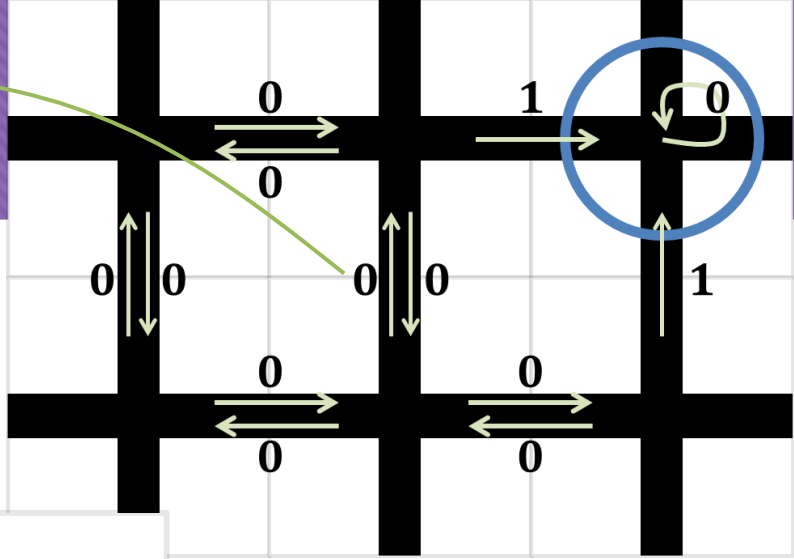


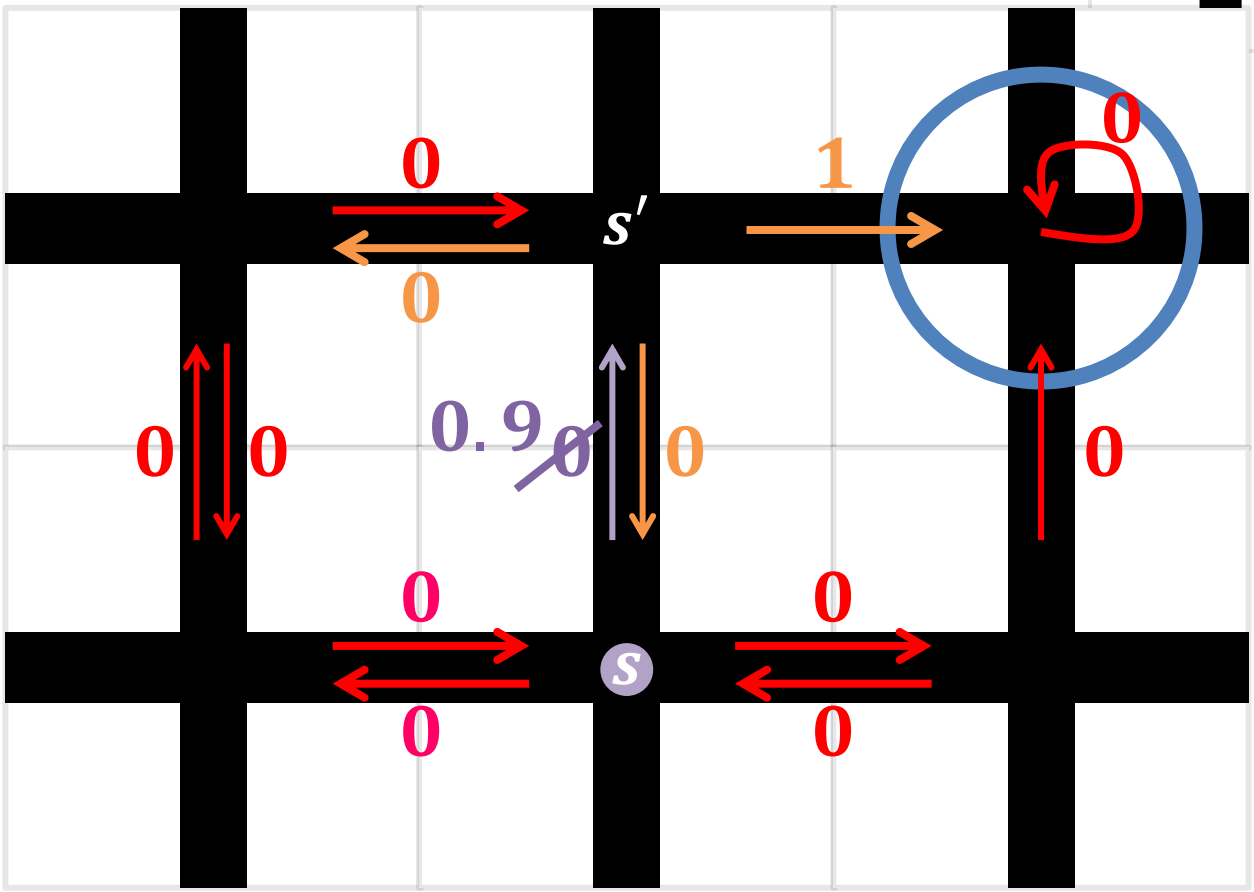


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 1

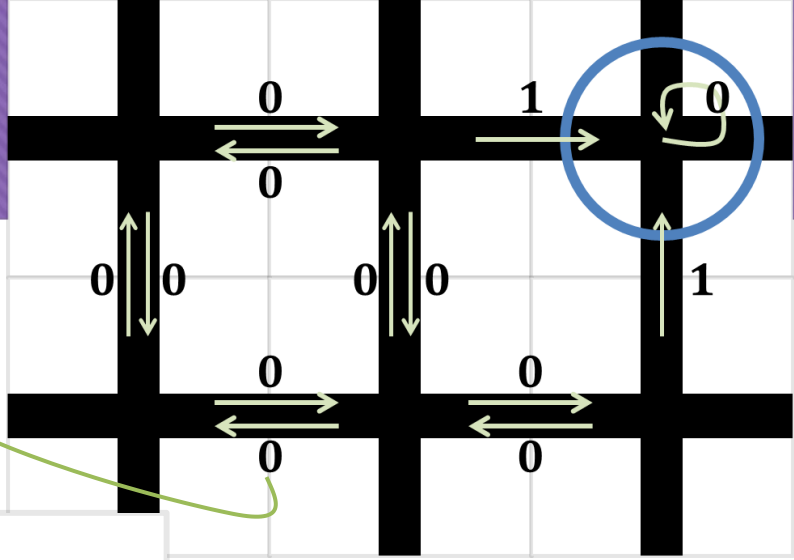


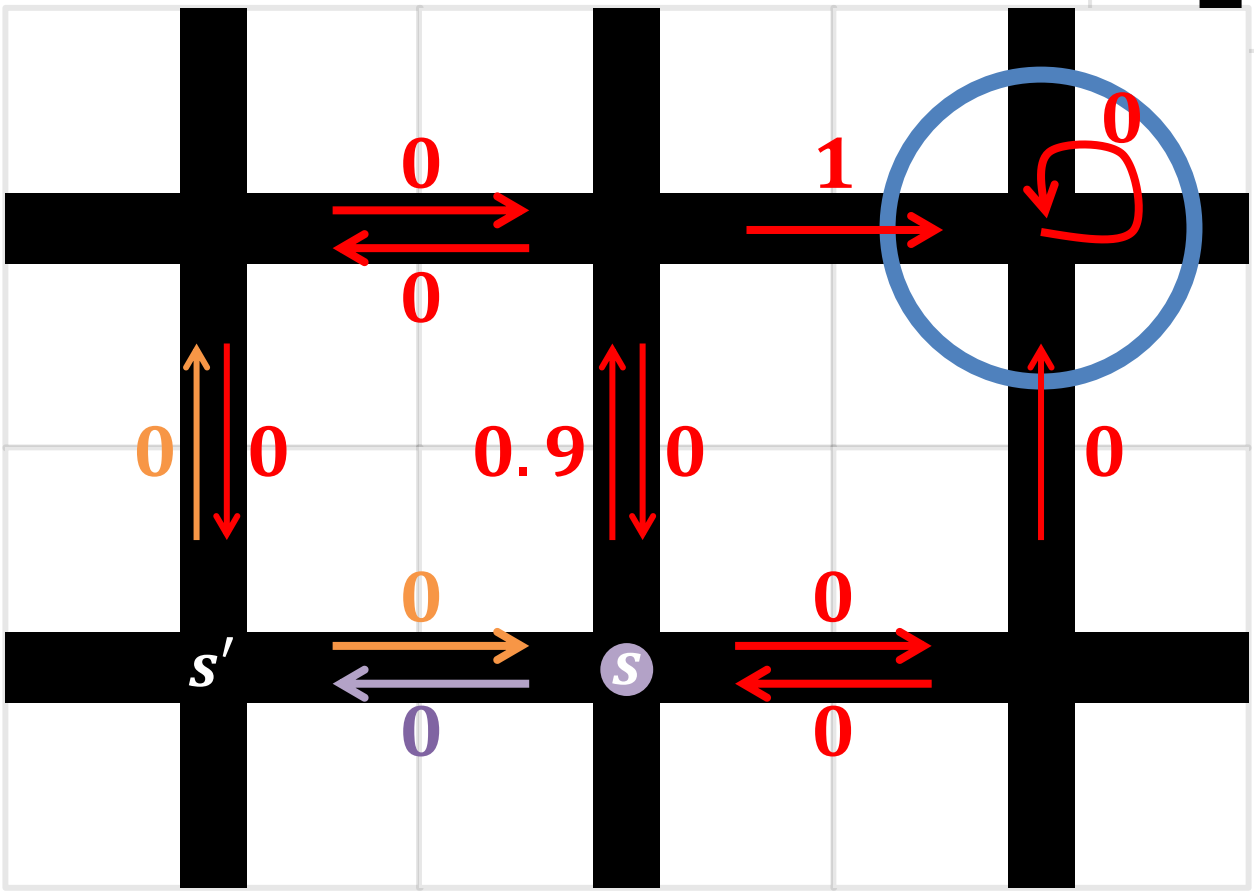


Q-러닝

$$\widehat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \widehat{Q}(s',a')$$

0 0

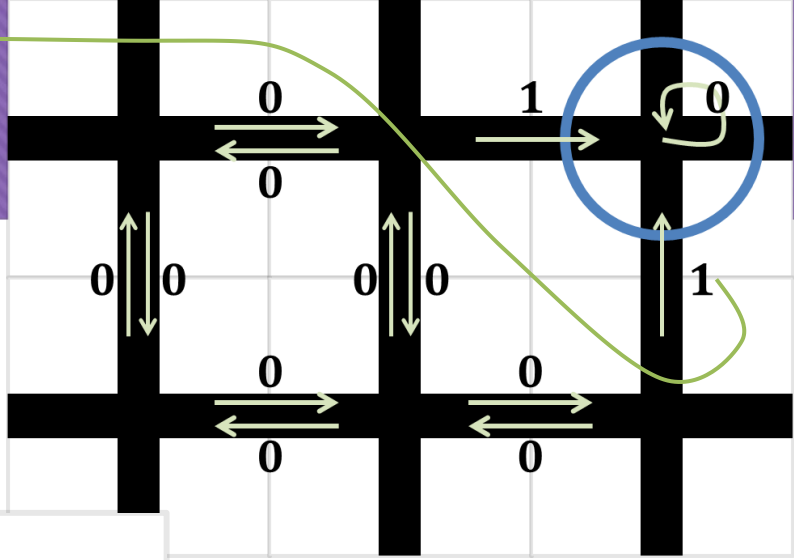


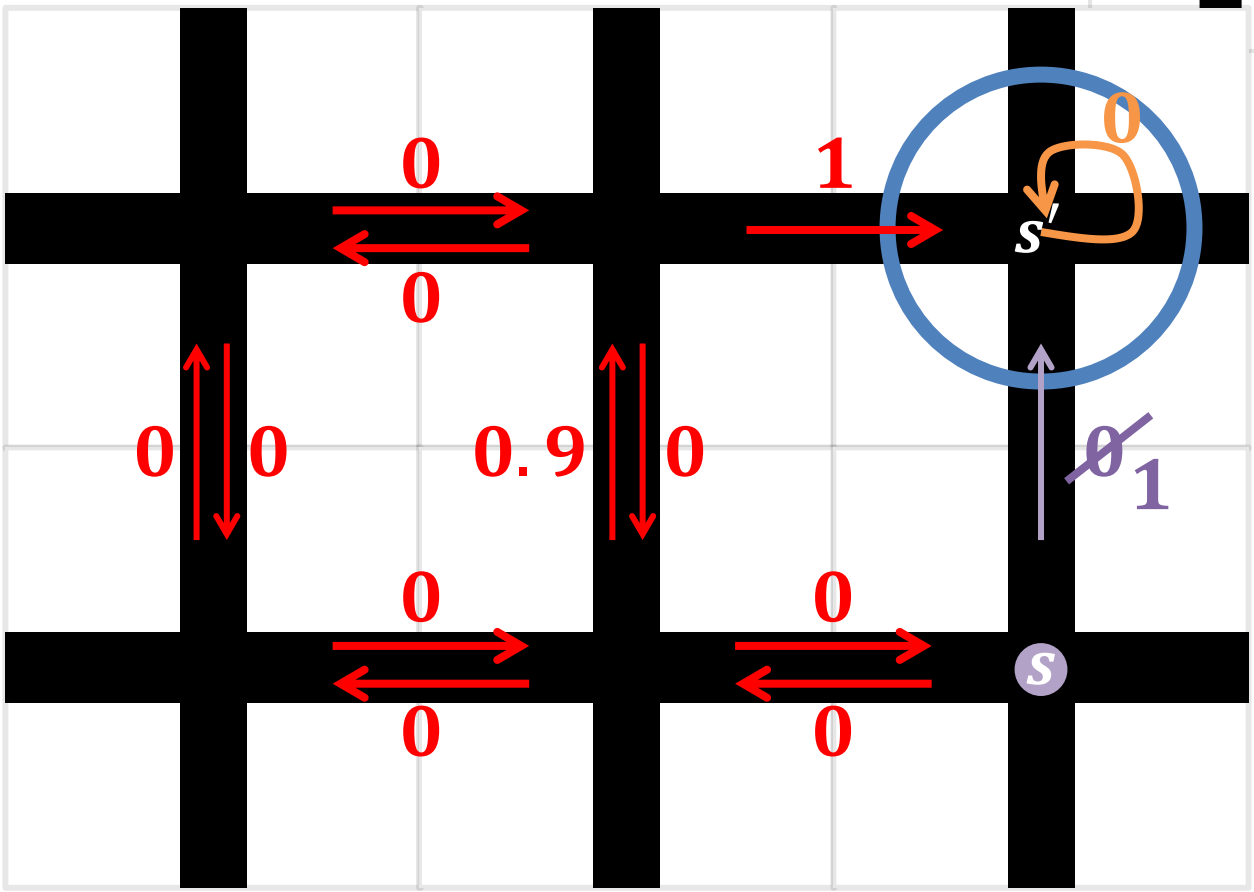


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

1 0

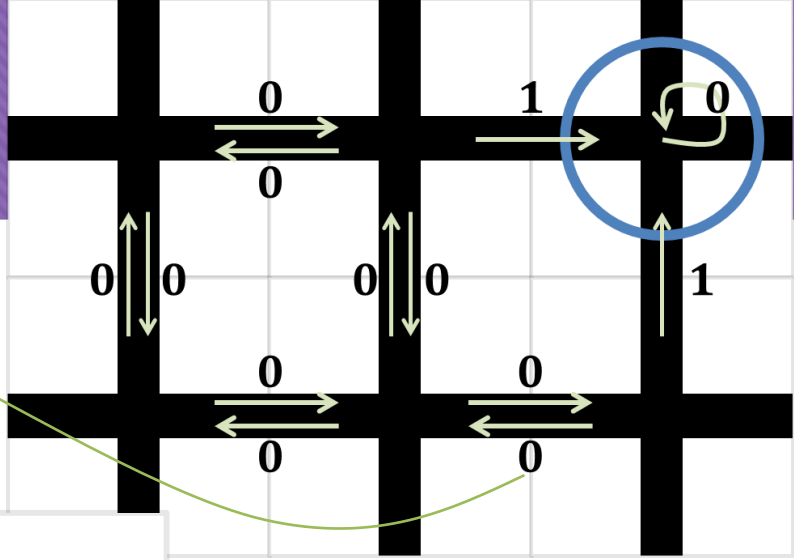


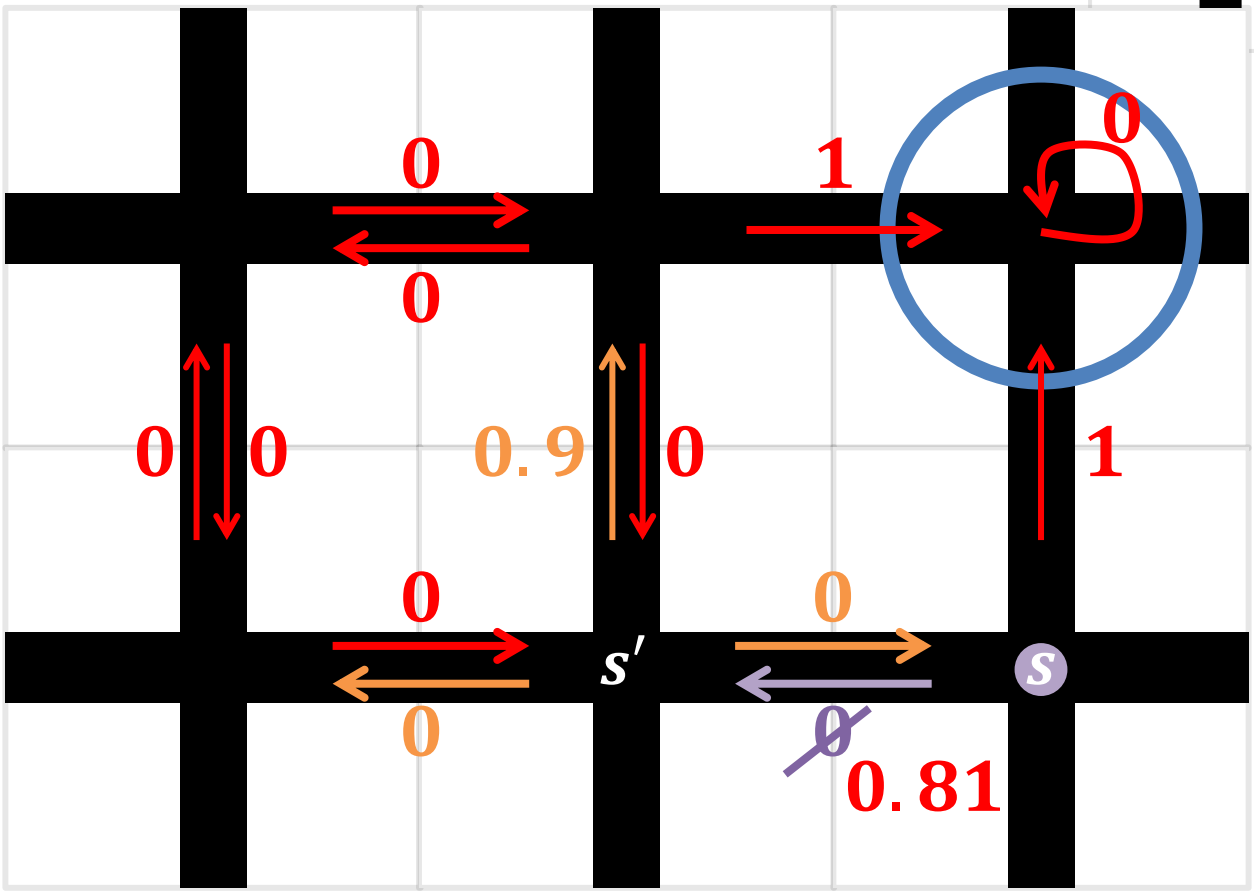


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 0.9

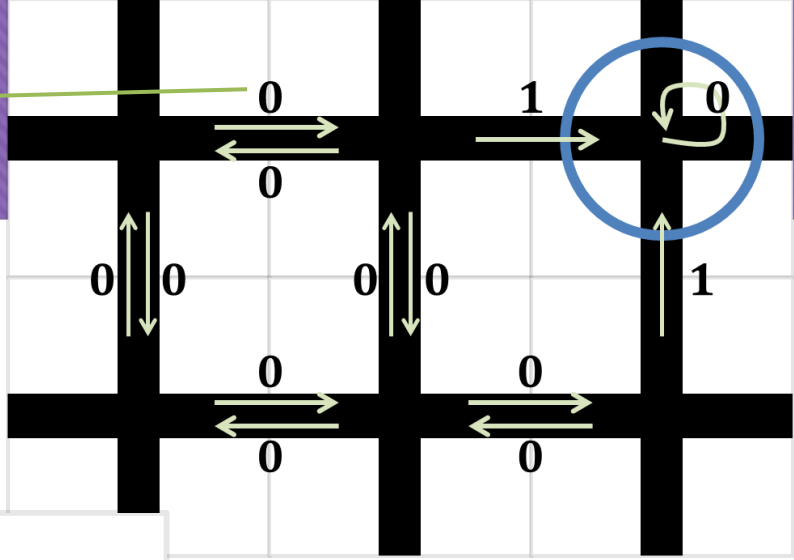


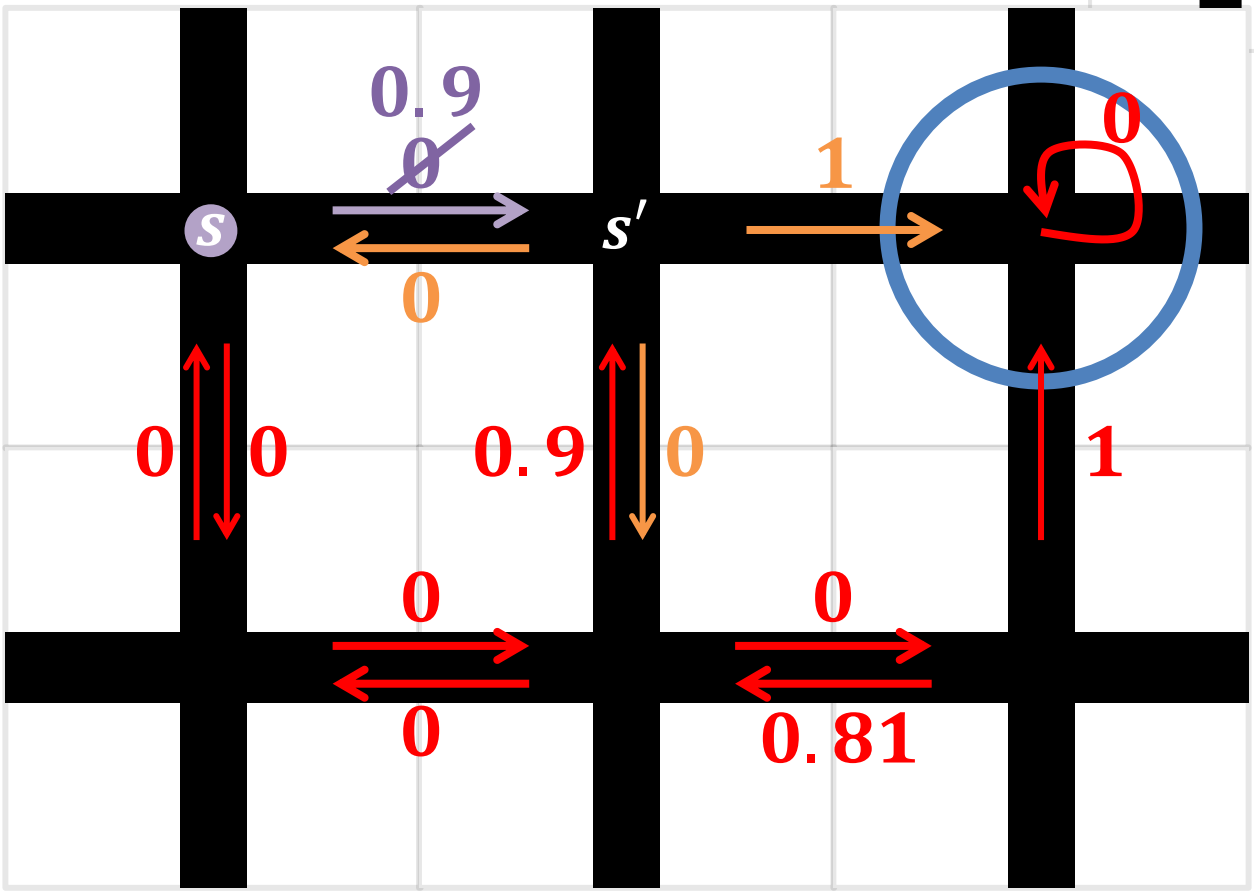


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 1

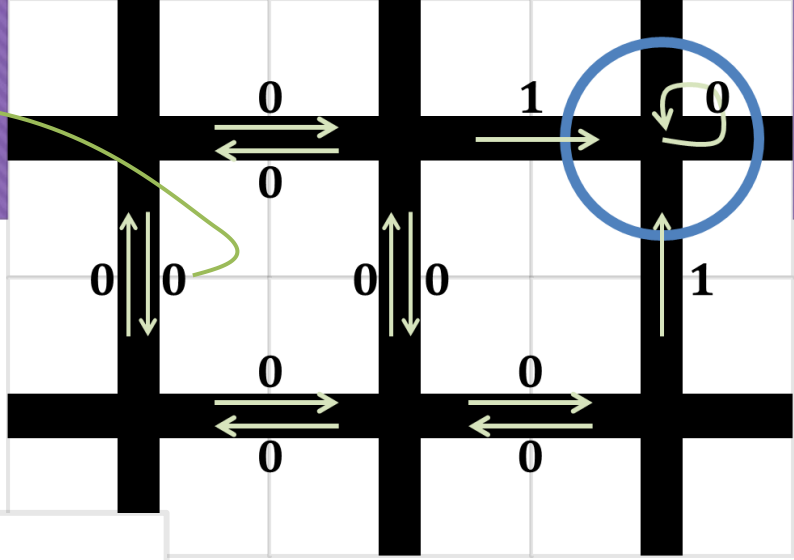


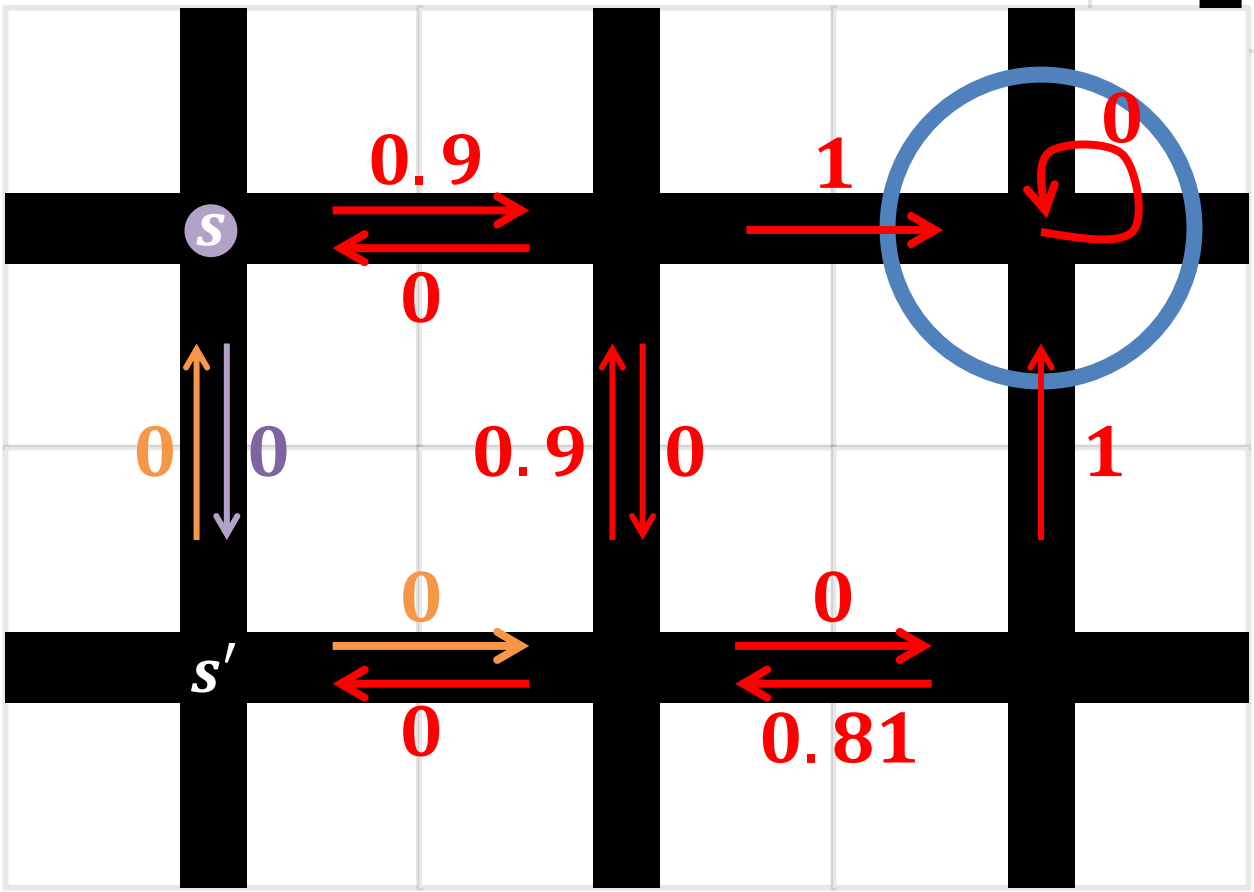


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 0

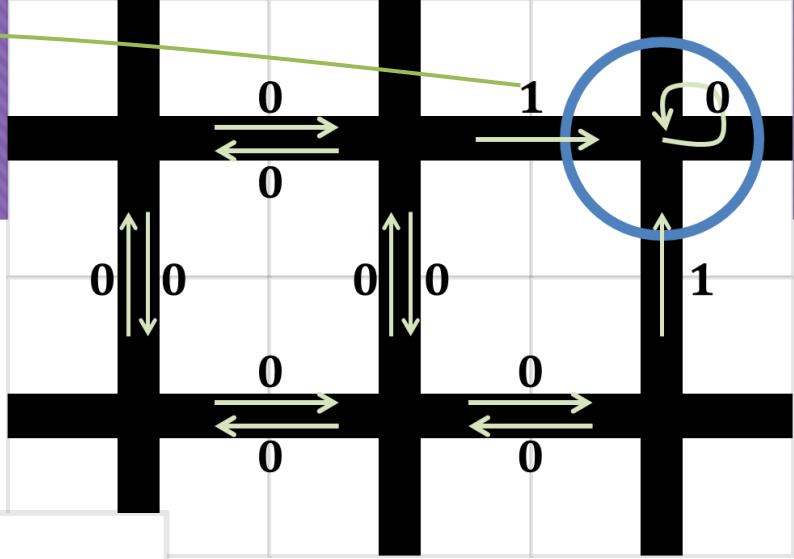


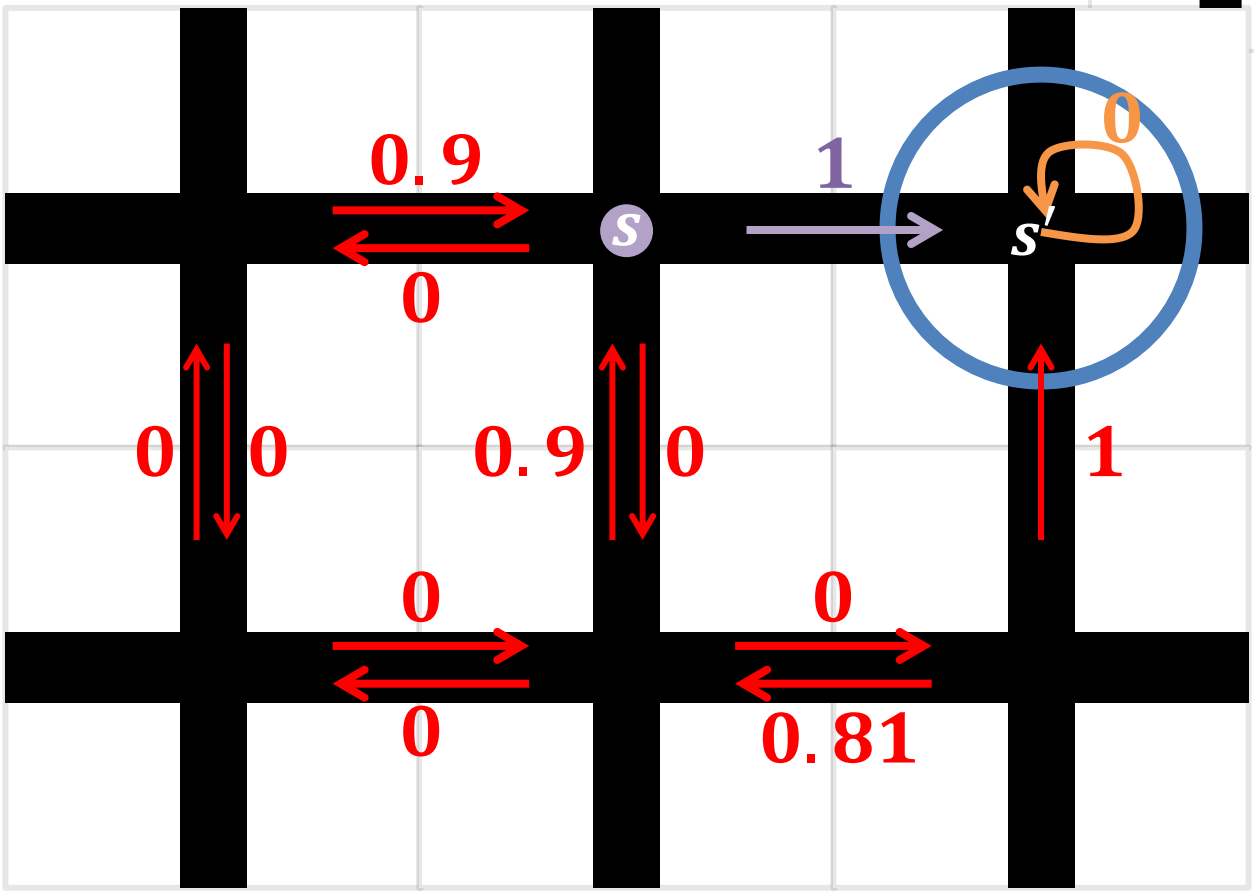


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 0

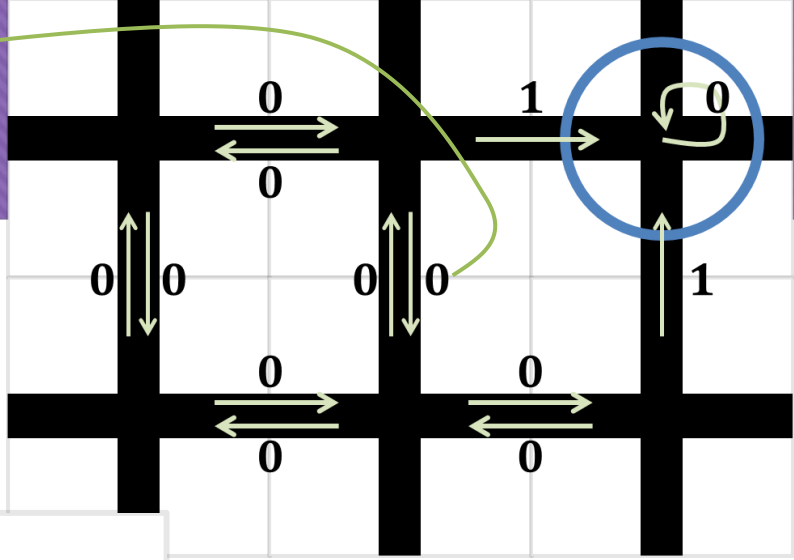


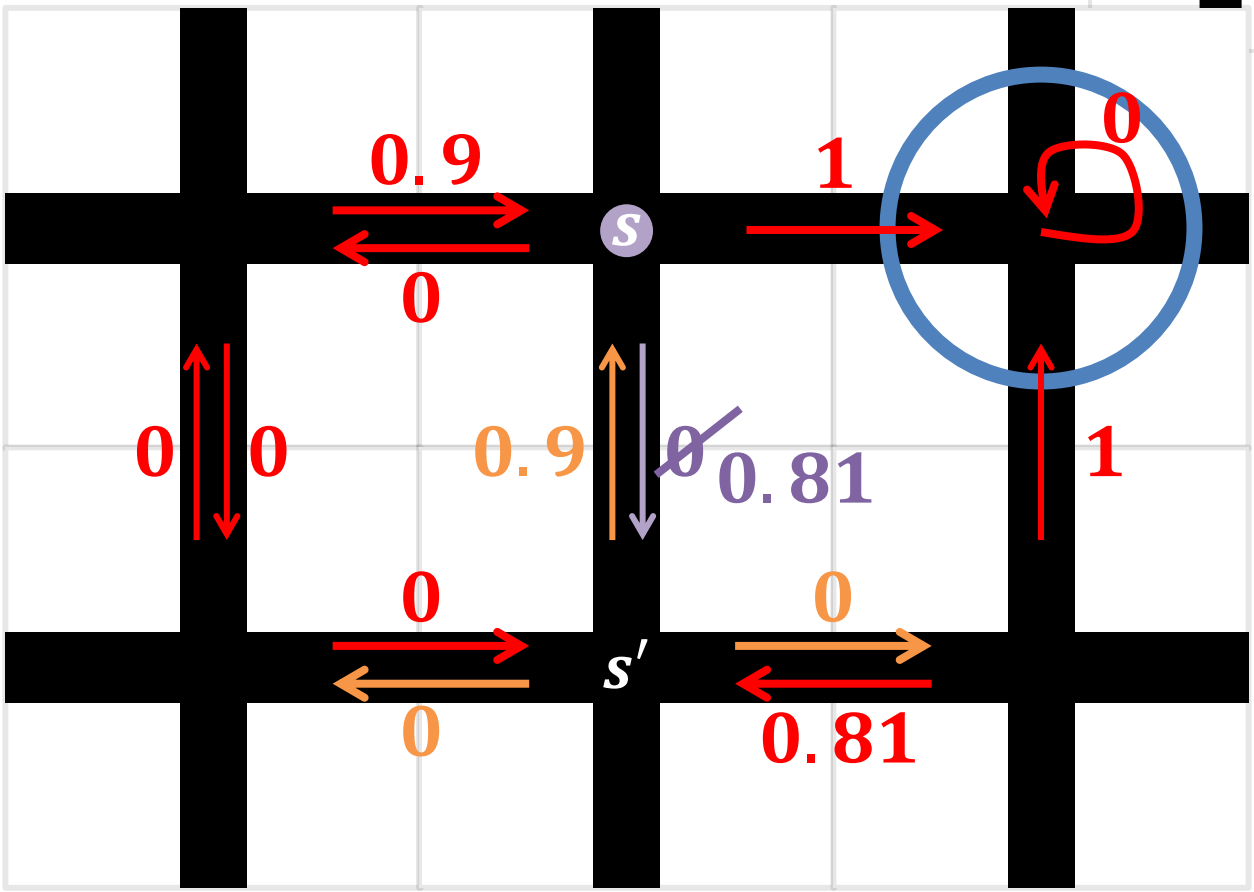


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 0.9

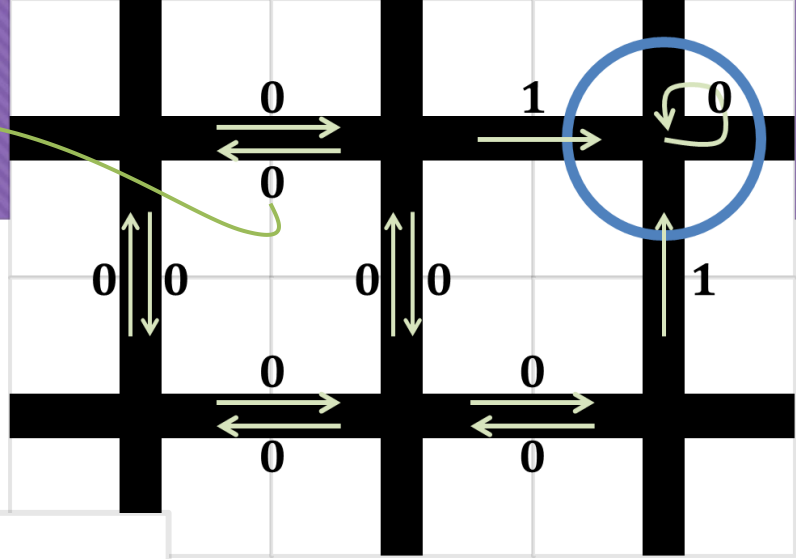


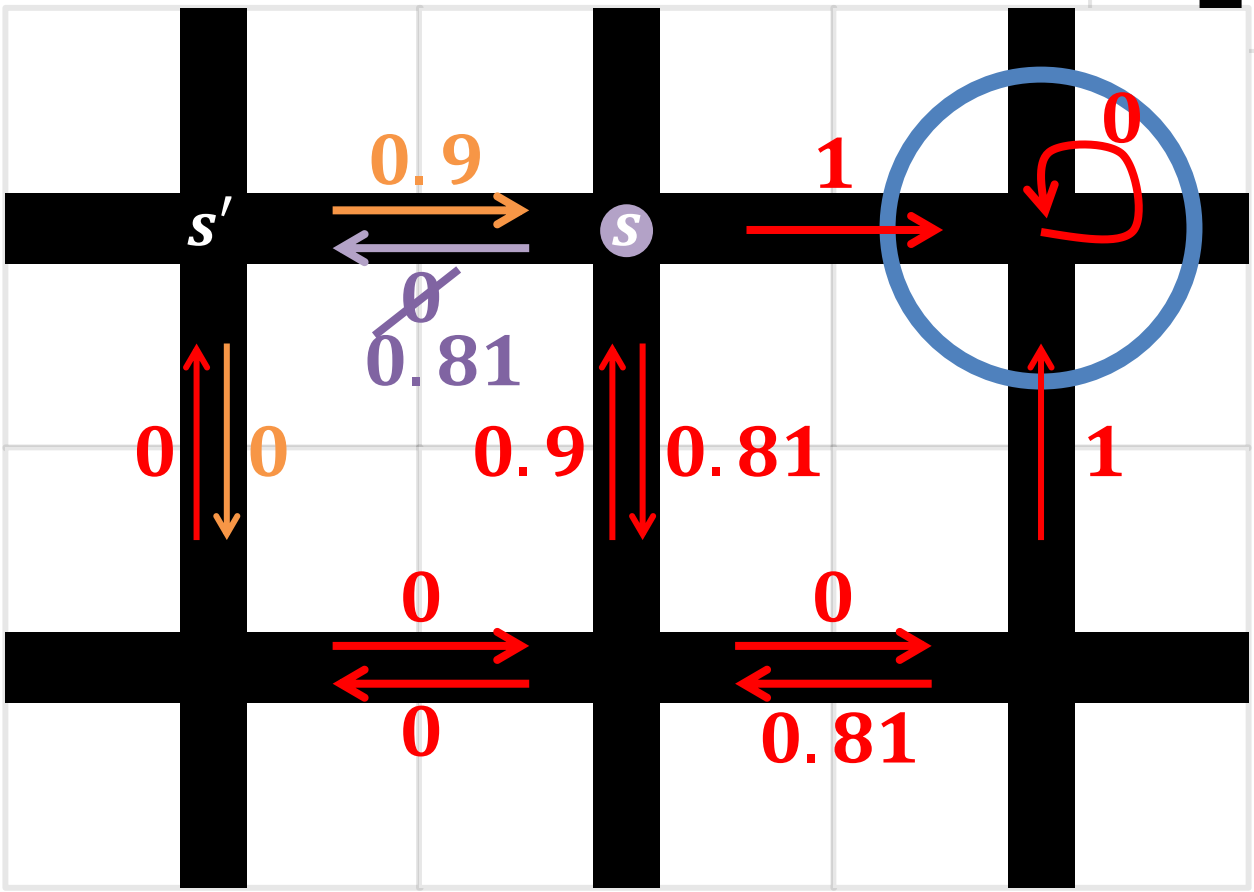


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 0.9

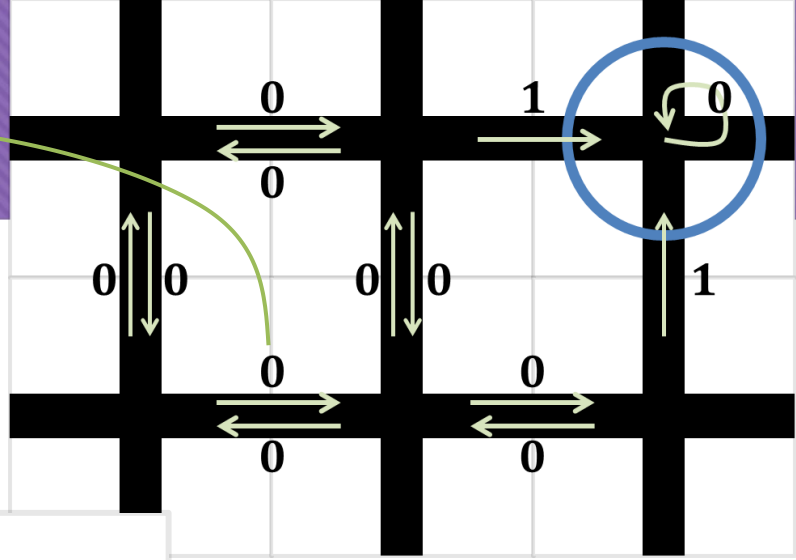


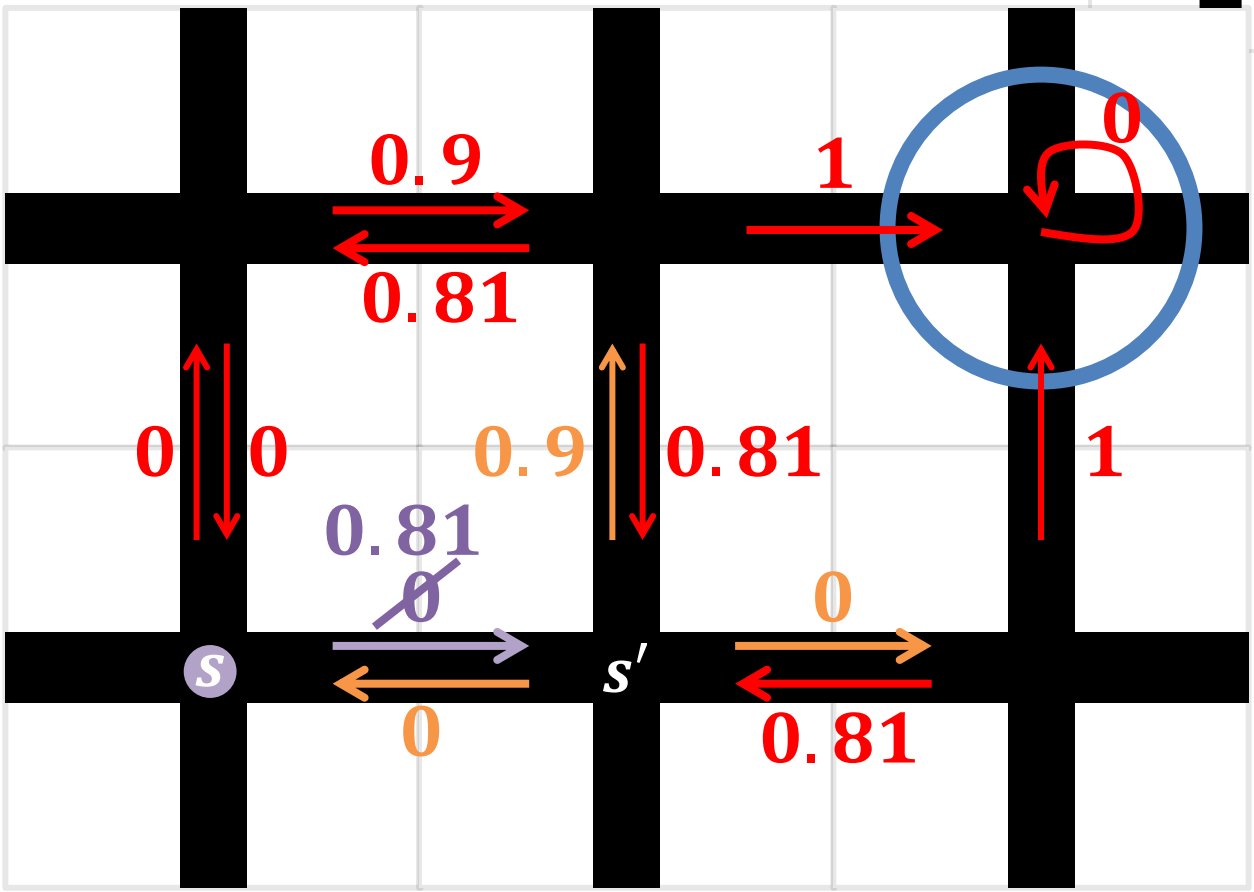


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 0.9

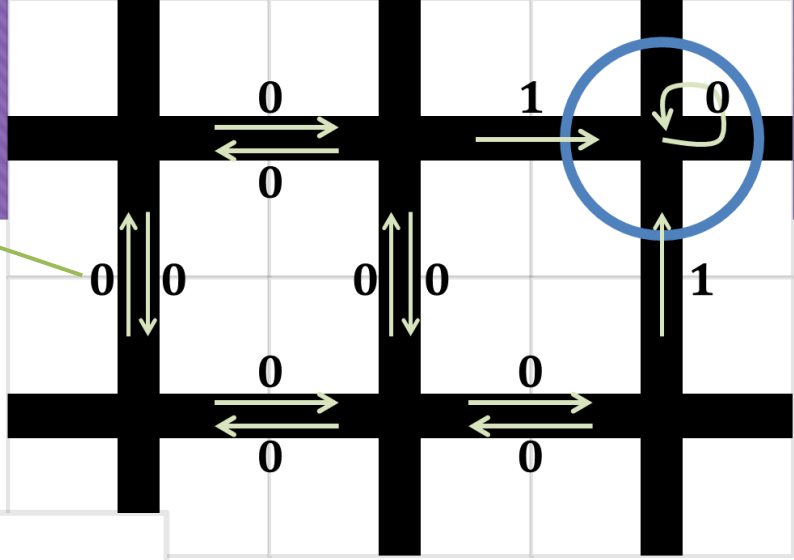


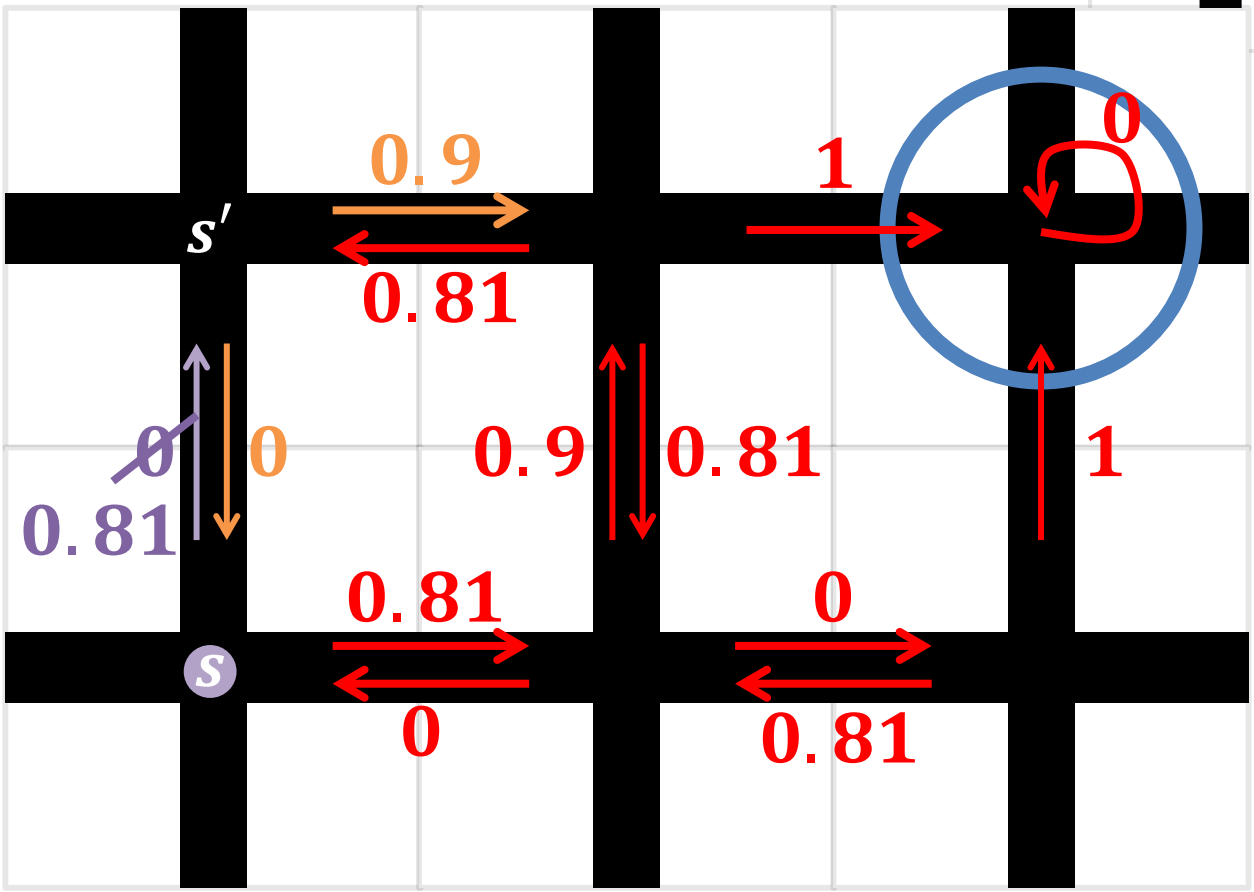


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 0.9

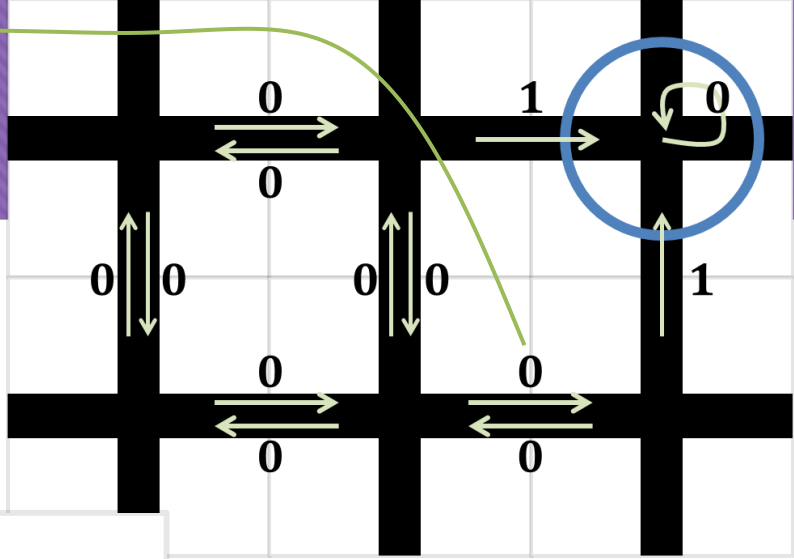


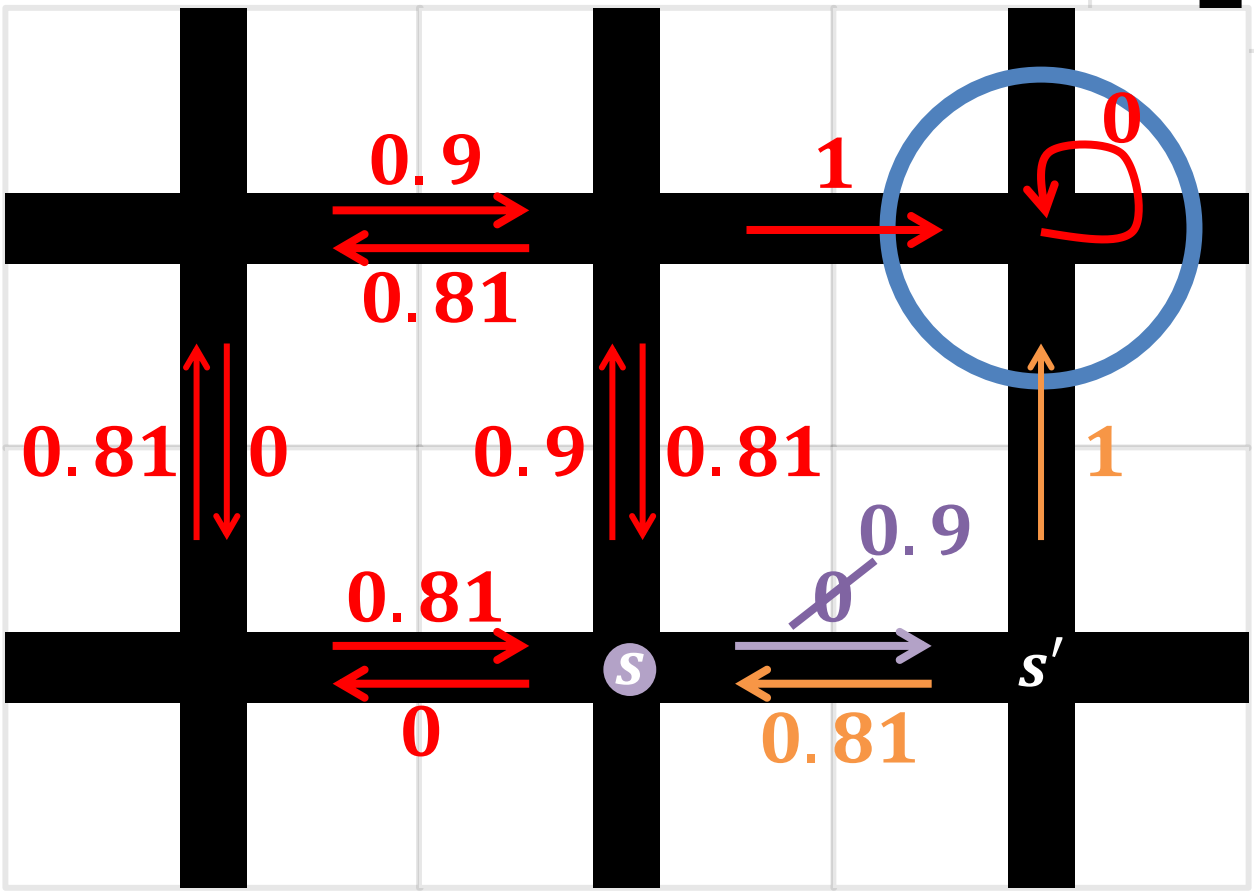


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 1

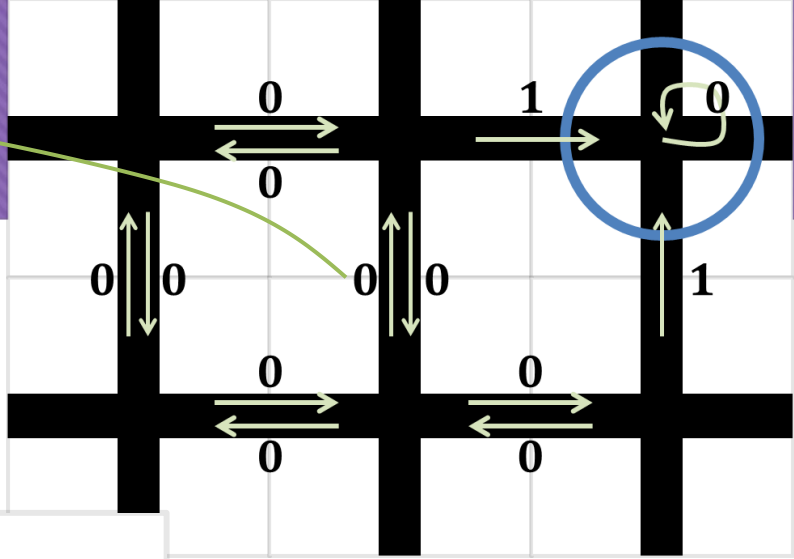


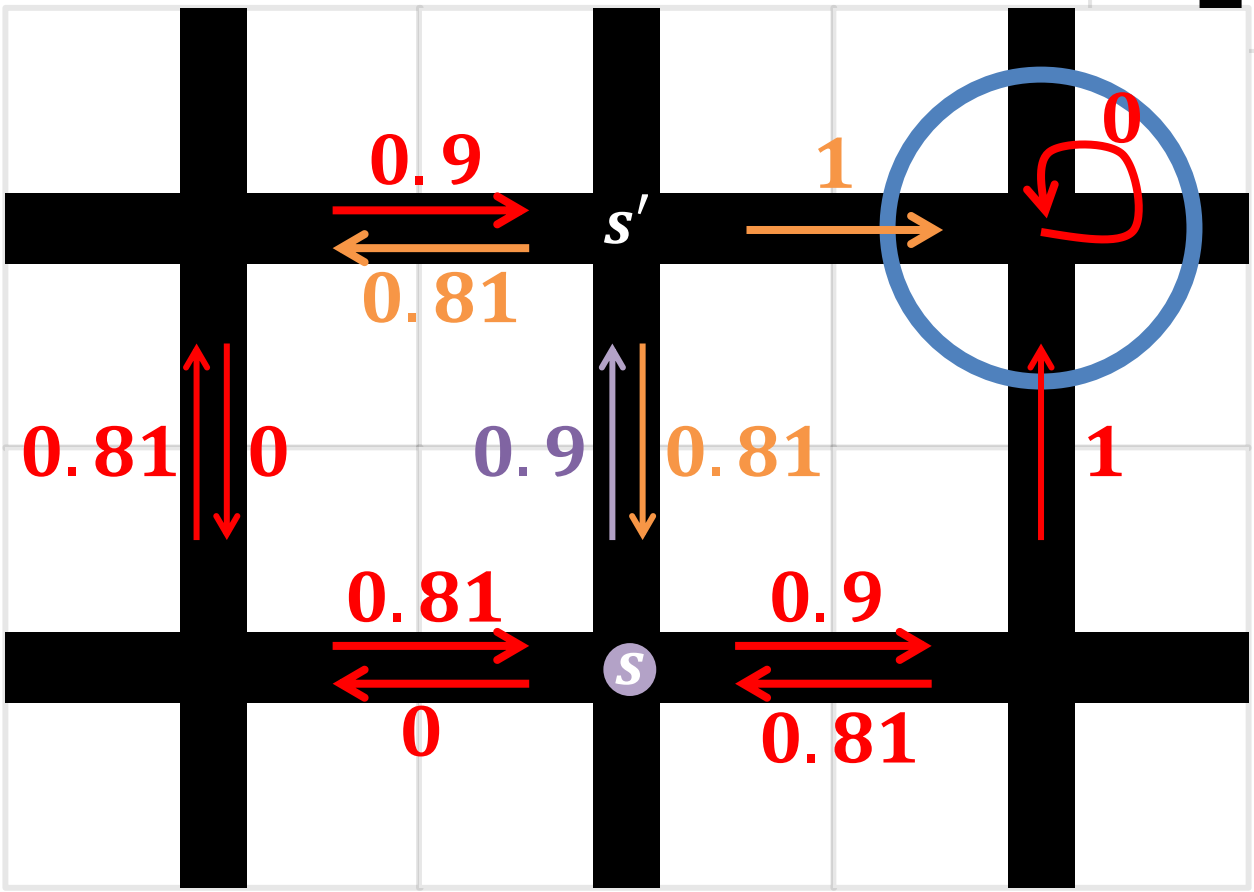


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 1

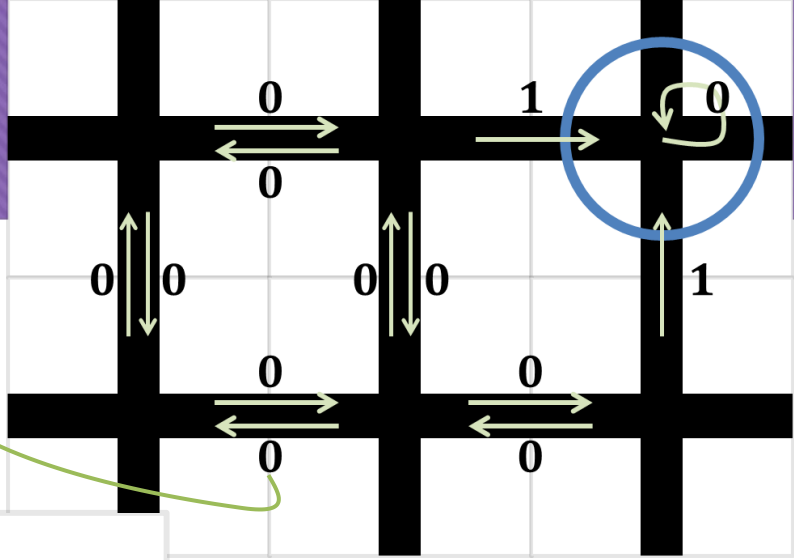


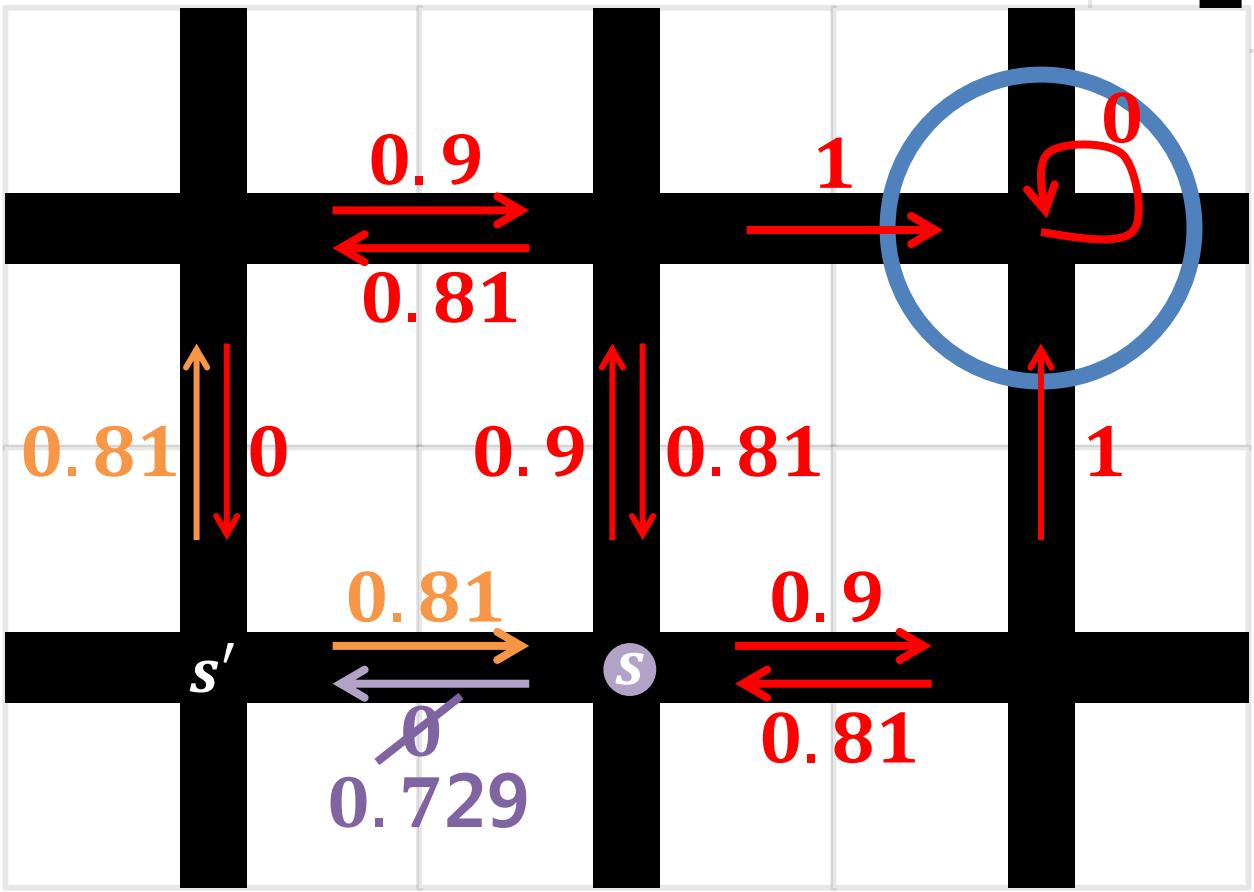


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 0.81

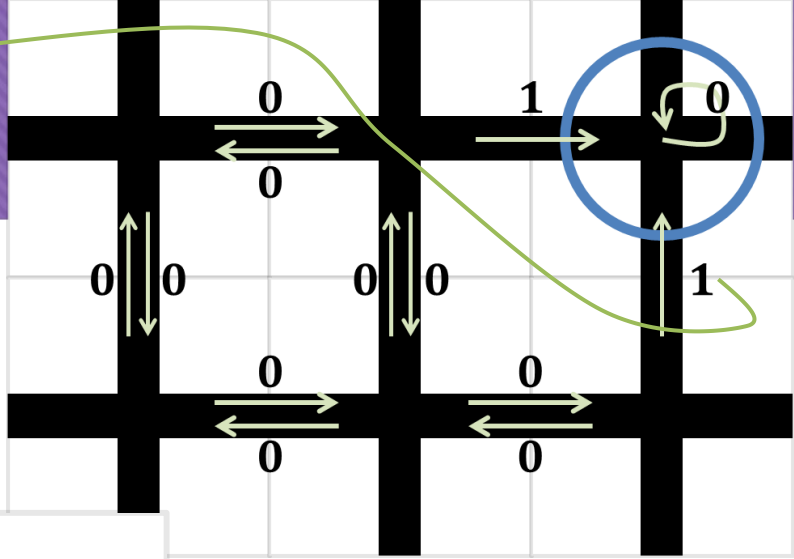


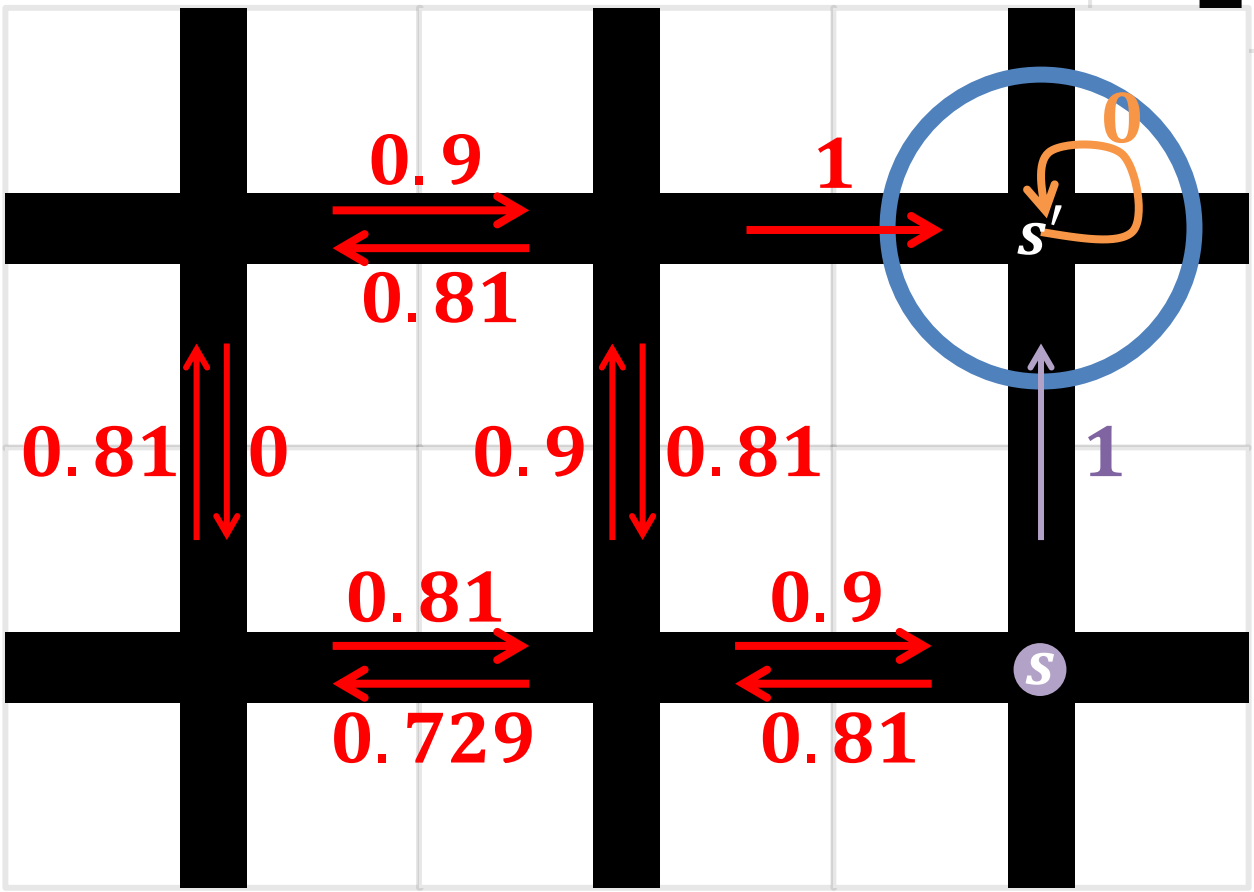


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

1 0

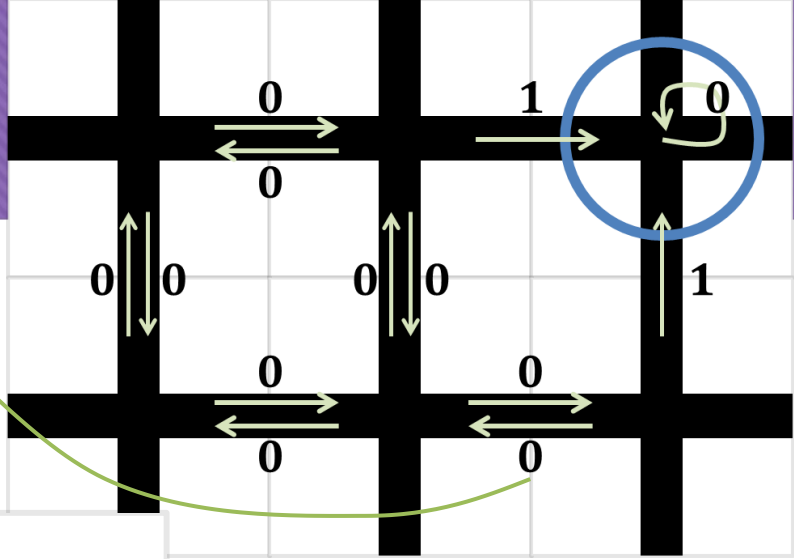


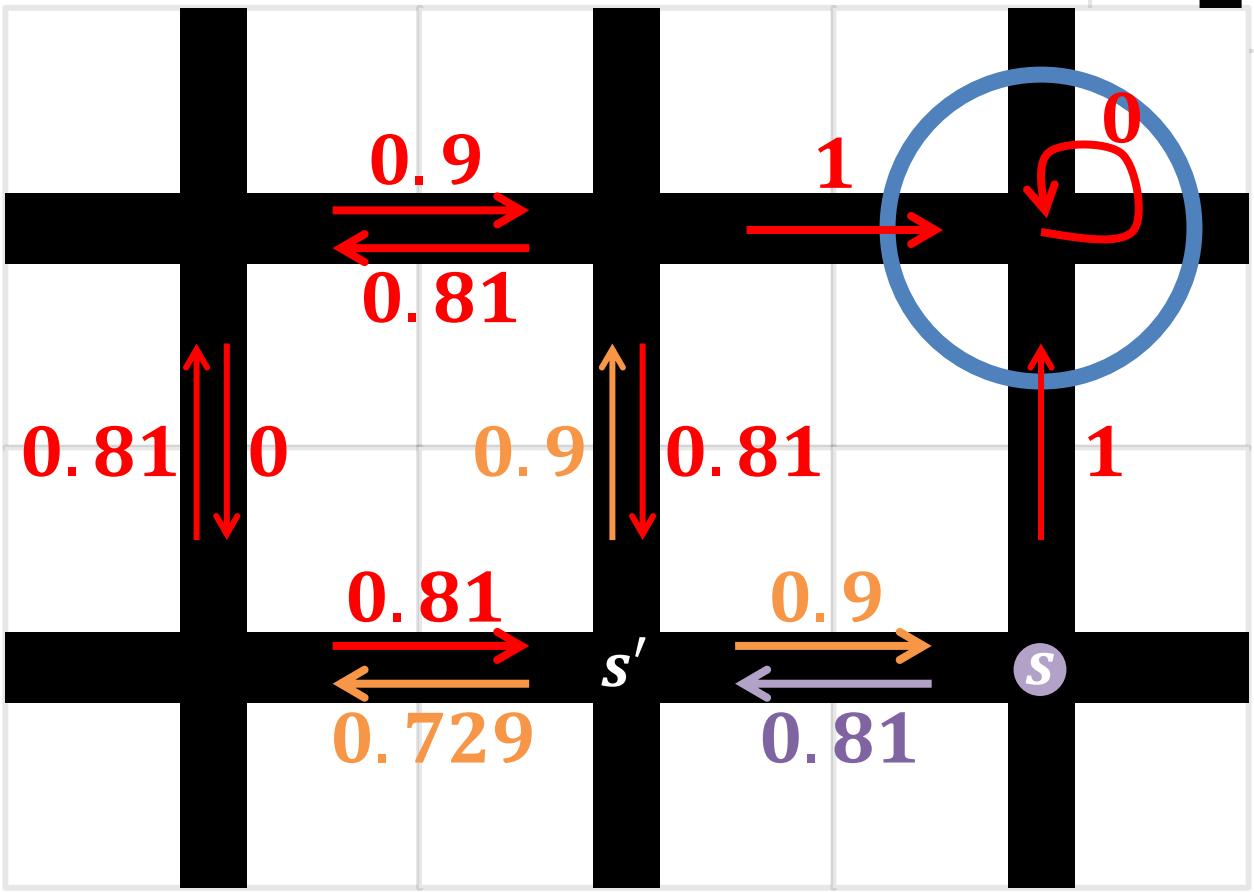


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 0.9

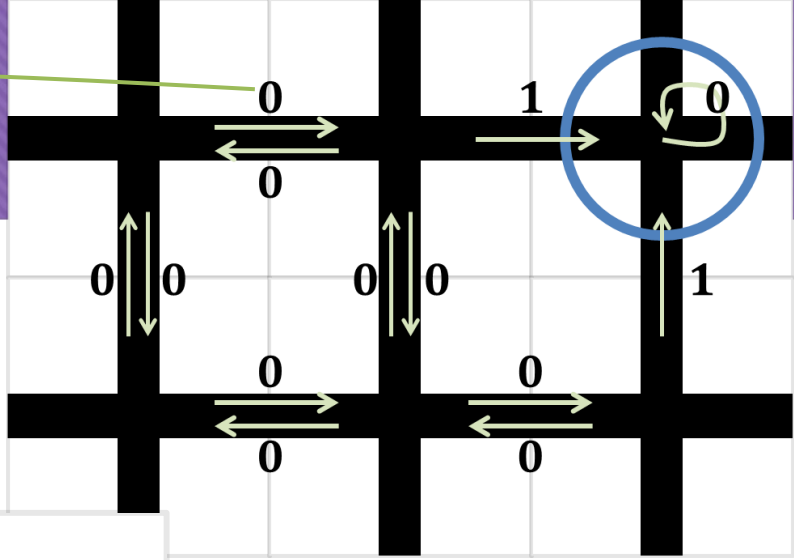


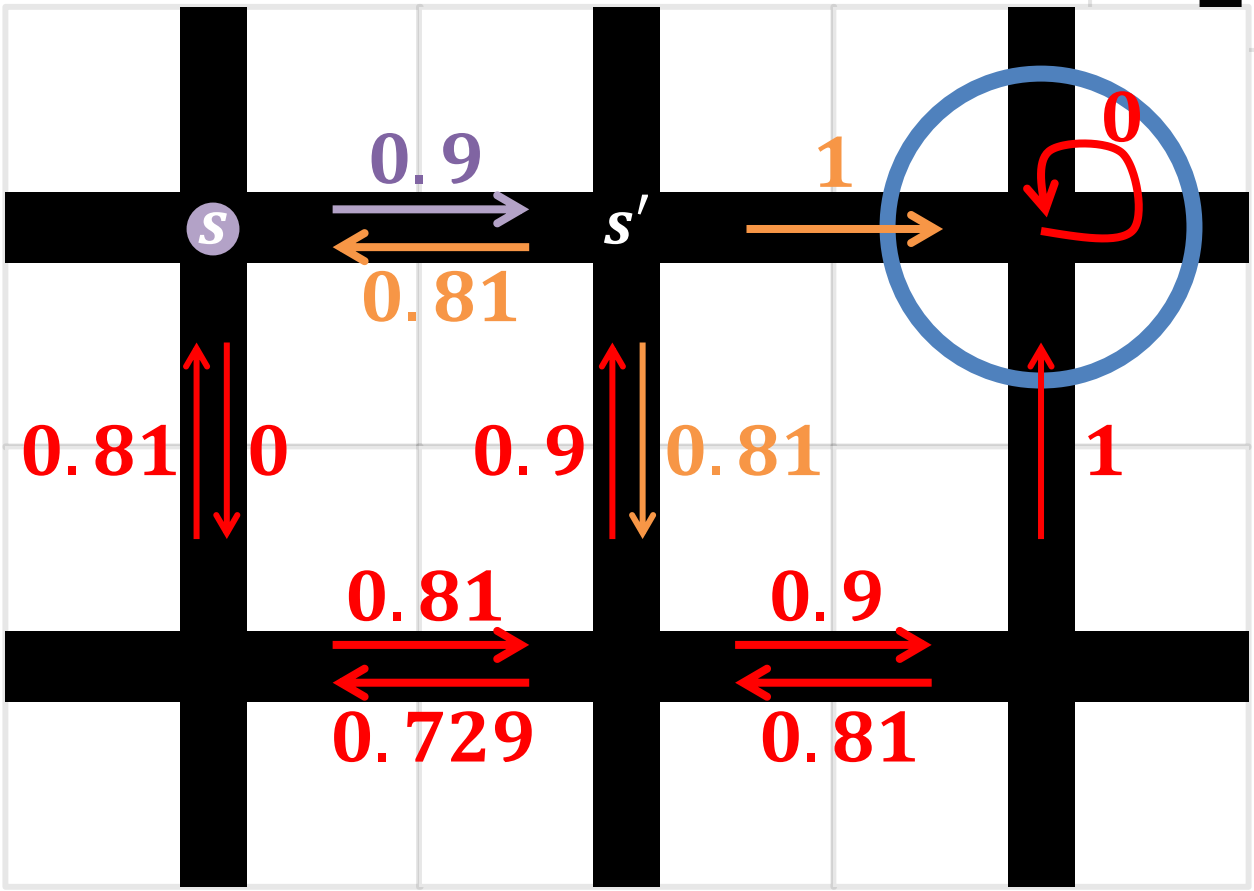


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 1

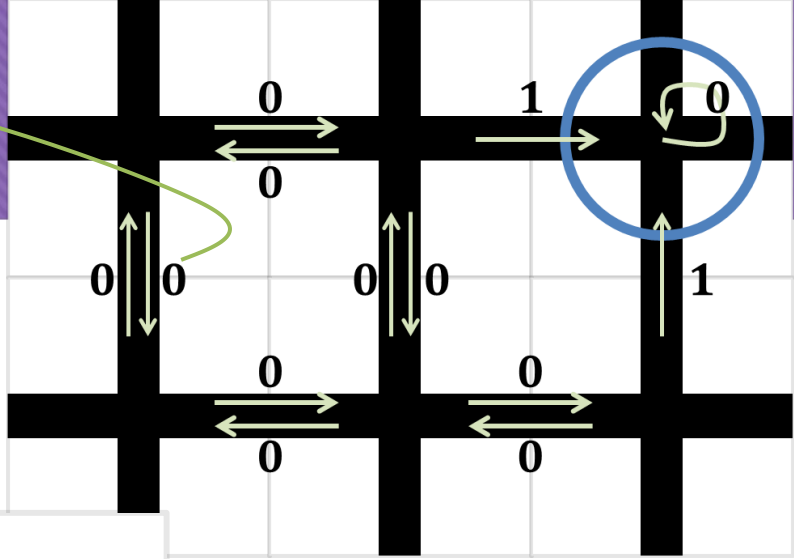


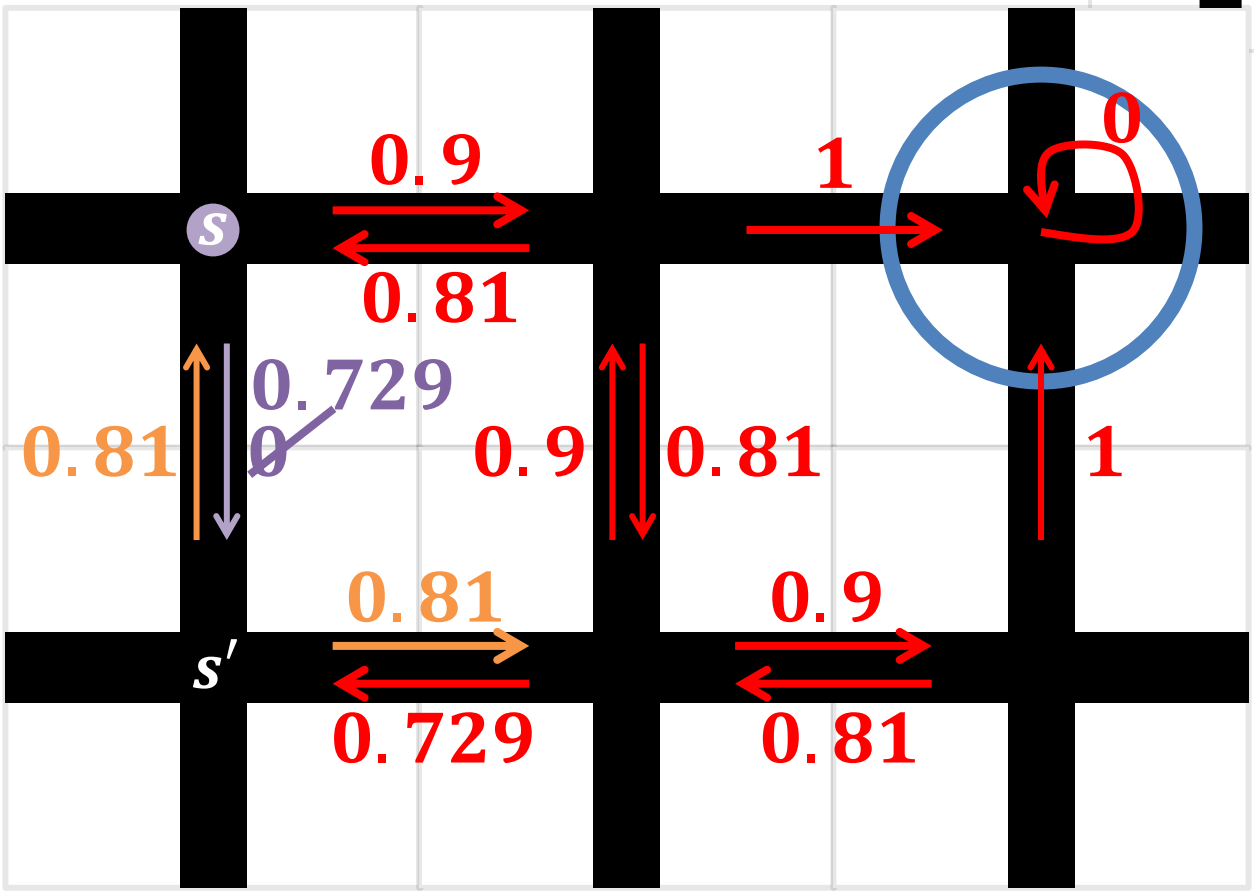


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 0.81

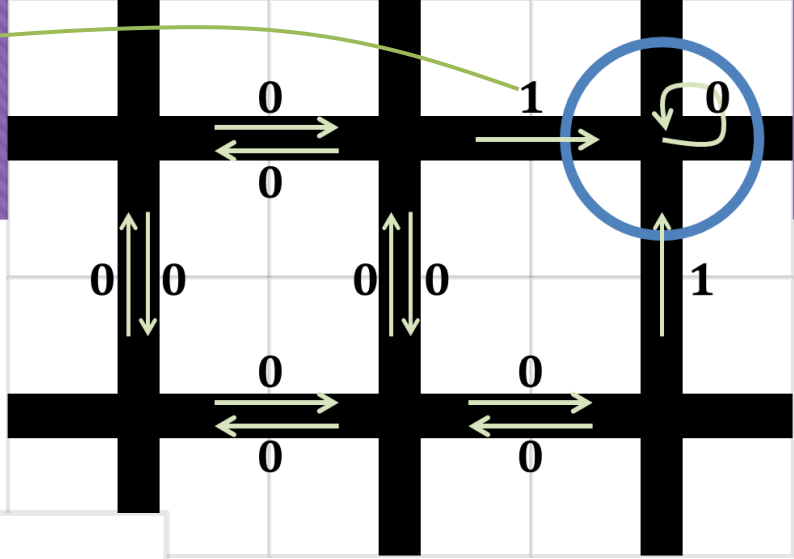


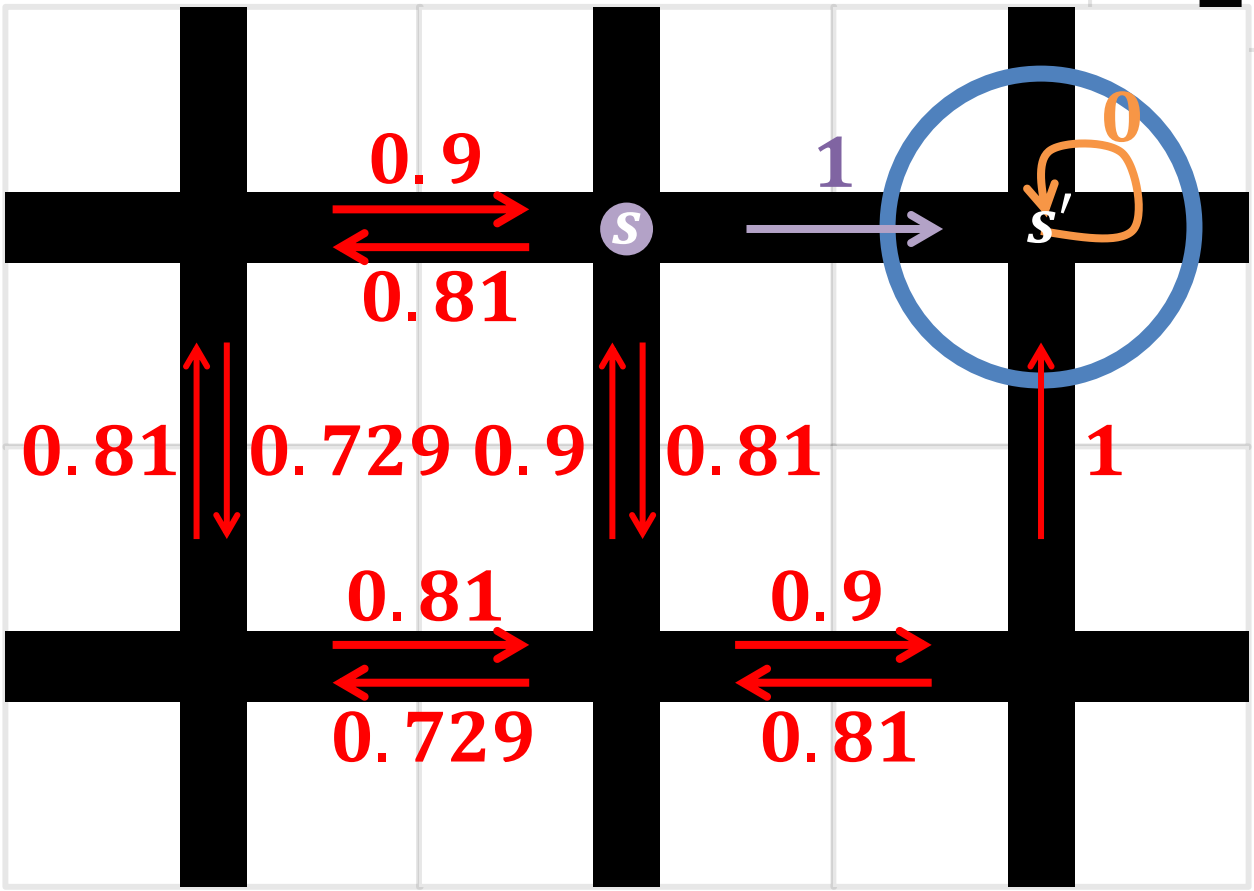


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

1 0

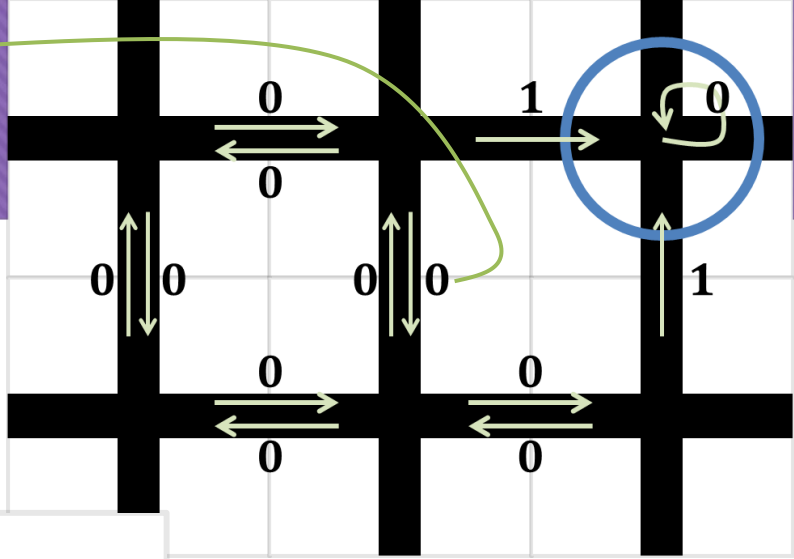


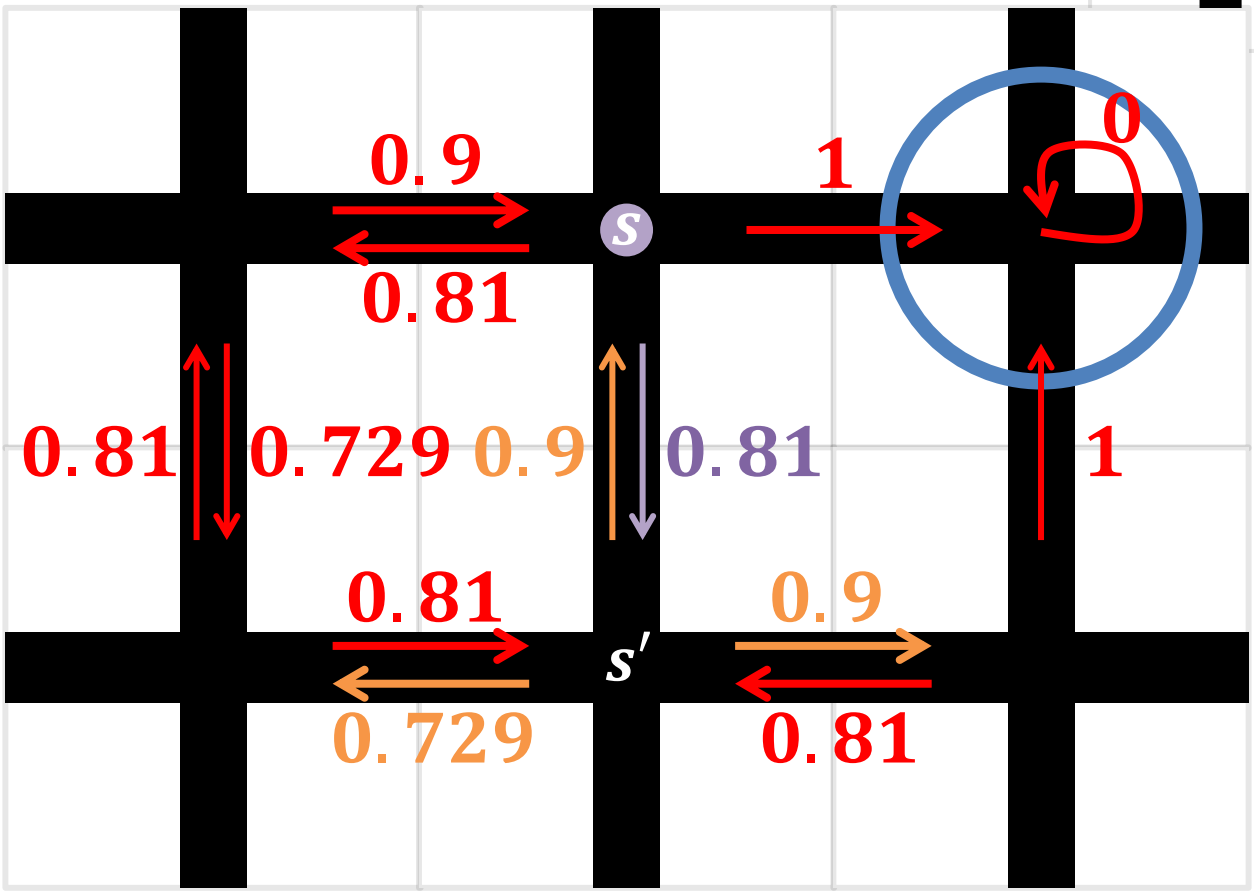


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 0.9

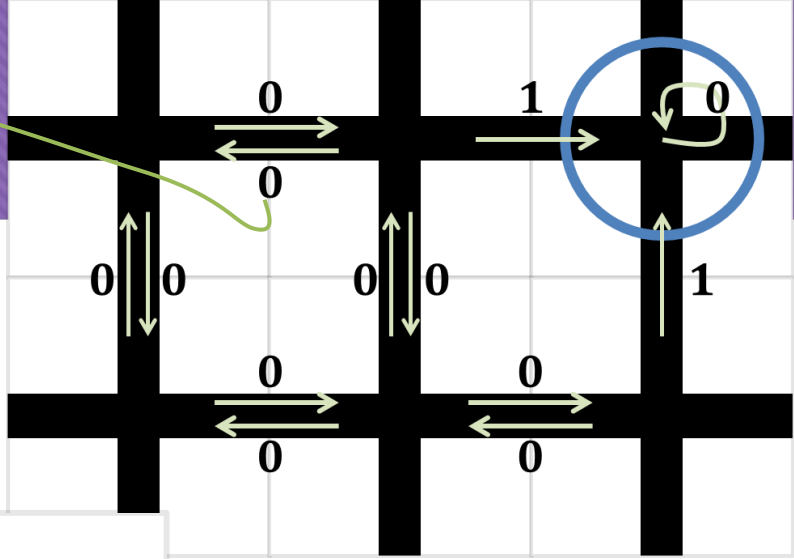


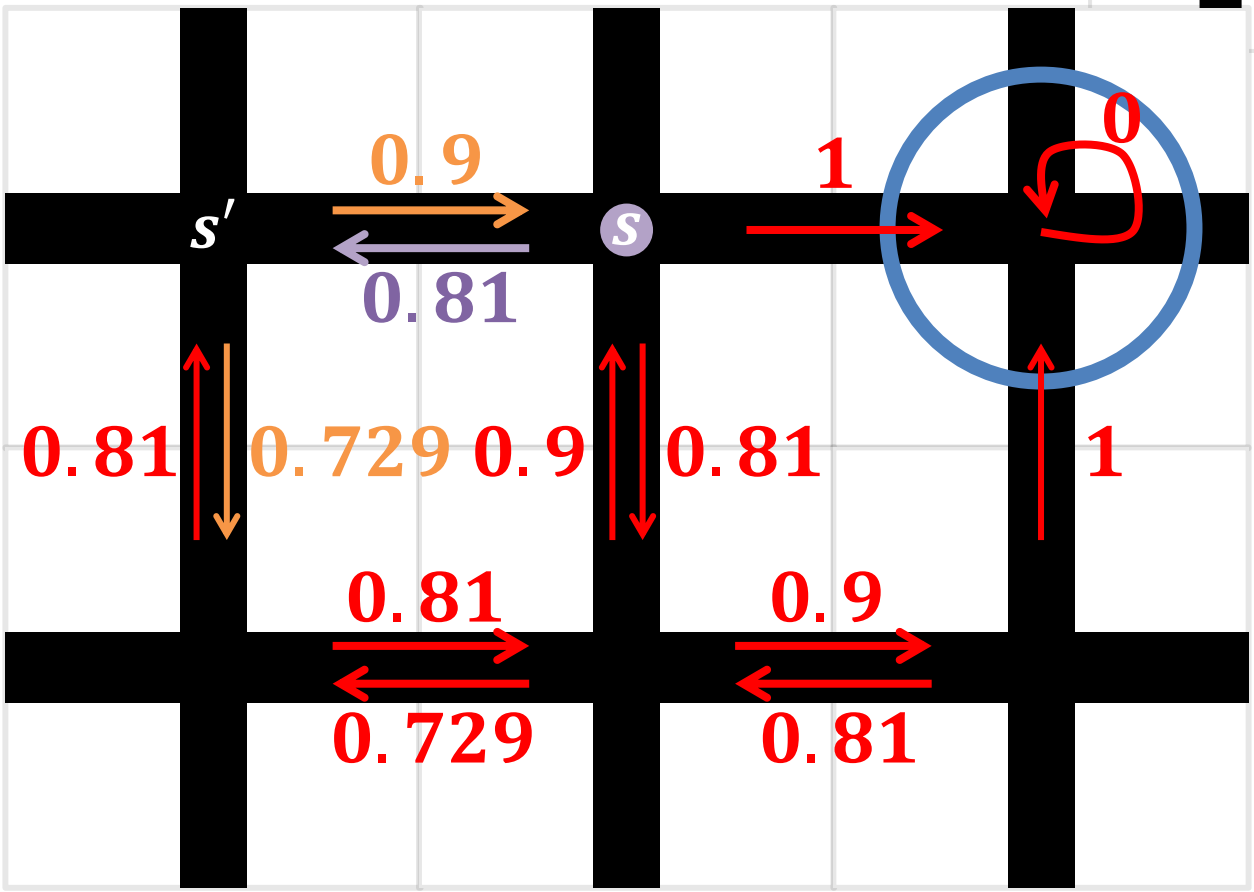


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 0.9

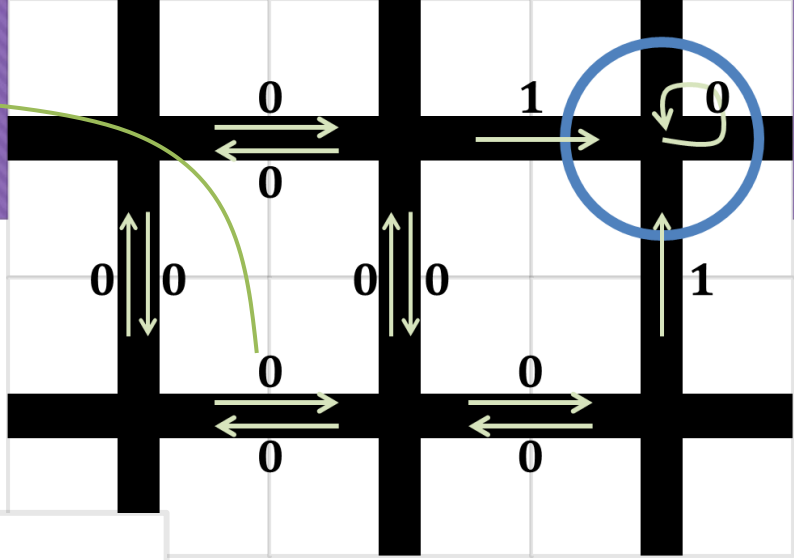


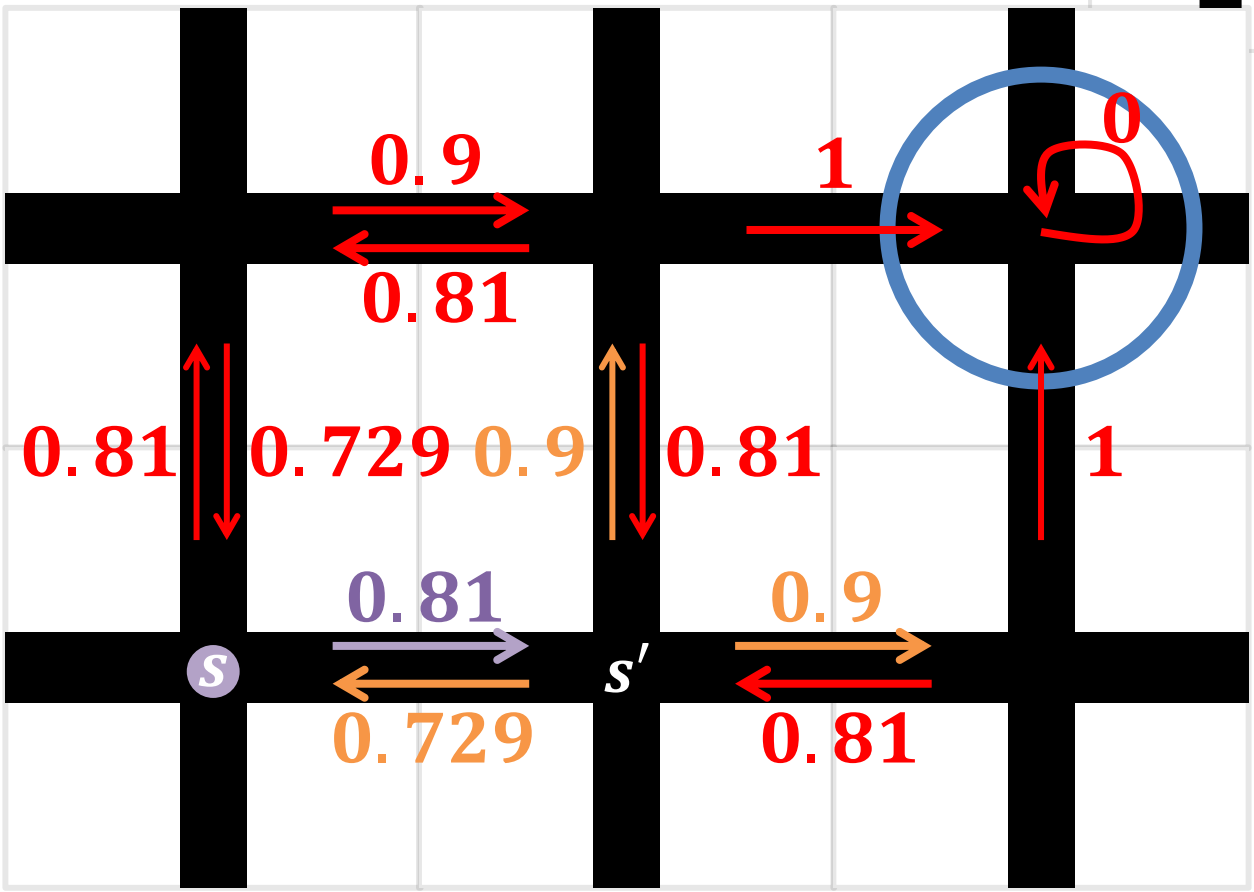


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 0.9

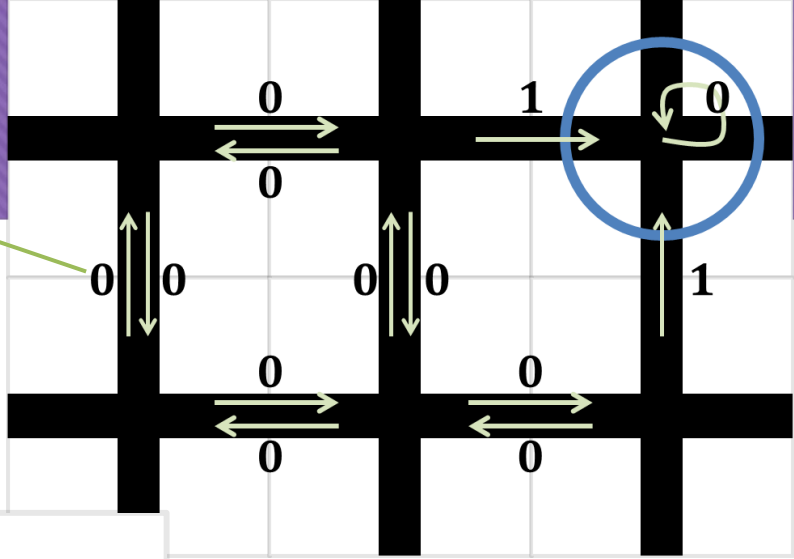


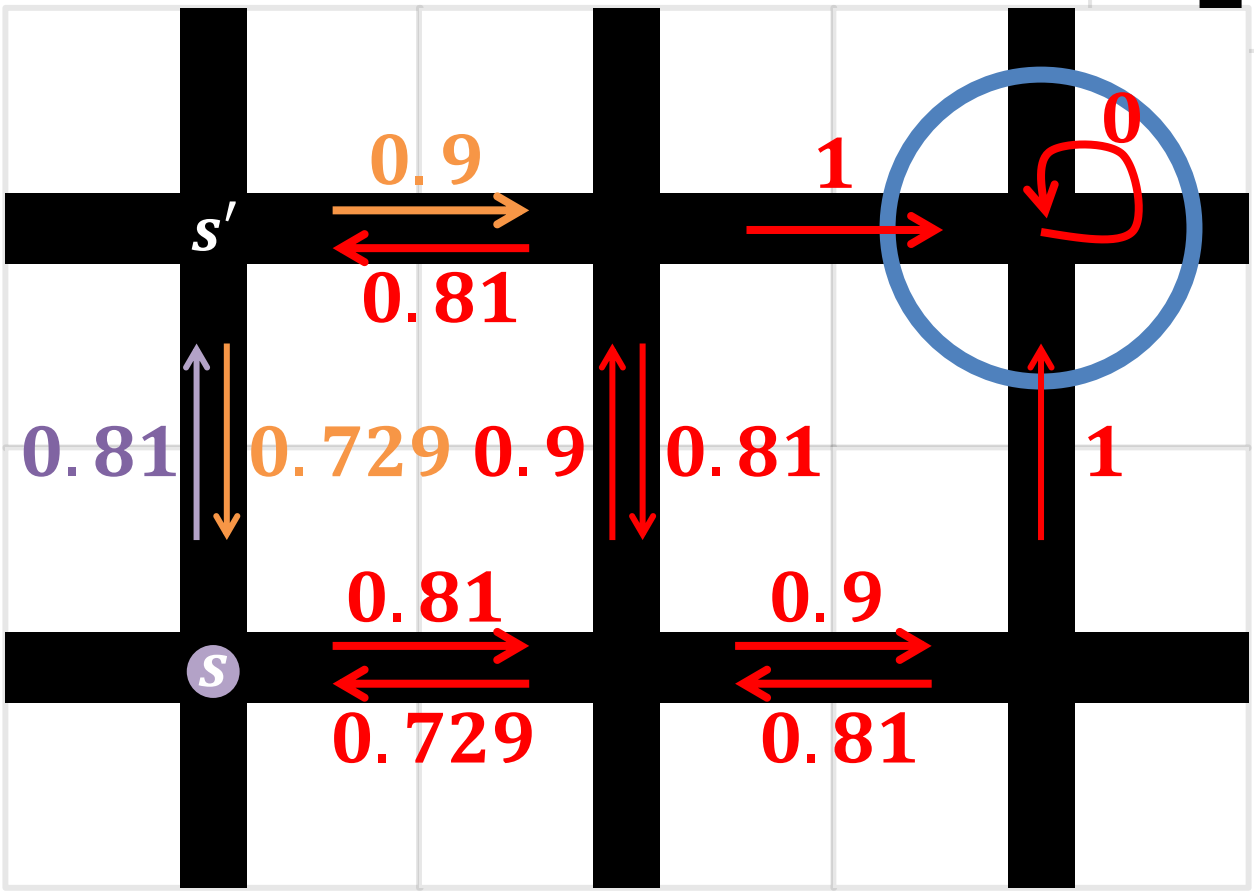


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 0.9

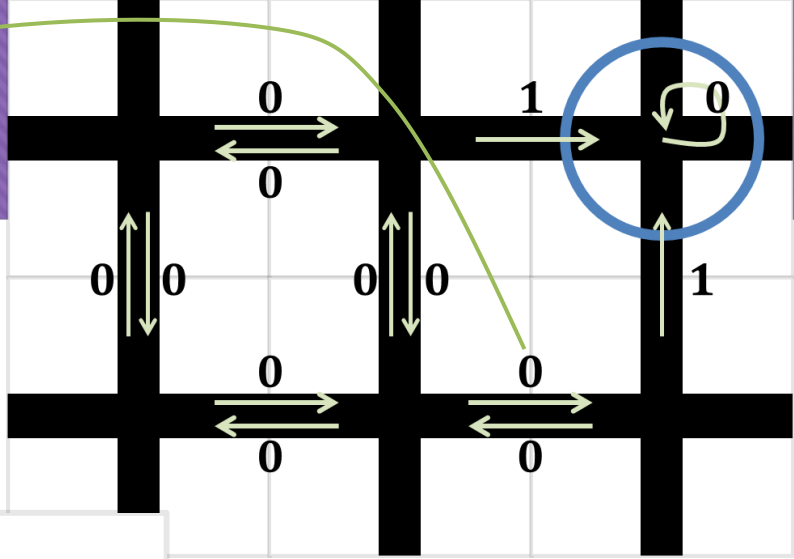


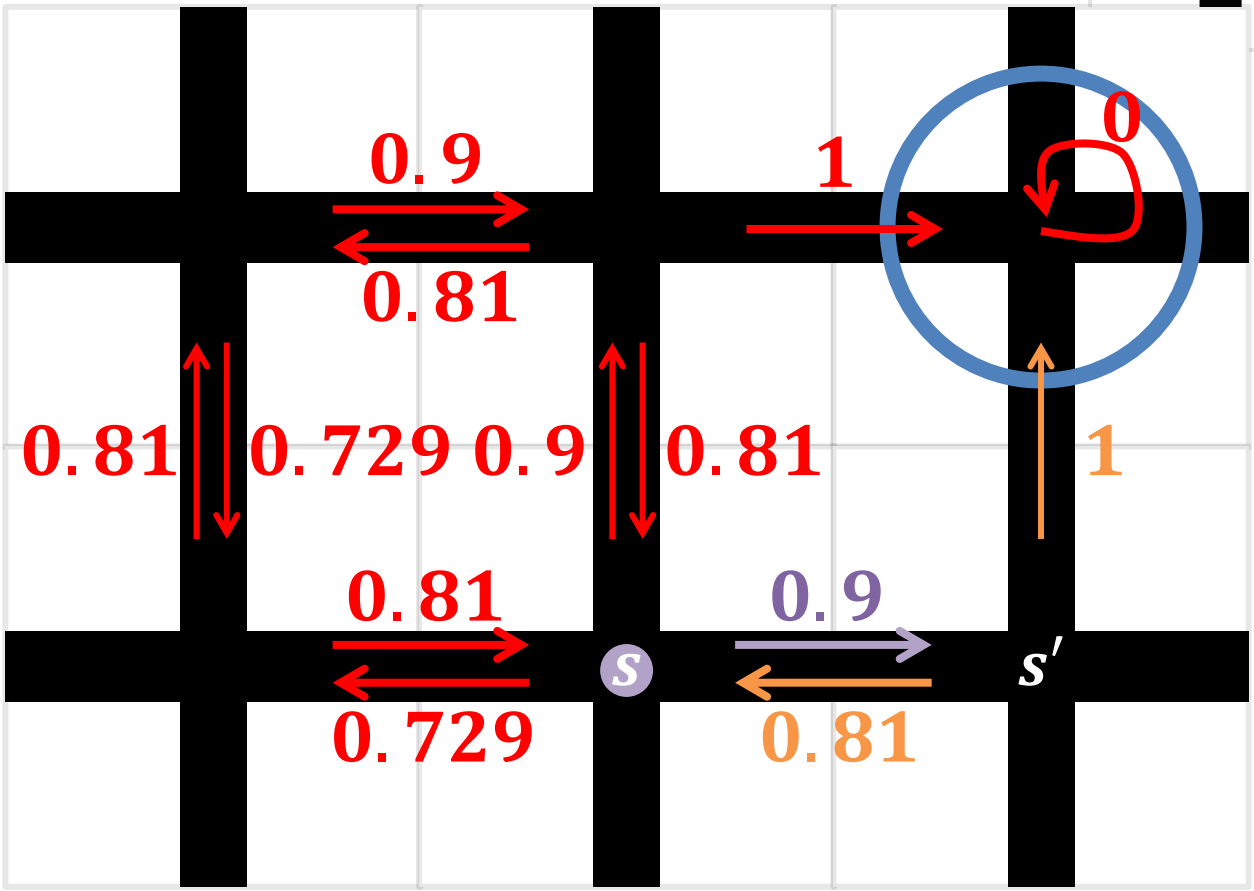


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 1

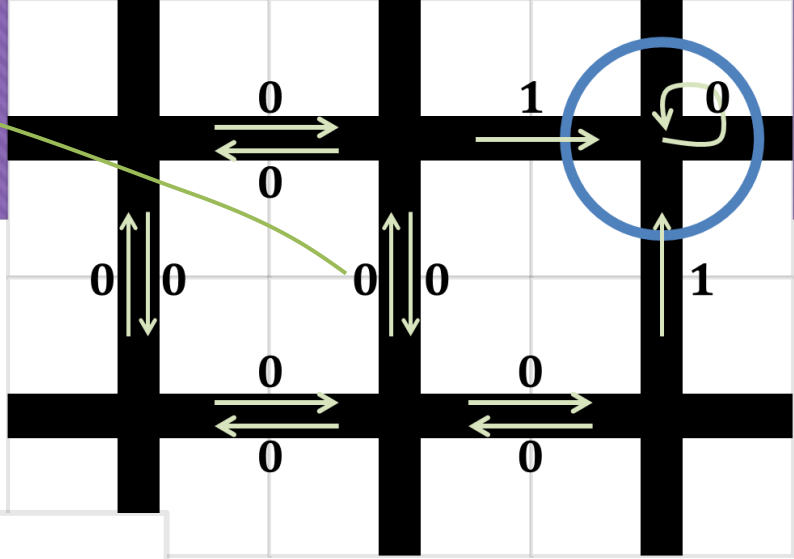


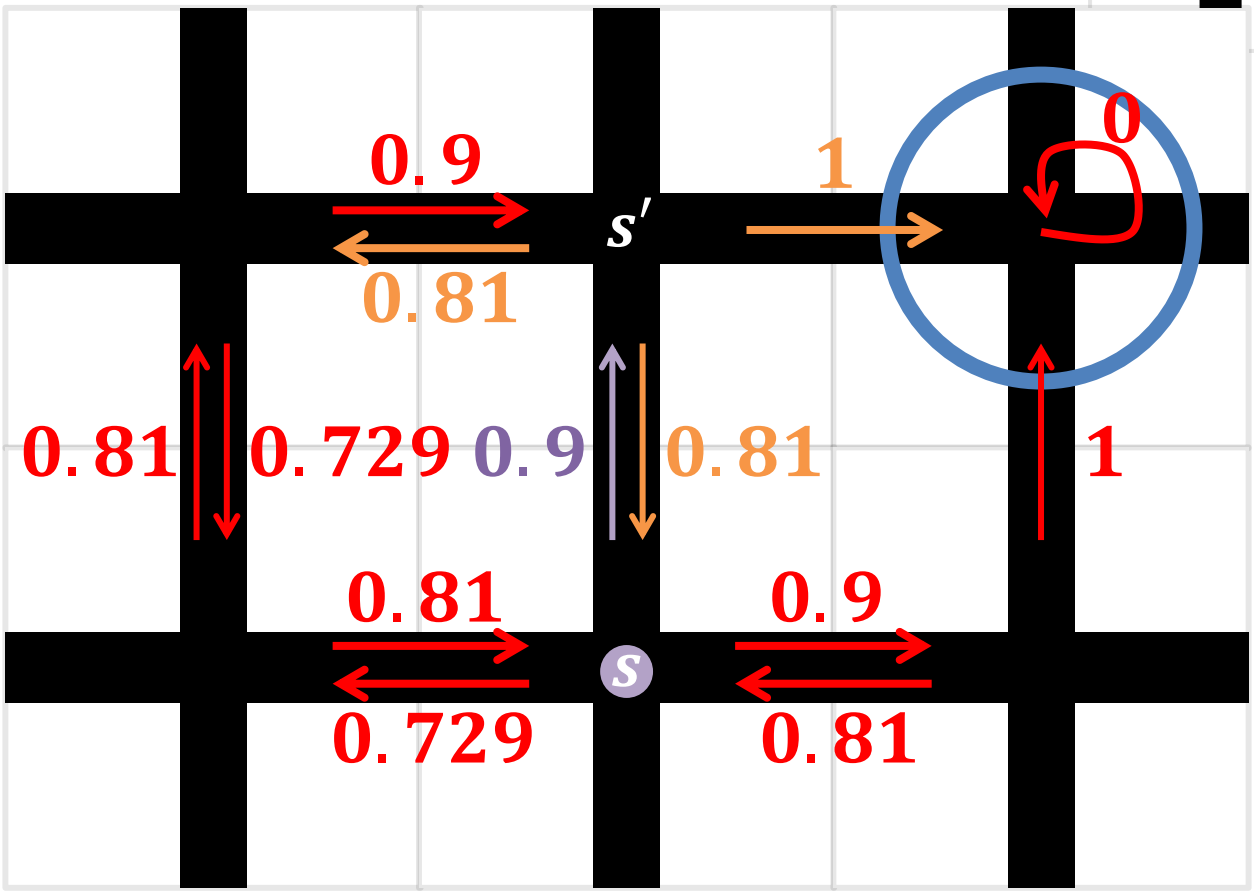


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 1

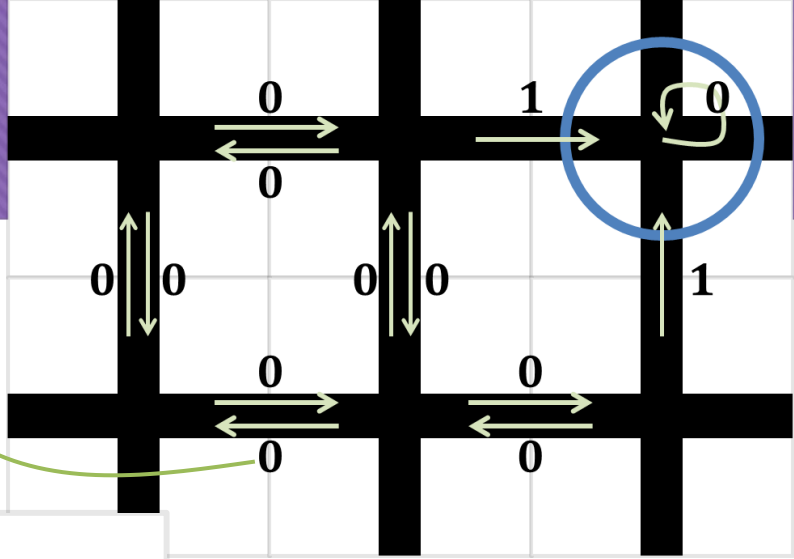


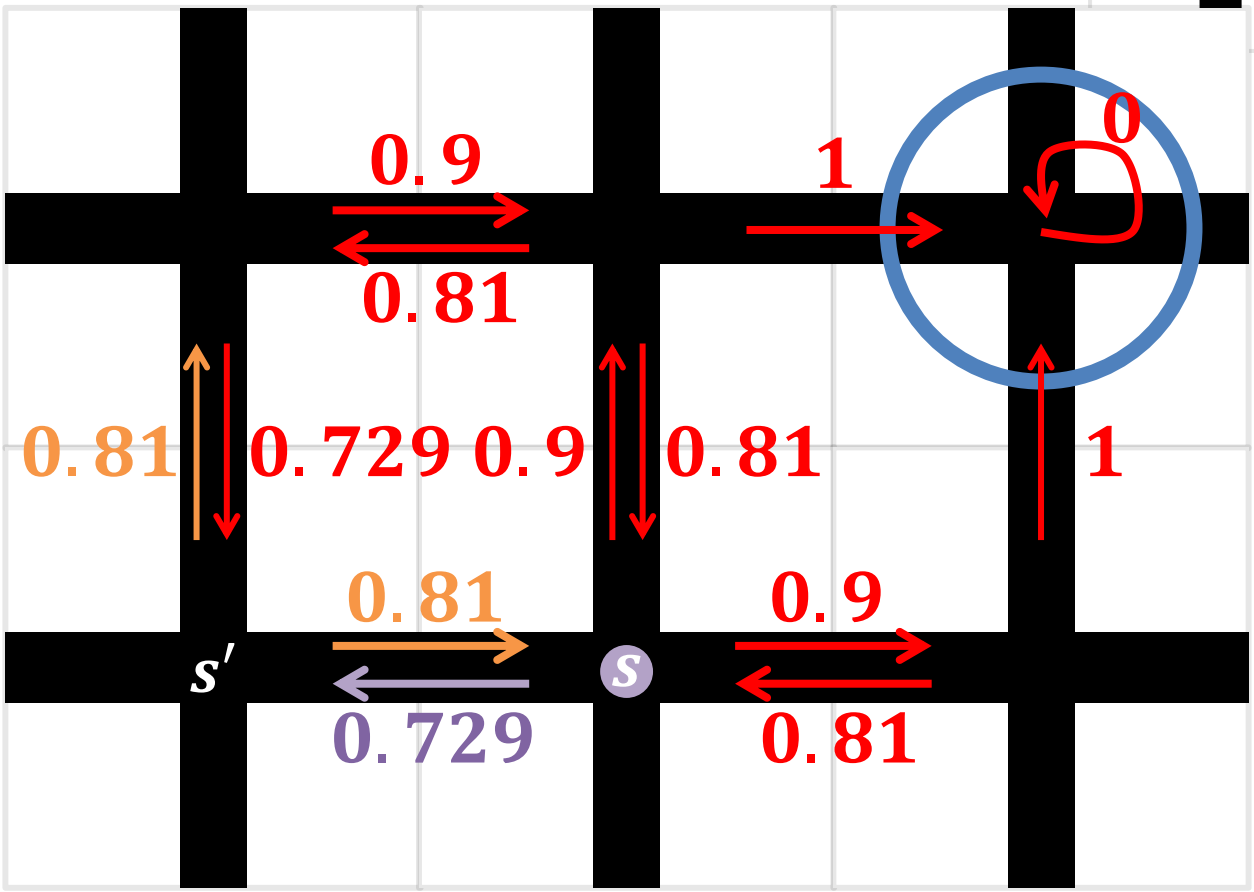


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

0 0.81

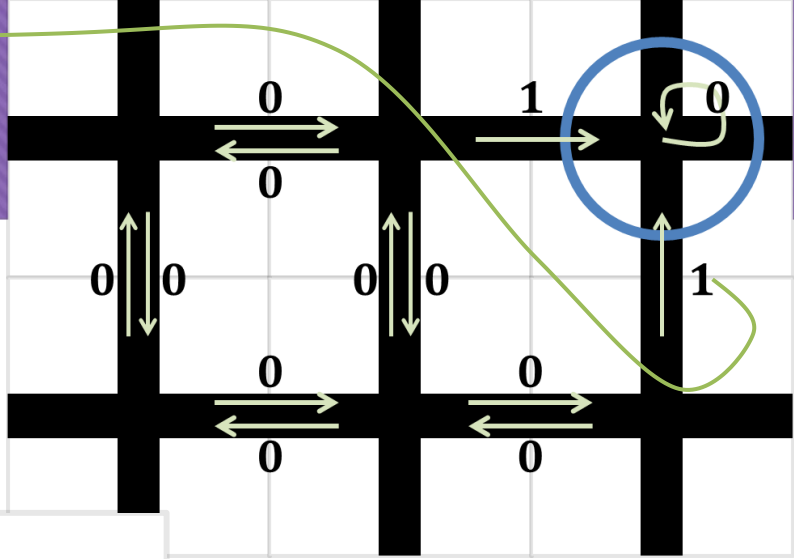


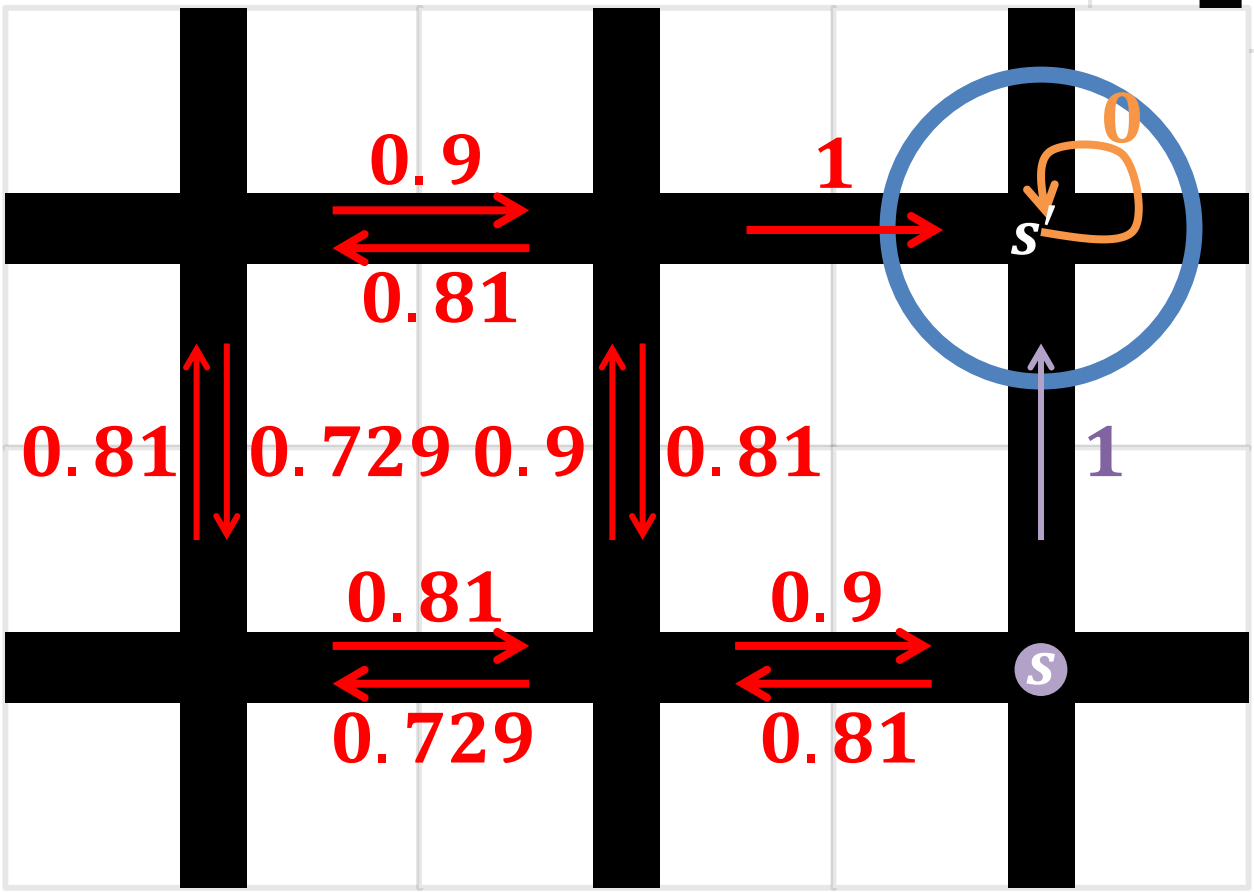


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

1 0

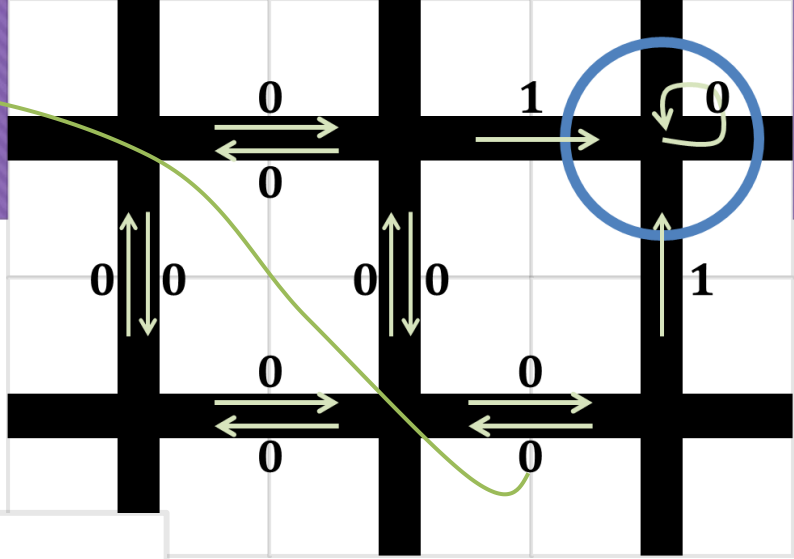


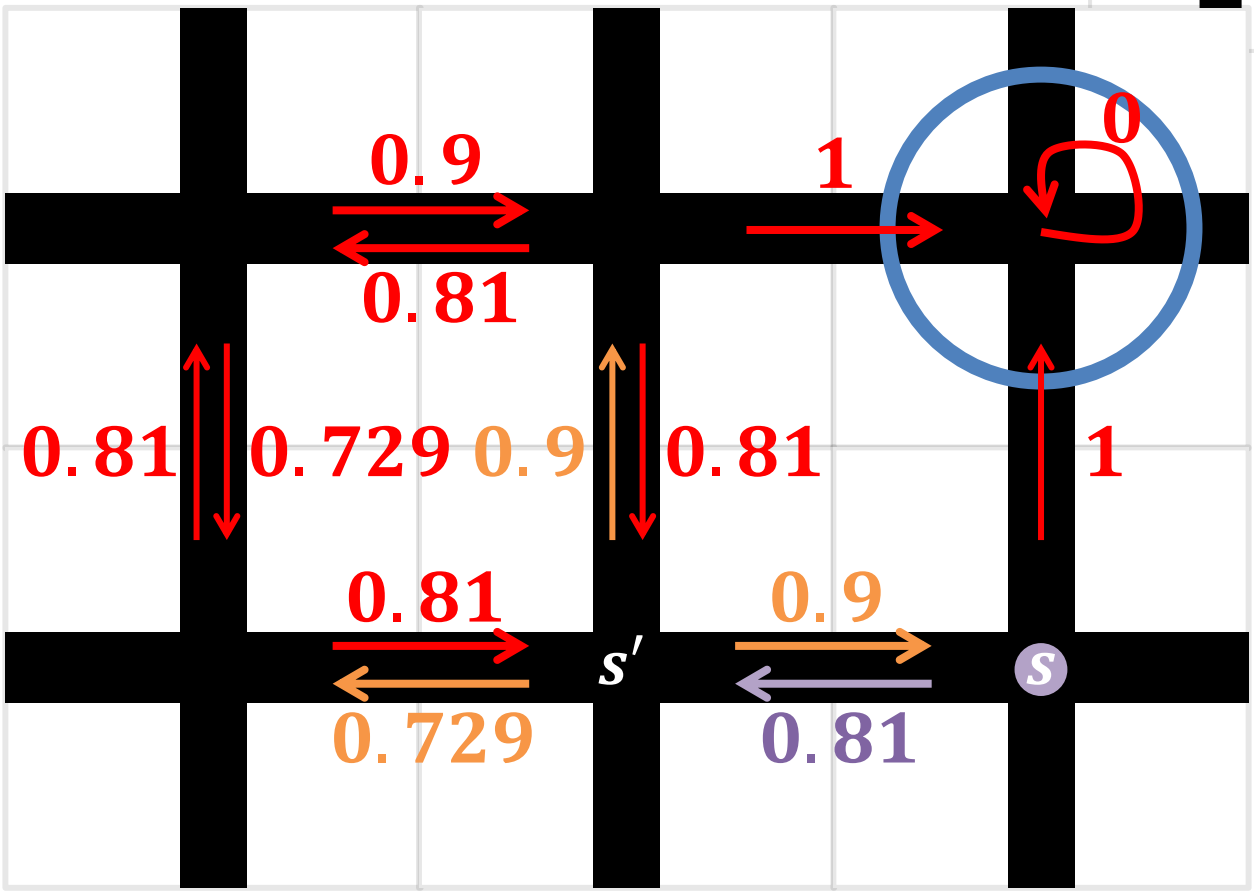


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

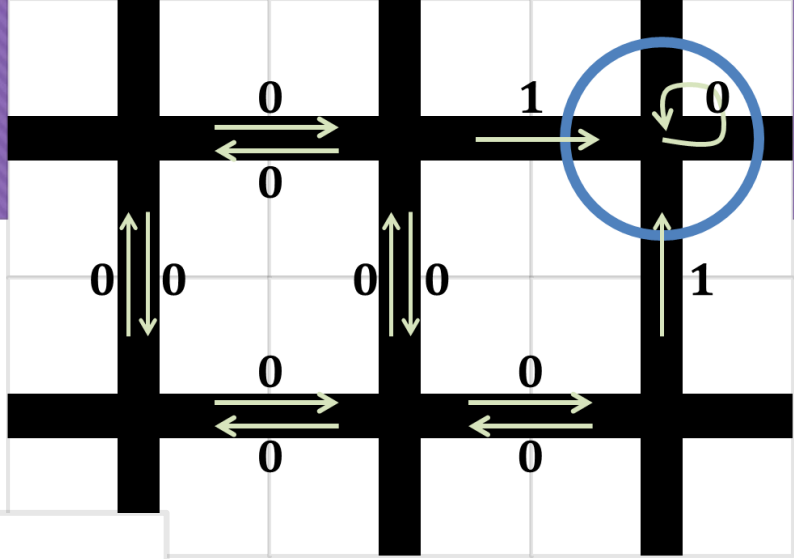
0 0.9

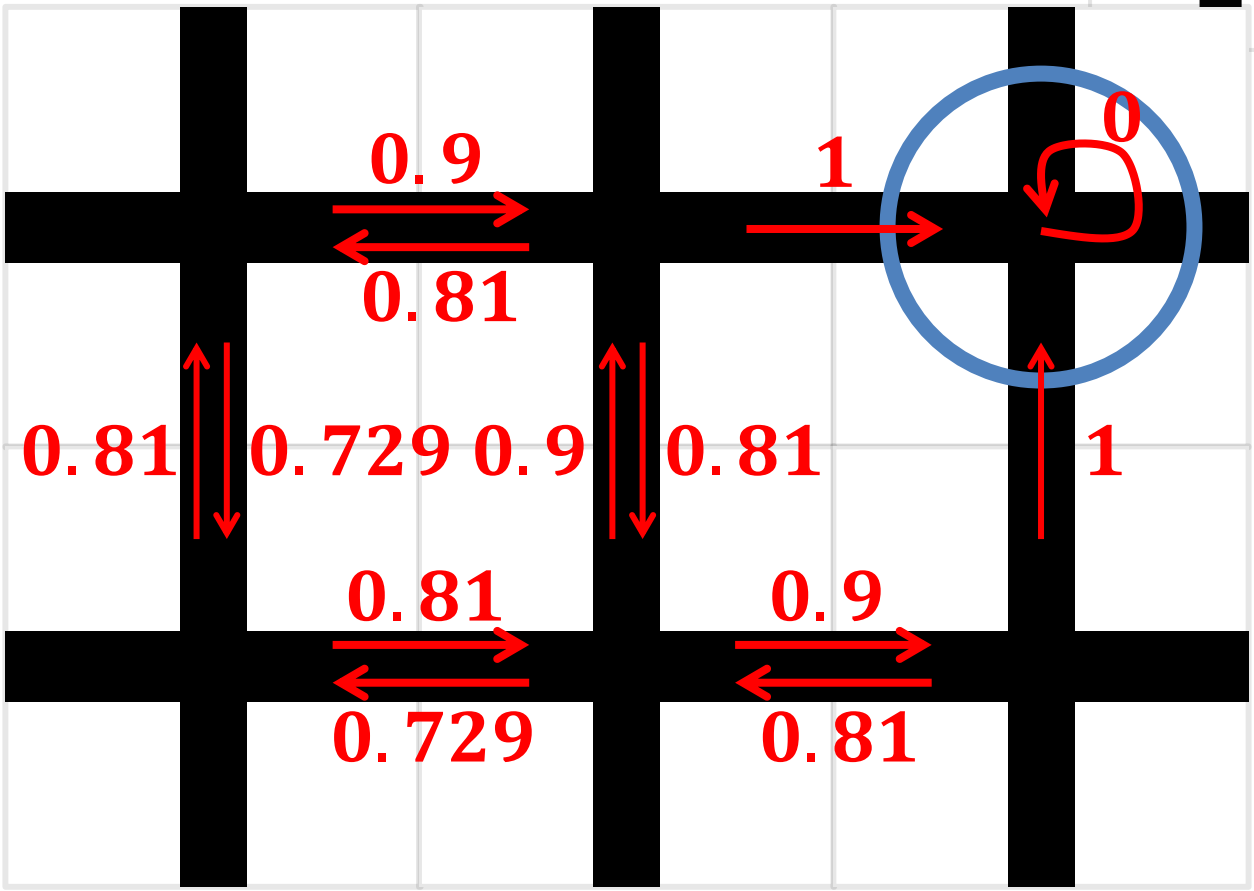




Q-러닝

변화가 없을 때까지 계속 반복



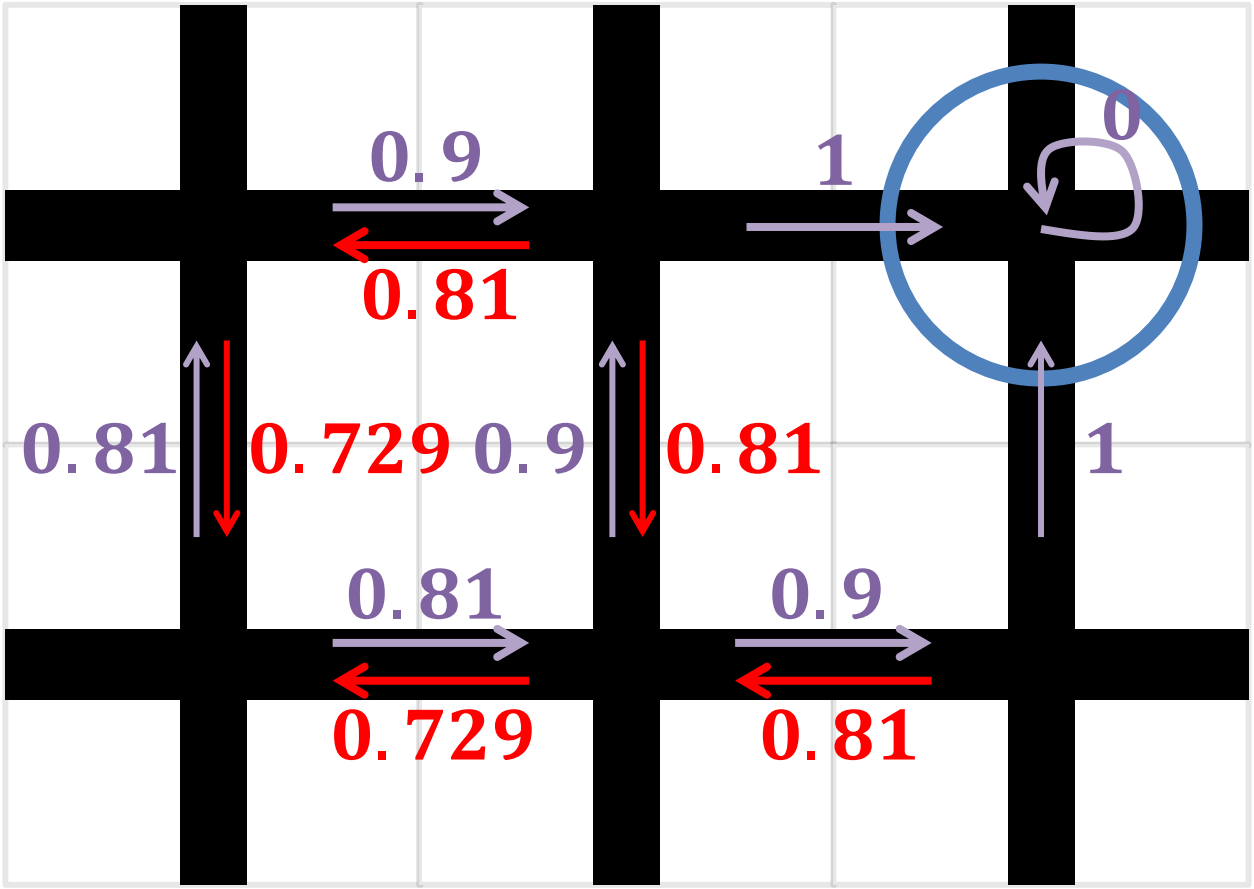


Q-러닝

$$\pi^*(s) = \underset{a}{\operatorname{argmax}}\, Q(s,a)$$

최적의 정책(policy) $\pi^*(s)$

각 상태에서 Q값이 가장 큰 행동

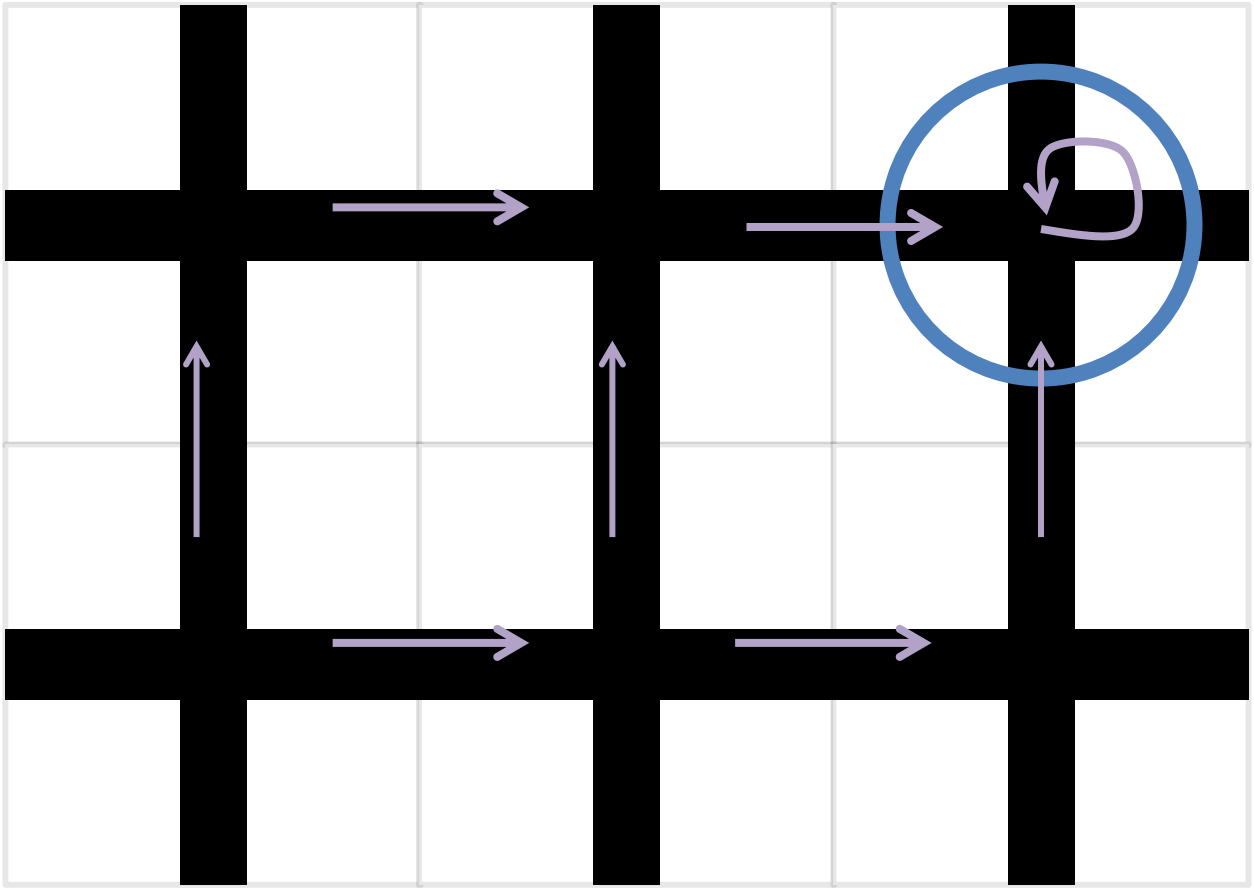


Q-러닝

$$\pi^*(s) = \underset{a}{\operatorname{argmax}}\, Q(s,a)$$

최적의 정책(policy) $\boldsymbol{\pi^*(s)}$

각 상태에서 Q값이 가장 큰 행동

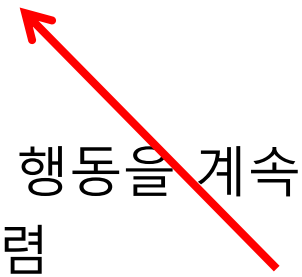


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + \gamma \max_{a'} \hat{Q}(s',a')$$

0.9

실제 상황에서는 학습을 완료한 후
행동하는 것이 아닐 수 있다.
$Q(s,a)$가 아니라 $\hat{Q}(s,a)$가 최대인 행동 선택



모든 상태에서의 모든 행동을 계속 경험하면
$\hat{Q}(s,a) \rightarrow Q(s,a)$로 수렴

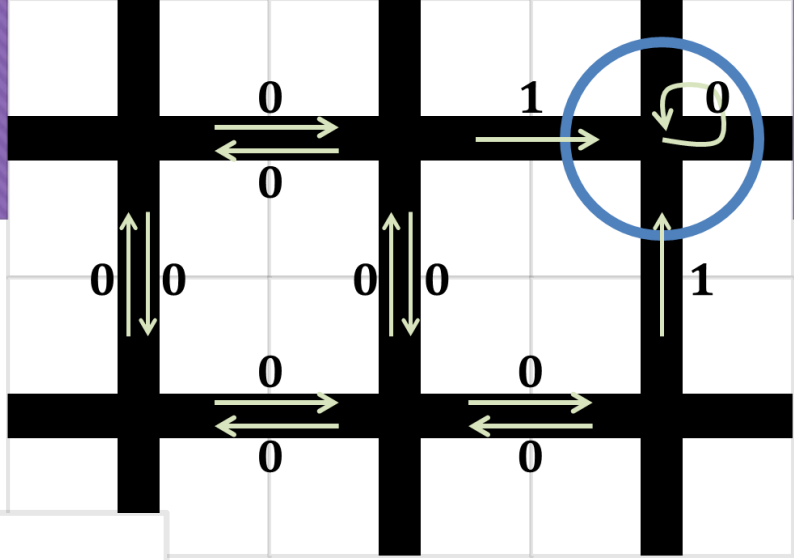
$$\pi^*(s) = \operatorname*{argmax}_a Q(s,a)$$

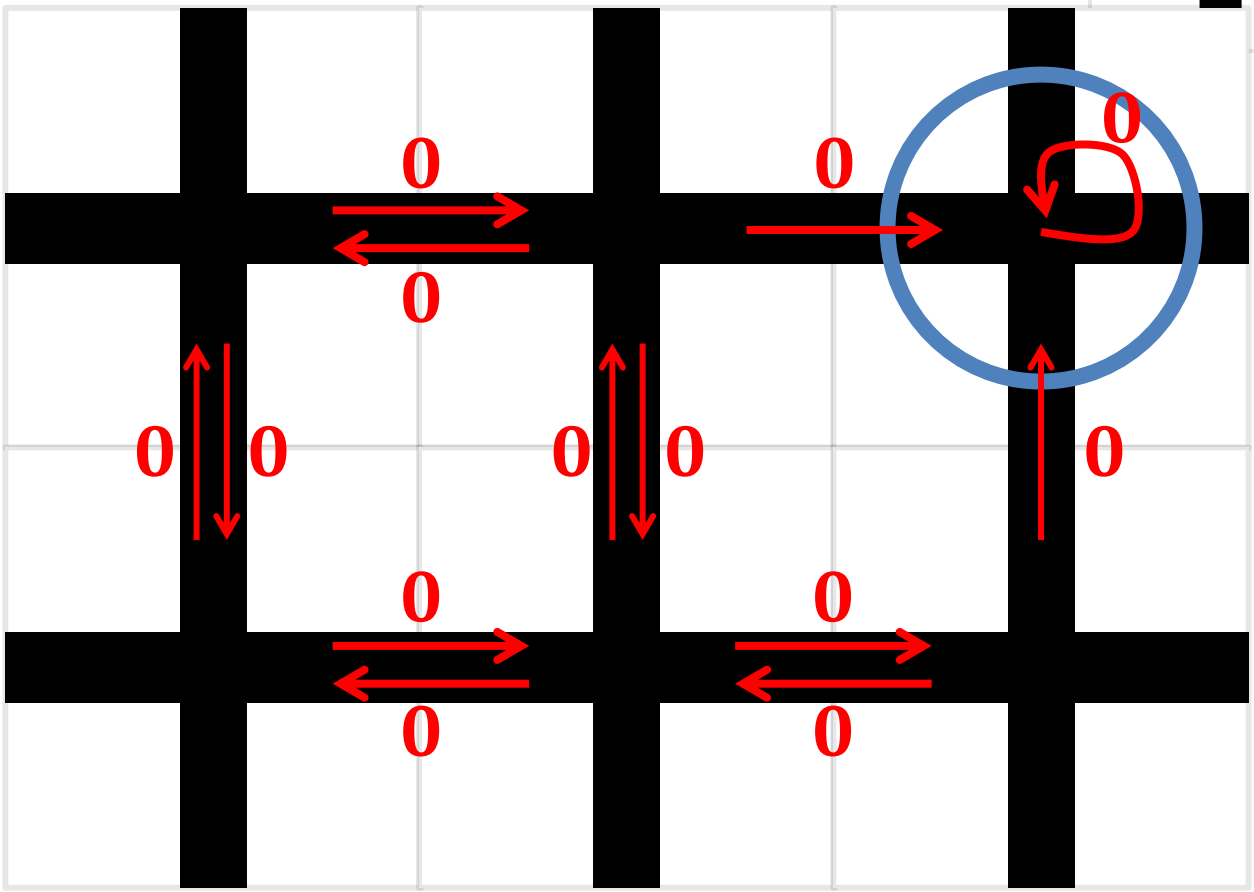
각 상태에서 $Q(s,a)$가 최대인 행동(a)을
취하면 됨

Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

$\hat{Q}$ 초기 값 : 모두 0

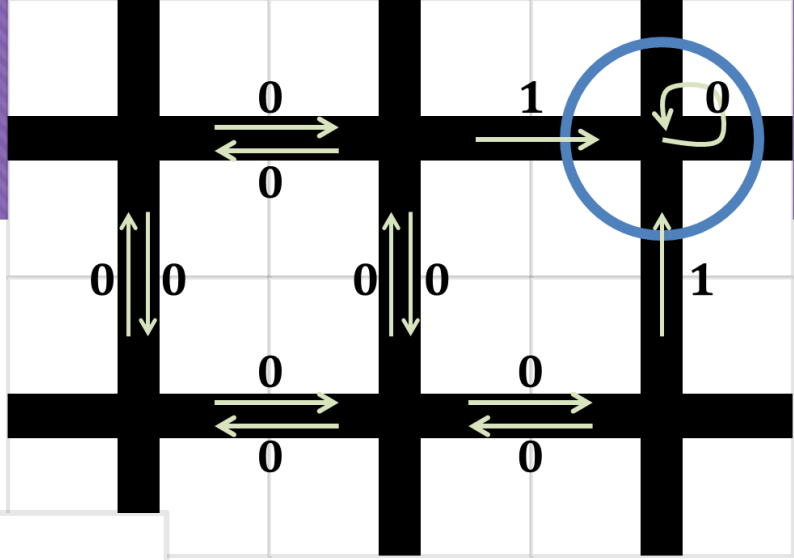


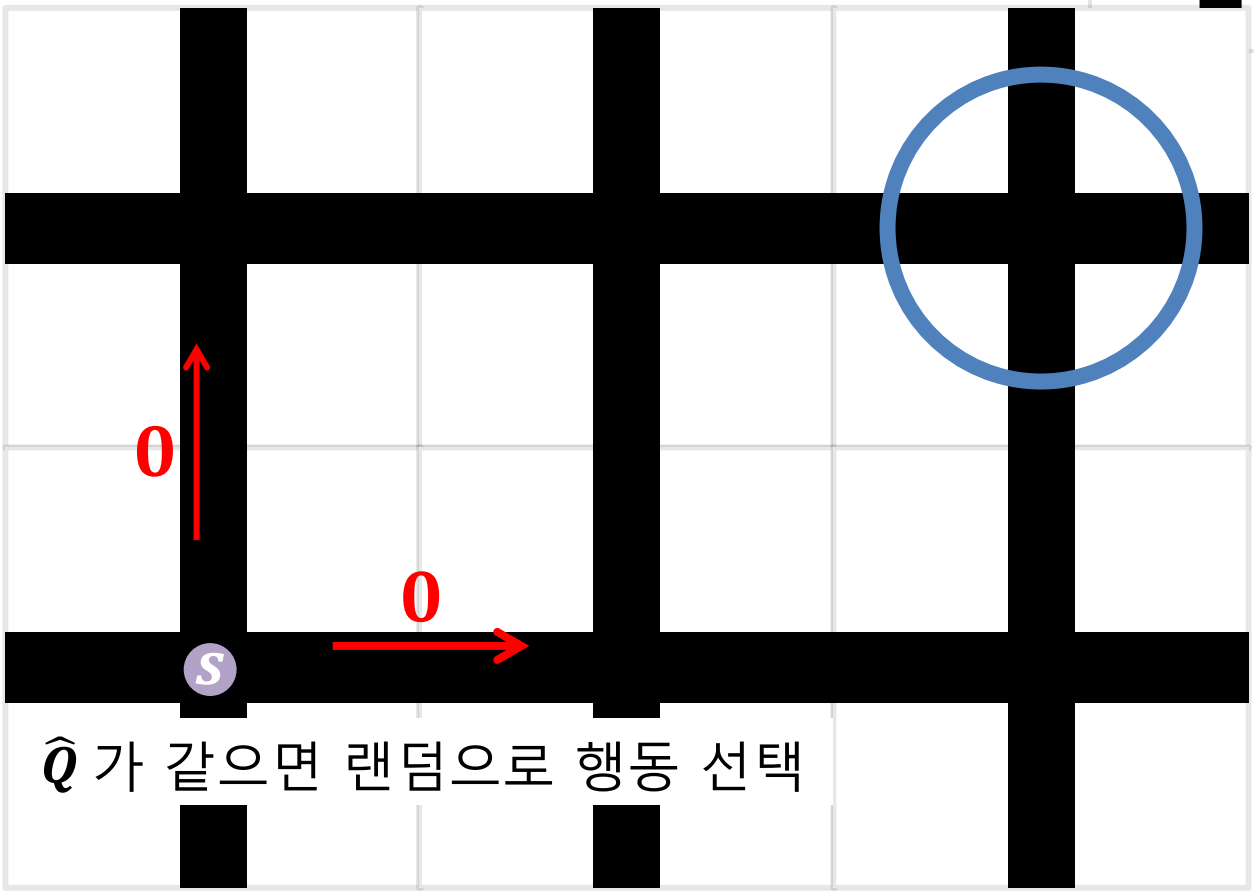


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

출발 위치: 왼쪽 아래



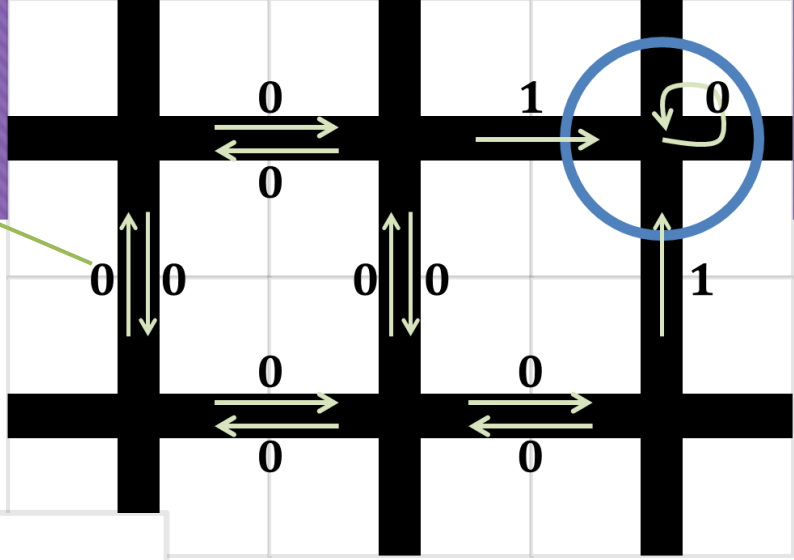


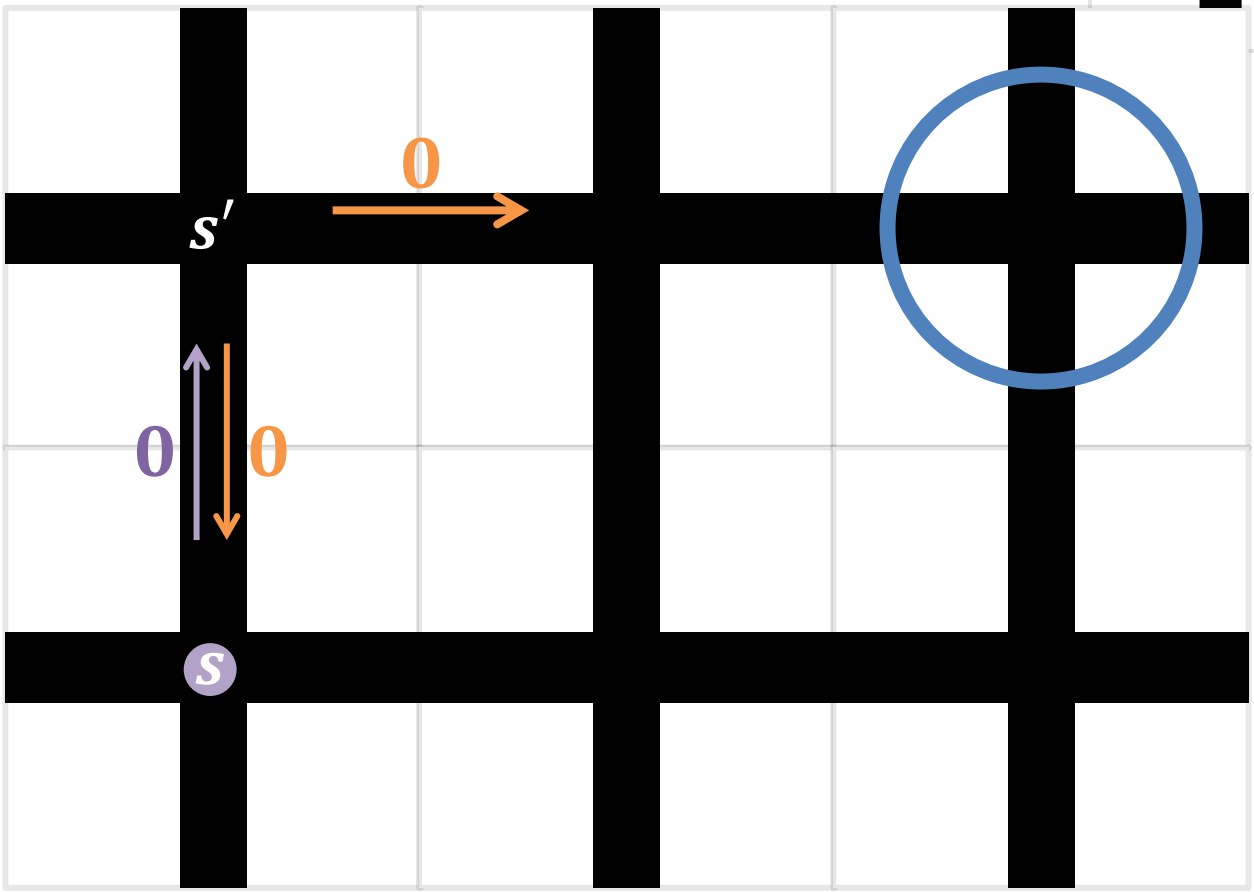
$\hat{Q}$ 가 같으면 랜덤으로 행동 선택

Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

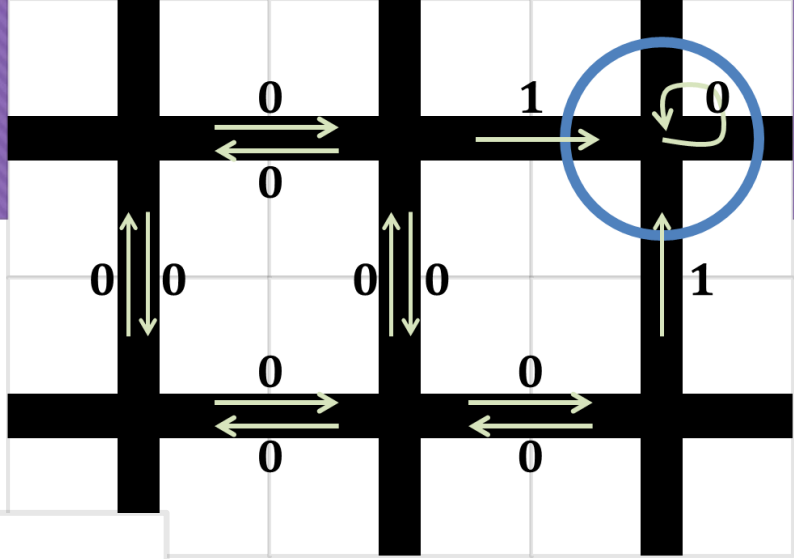
0 0

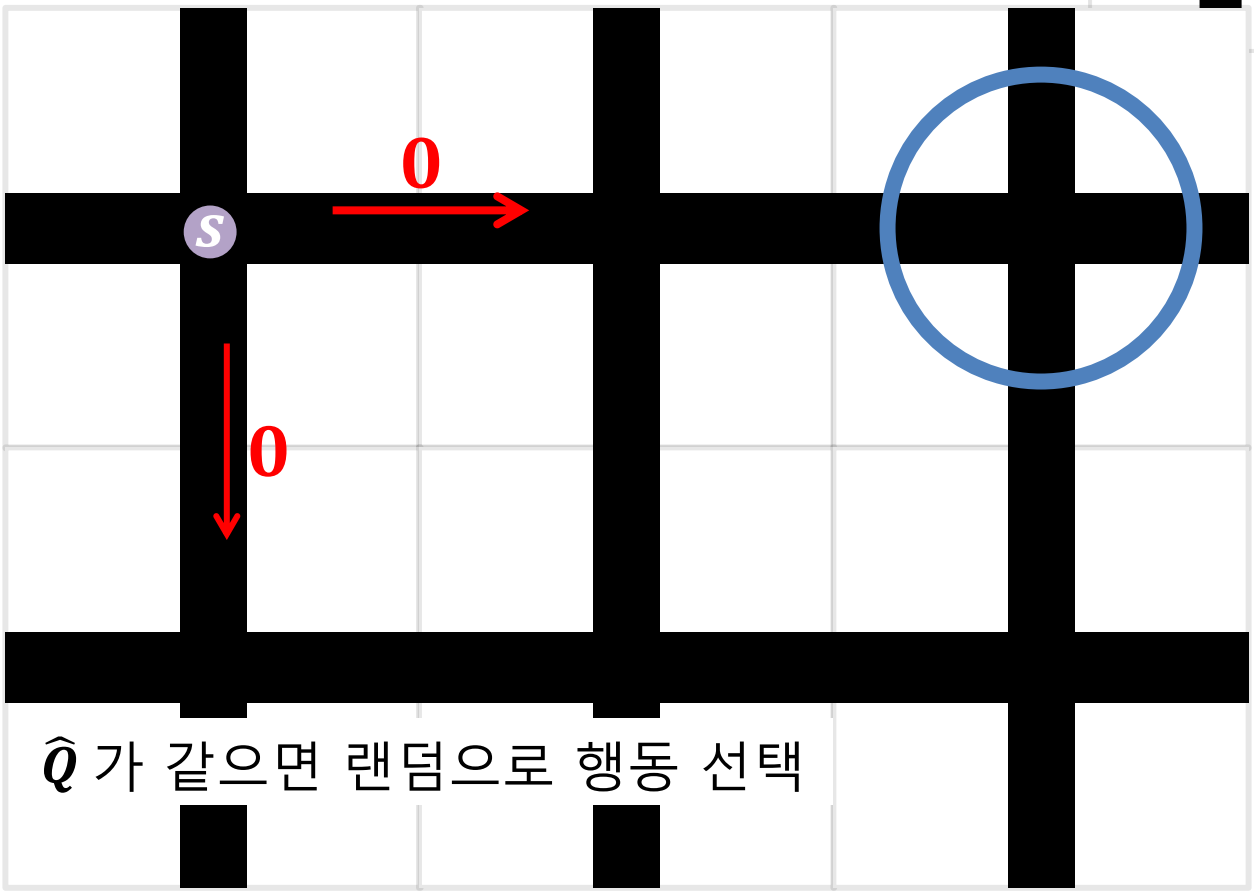




Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$



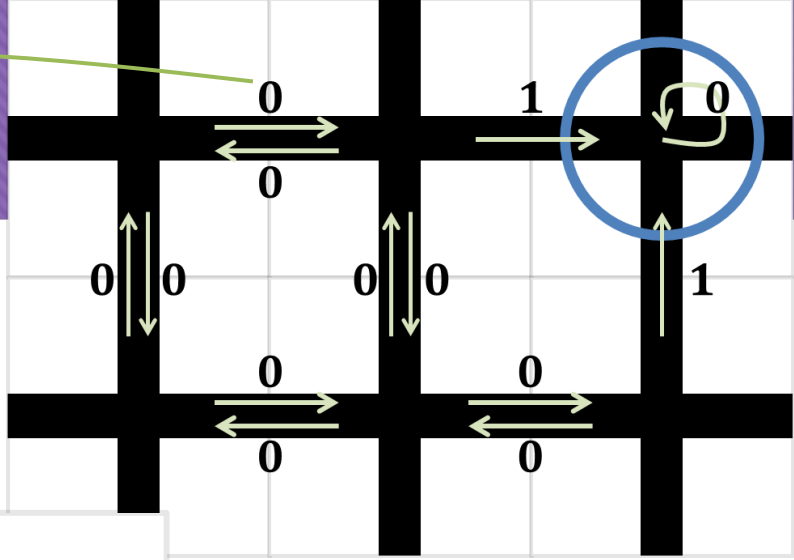


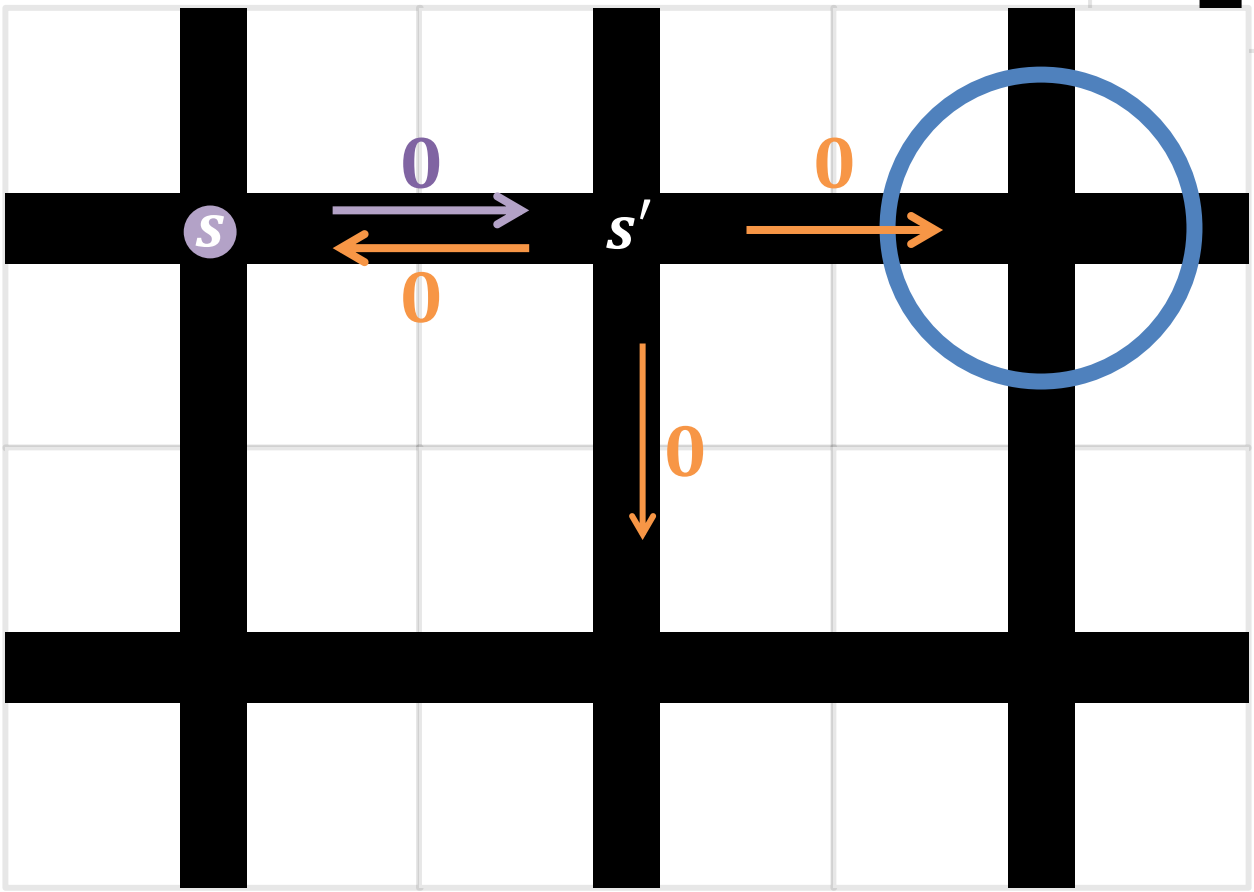
$\hat{Q}$ 가 같으면 랜덤으로 행동 선택

Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

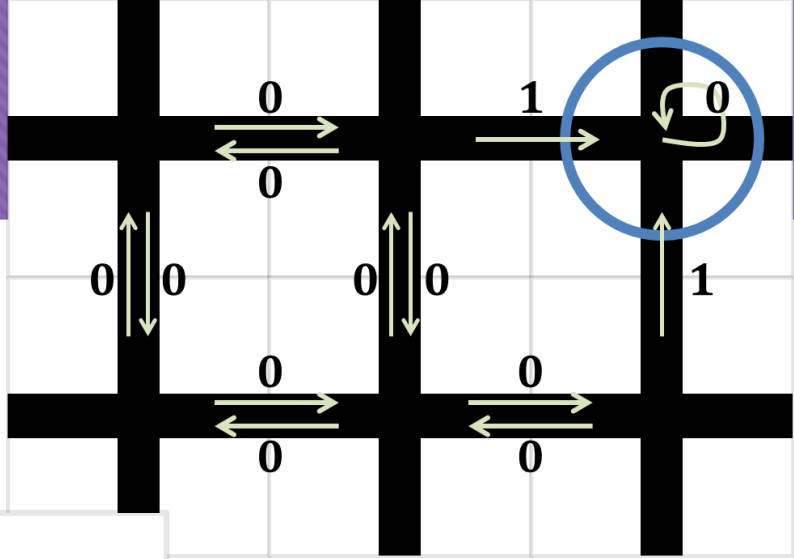
0 0

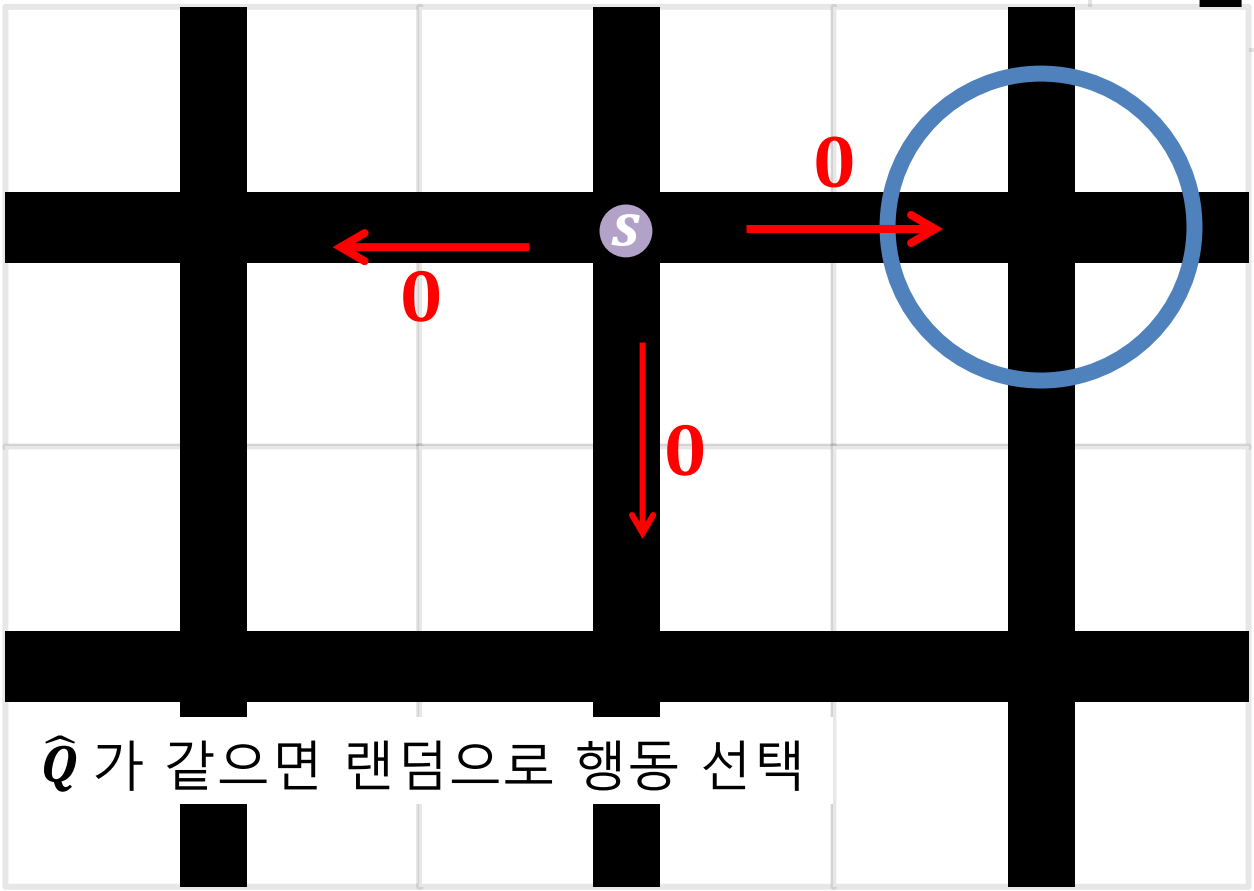




Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$



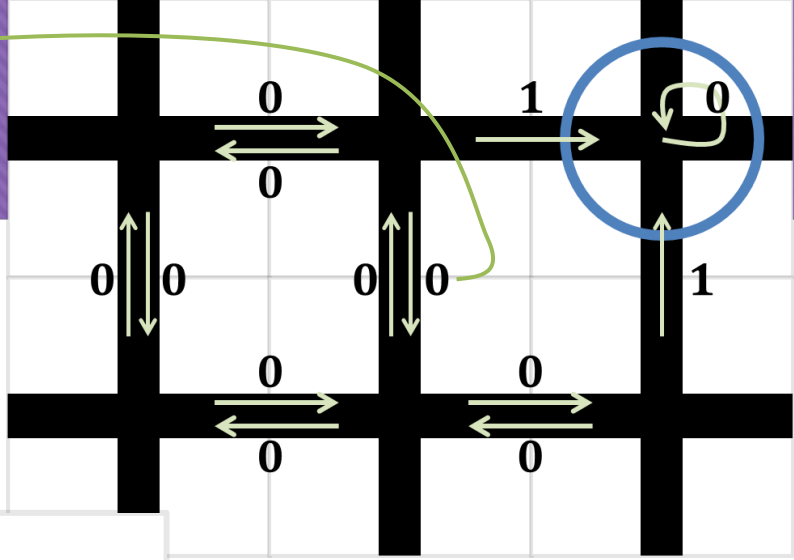


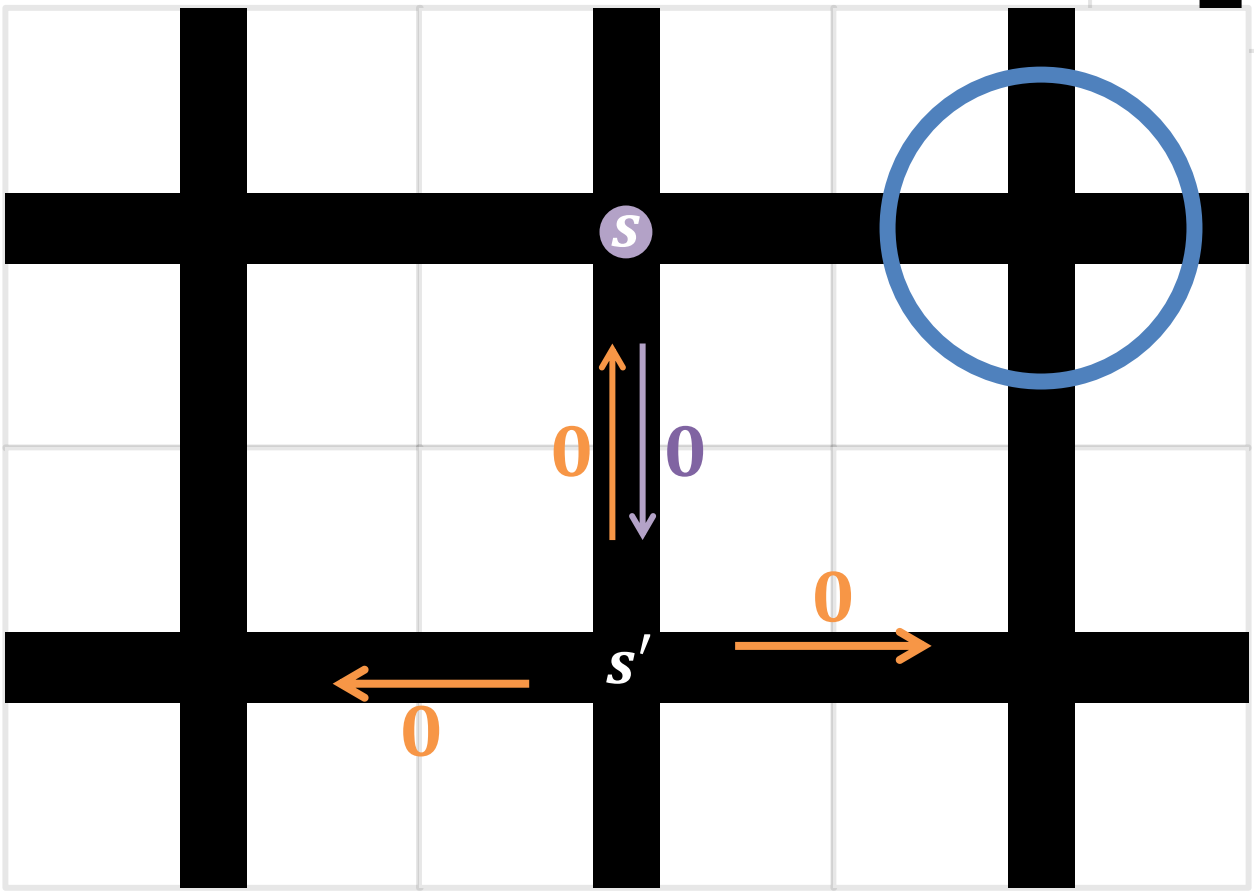
$\hat{Q}$ 가 같으면 랜덤으로 행동 선택

Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

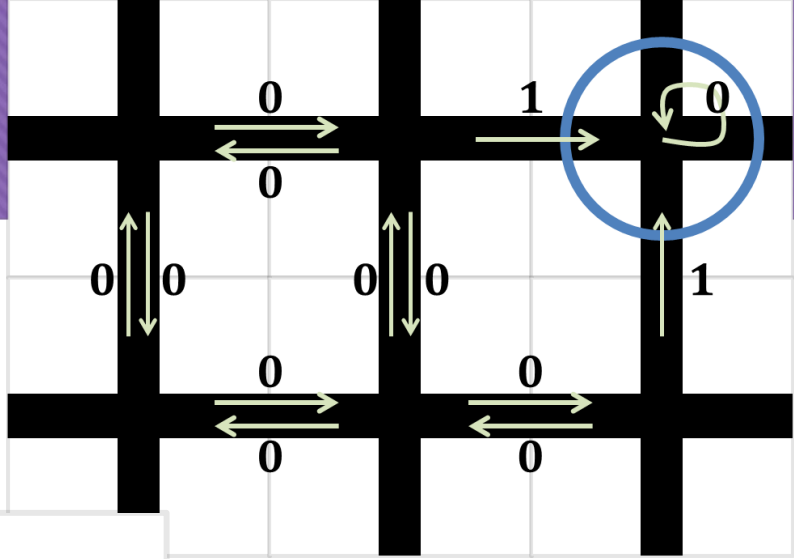
0 0

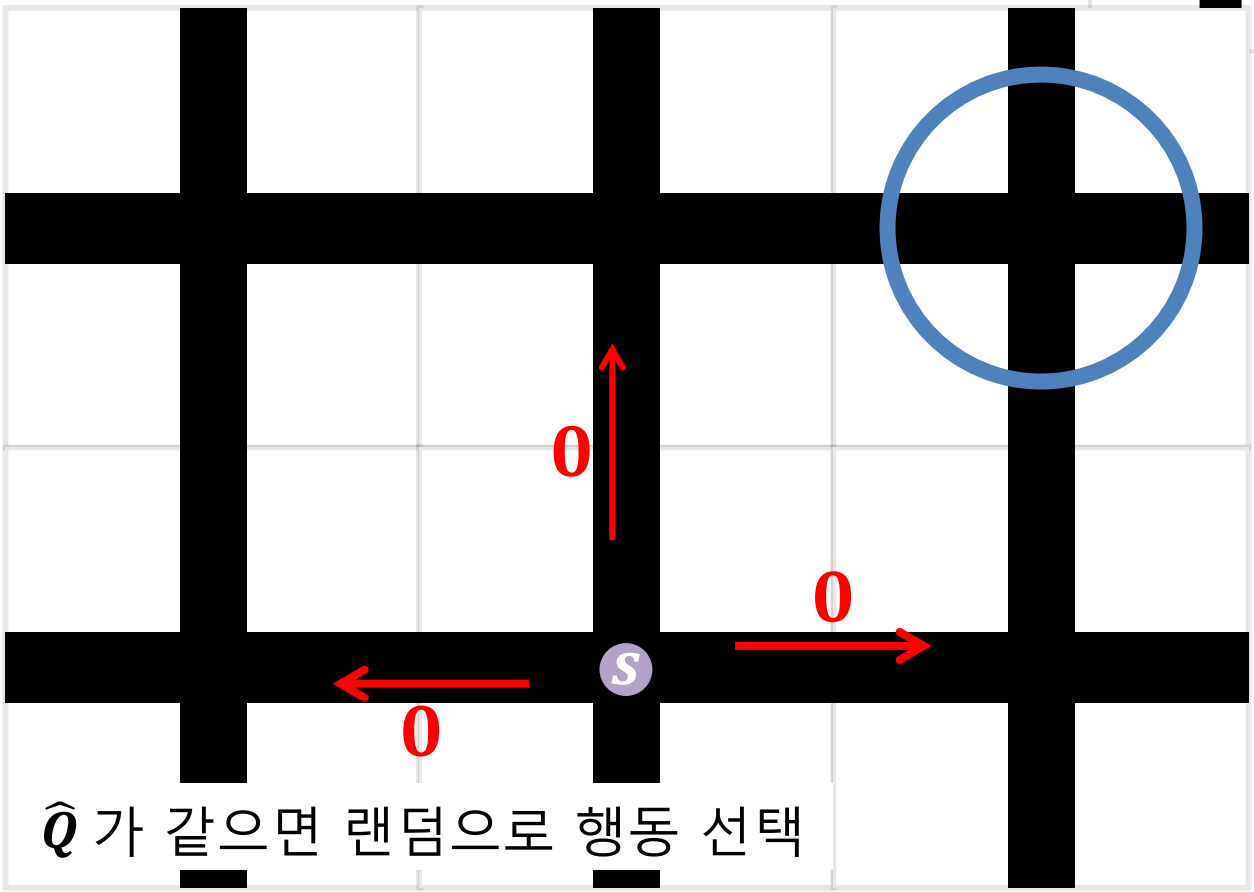




Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$



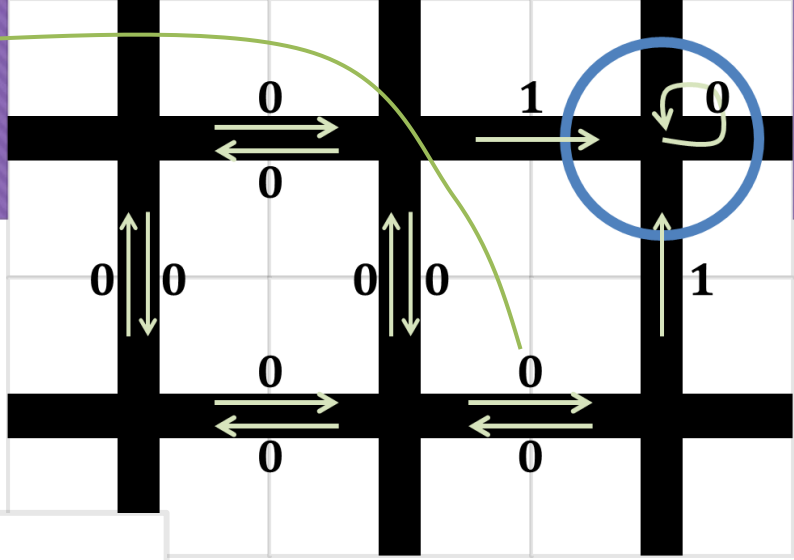


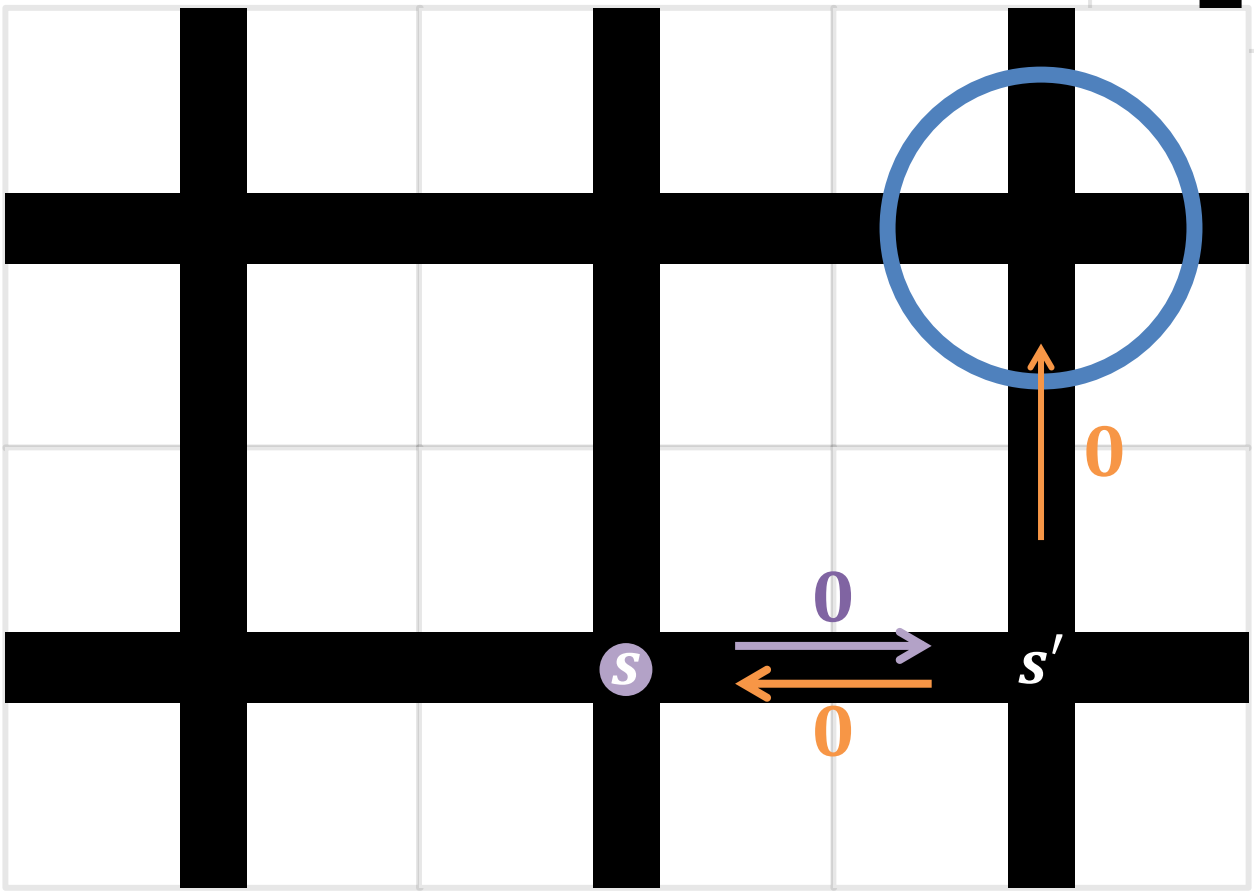
$\hat{Q}$ 가 같으면 랜덤으로 행동 선택

Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

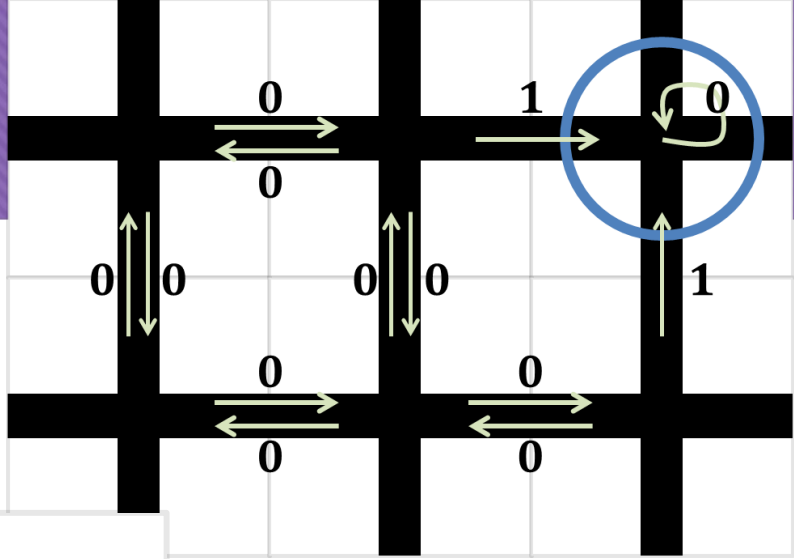
0 0

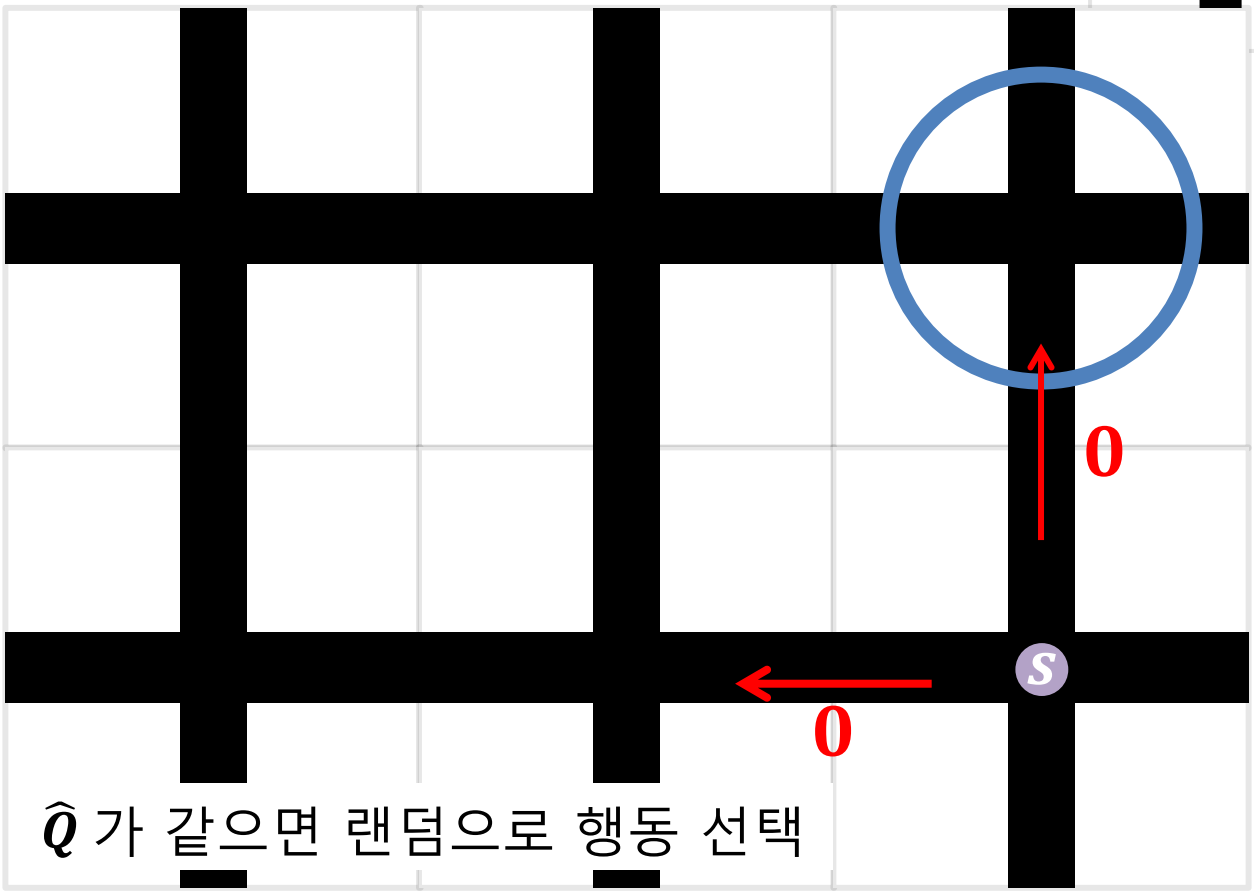




Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$



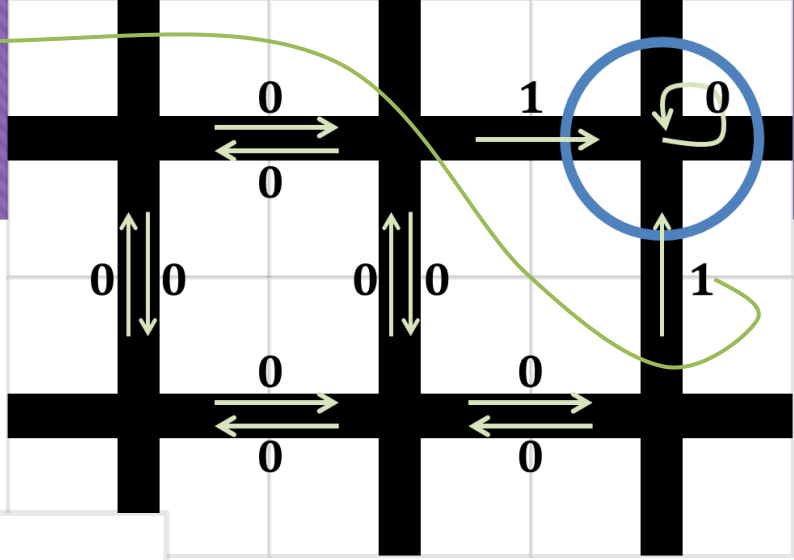


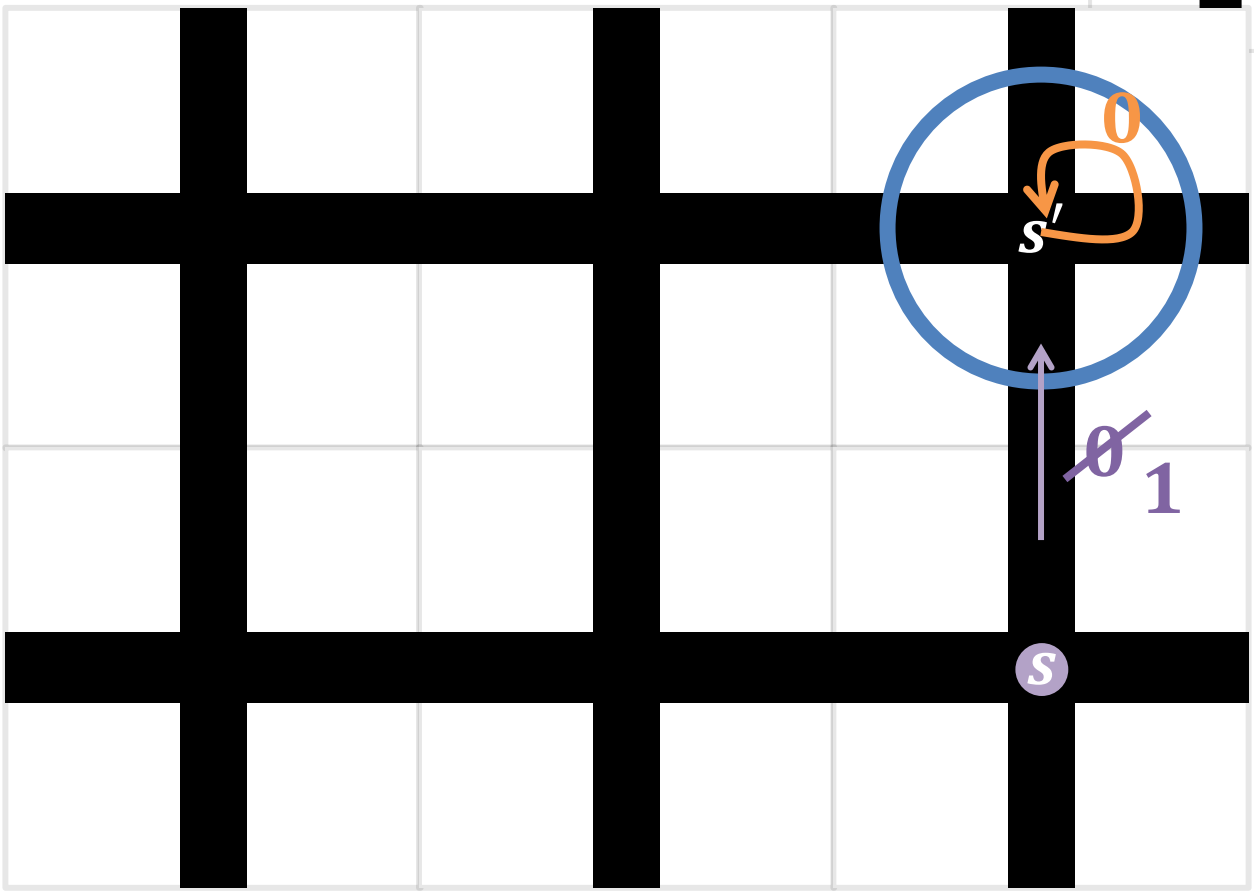
$\hat{Q}$ 가 같으면 랜덤으로 행동 선택

Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

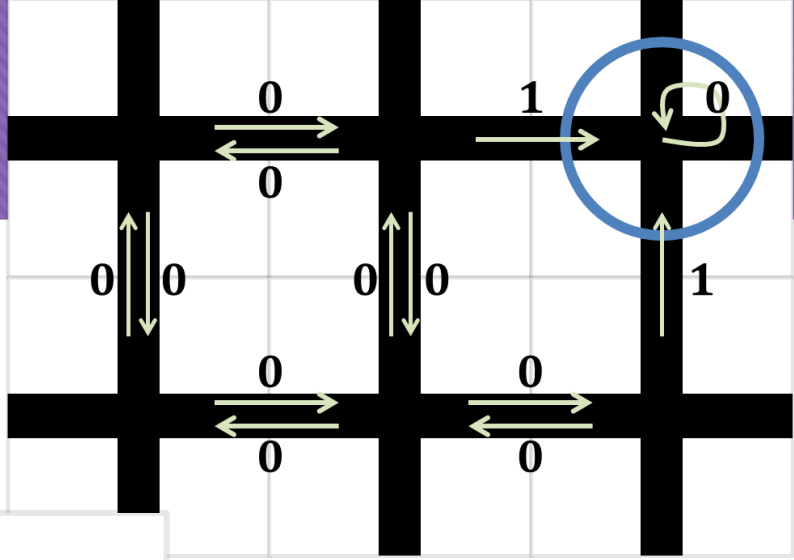
1 0

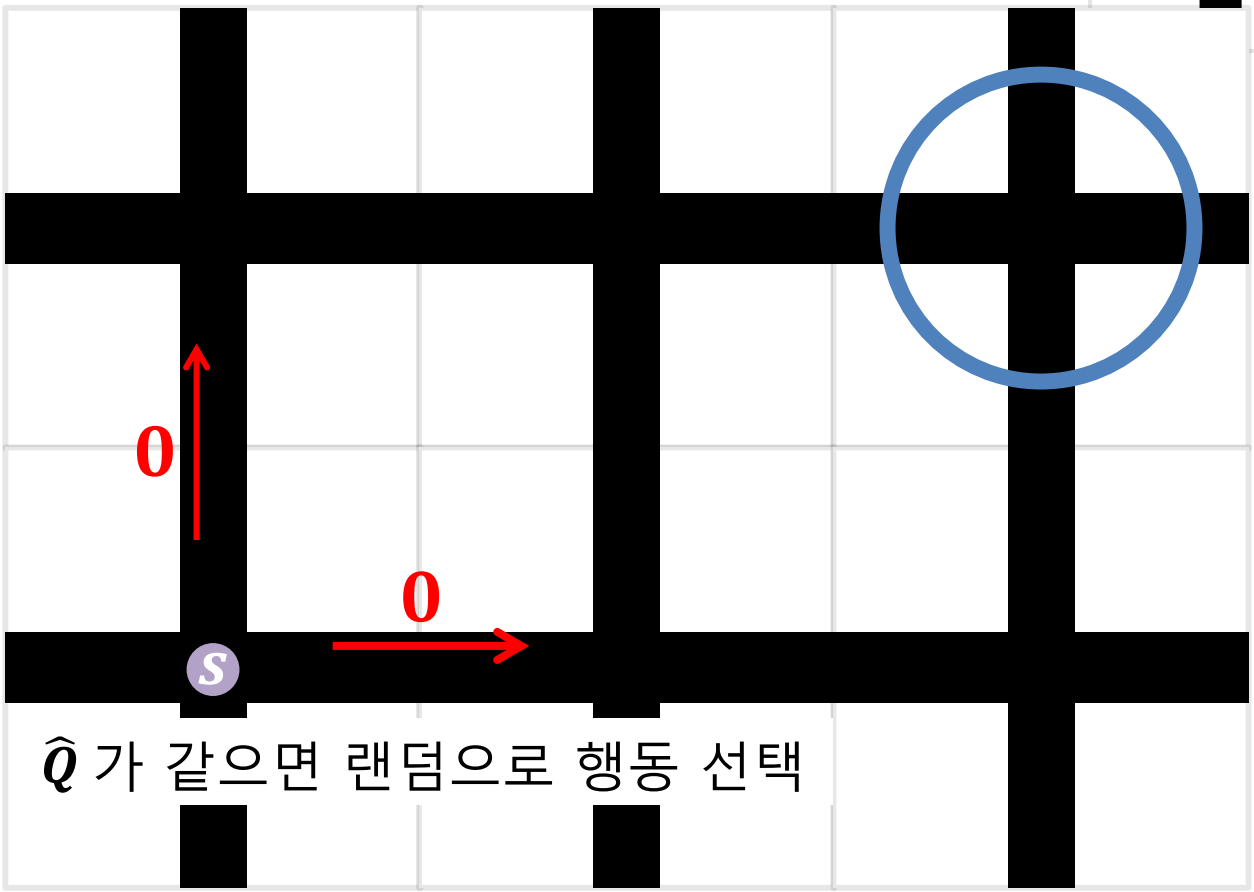




Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$



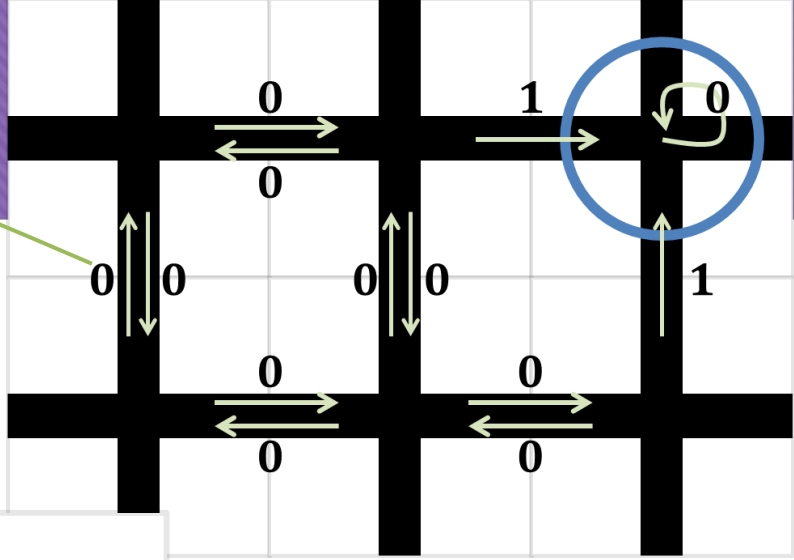


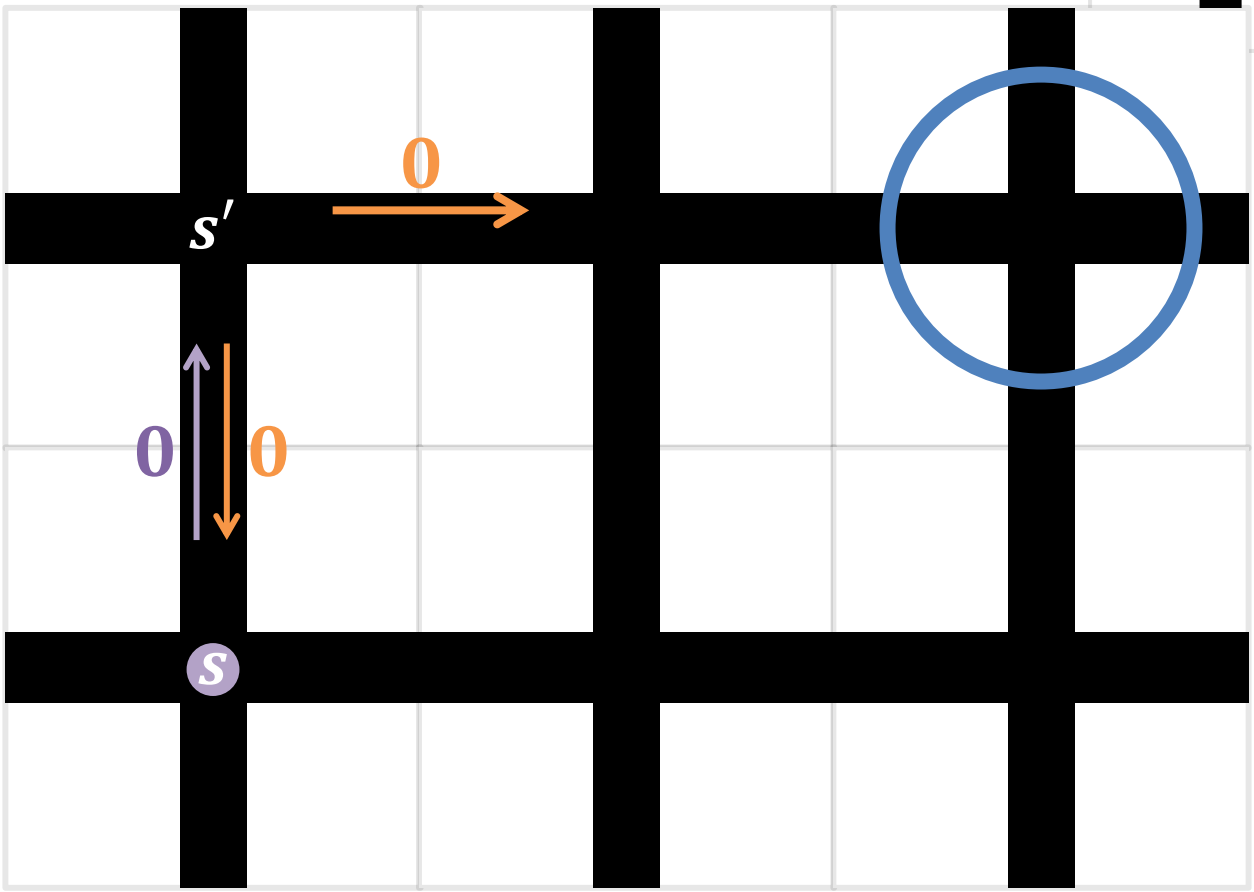
$\hat{Q}$ 가 같으면 랜덤으로 행동 선택

Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

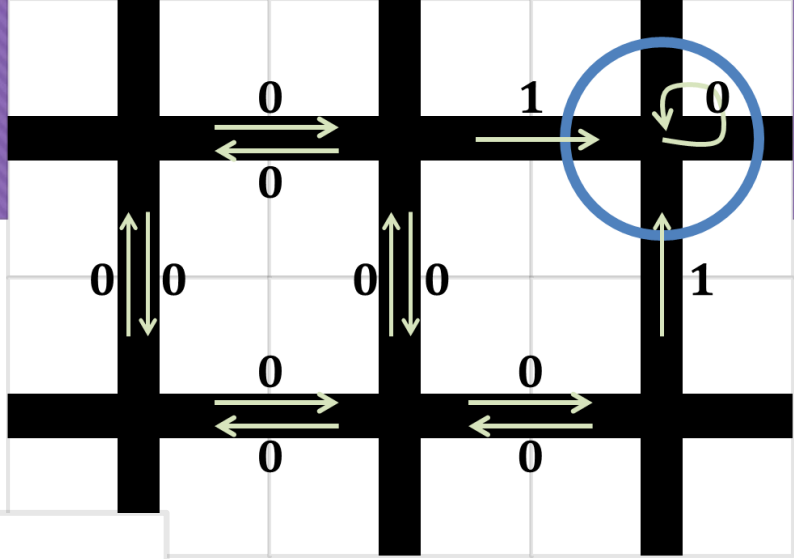
0 0

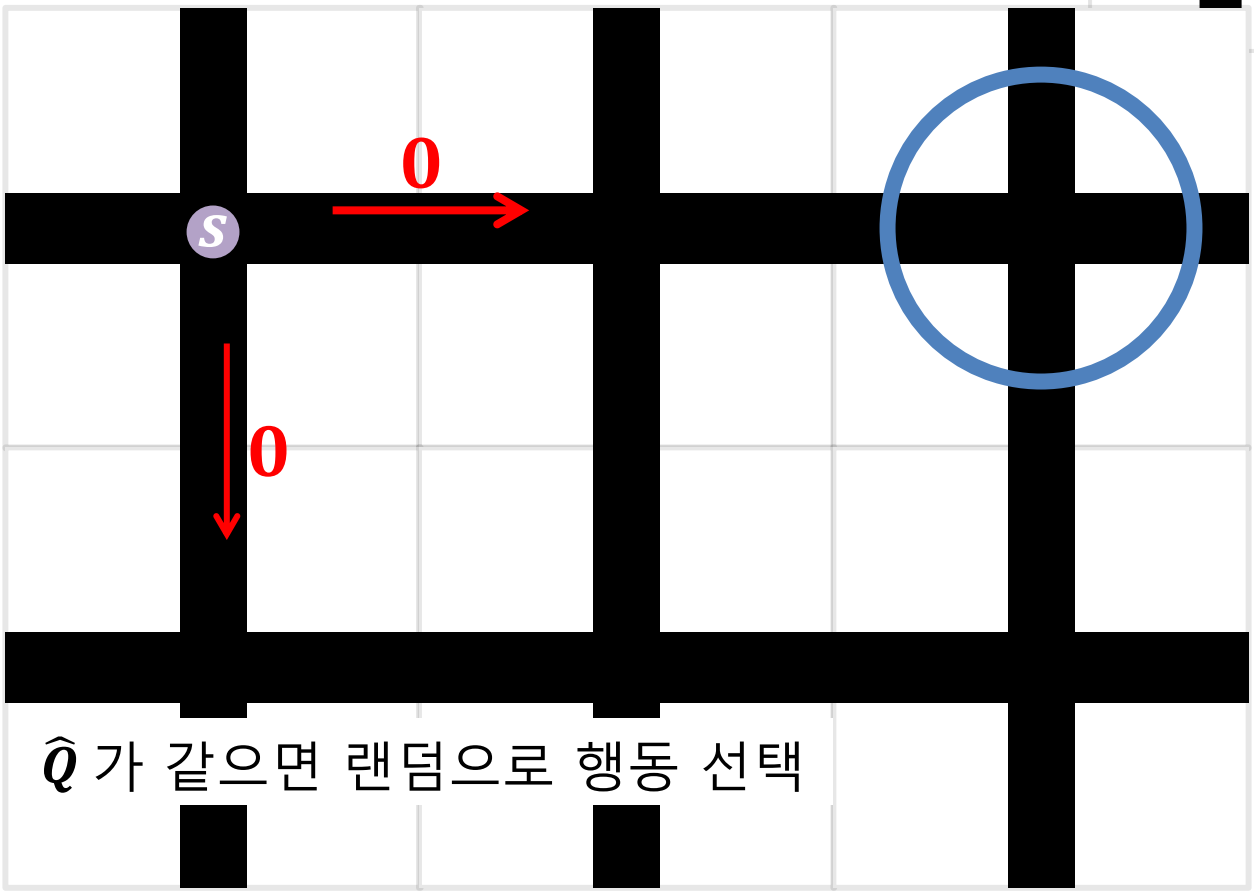




Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$



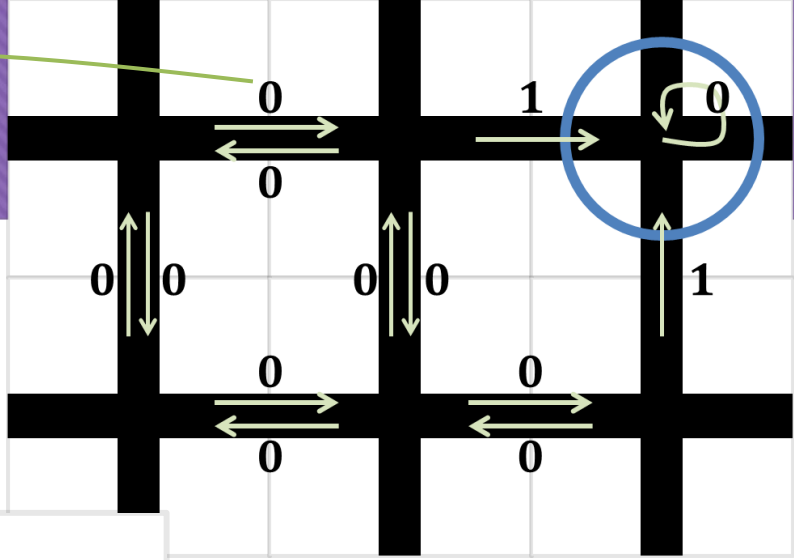


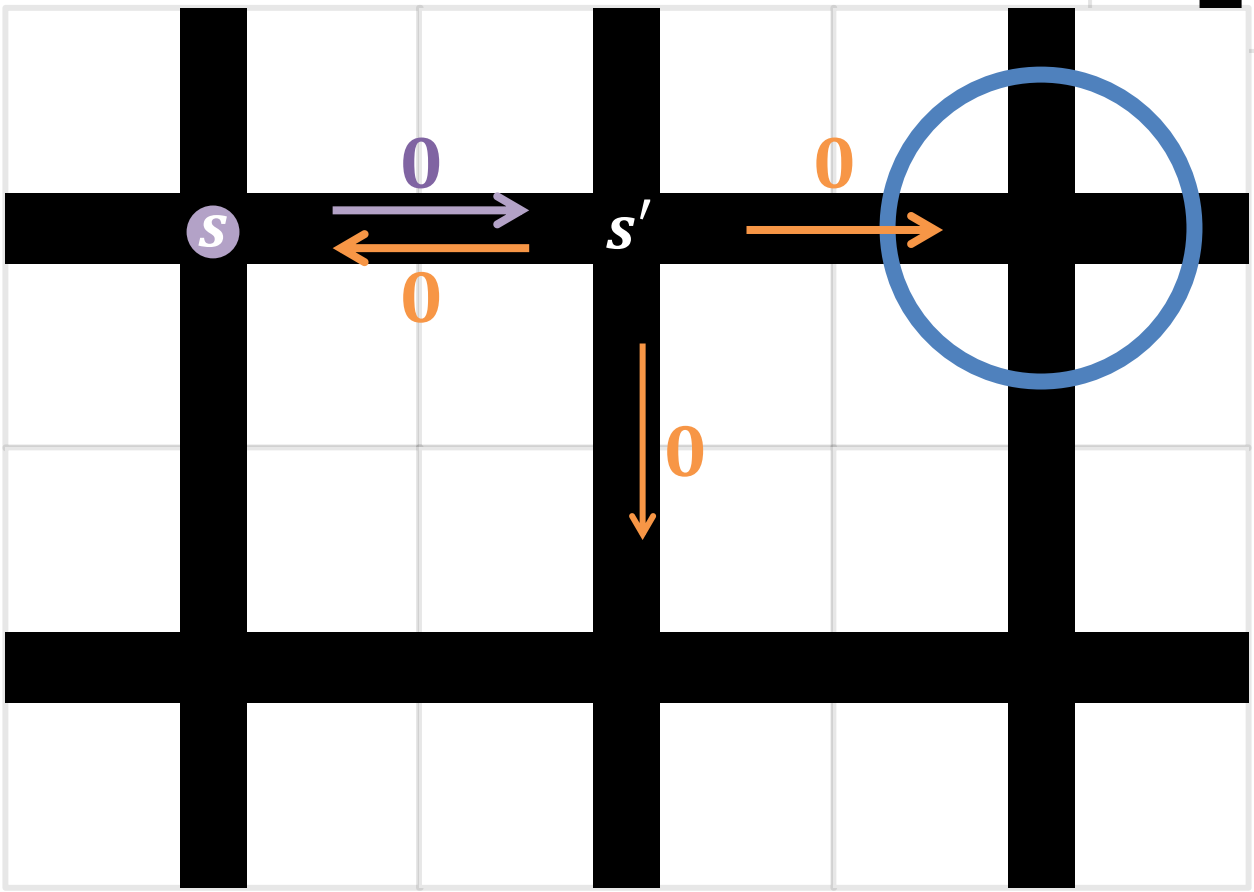
$\hat{Q}$ 가 같으면 랜덤으로 행동 선택

Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

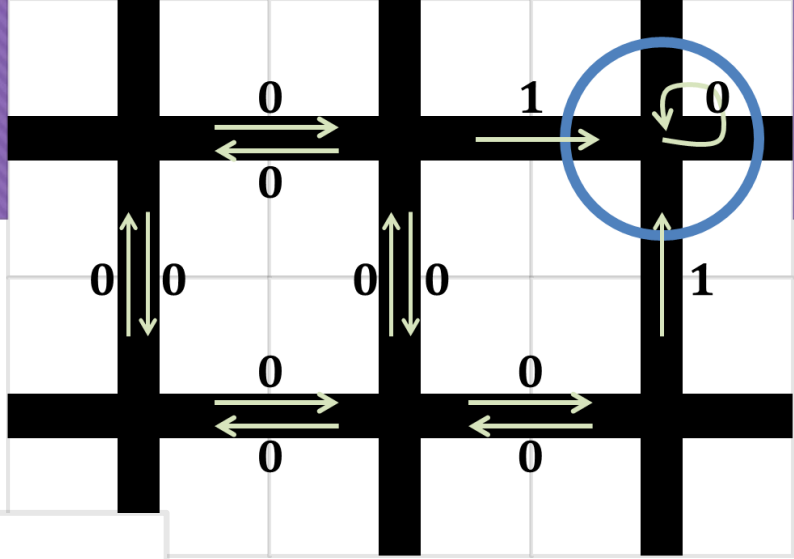
0 0

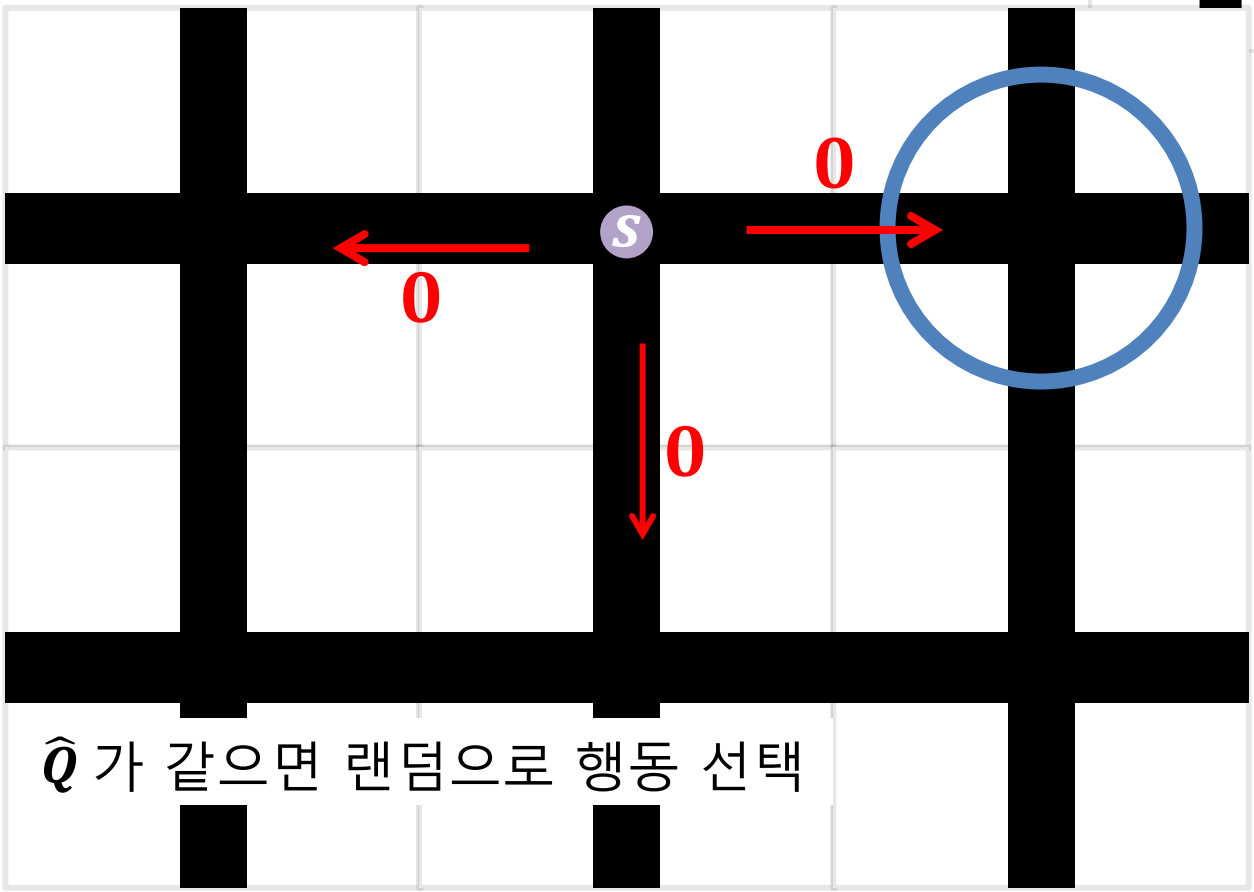




Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$



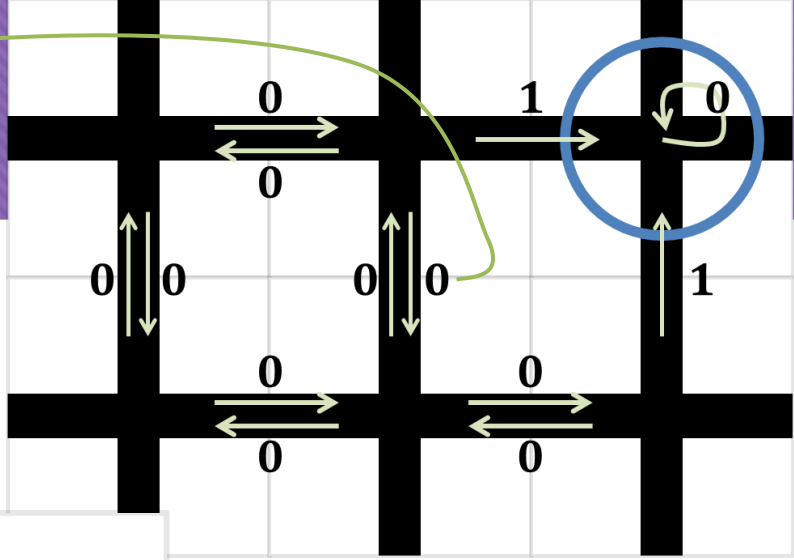


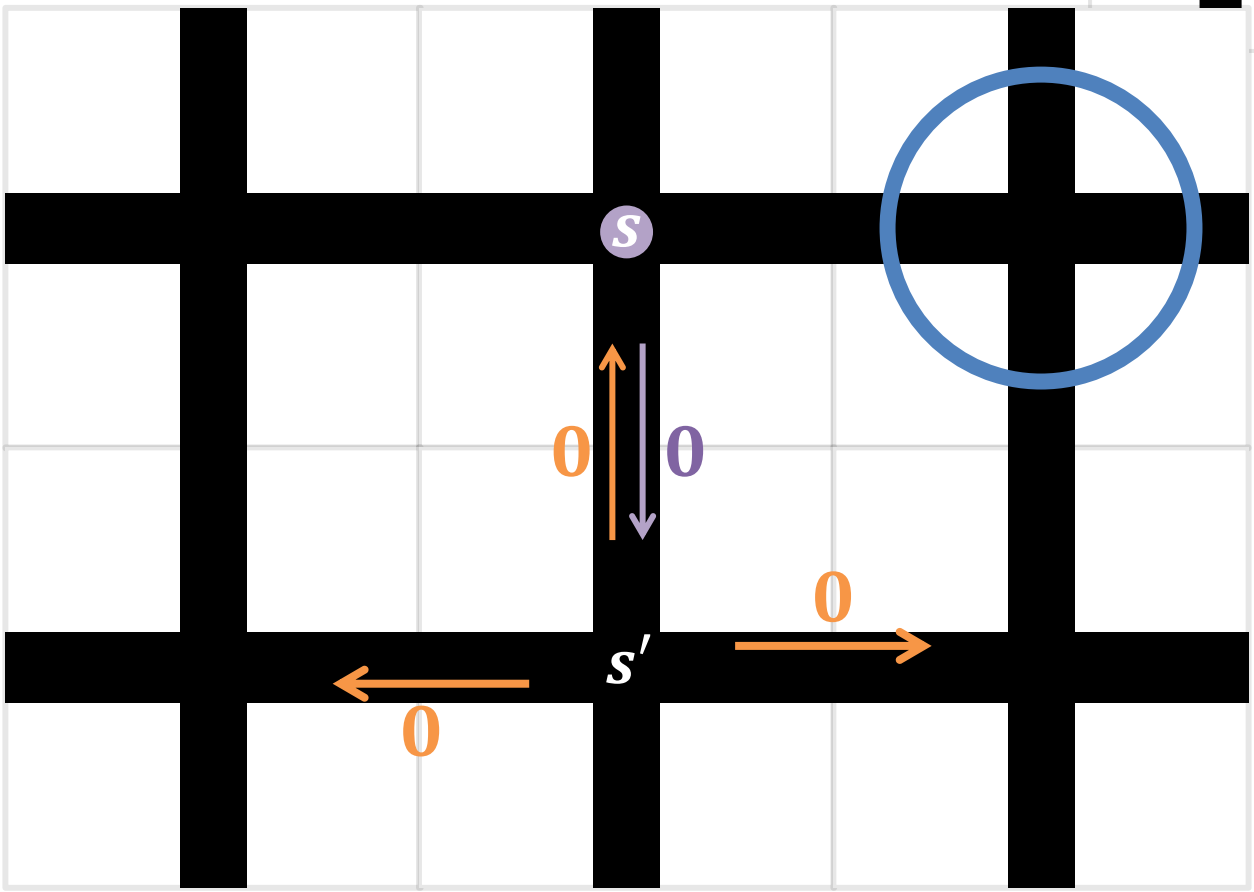
$\hat{Q}$ 가 같으면 랜덤으로 행동 선택

Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

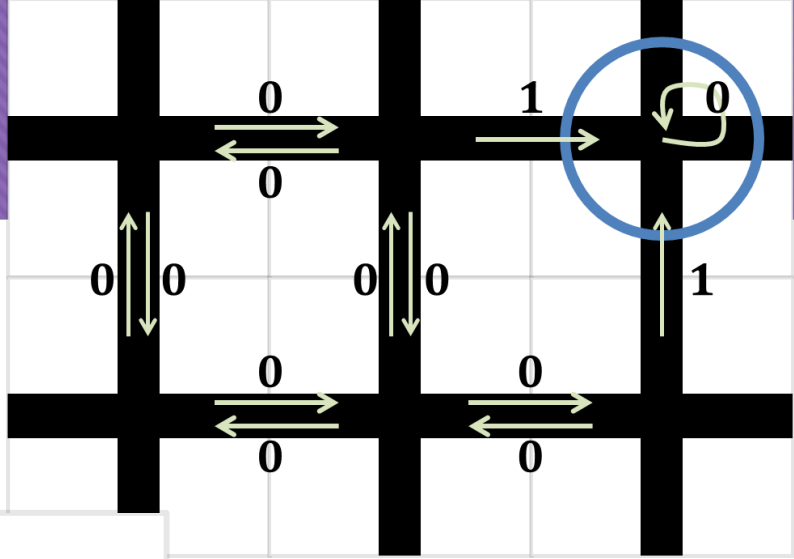
0 0

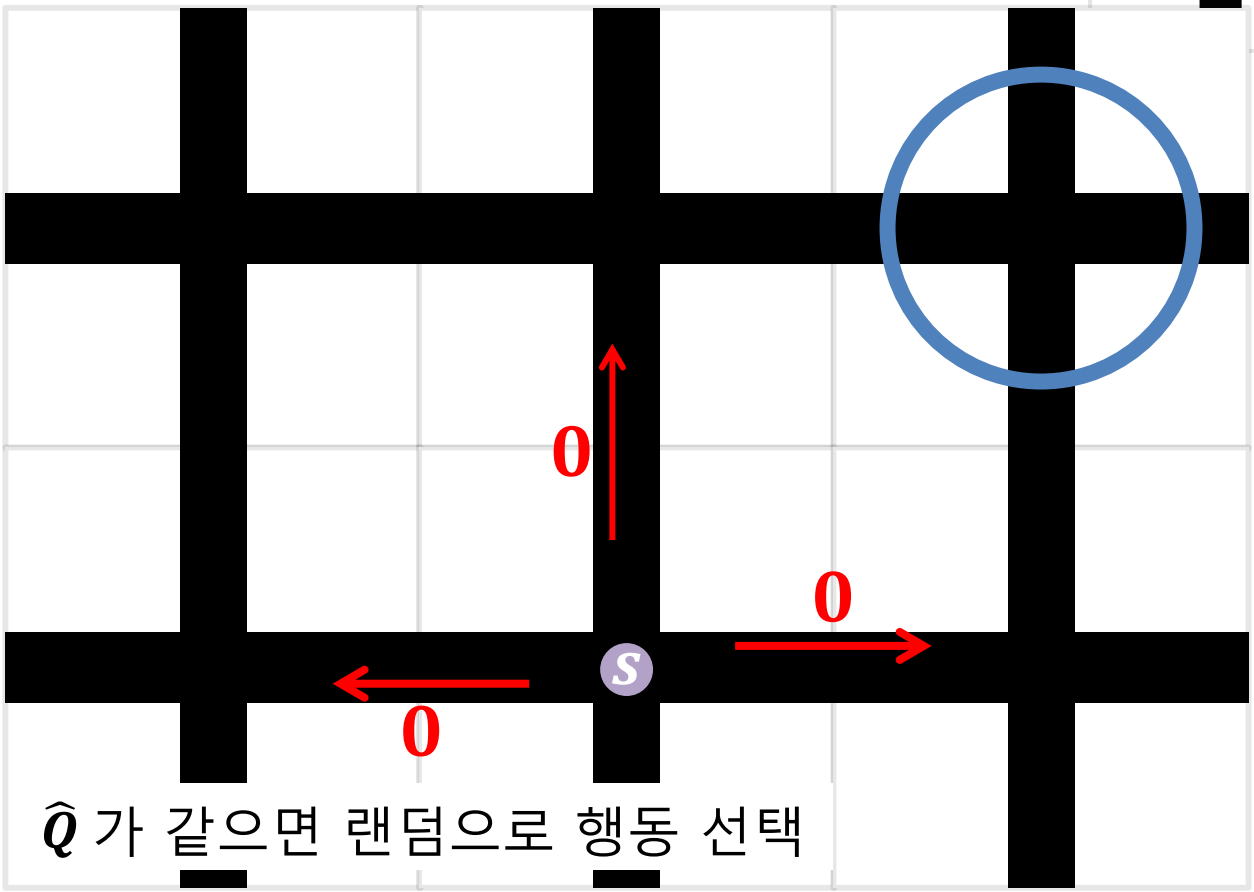




Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$



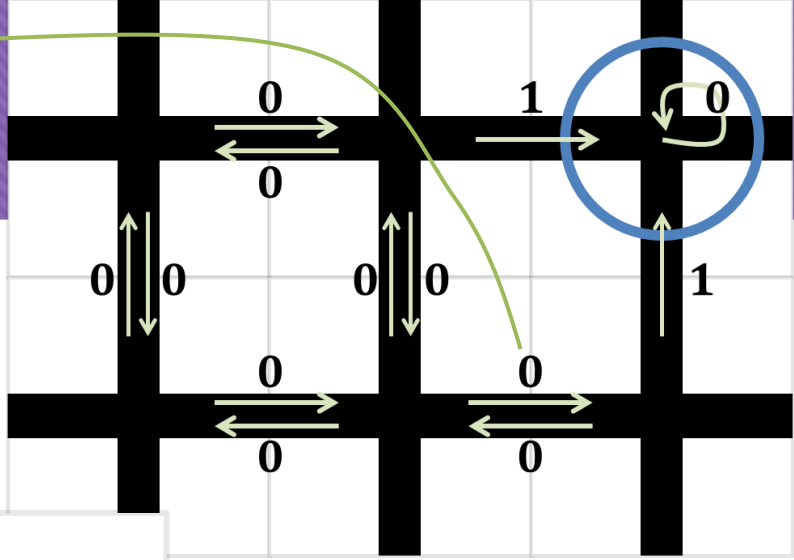


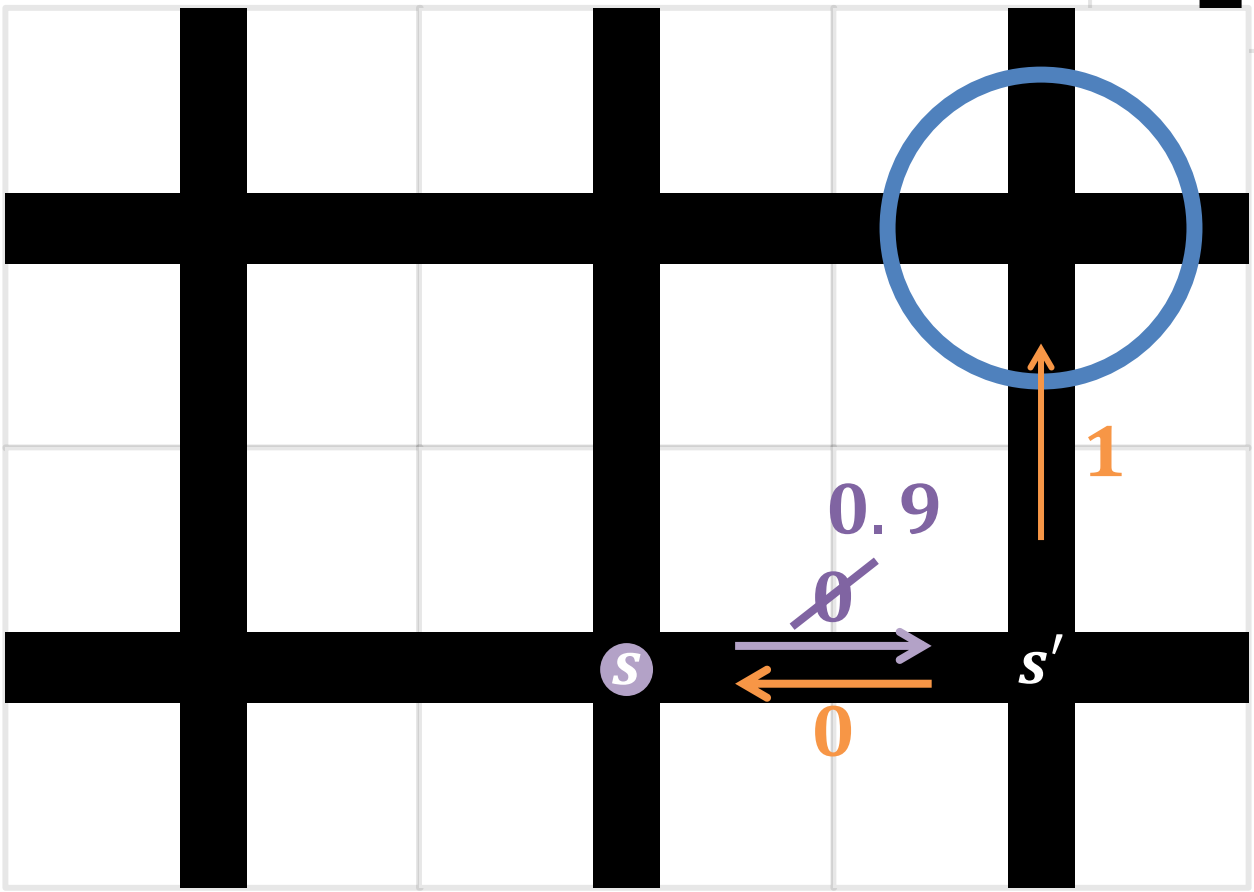
$\hat{Q}$ 가 같으면 랜덤으로 행동 선택

Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

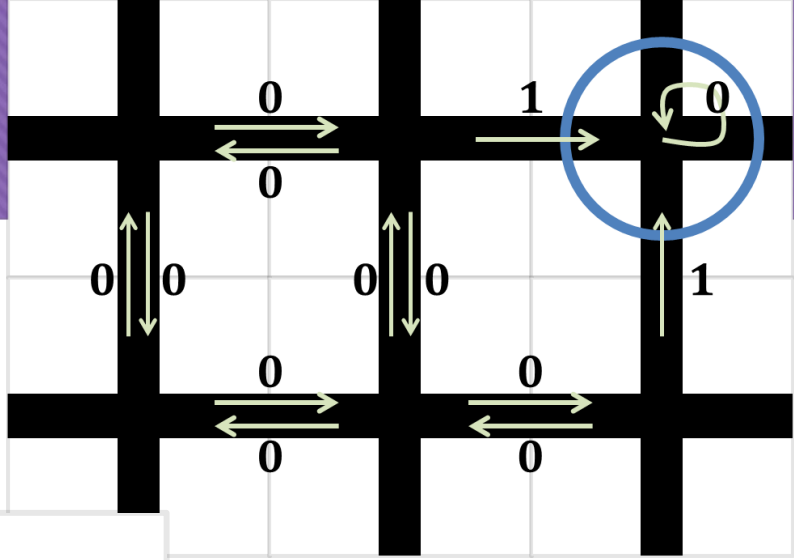
0 1

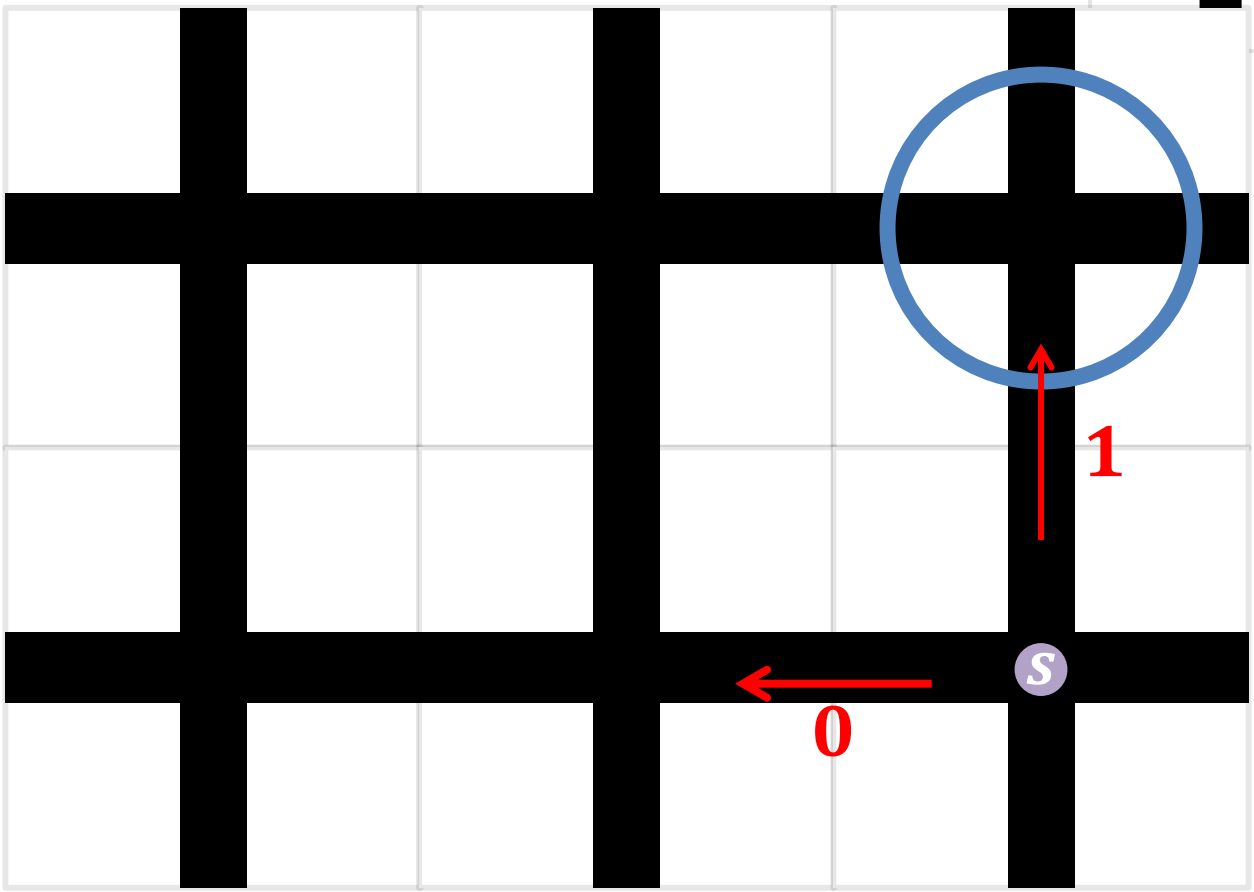




Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

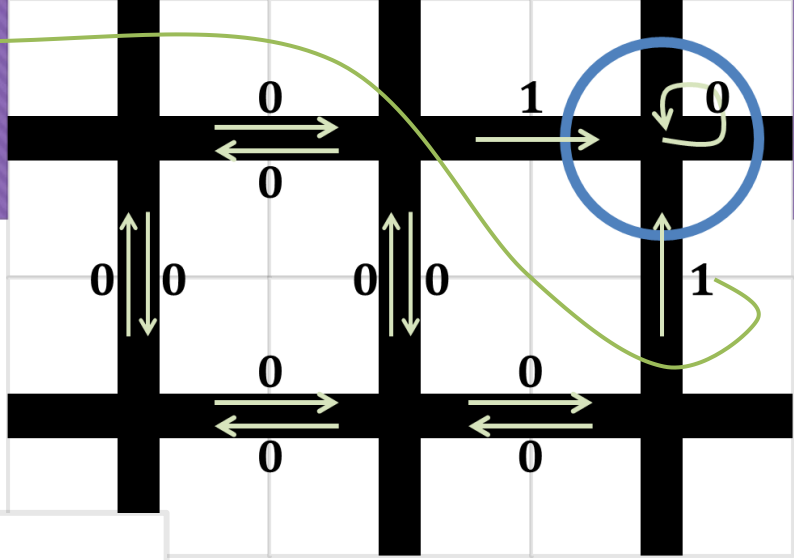


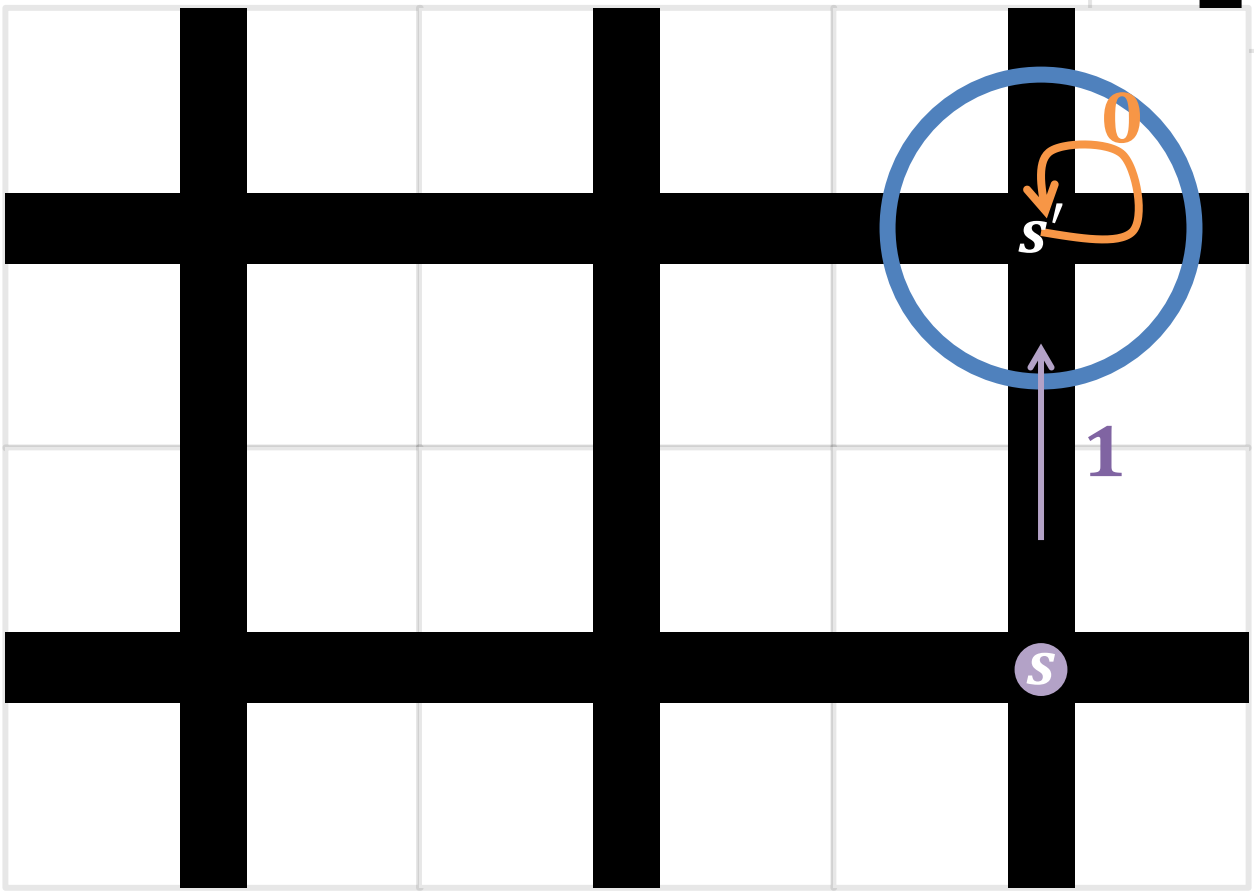


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

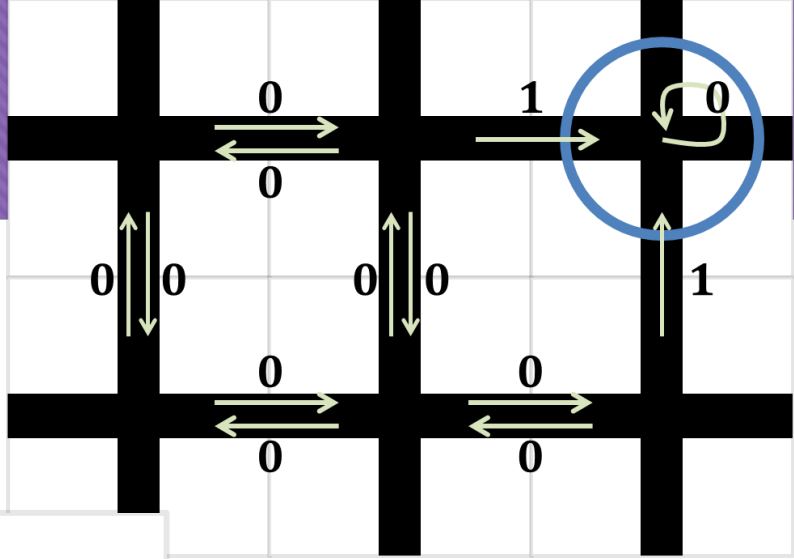
1 0

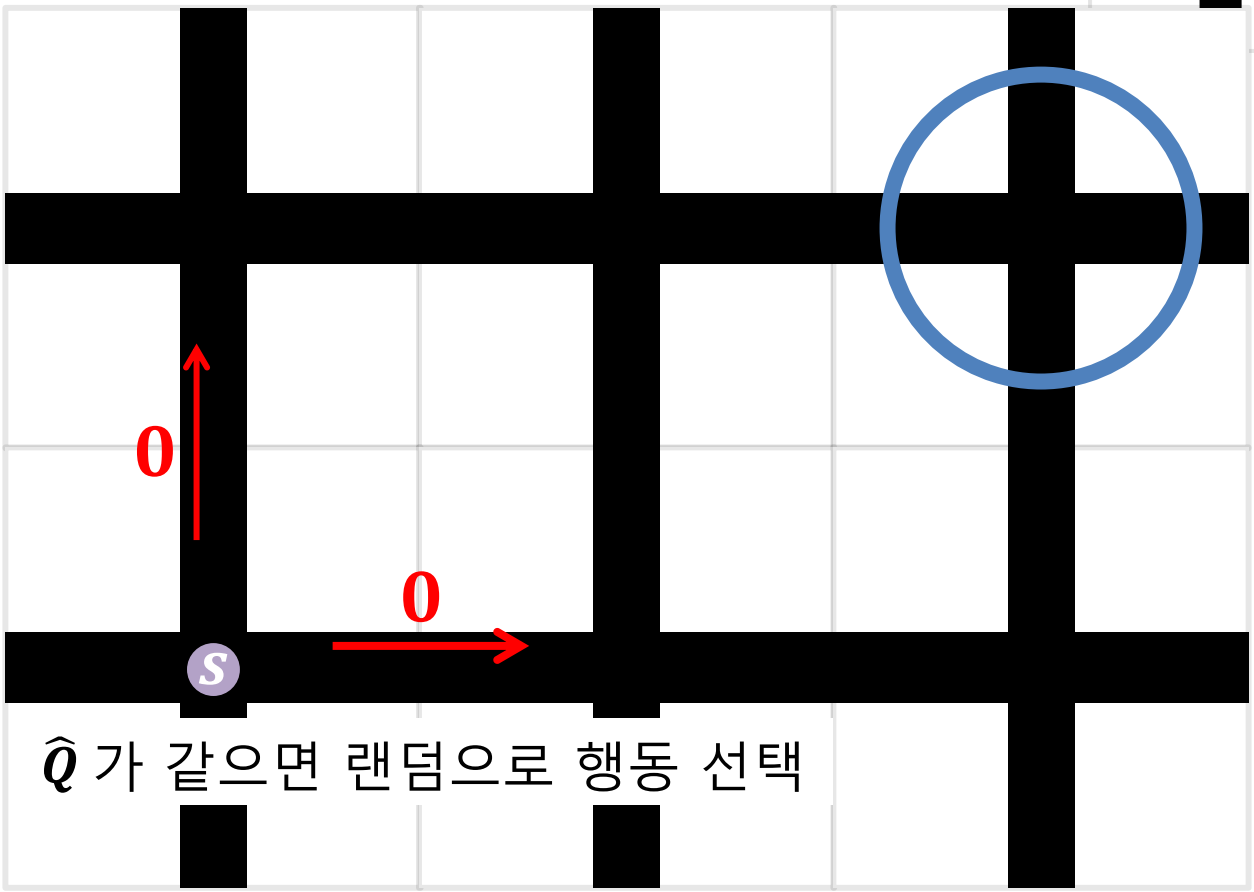




Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$



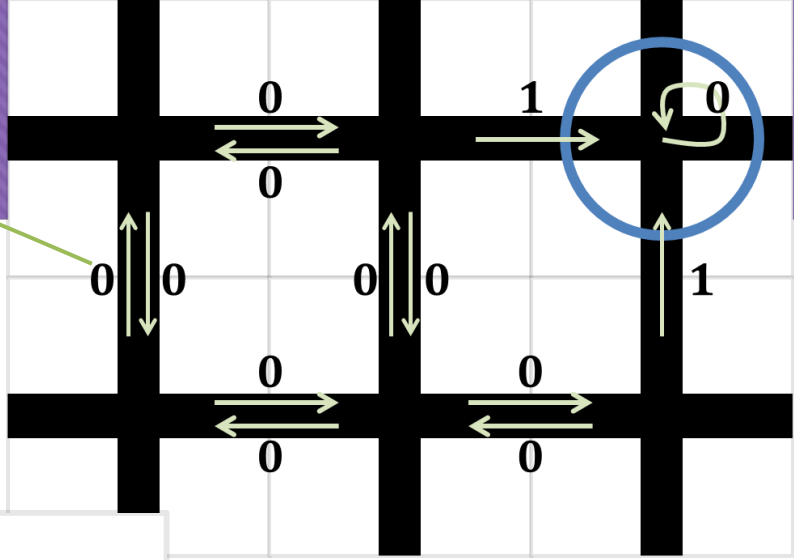


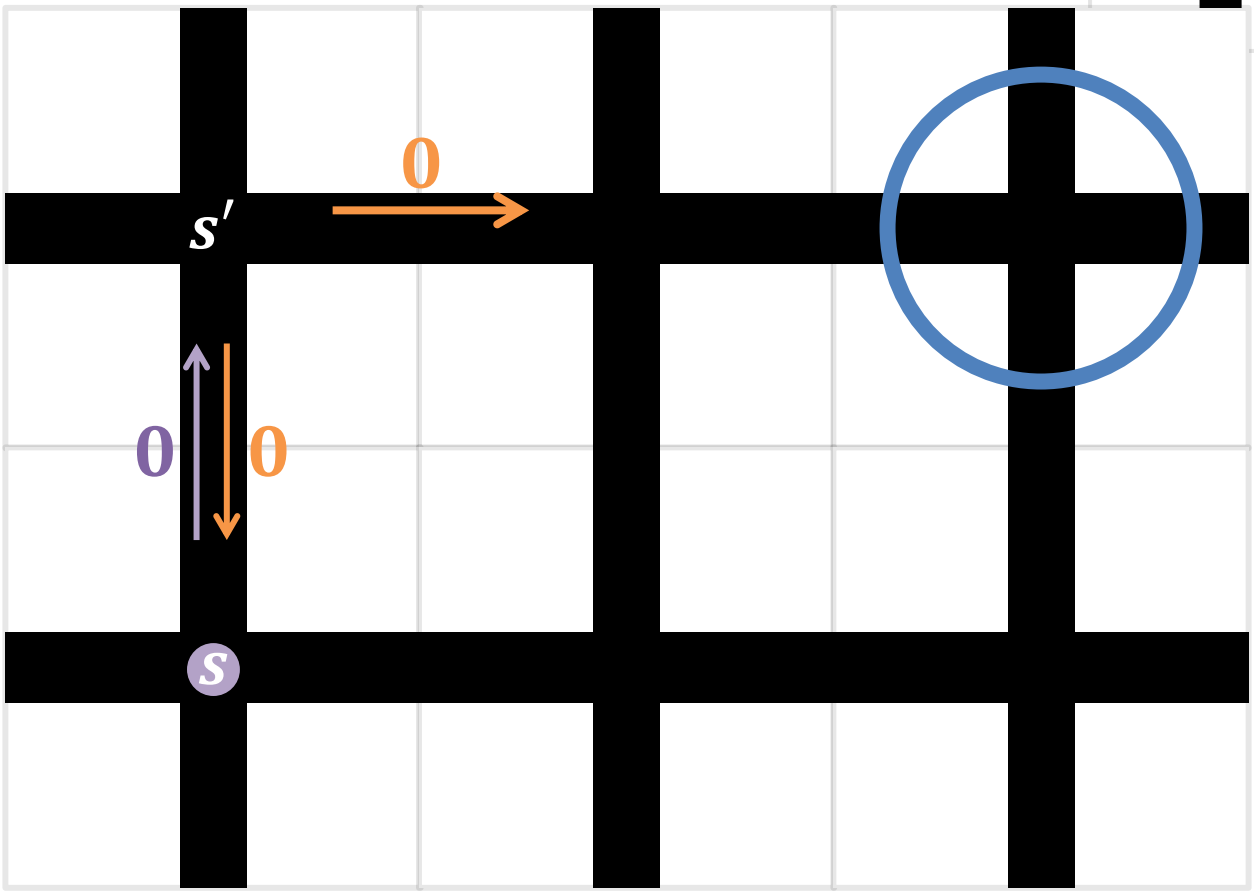
$\hat{Q}$ 가 같으면 랜덤으로 행동 선택

Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

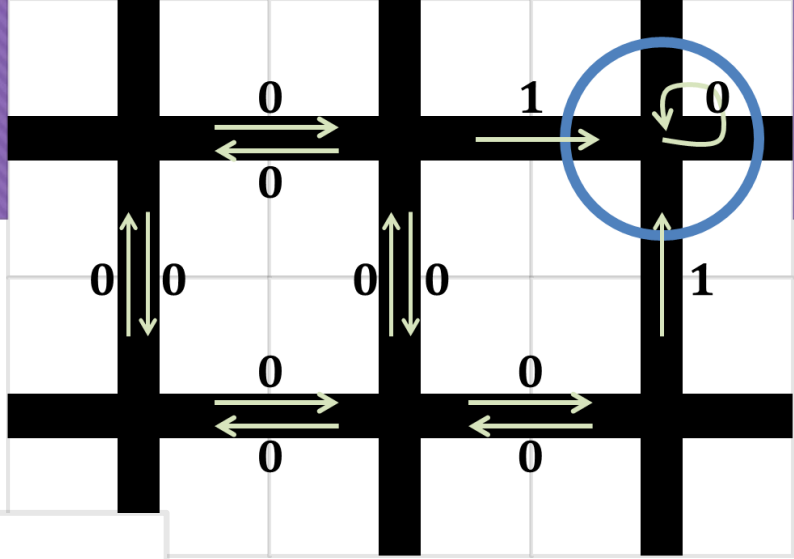
0 0

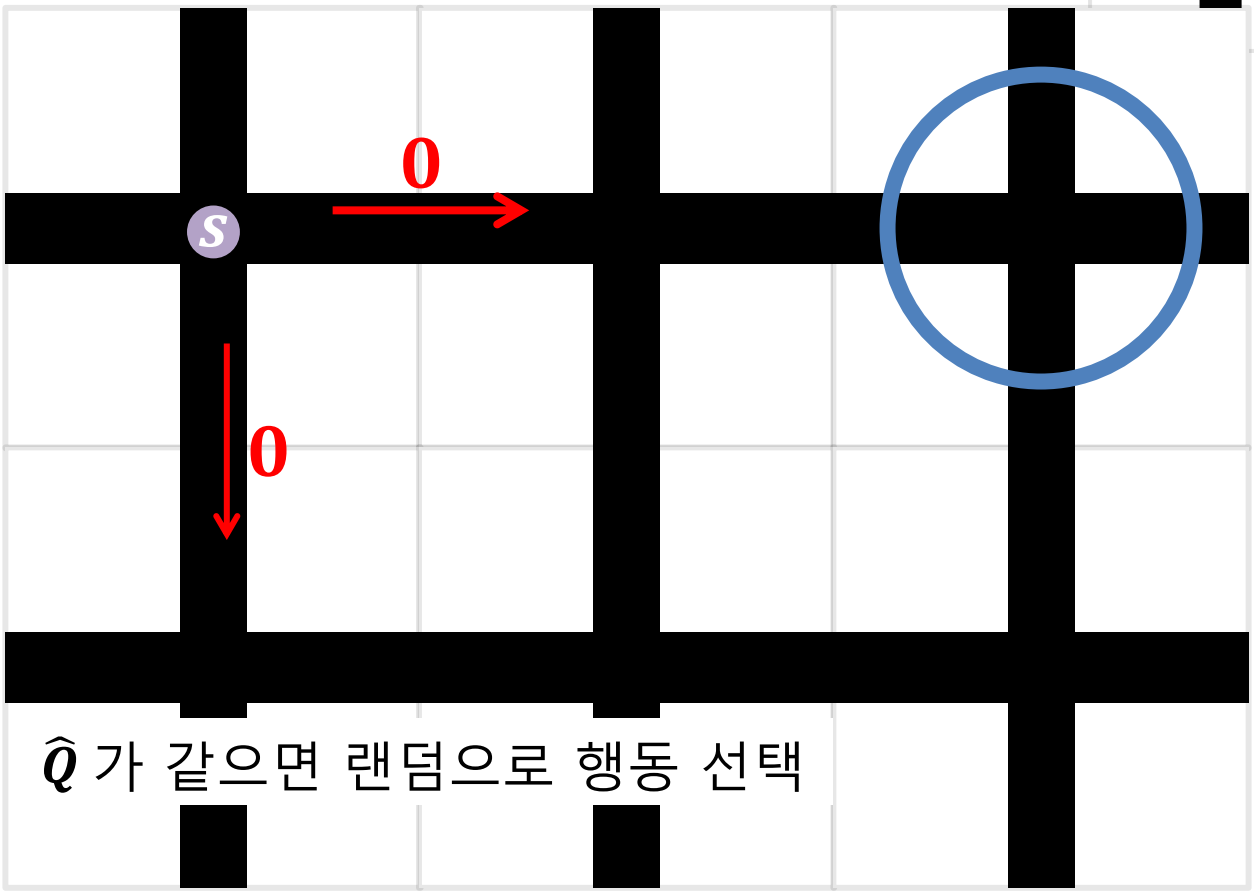




Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$



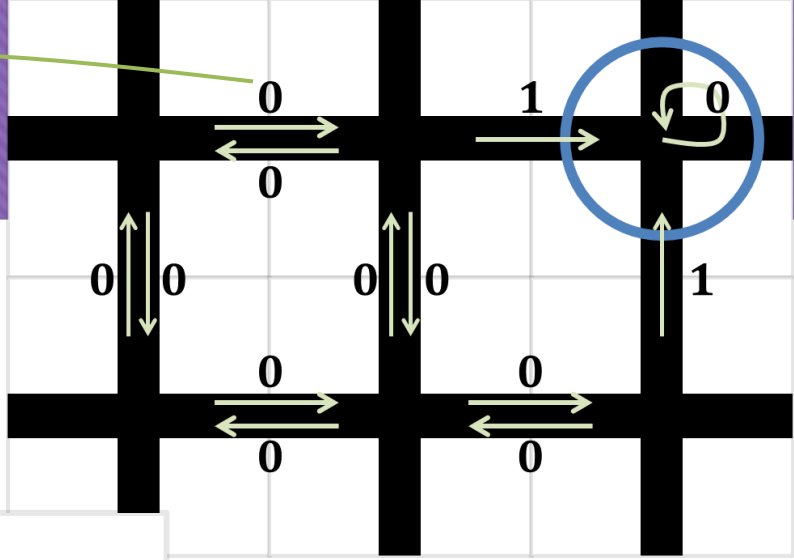


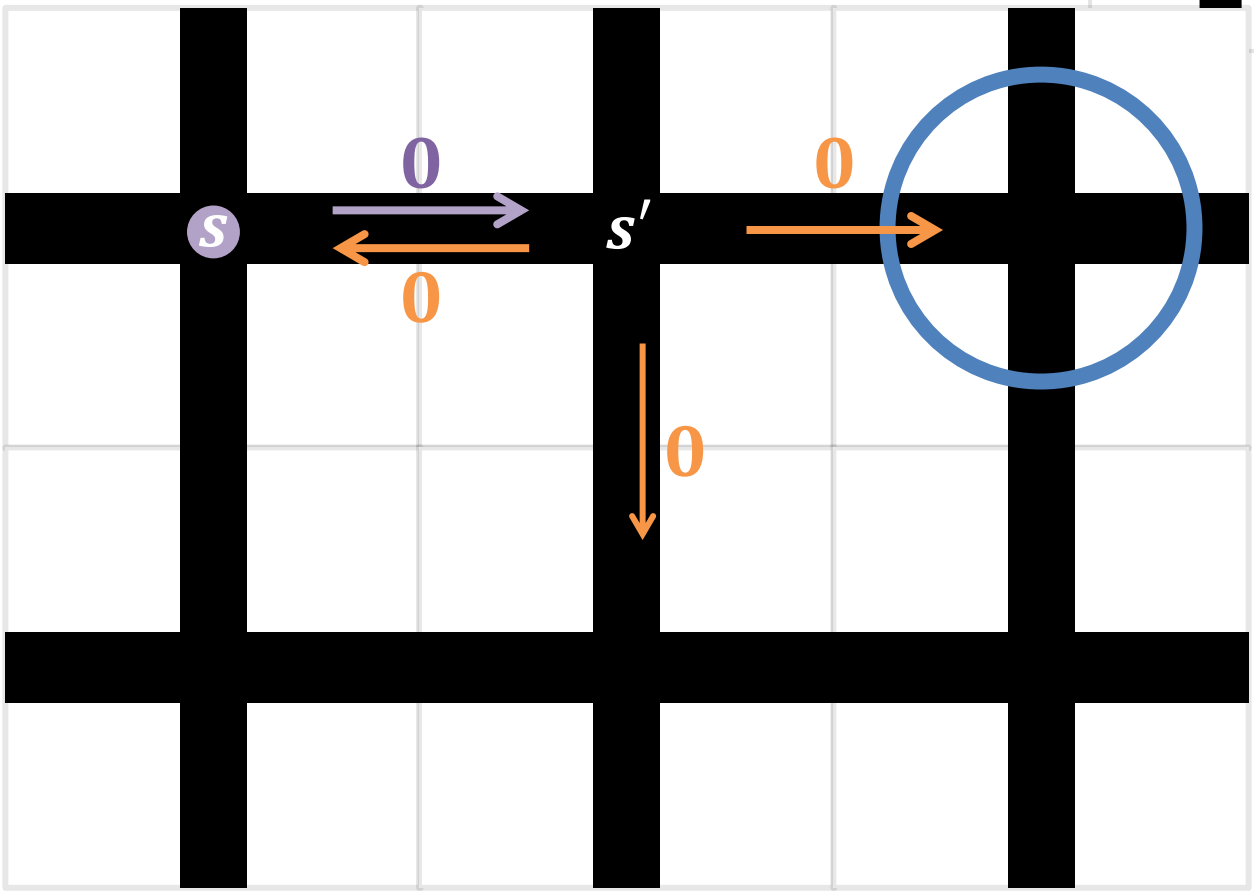
$\hat{Q}$ 가 같으면 랜덤으로 행동 선택

Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

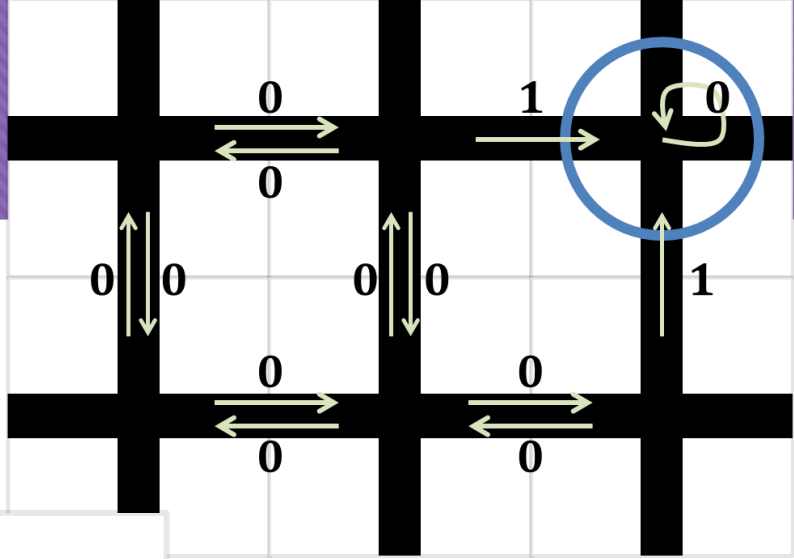
0 0

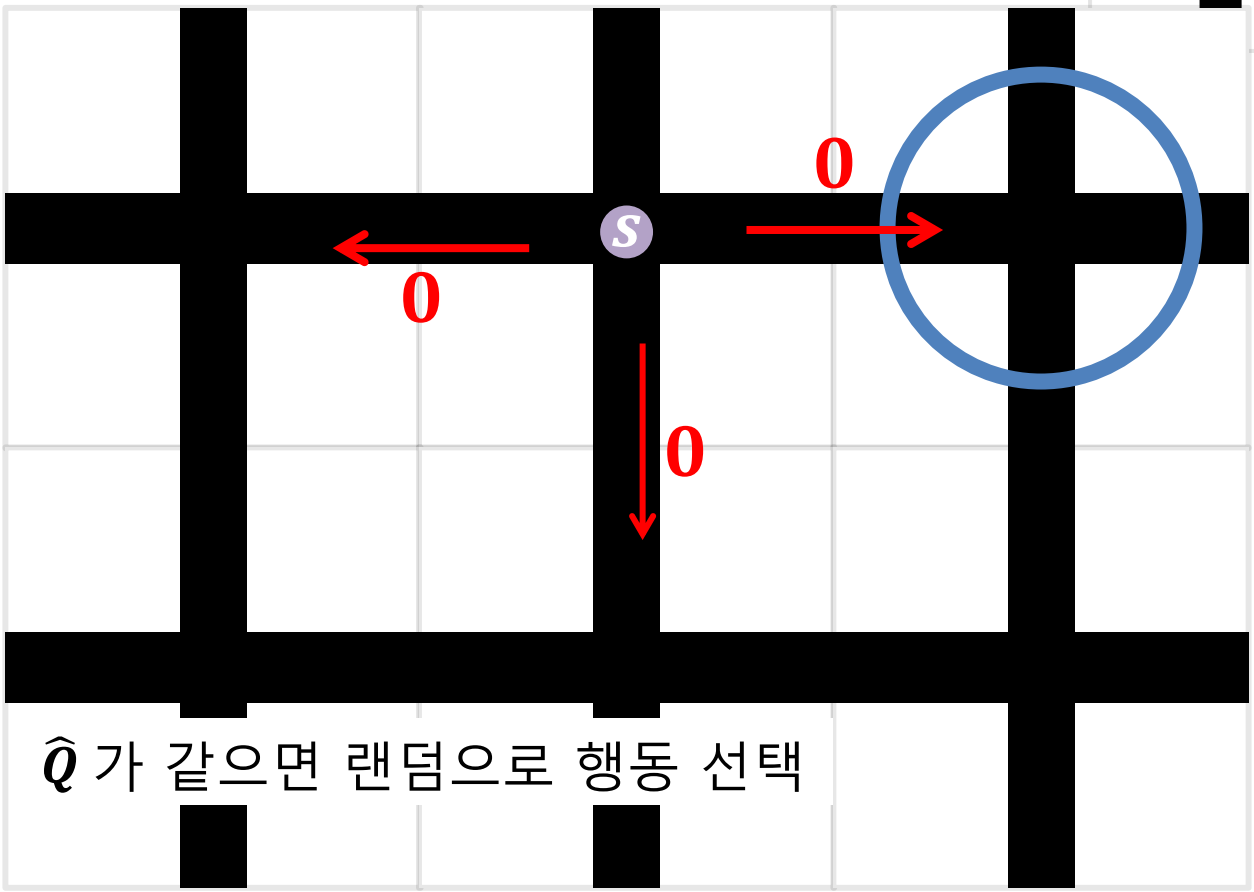




Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$



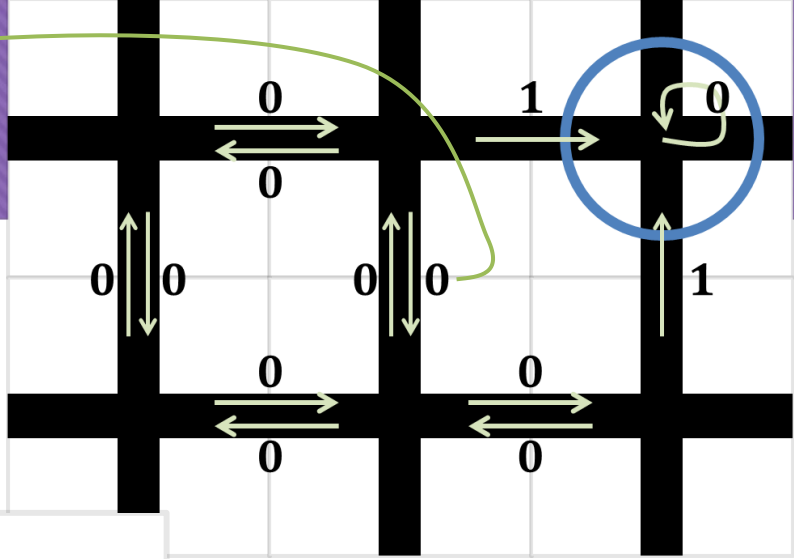


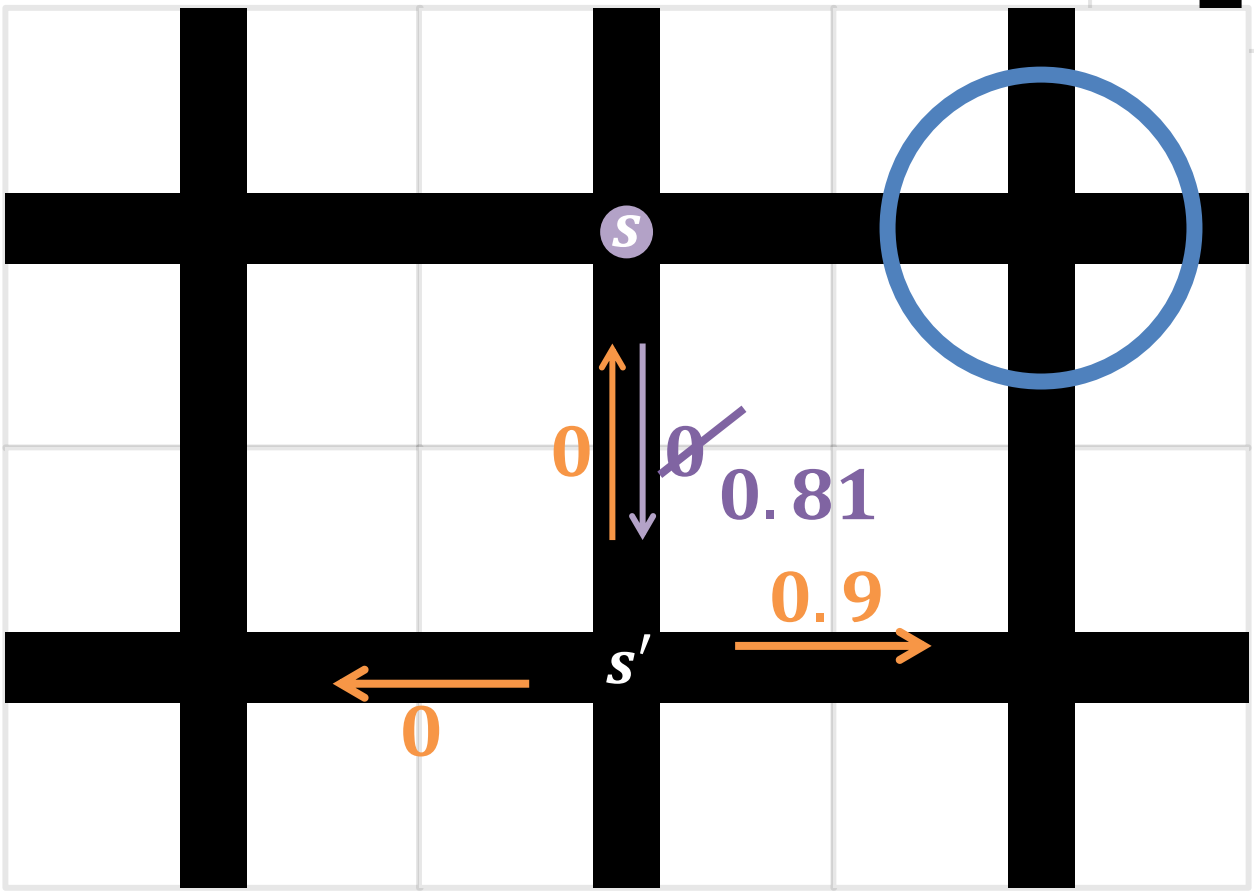
$\hat{Q}$ 가 같으면 랜덤으로 행동 선택

Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

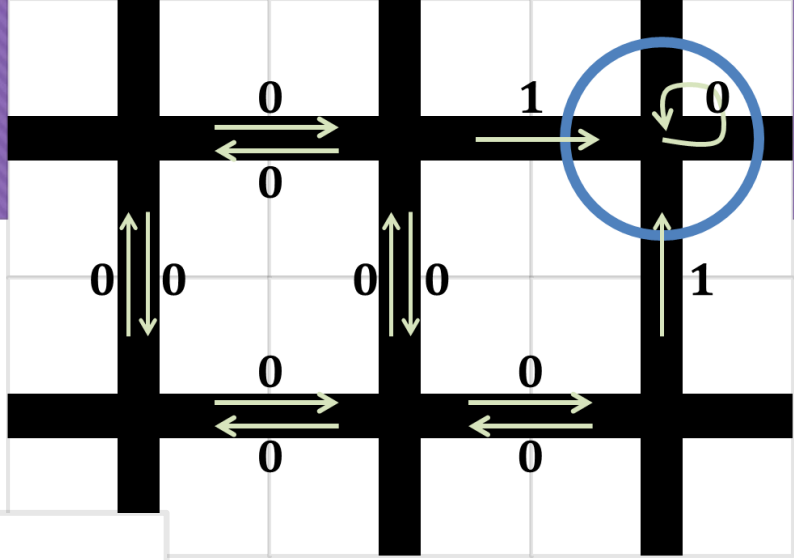
0 0.9

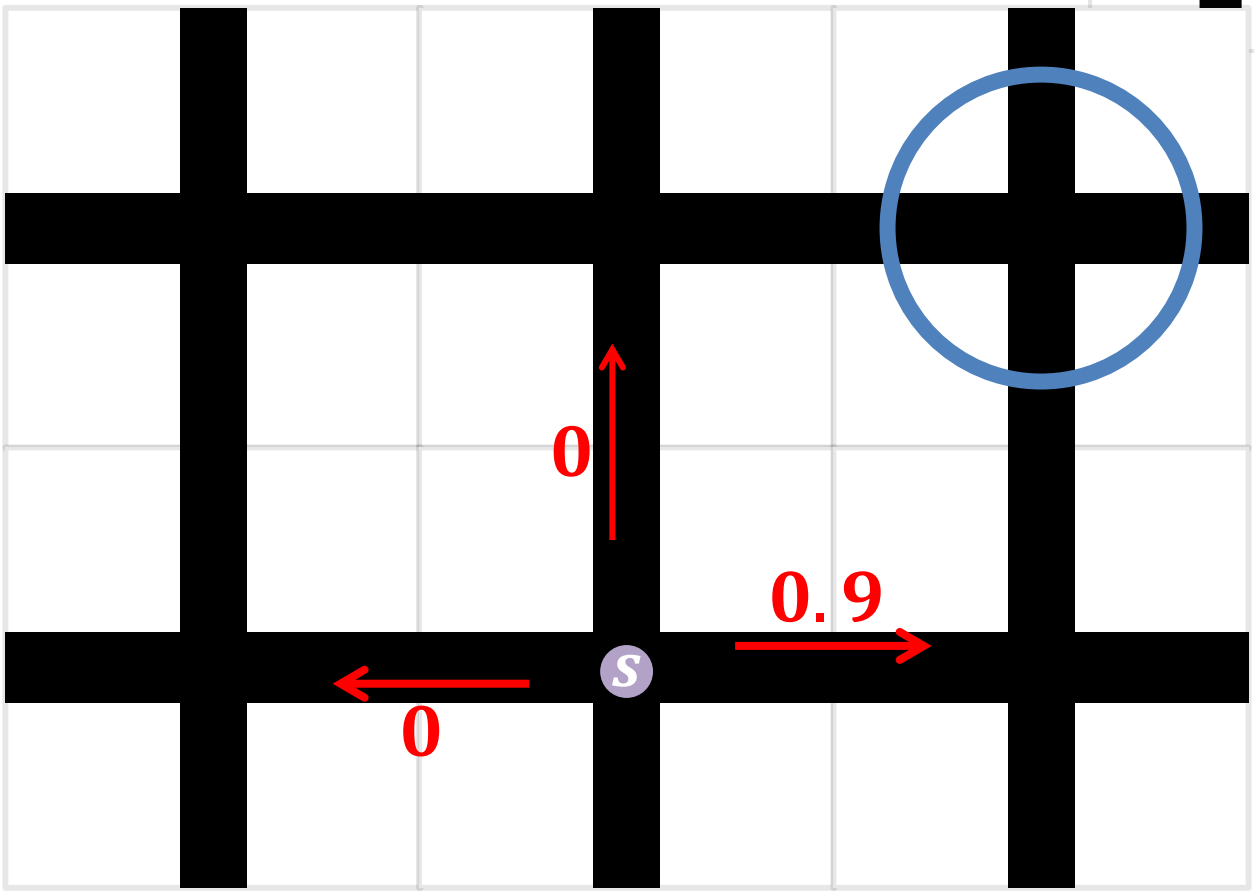




Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

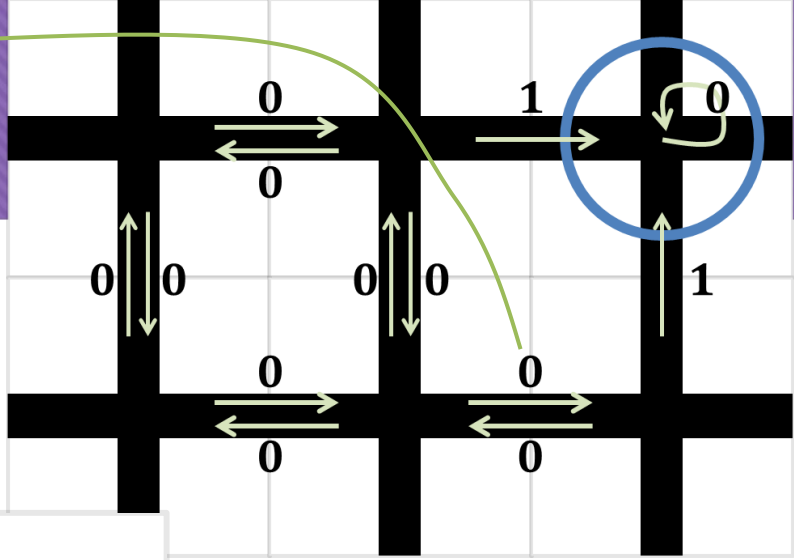


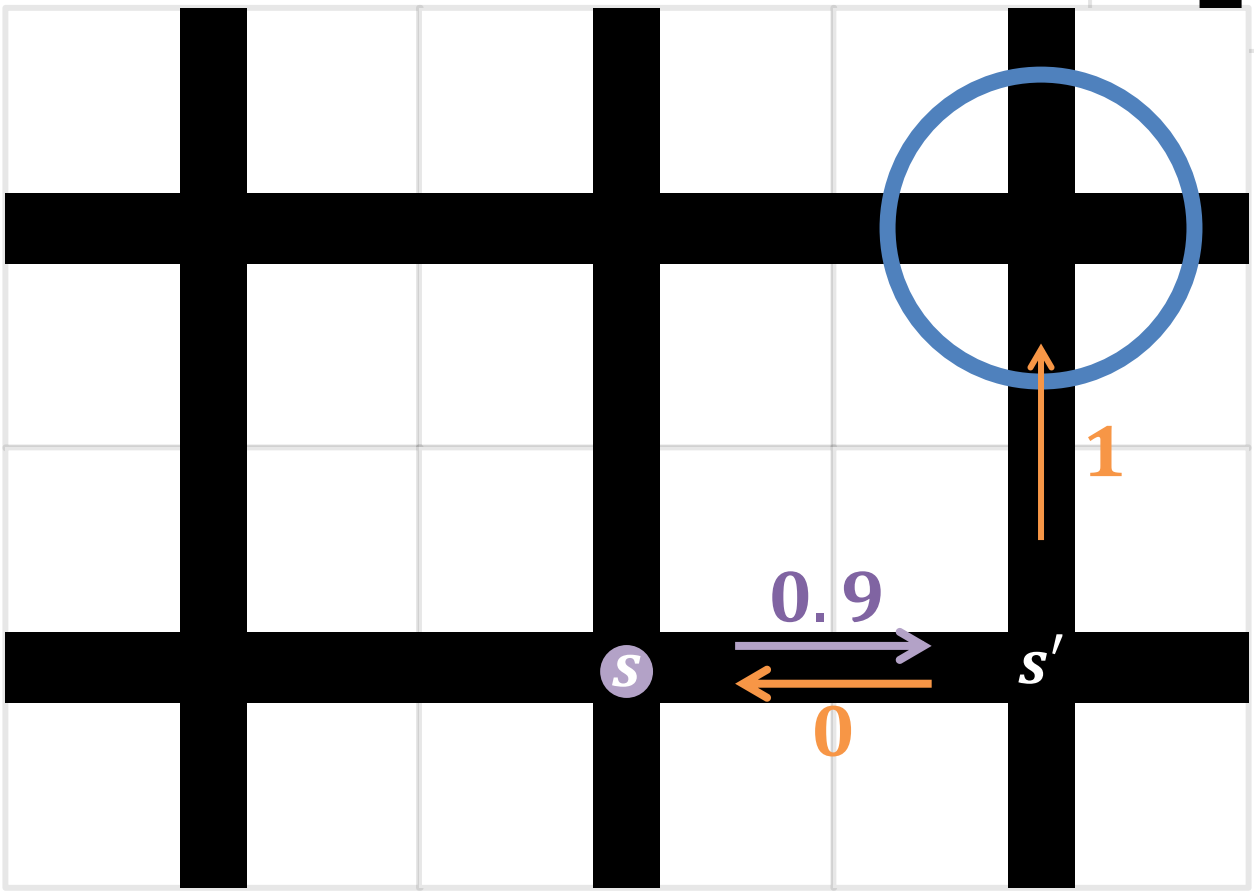


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

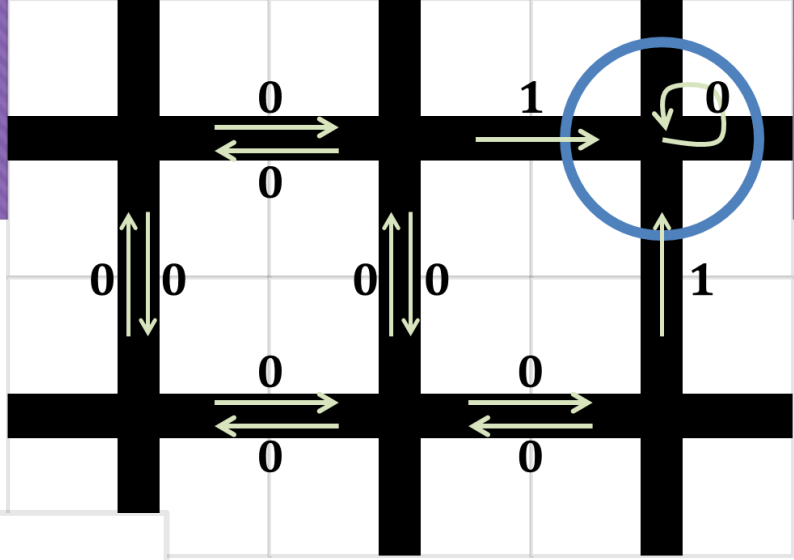
0 1

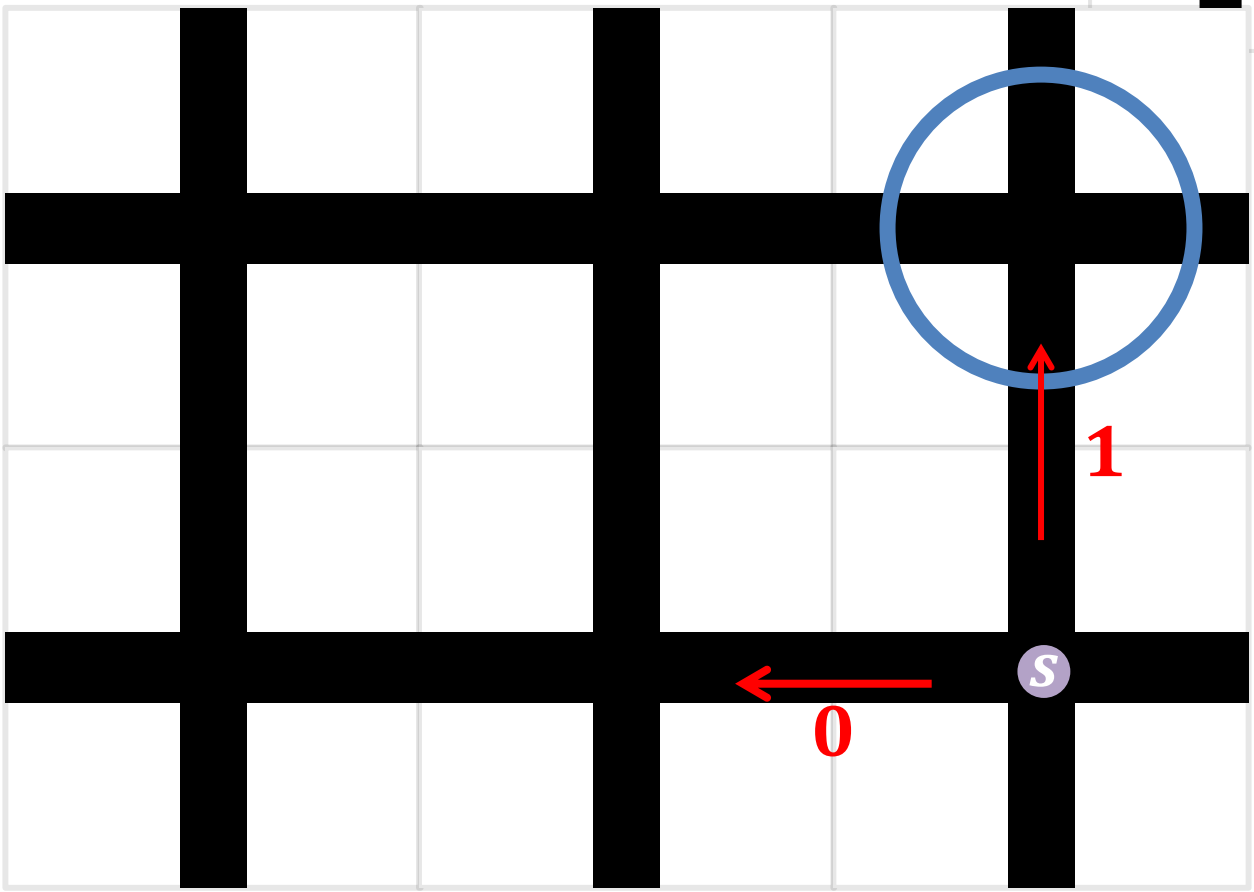




Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

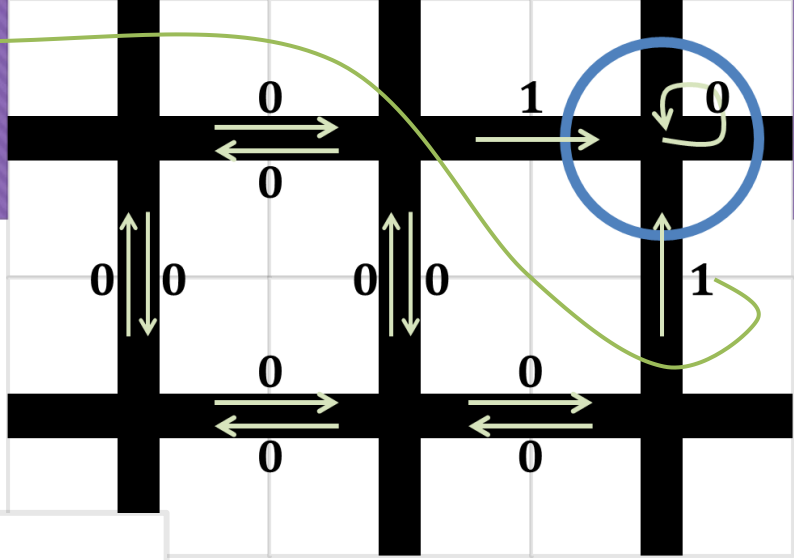


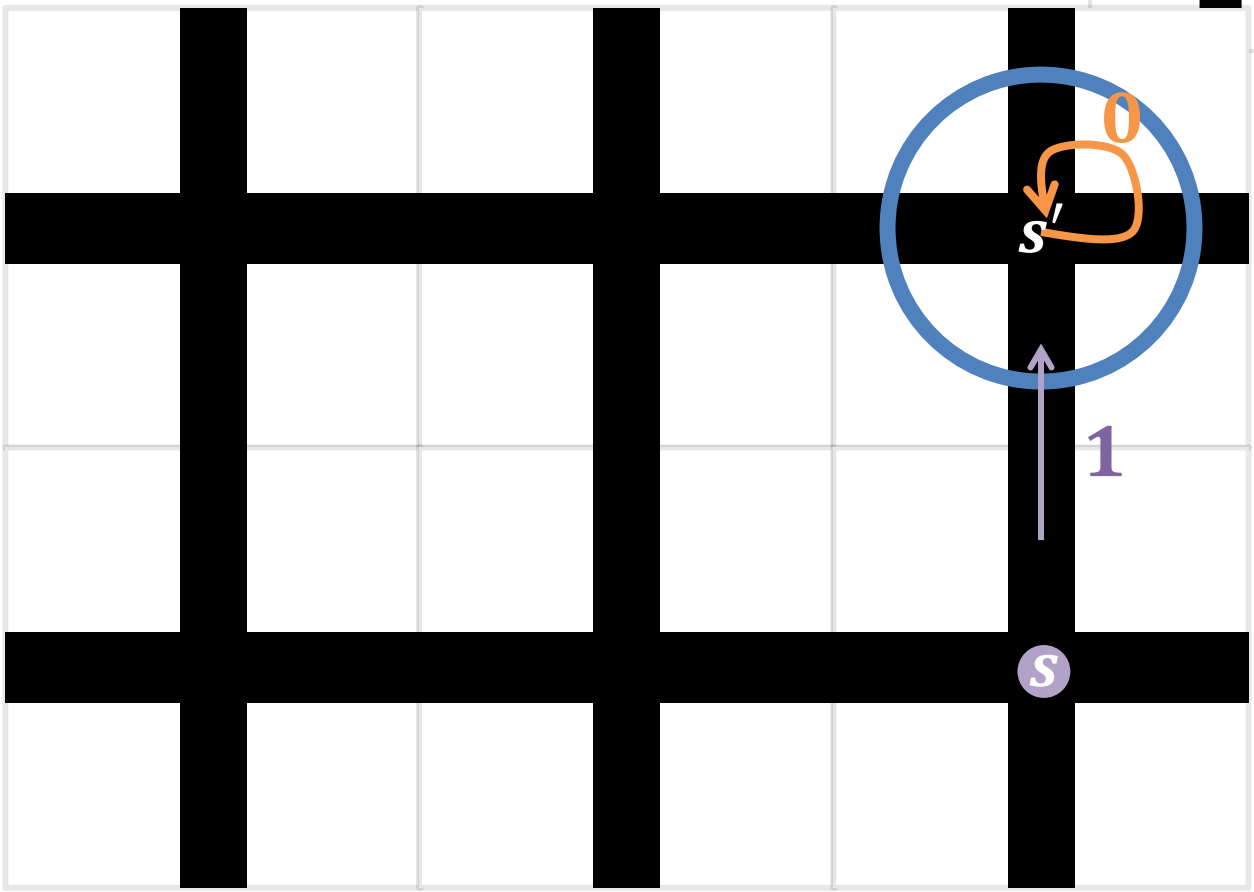


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

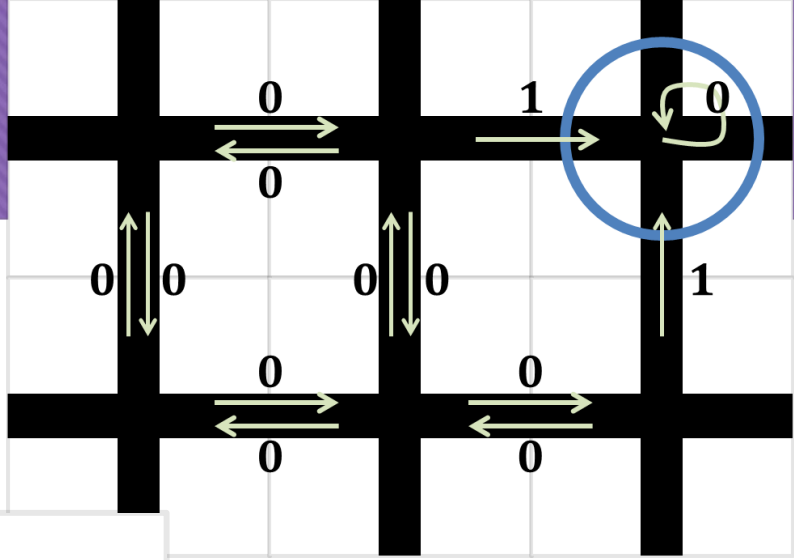
1 0

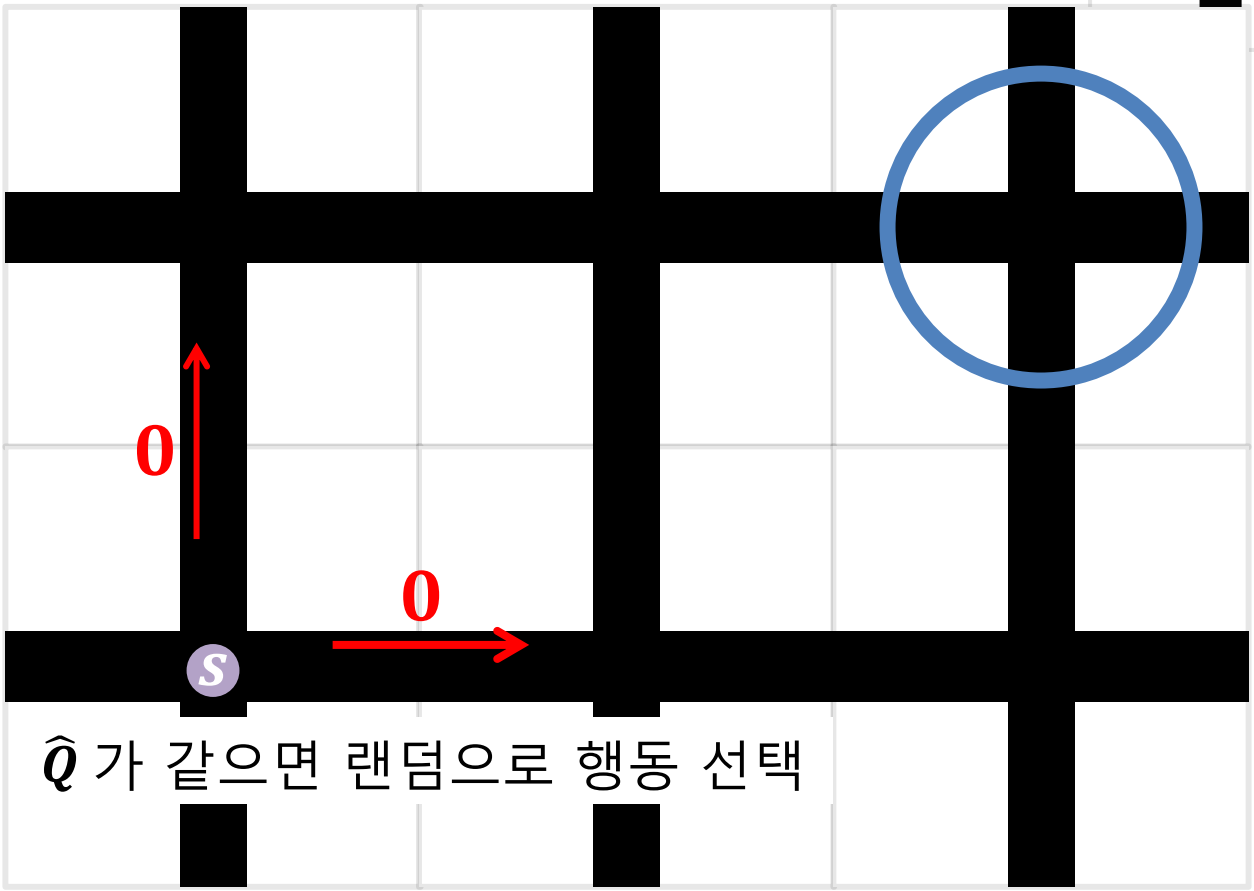




Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$



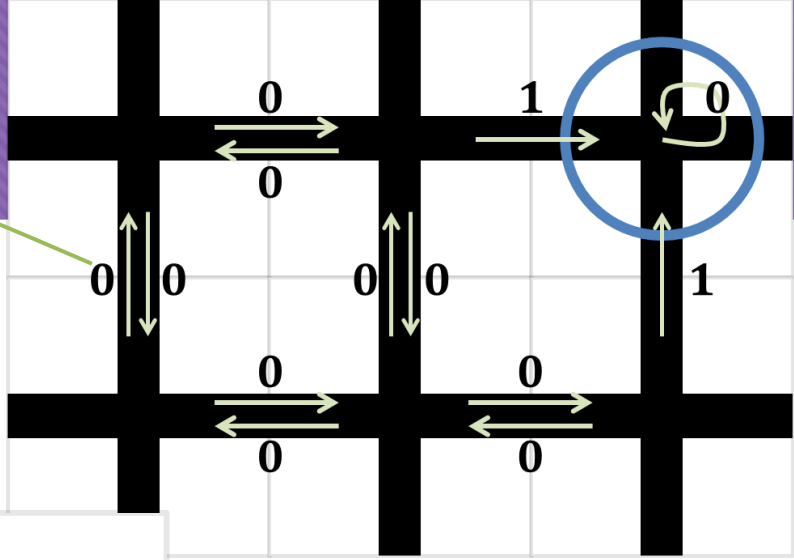


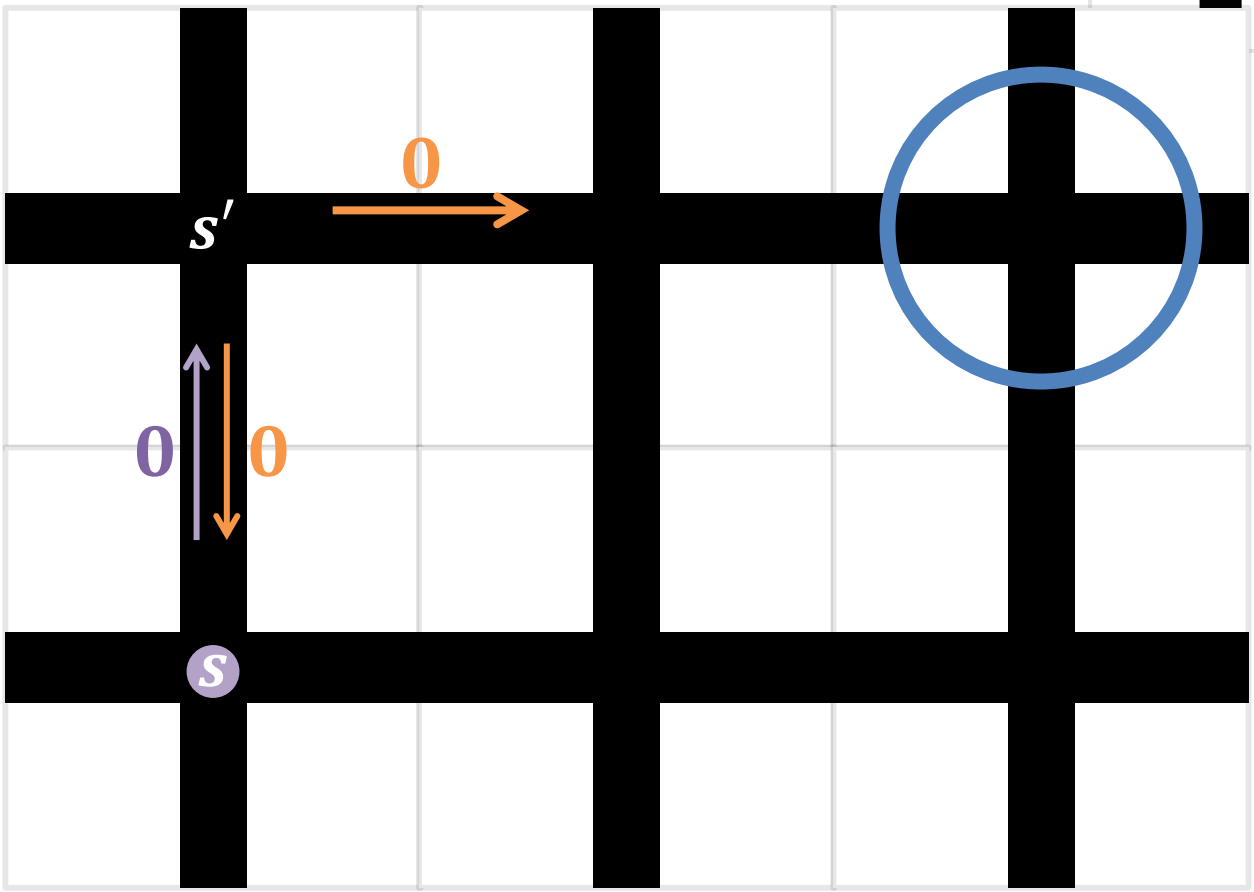
$\hat{Q}$ 가 같으면 랜덤으로 행동 선택

Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

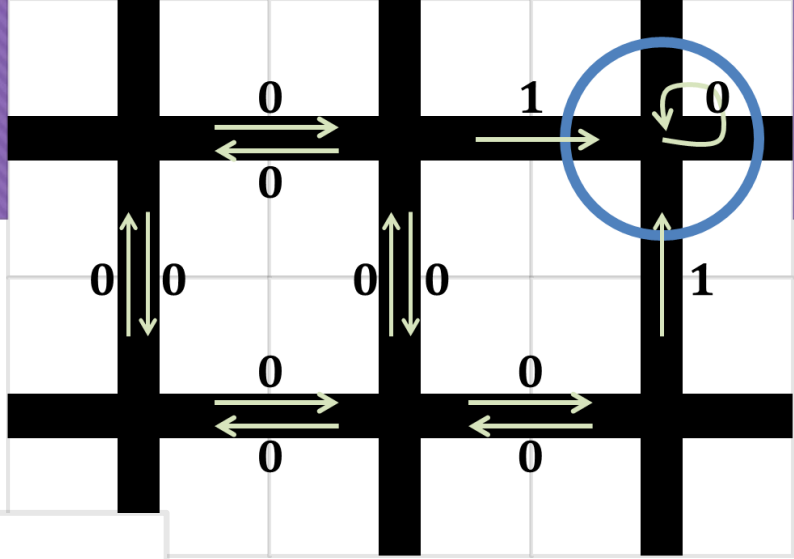
0 0

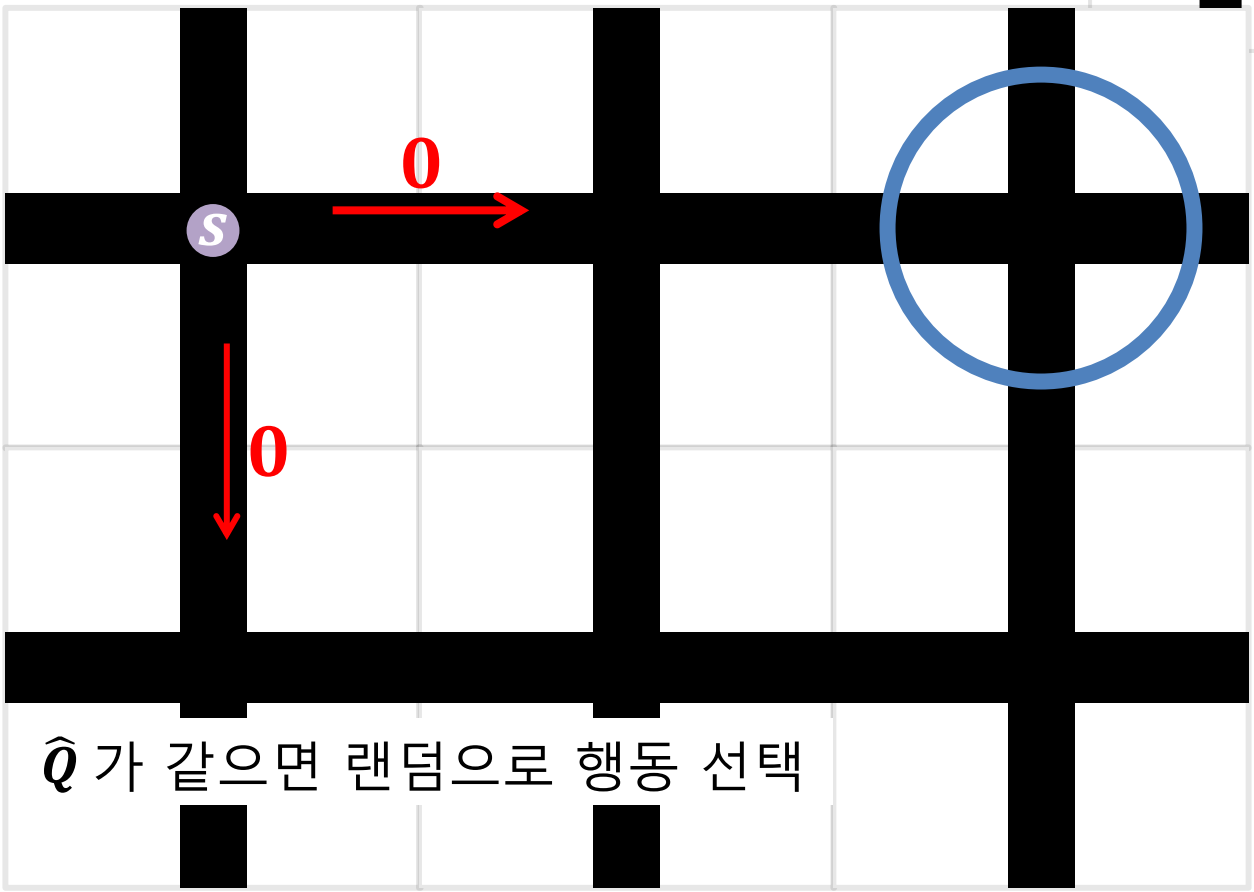




Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$



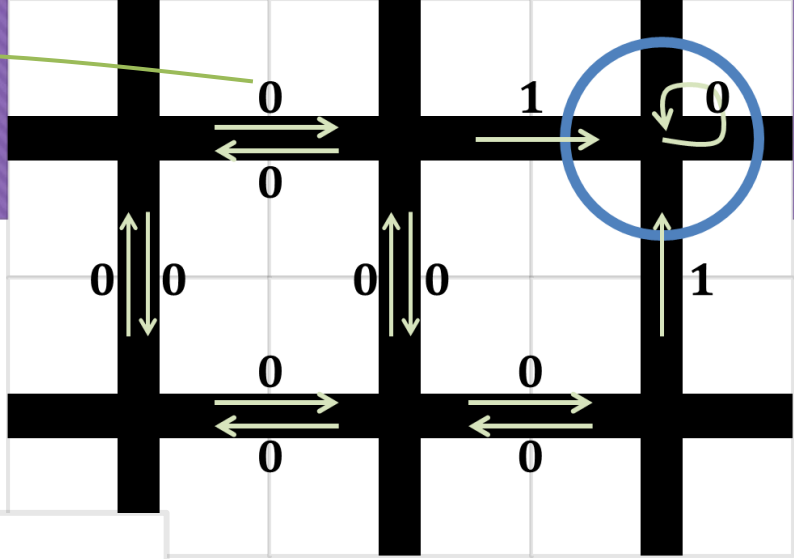


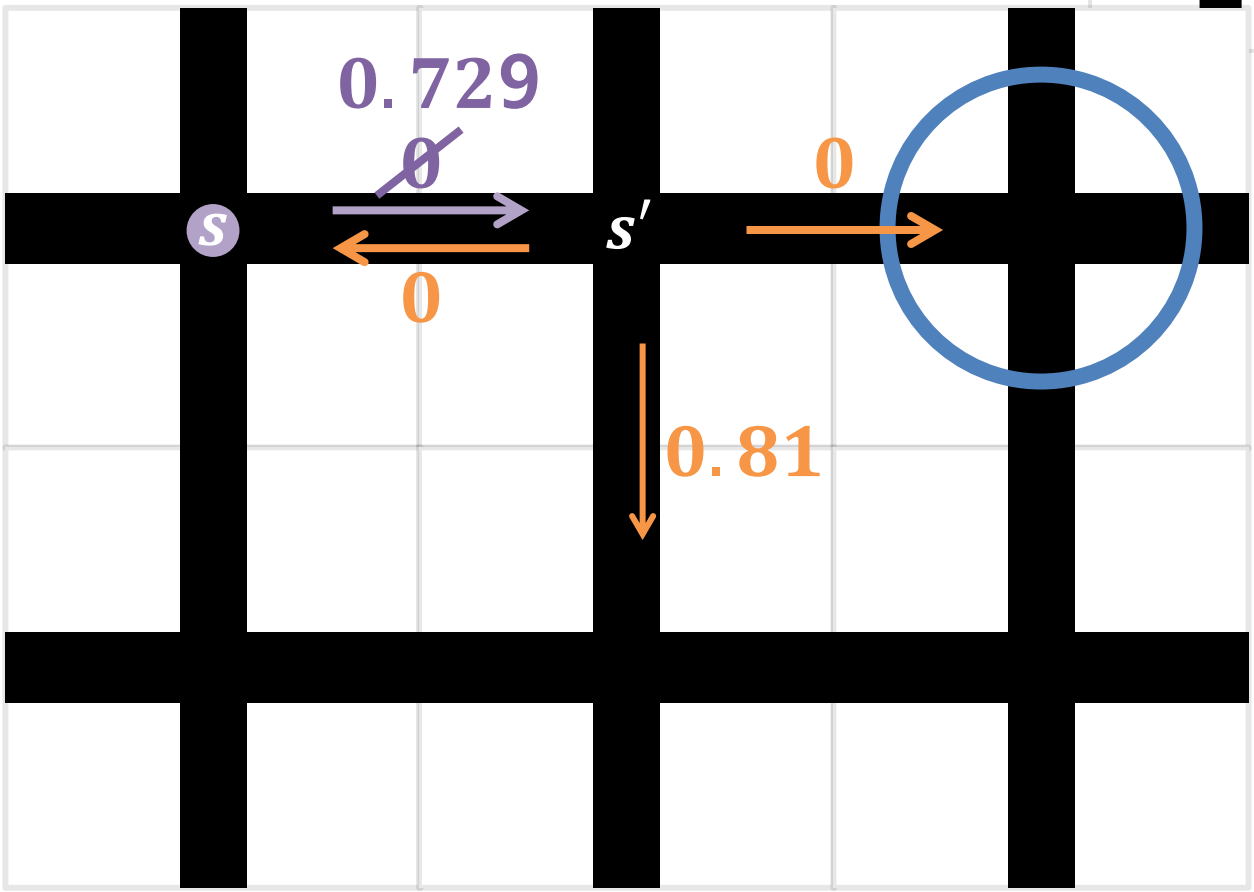
$\hat{Q}$ 가 같으면 랜덤으로 행동 선택

Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

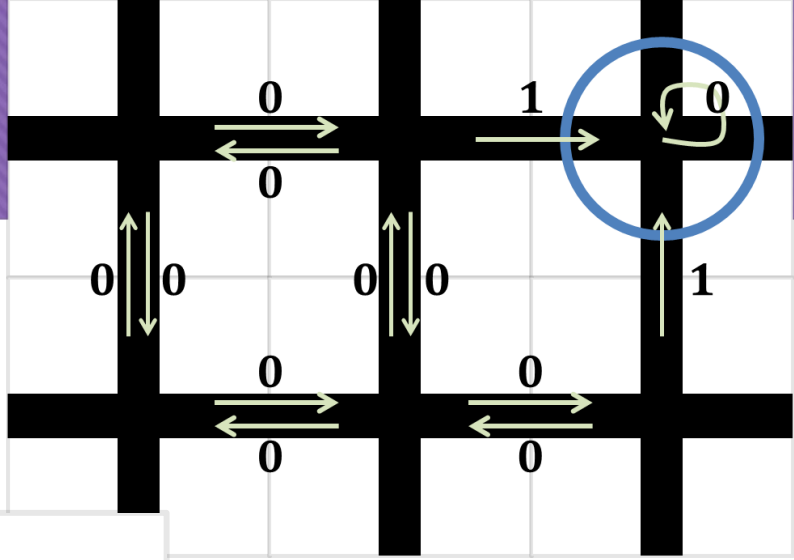
0 0.81

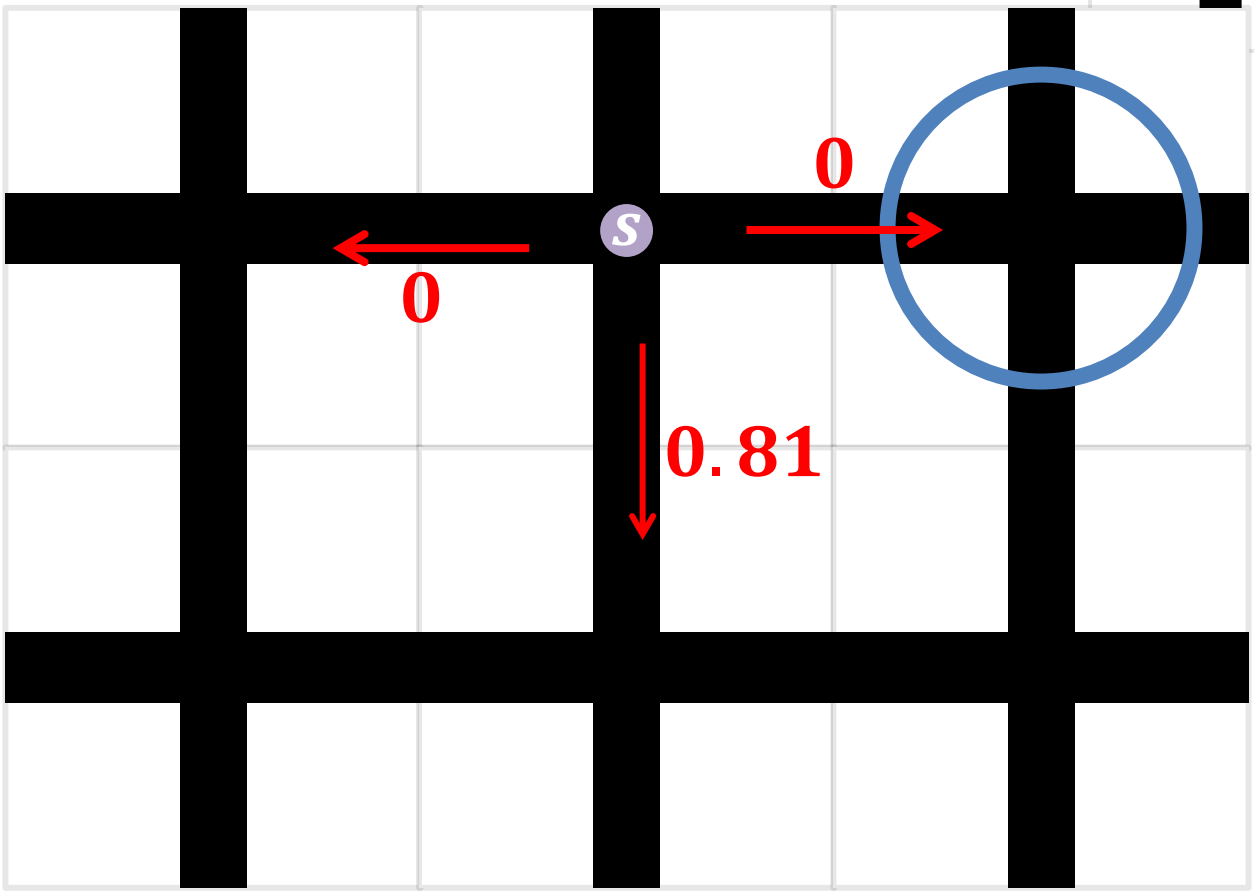




Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

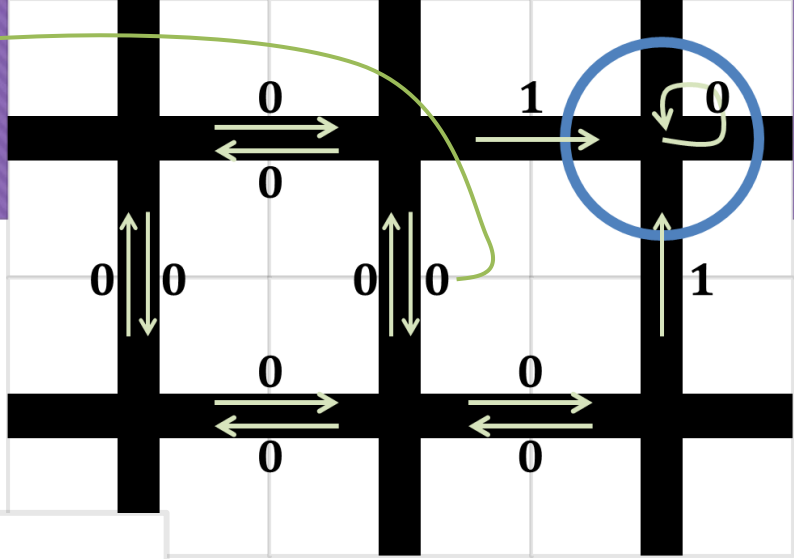


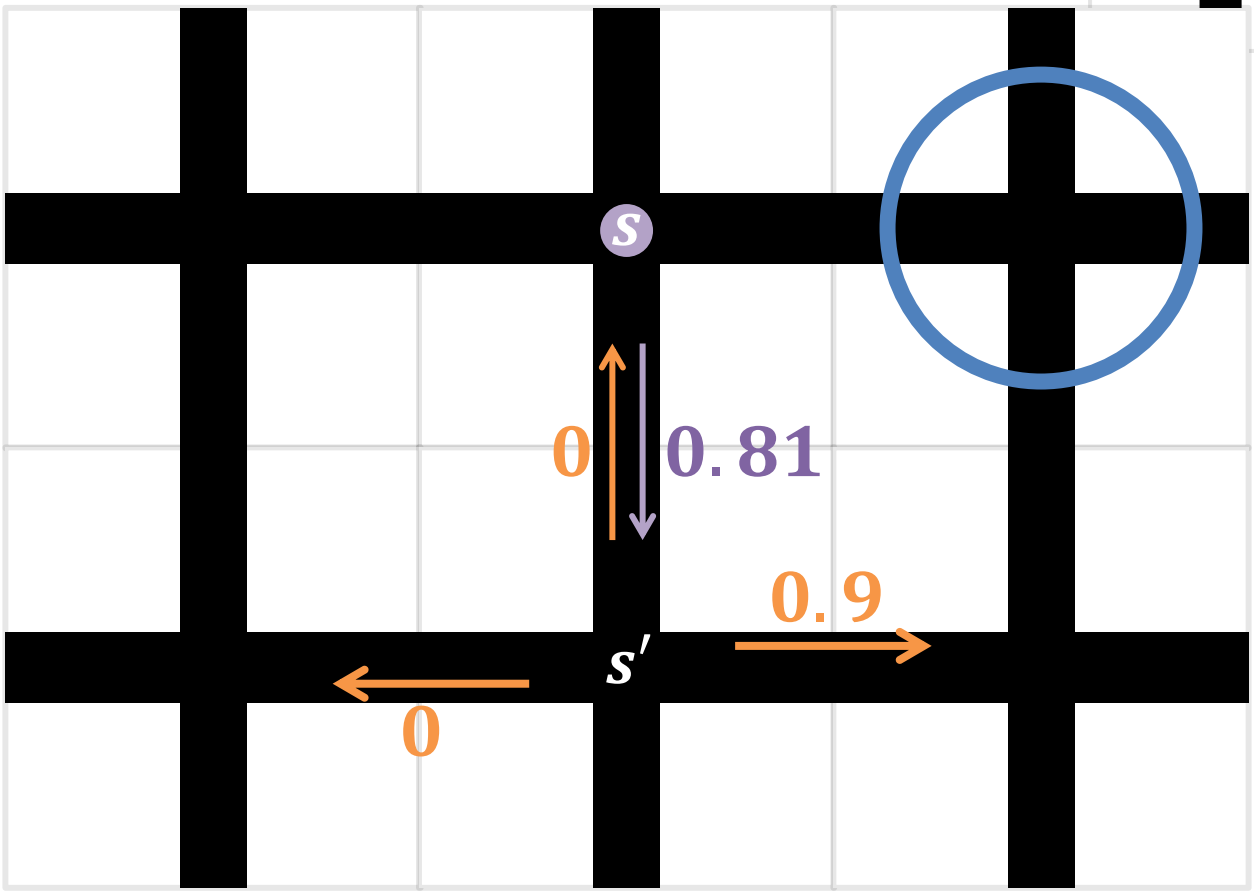


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

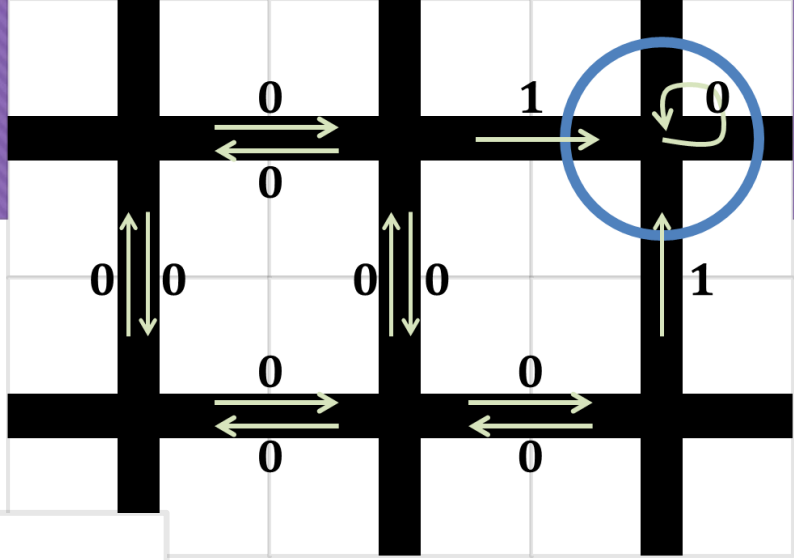
0 0.9

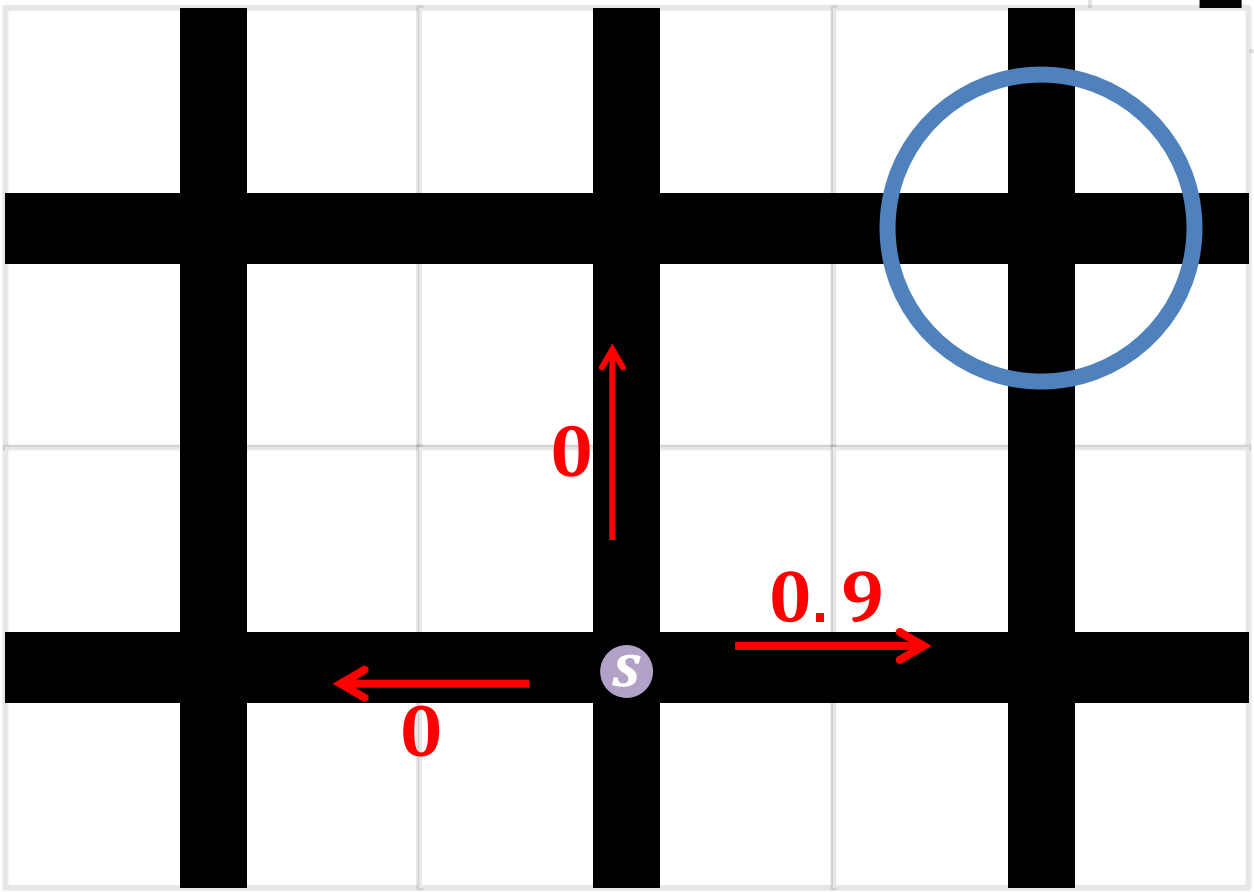




Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

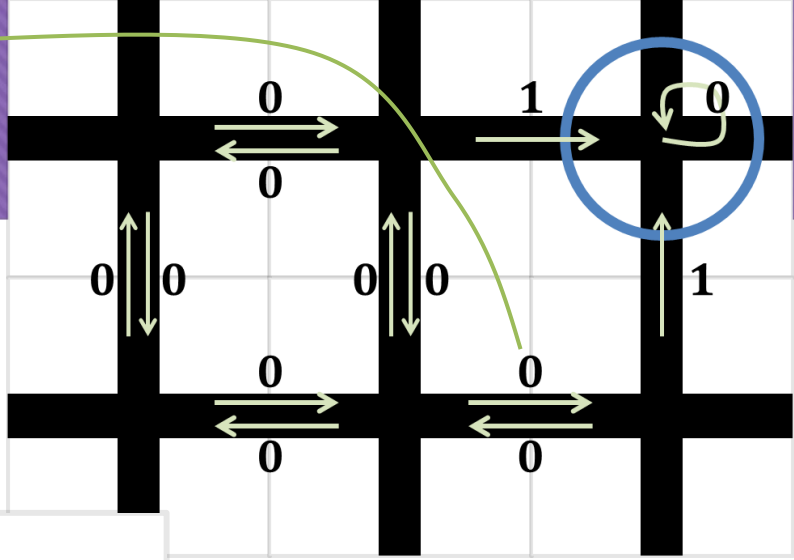


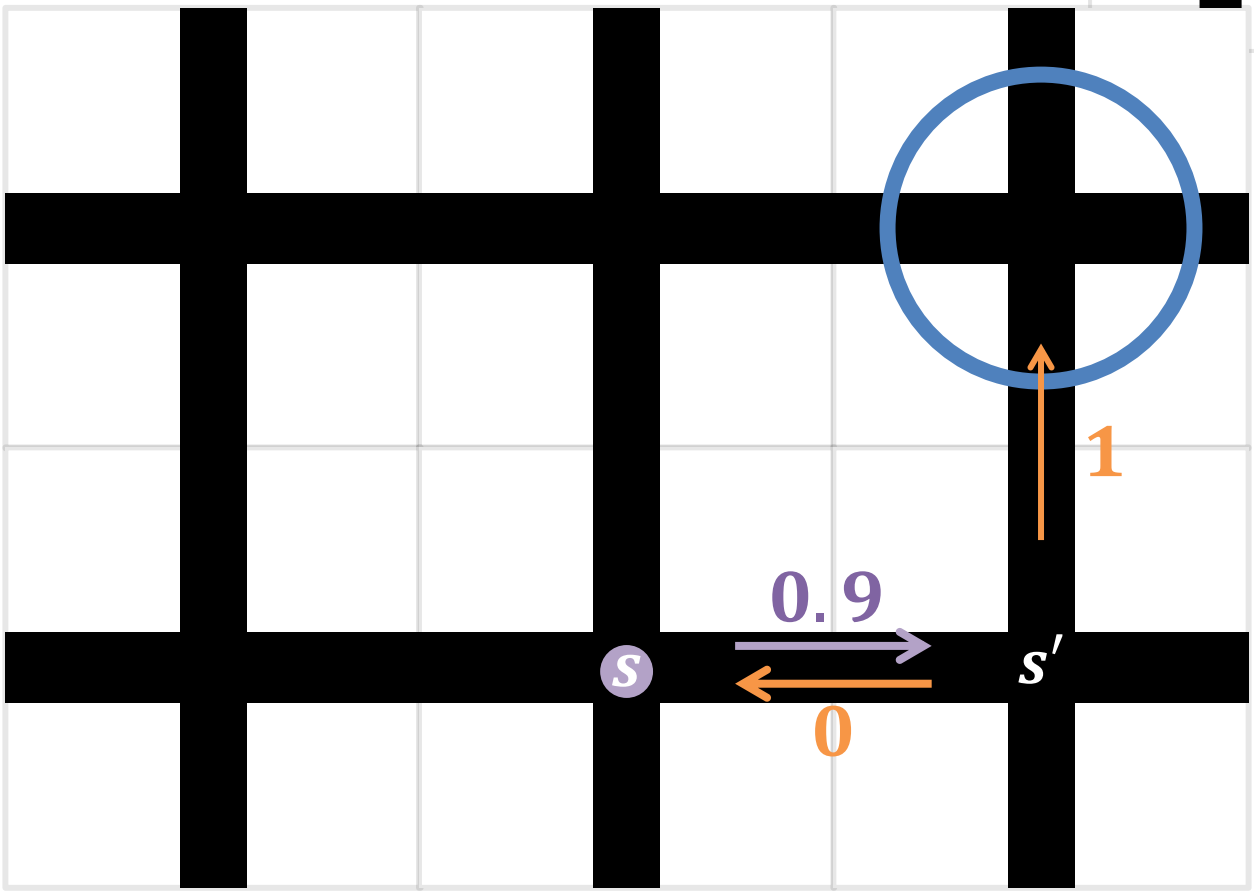


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

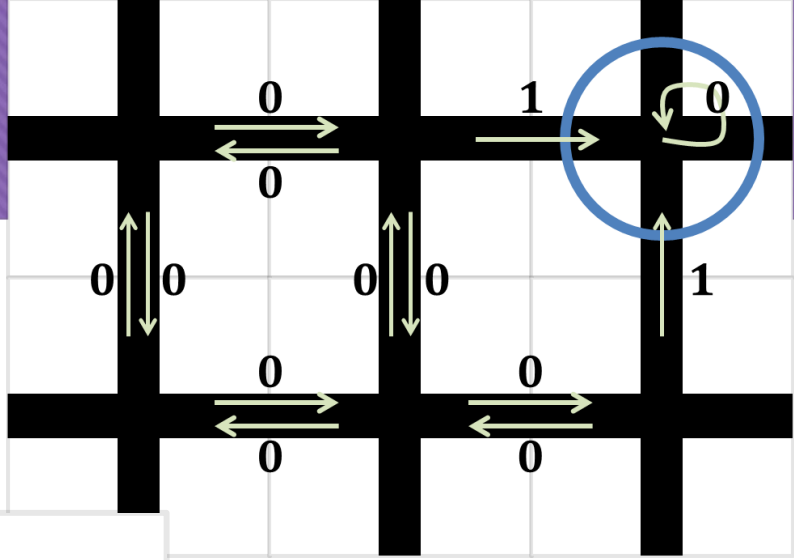
0 1

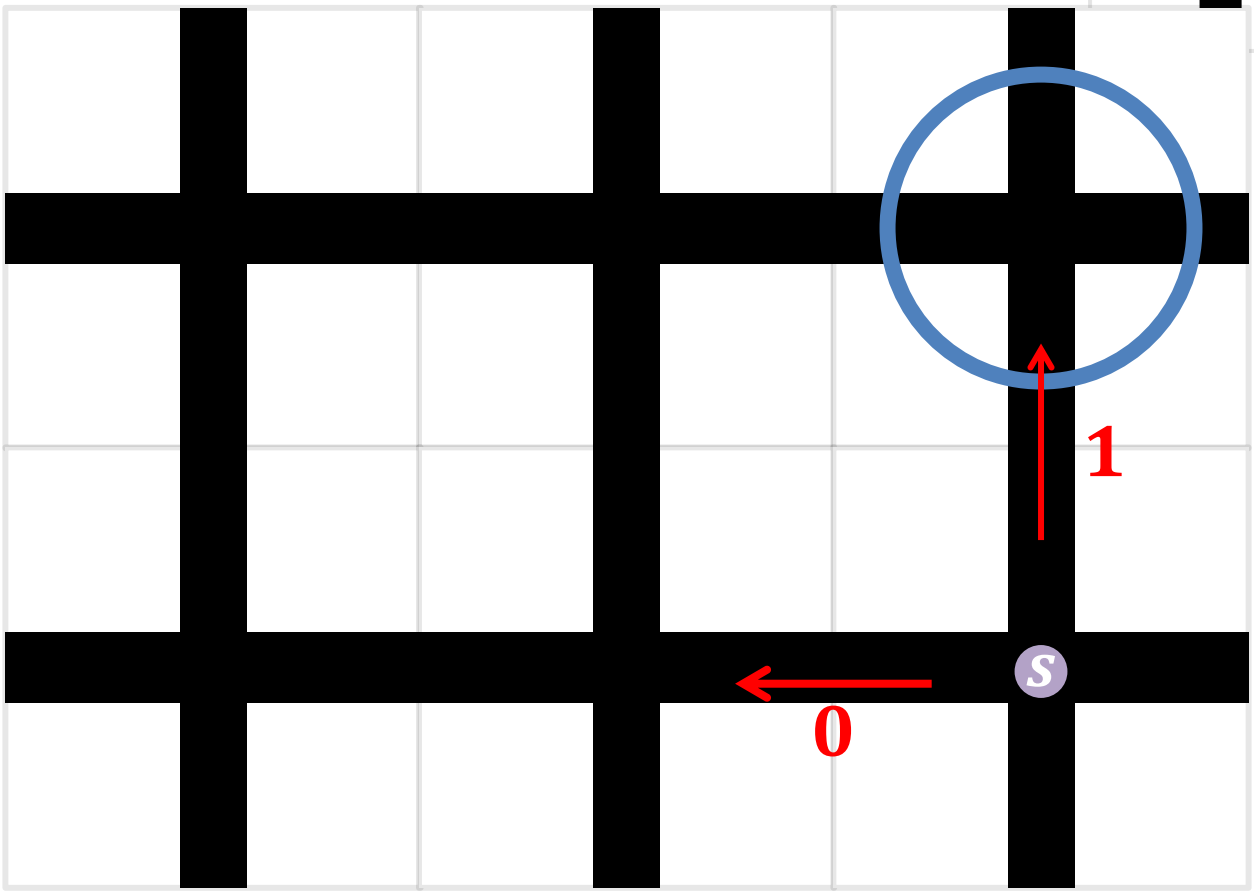




Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

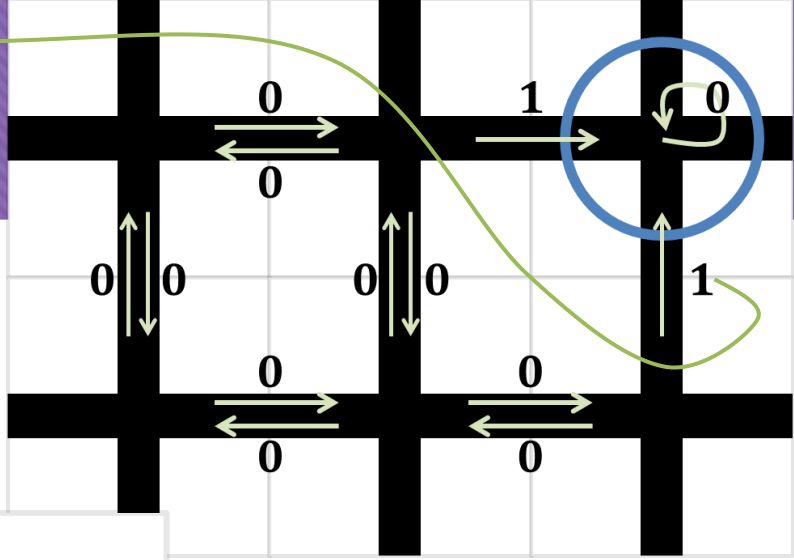


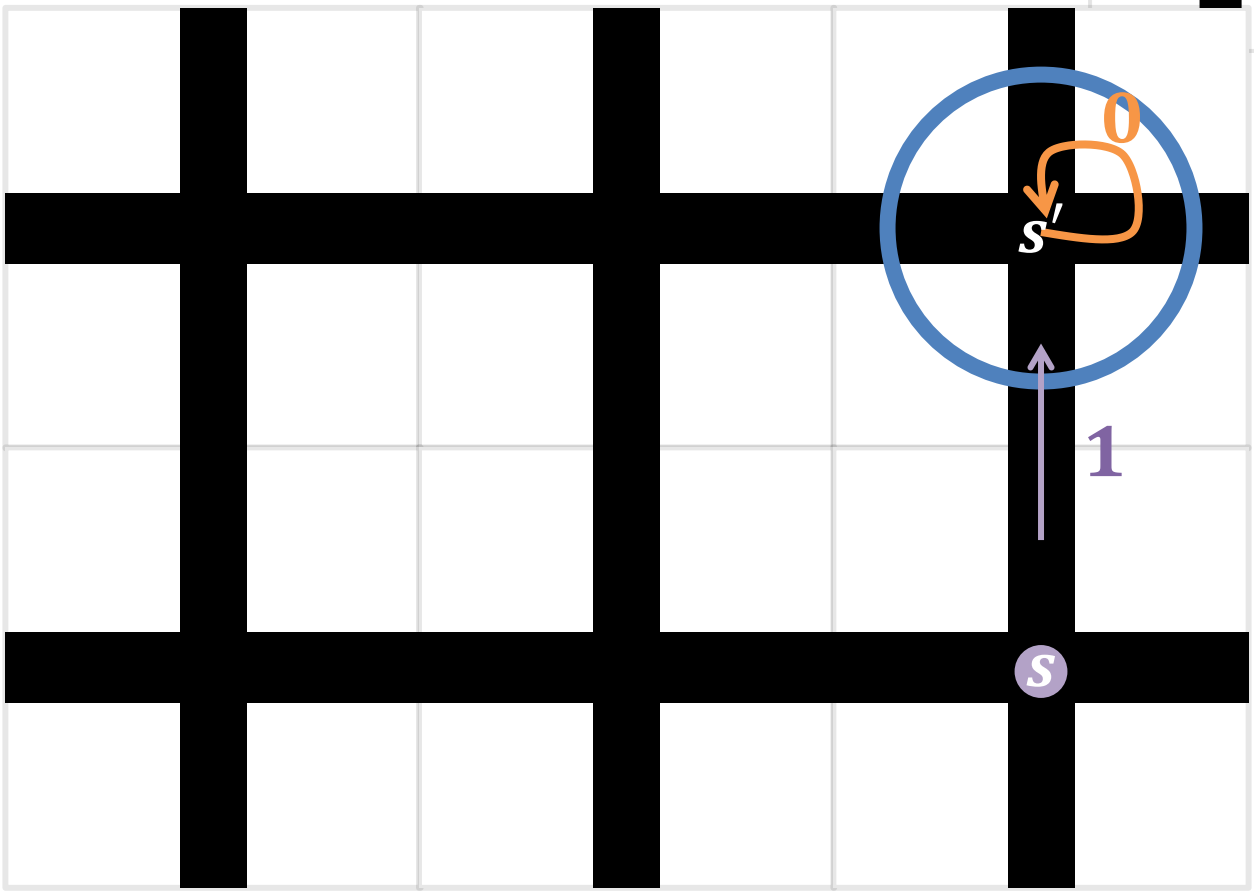


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

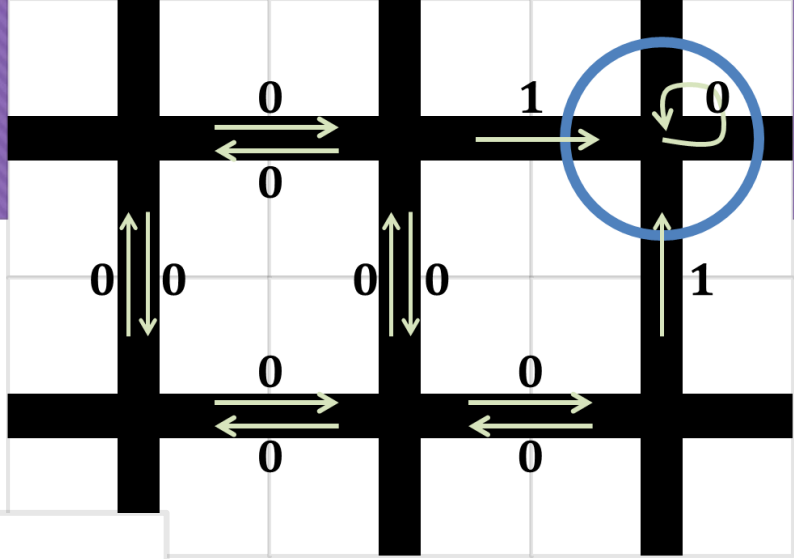
1 0

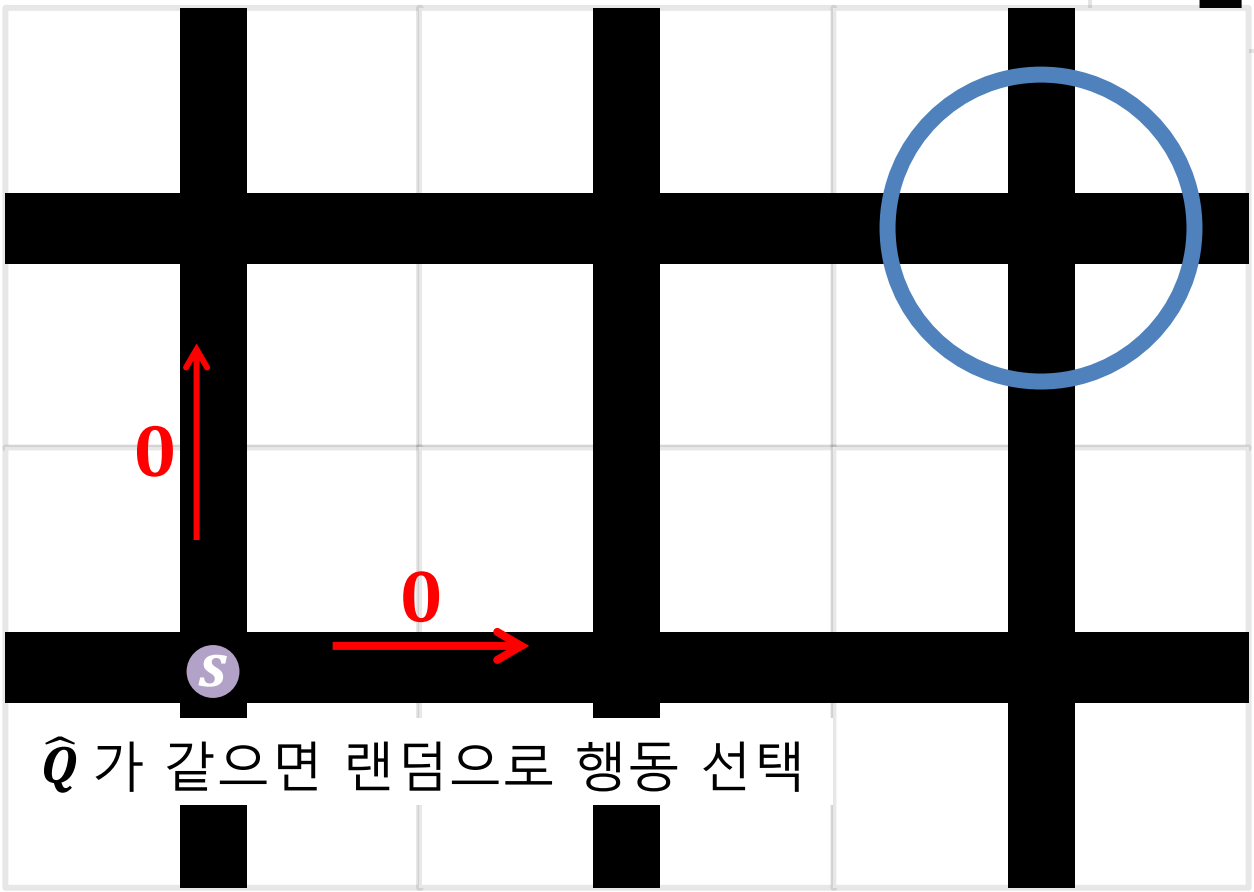




Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$



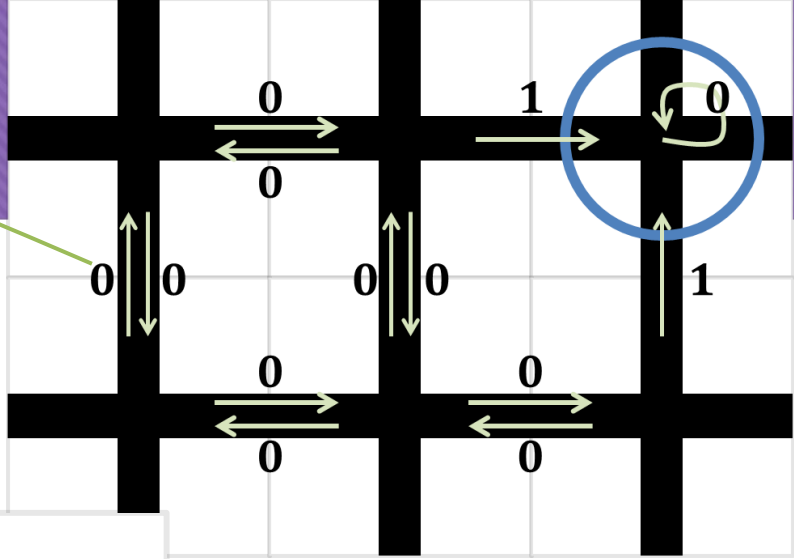


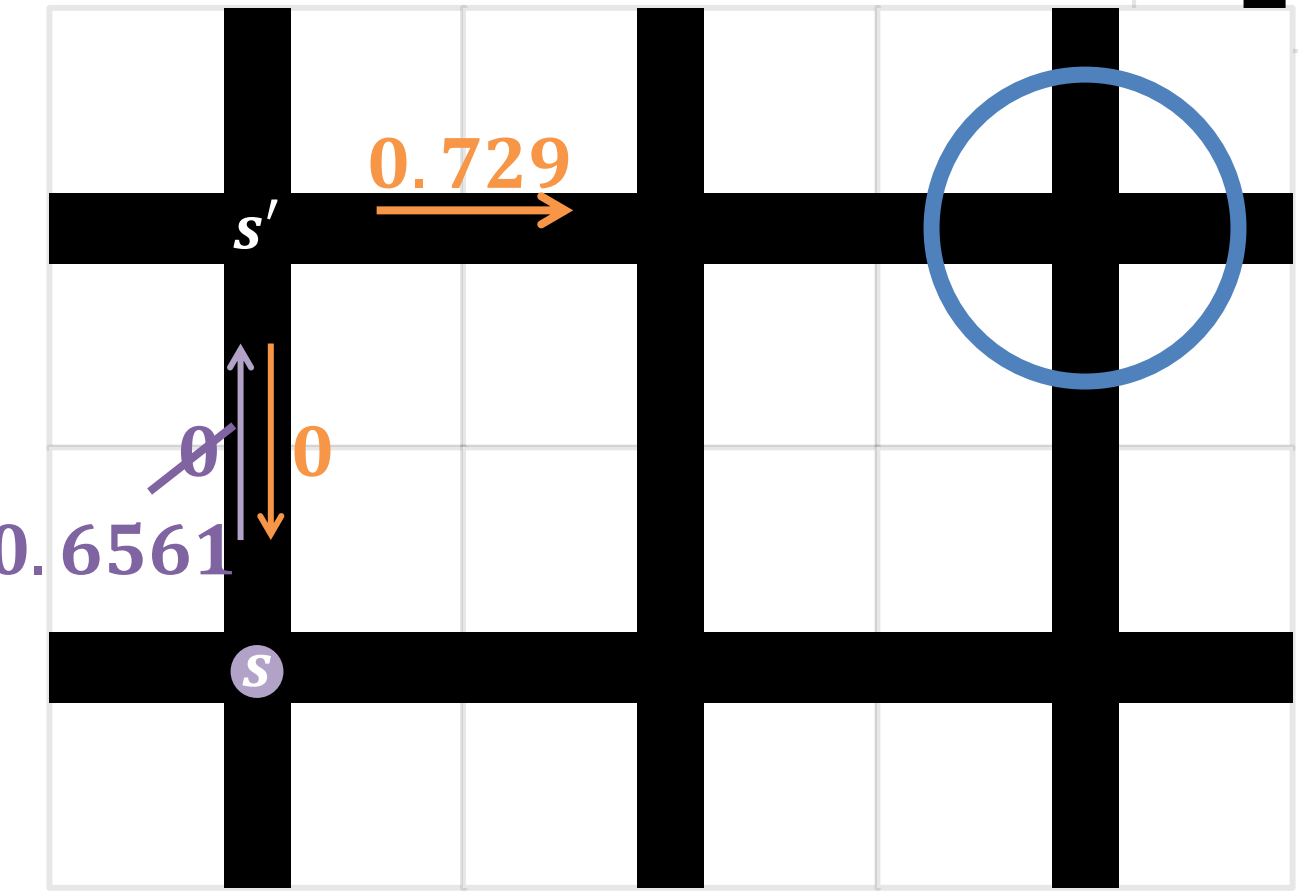
$\hat{Q}$ 가 같으면 랜덤으로 행동 선택

Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

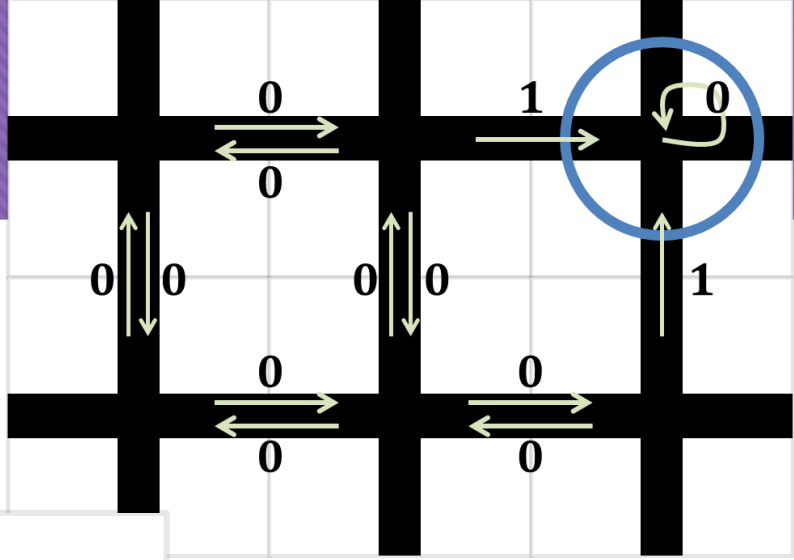
0 0.729

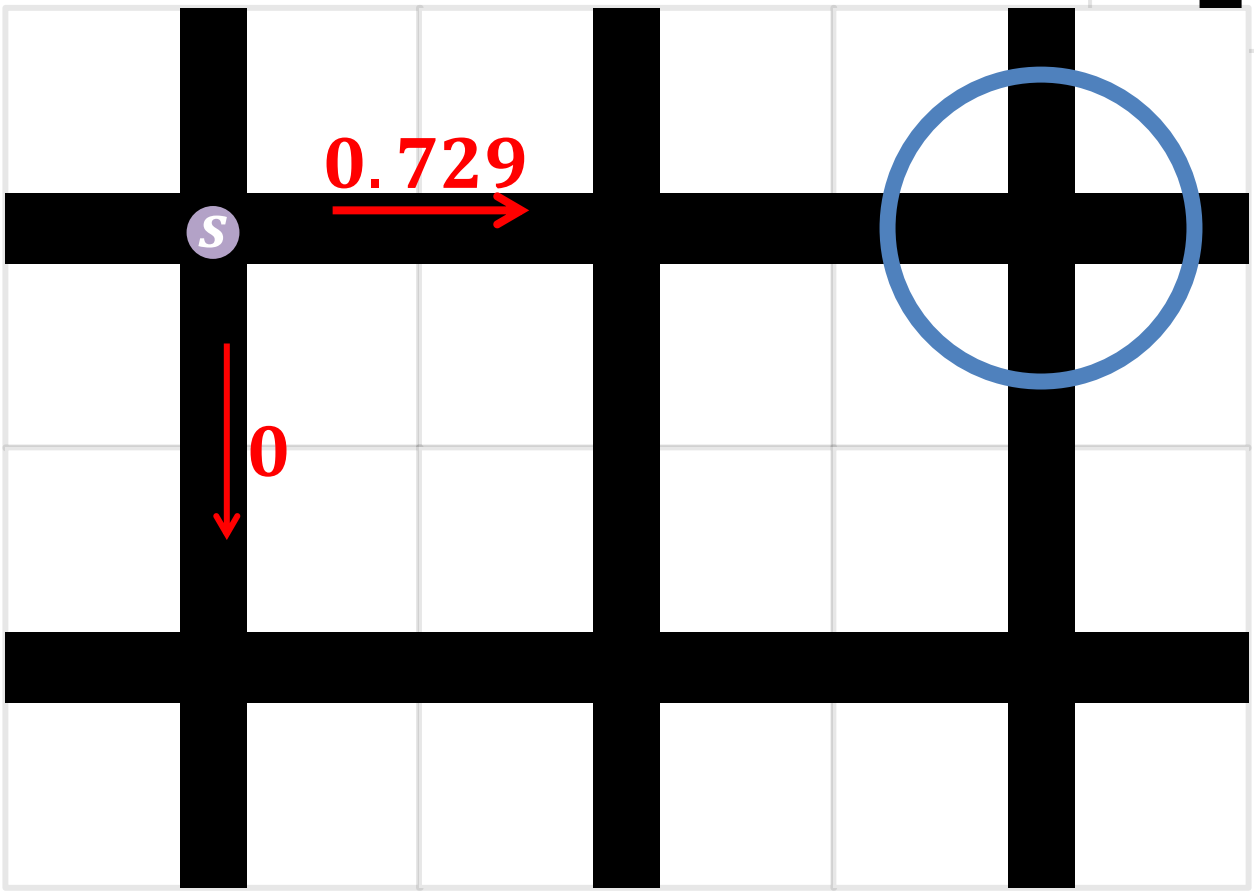




Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

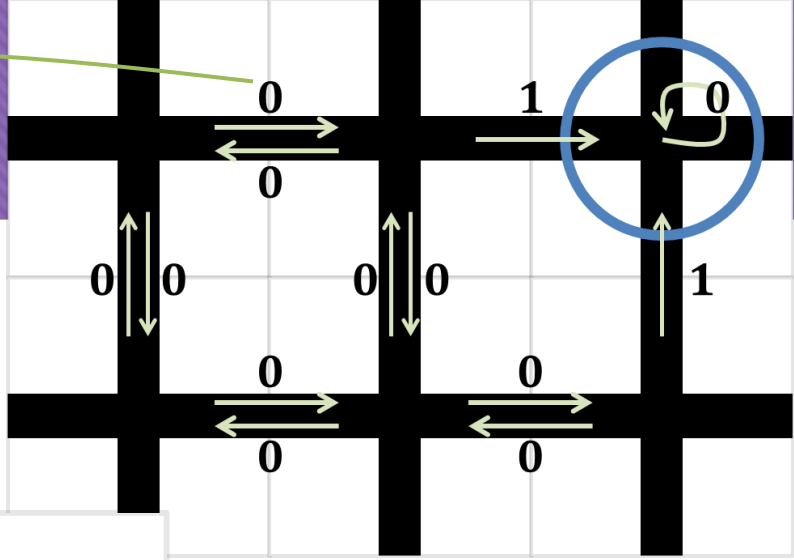


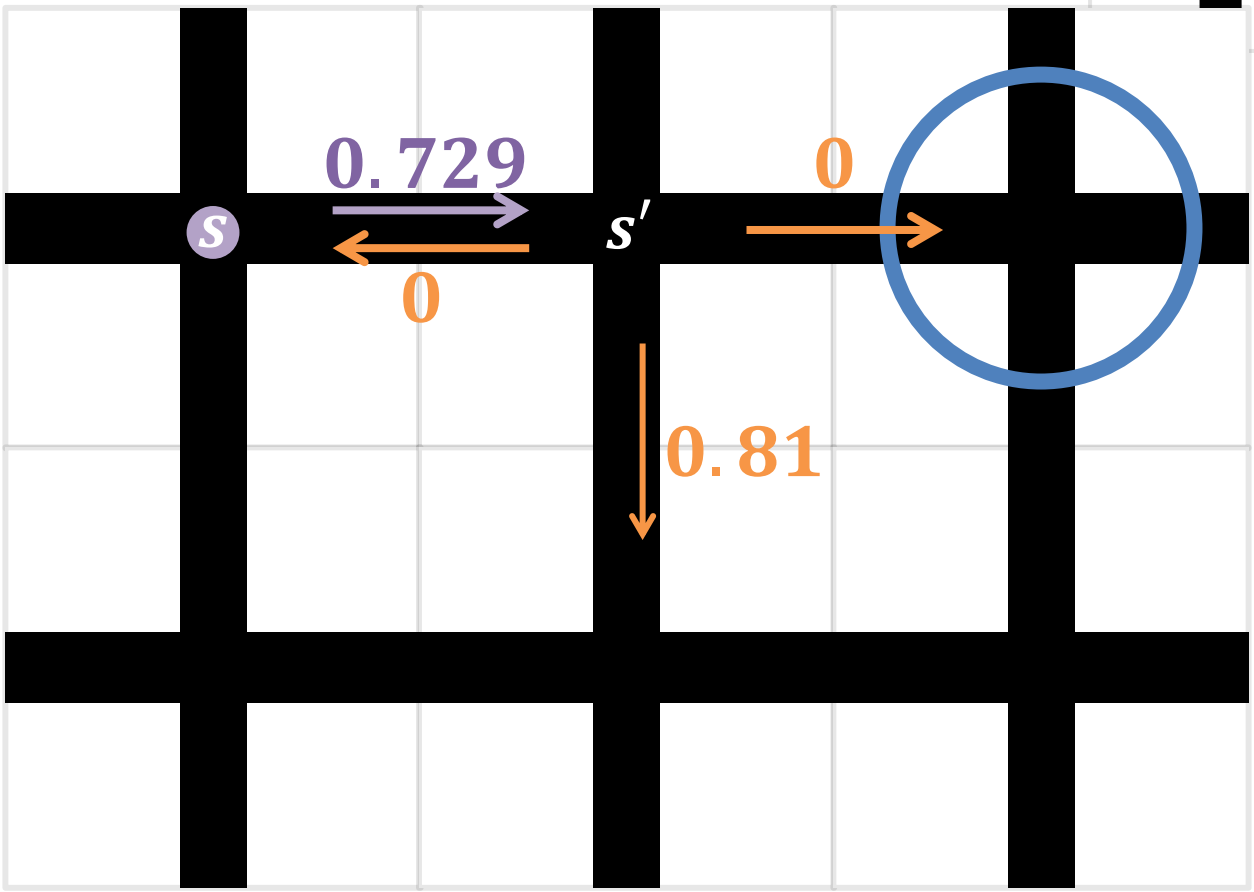


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

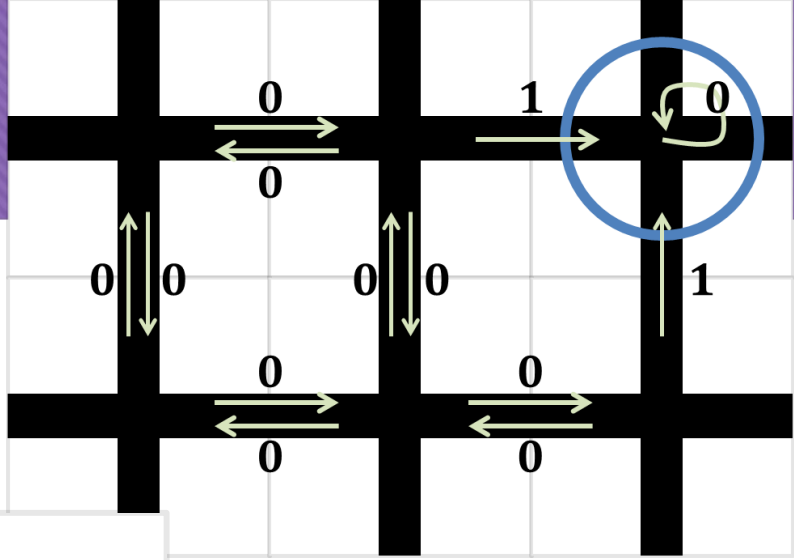
0 0.81

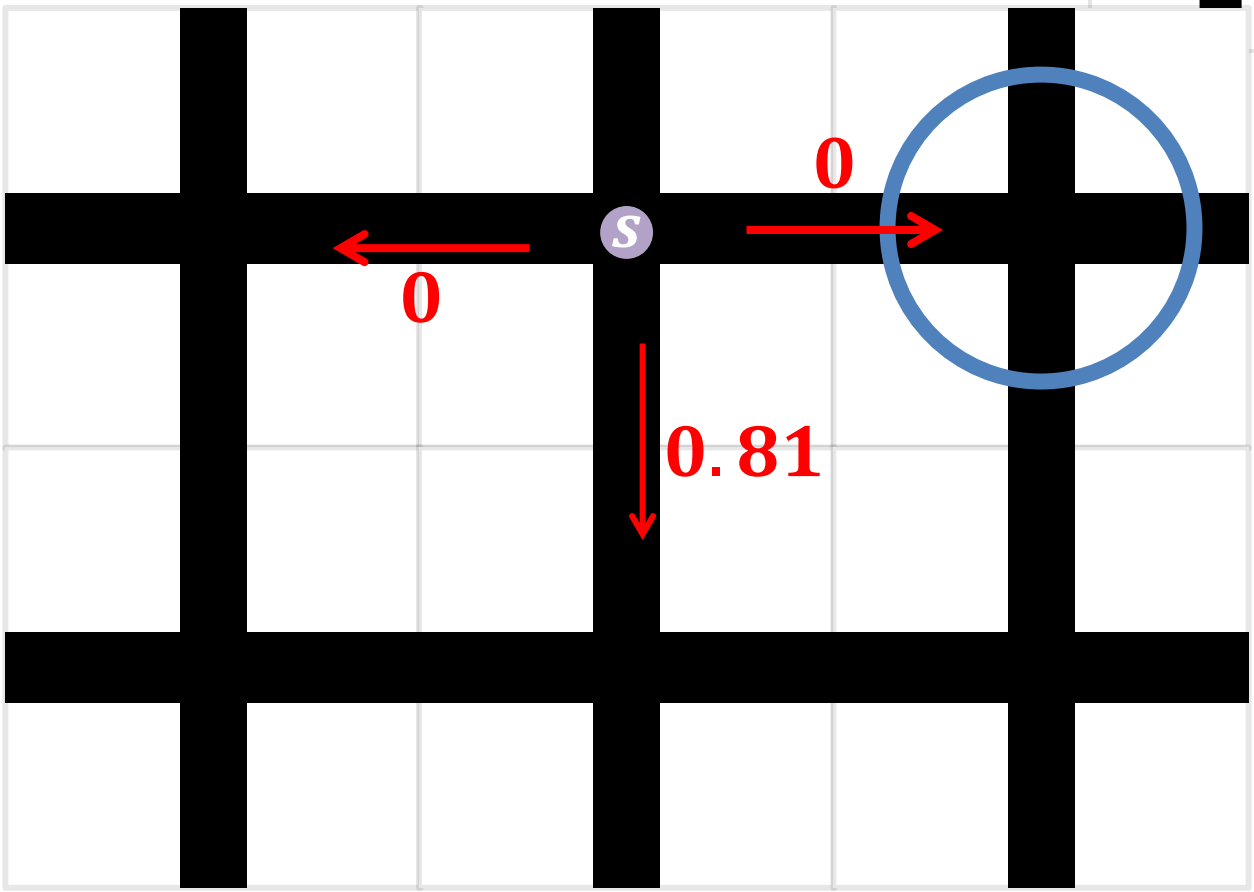




Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

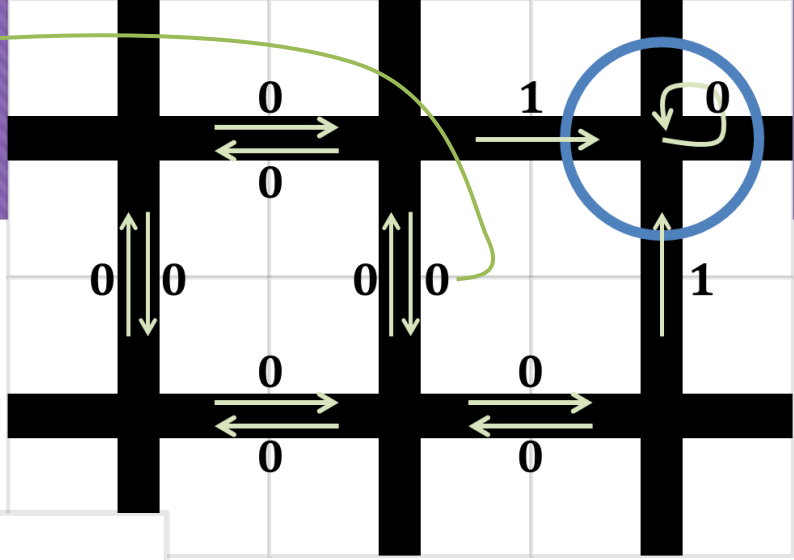


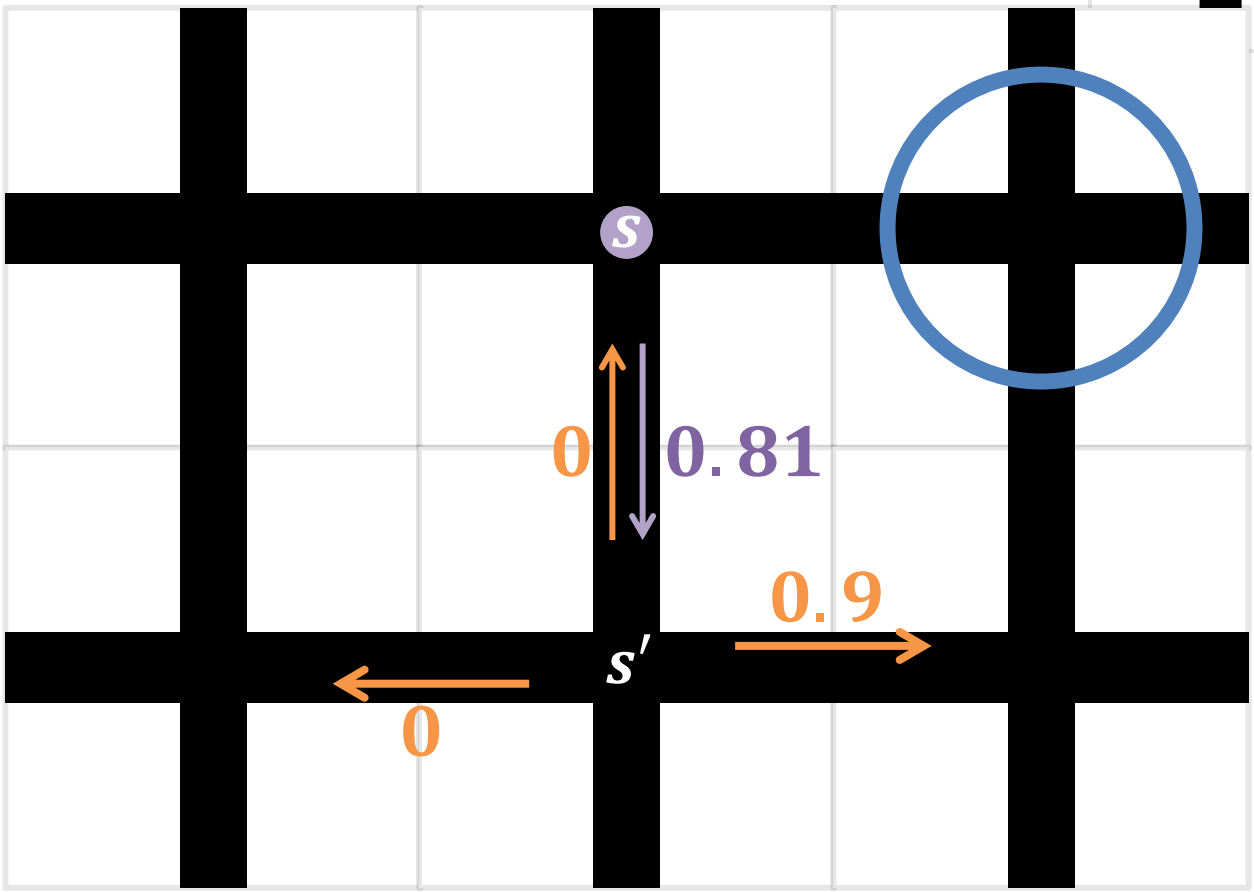


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

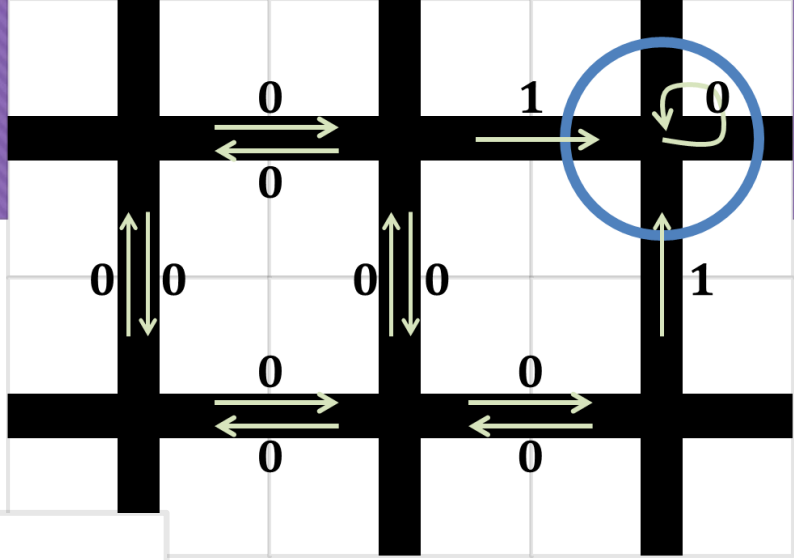
0 0.9

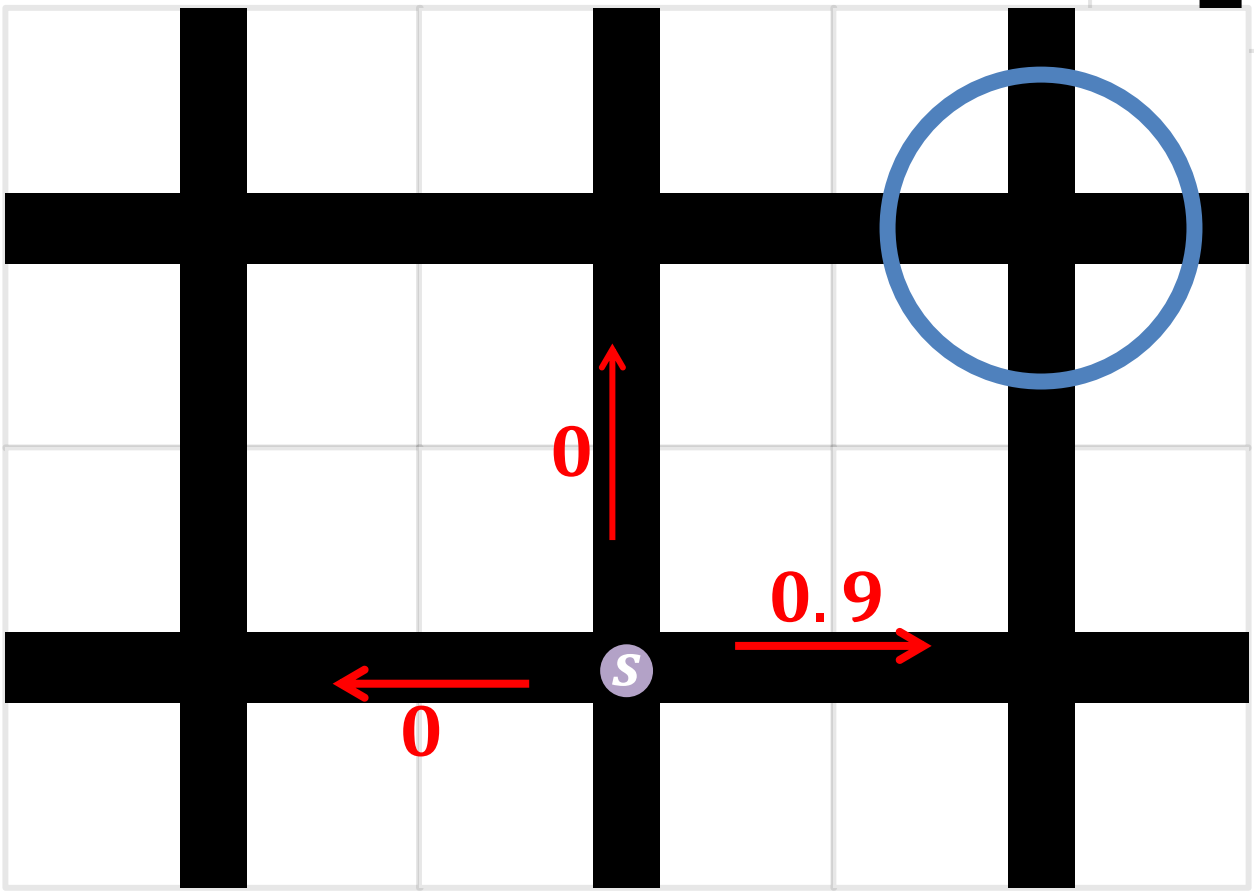




Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

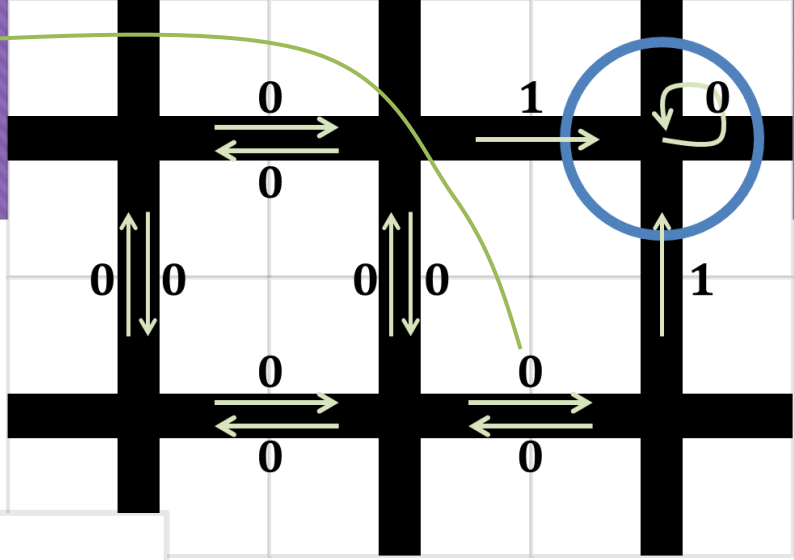


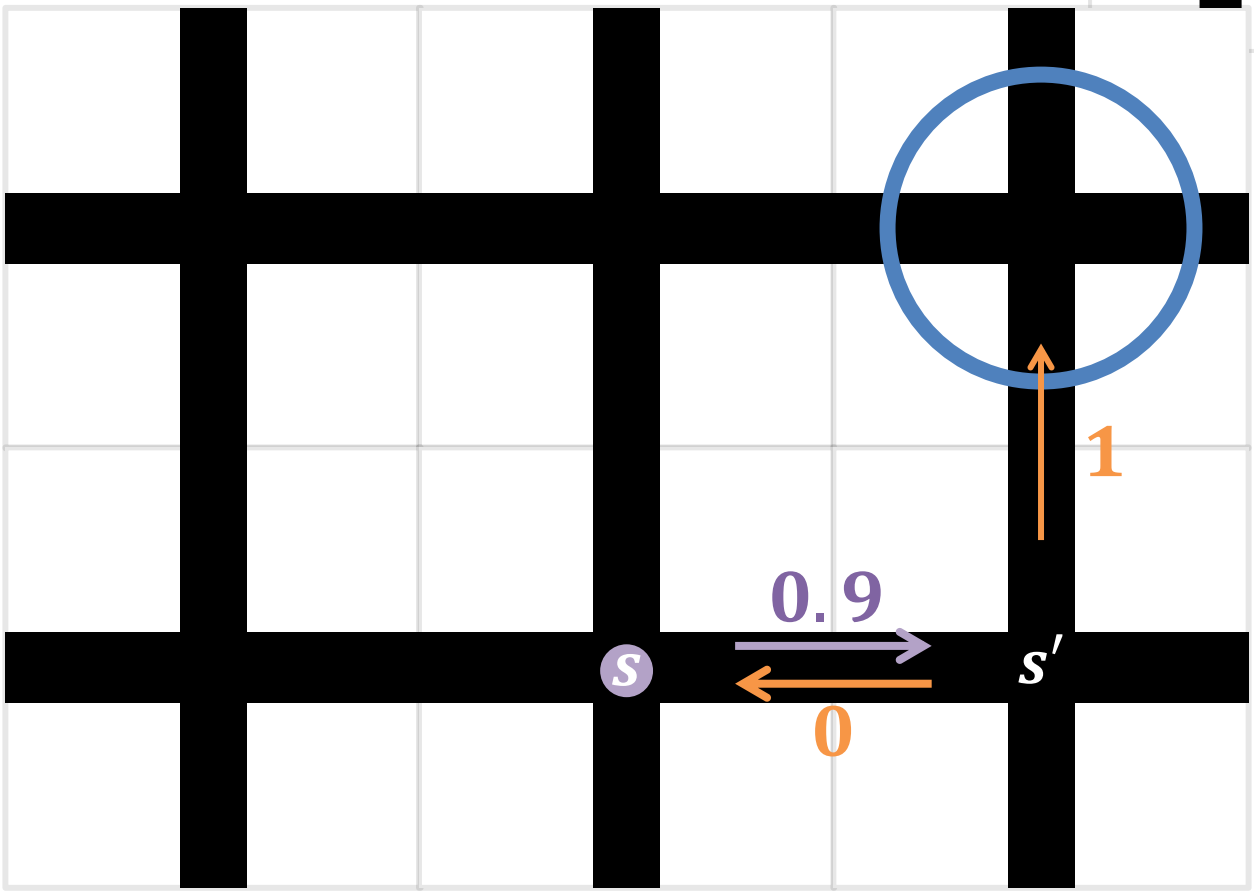


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

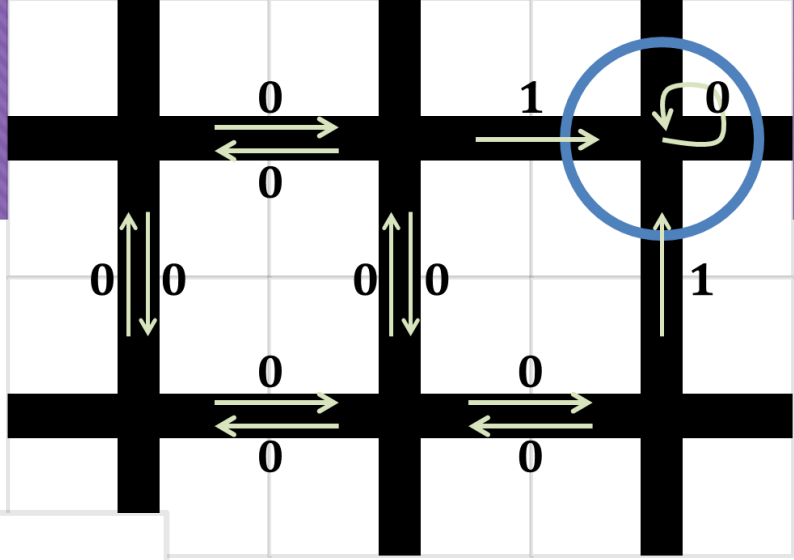
0 1

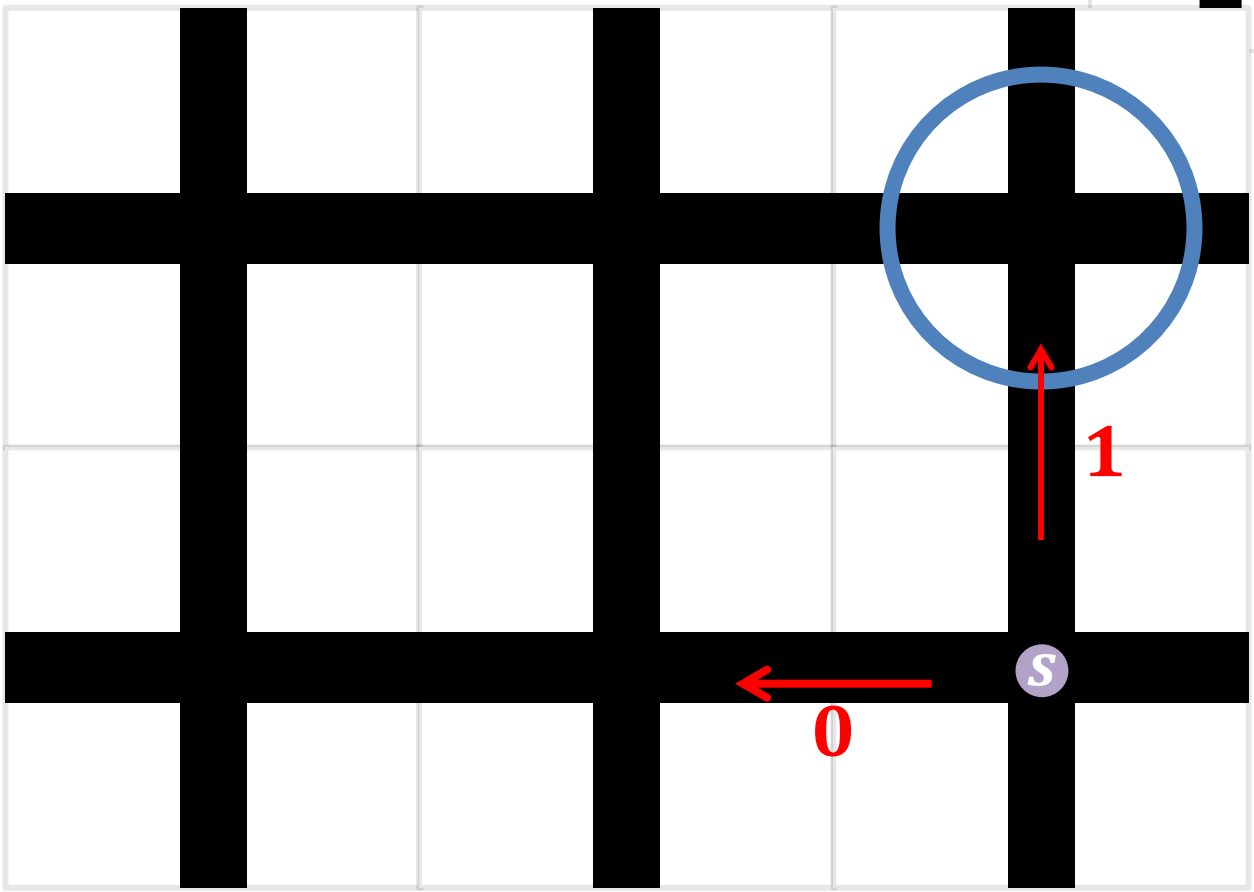




Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

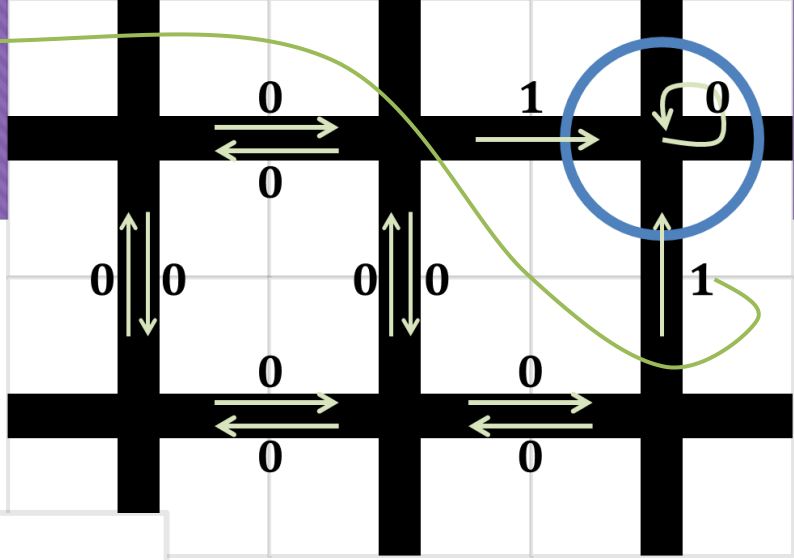


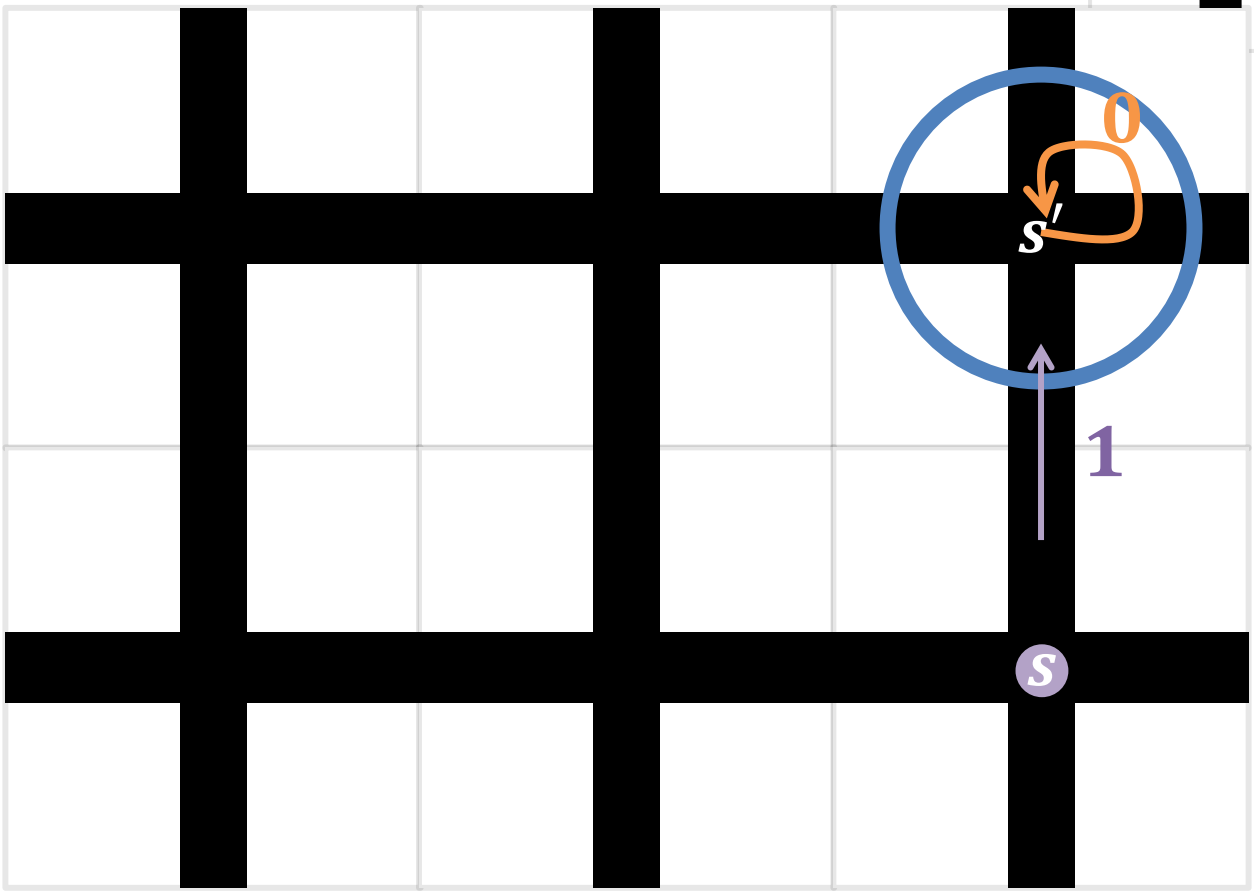


Q-러닝

$$\hat{Q}(s,a) \leftarrow r + 0.9 \times \max_{a'} \hat{Q}(s',a')$$

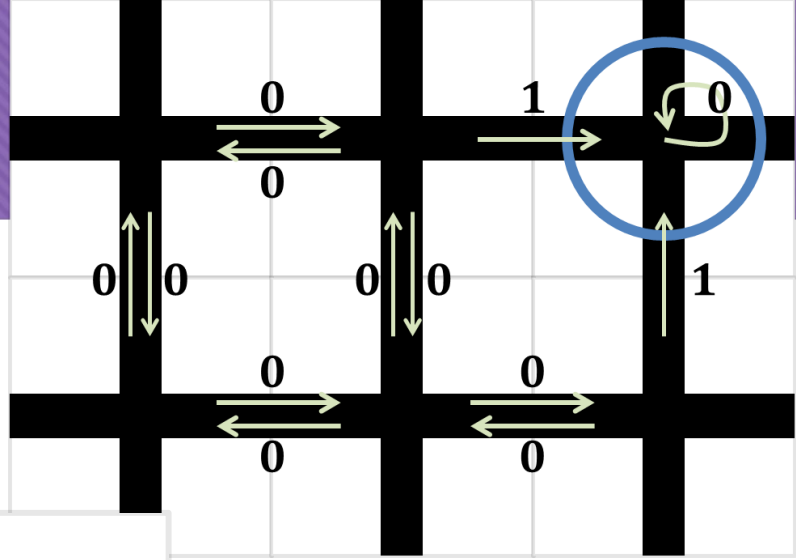
1 0

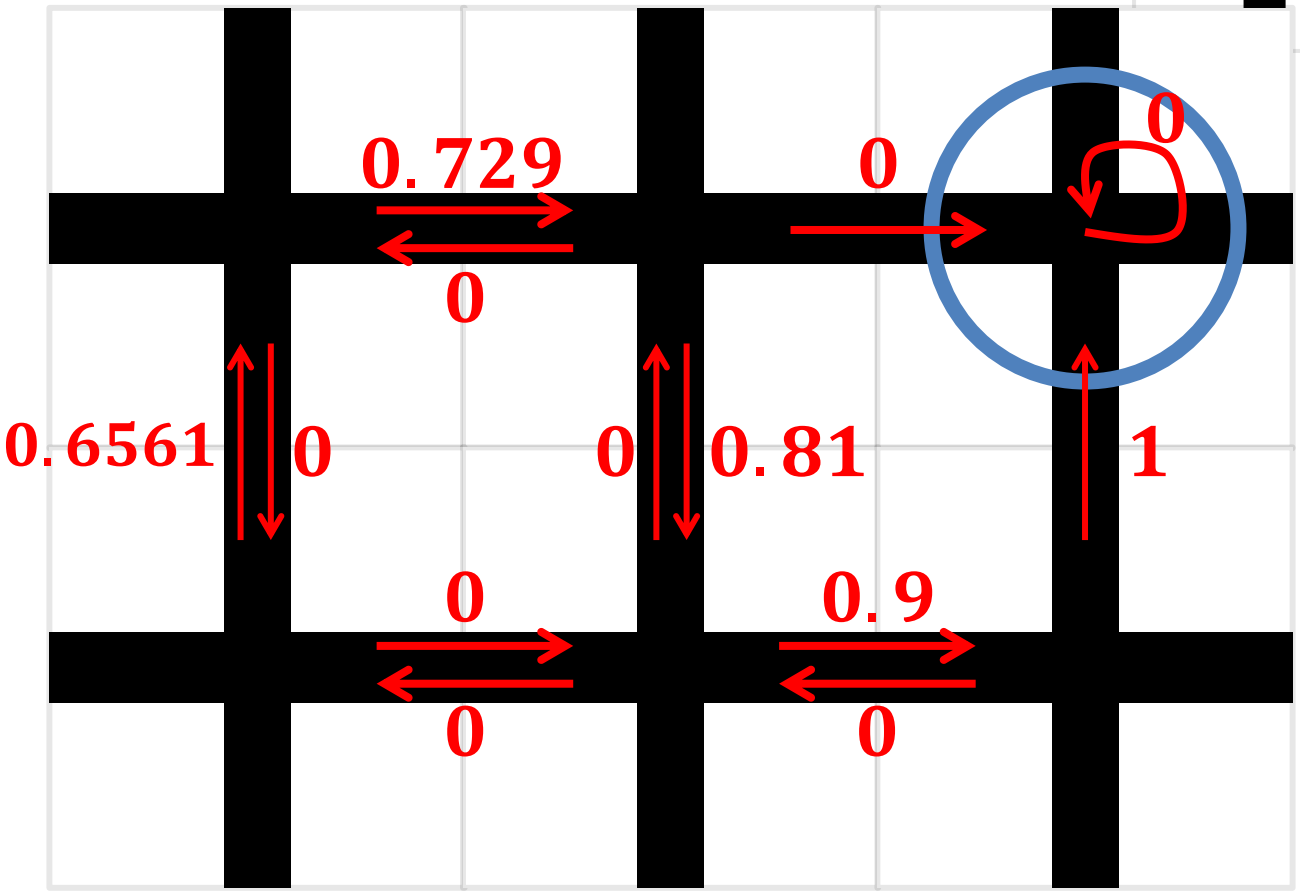




Q-러닝

변화가 없을 때까지 계속 반복



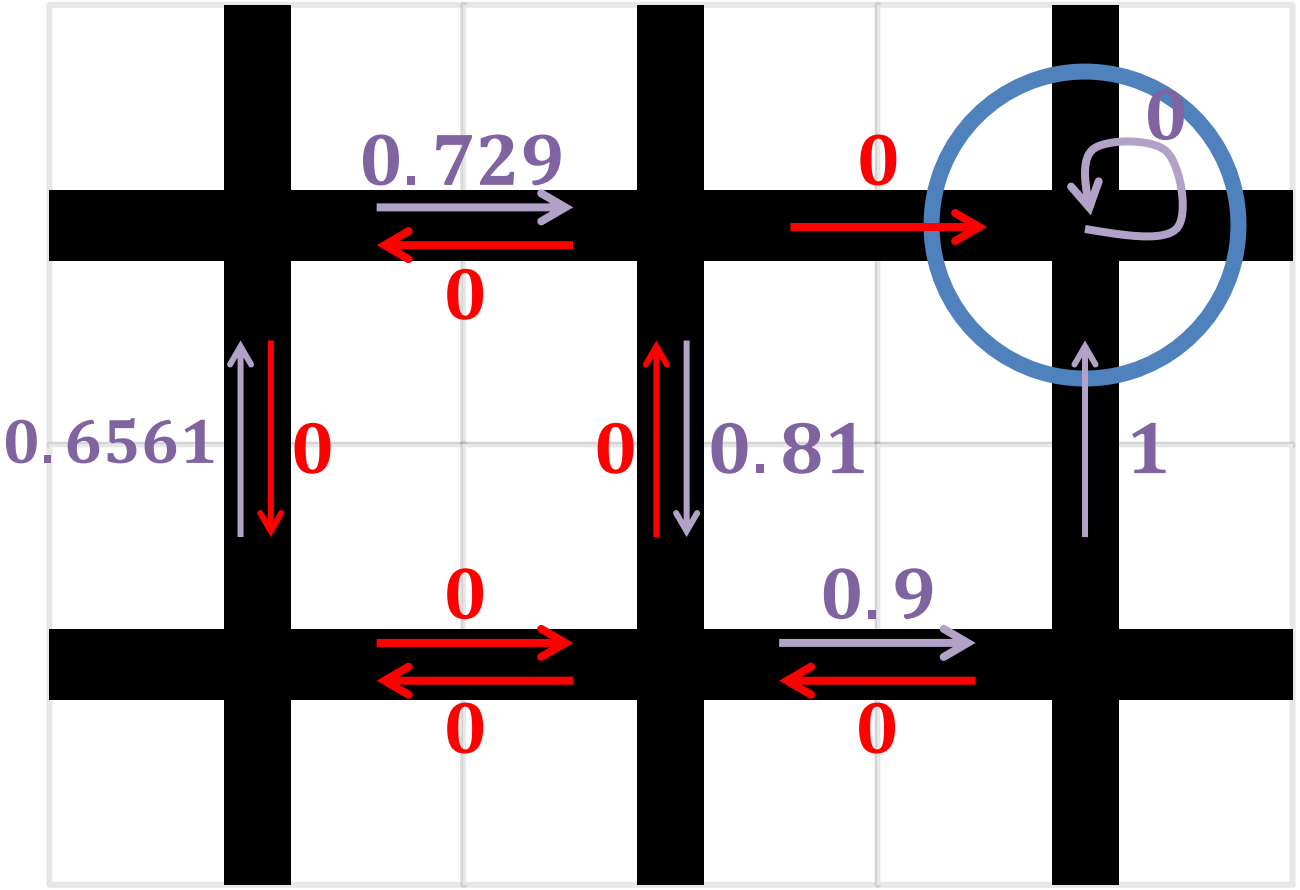


Q-러닝

$$\pi^*(s) = \underset{a}{\operatorname{argmax}}\, Q(s,a)$$

최적의 정책(policy) $\boldsymbol{\pi^*(s)}$

각 상태에서 Q값이 가장 큰 행동

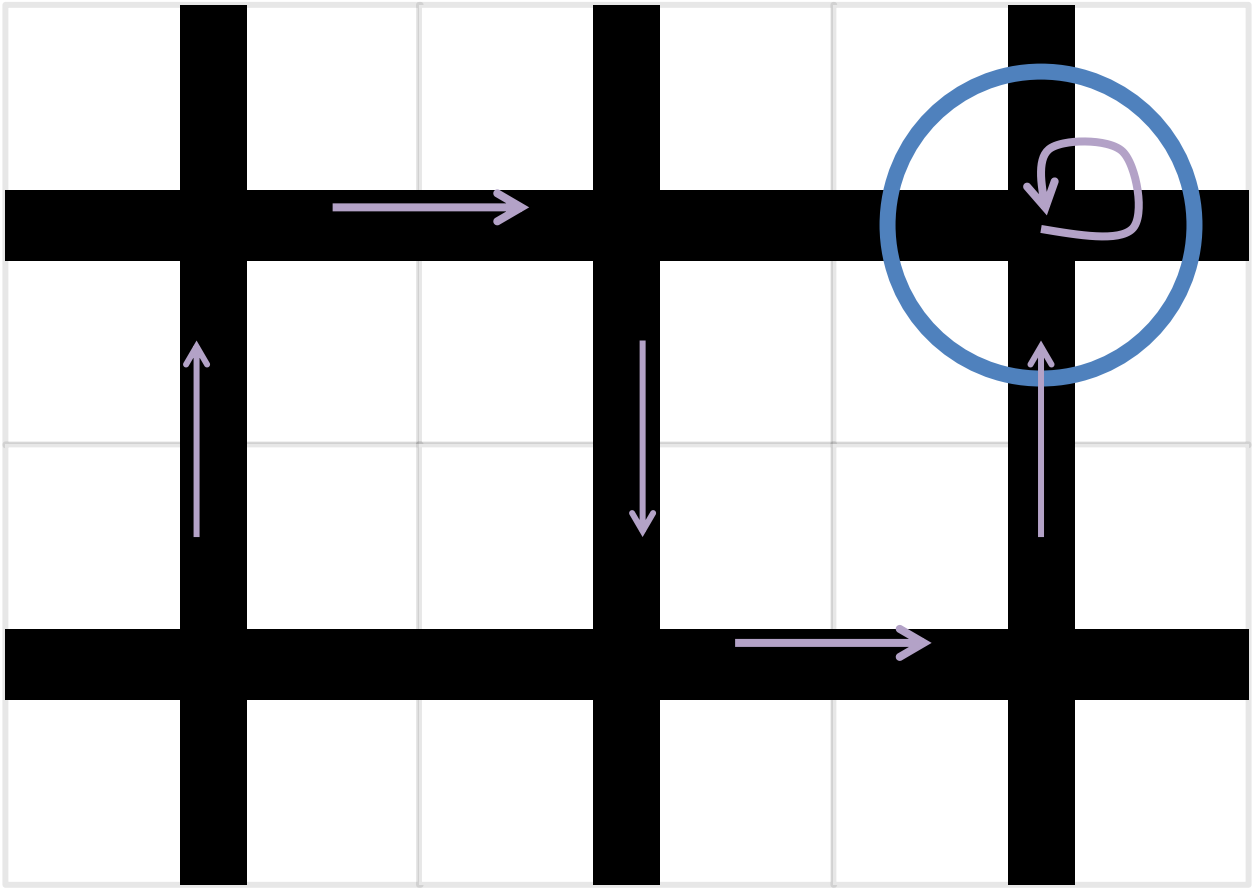


Q-러닝

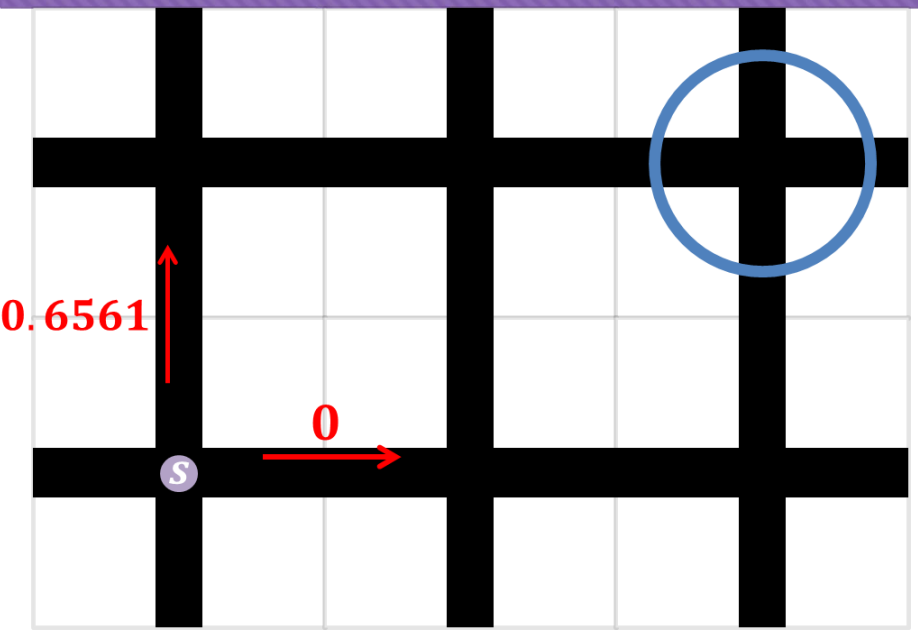
$$\pi^*(s) = \underset{a}{\operatorname{argmax}}\, Q(s,a)$$

최적의 정책(policy) $\boldsymbol{\pi^*(s)}$

각 상태에서 Q값이 가장 큰 행동



탐험(exploration)과 착취(exploitation)



$$p(a_i|s) = \frac{\kappa^{\hat{Q}(s,a_i)}}{\sum_j \kappa^{\hat{Q}(s,a_j)}} \qquad \kappa > 0$$

$\kappa = 0.1$

$$\kappa^{\hat{Q}(s,a_{up})} = 0.1^{0.6561} = 0.22$$

$$\kappa^{\hat{Q}(s,a_{right})} = 0.1^{0} = 1$$

$$p(a_{up}|s) = \frac{0.22}{0.22+1} = 0.18$$

$$p(a_{right}|s) = \frac{1}{0.22+1} = 0.82$$

$\kappa = 10$

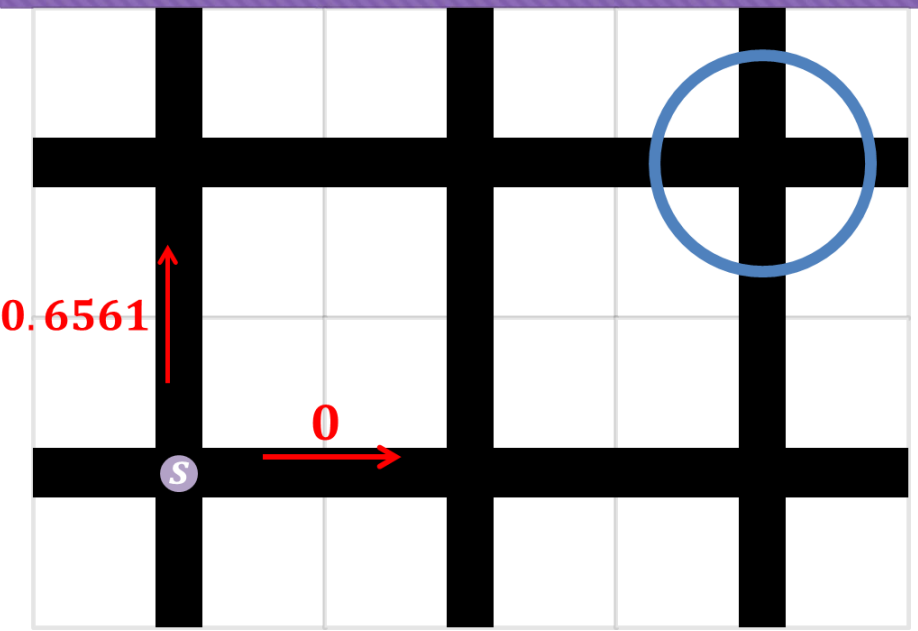
$$\kappa^{\hat{Q}(s,a_{up})} = 10^{0.6561} = 4.53$$

$$\kappa^{\hat{Q}(s,a_{right})} = 10^{0} = 1$$

$$p(a_{up}|s) = \frac{4.53}{4.53+1} = 0.82$$

$$p(a_{right}|s) = \frac{1}{4.53+1} = 0.18$$

탐험(exploration)과 착취(exploitation)



랜덤 노이즈:

$\hat{Q}(s,a)$+노이즈

ε-그리디(greedy):

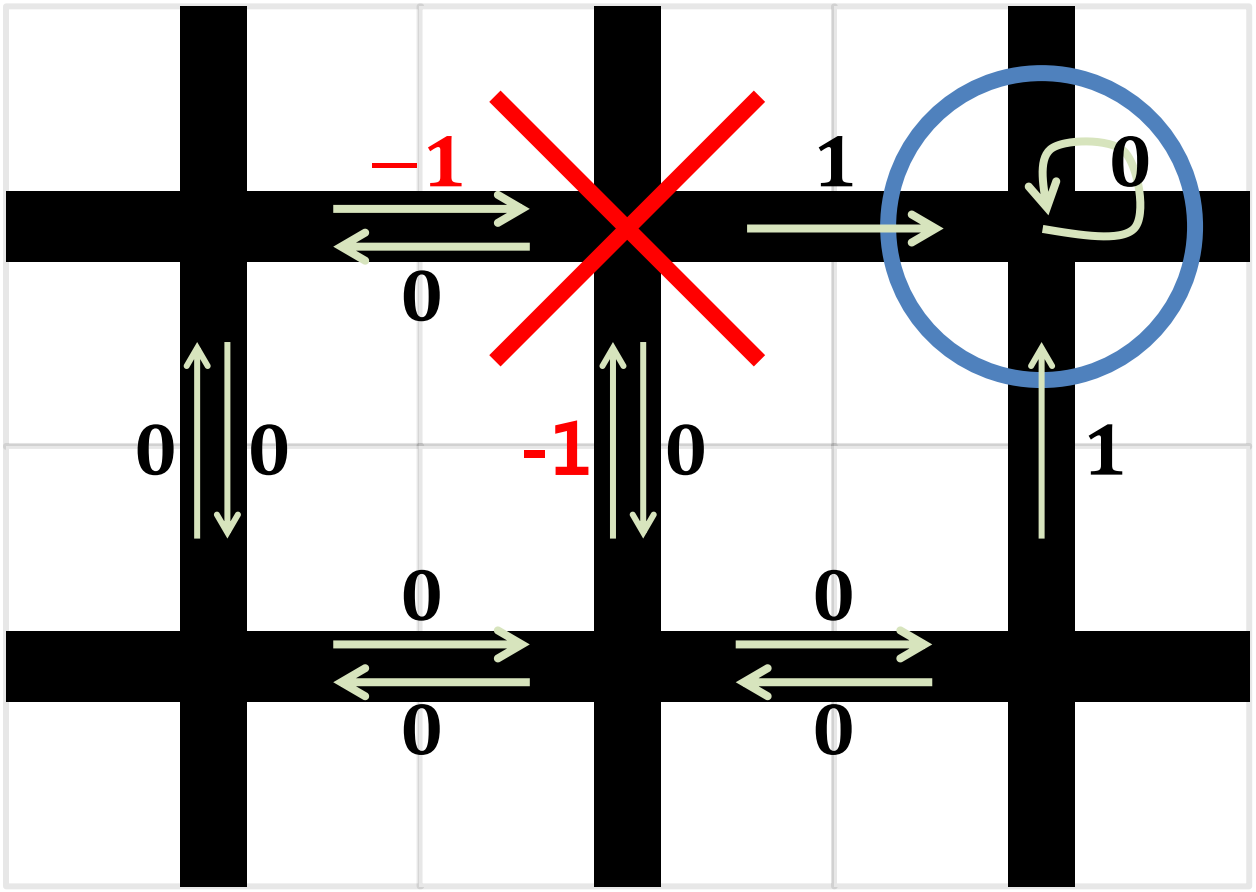
ε 확률로 탐험

$(1-\varepsilon)$ 확률로 착취(Q값에 따라 행동)

감소하는 ε-그리디(decaying ε-greedy):

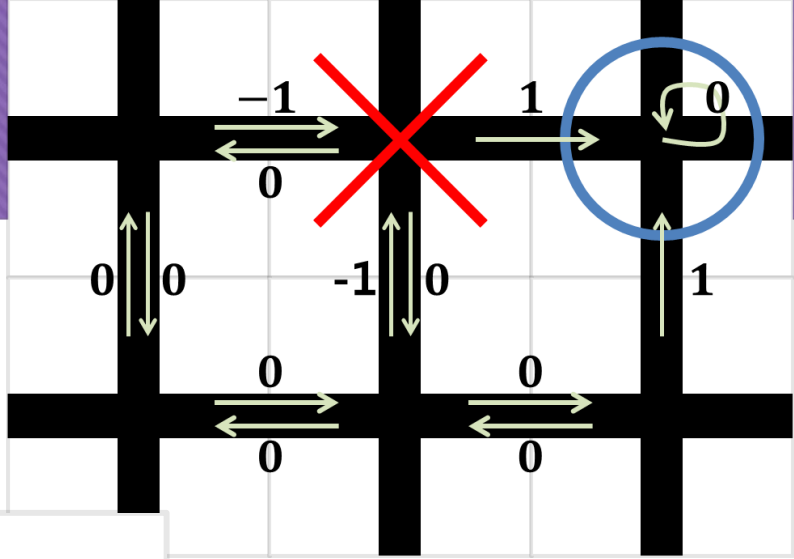
ε 값을 점점 줄여 가며 ε-그리디

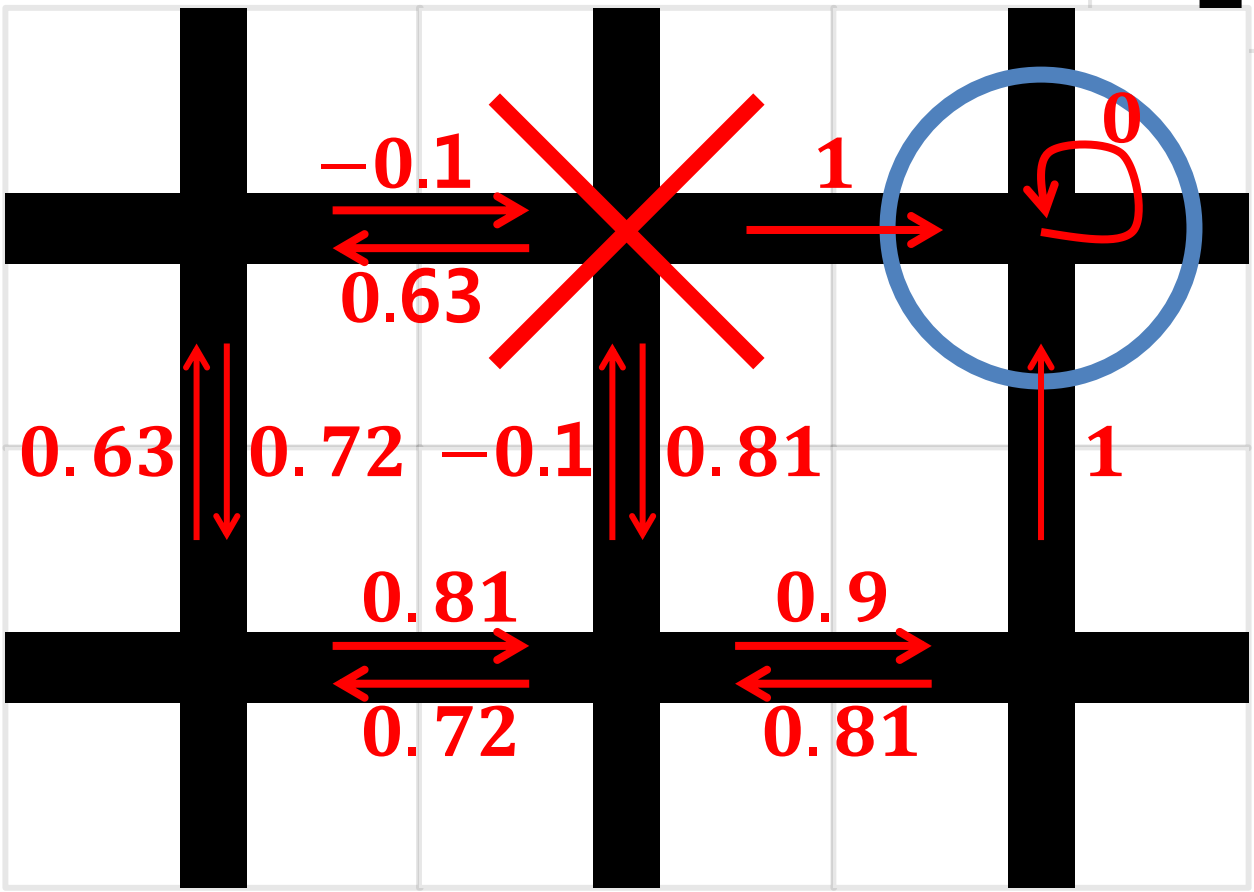
함정 추가



함정 추가

$$\hat{Q}(s,a) \leftarrow r + \gamma \max_{a'} \hat{Q}(s',a')$$



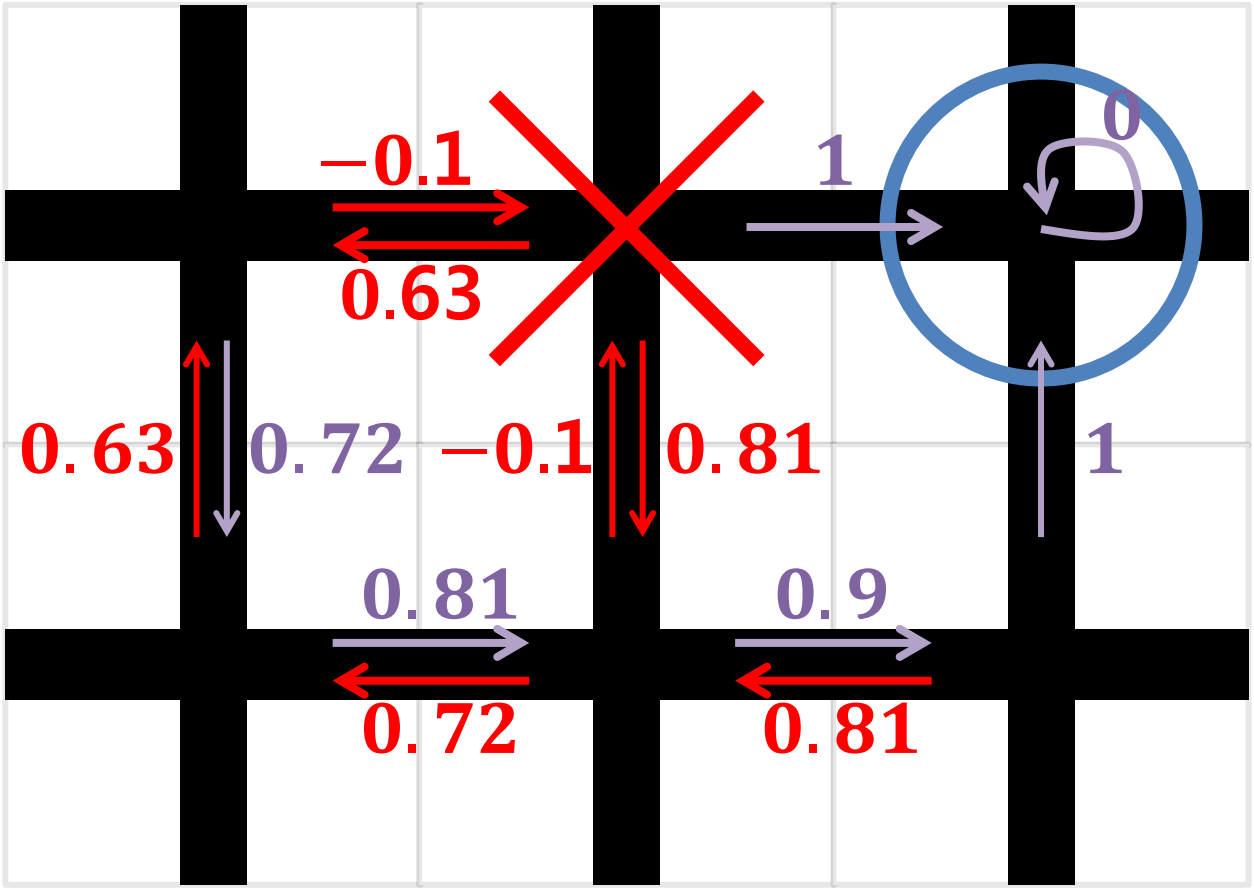


함정 추가

$$\pi^*(s) = \underset{a}{\operatorname{argmax}}\, Q(s,a)$$

최적의 정책(policy) $\boldsymbol{\pi^*(s)}$

각 상태에서 Q값이 가장 큰 행동

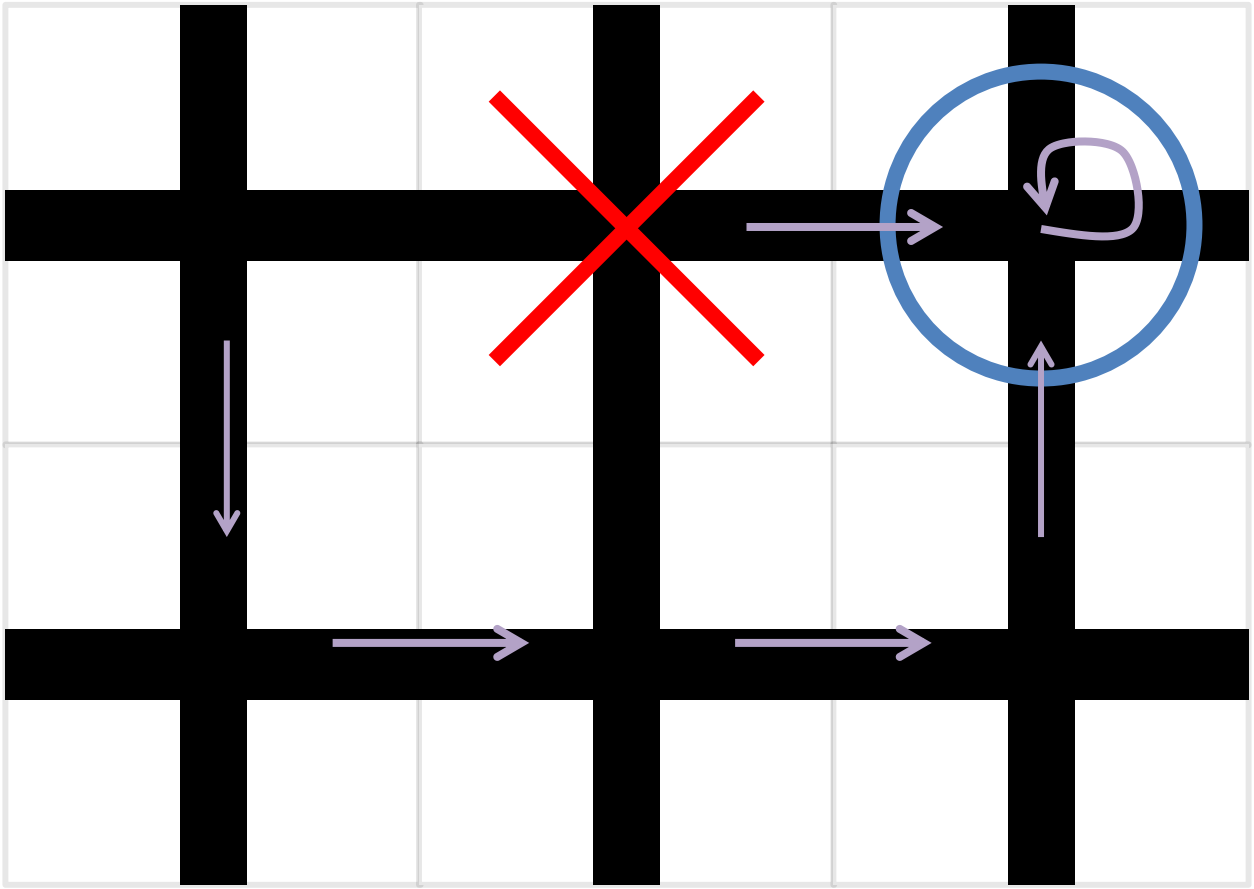


함정 추가

$$\pi^*(s) = \underset{a}{\operatorname{argmax}}\, Q(s,a)$$

최적의 정책(policy) $\boldsymbol{\pi^*(s)}$

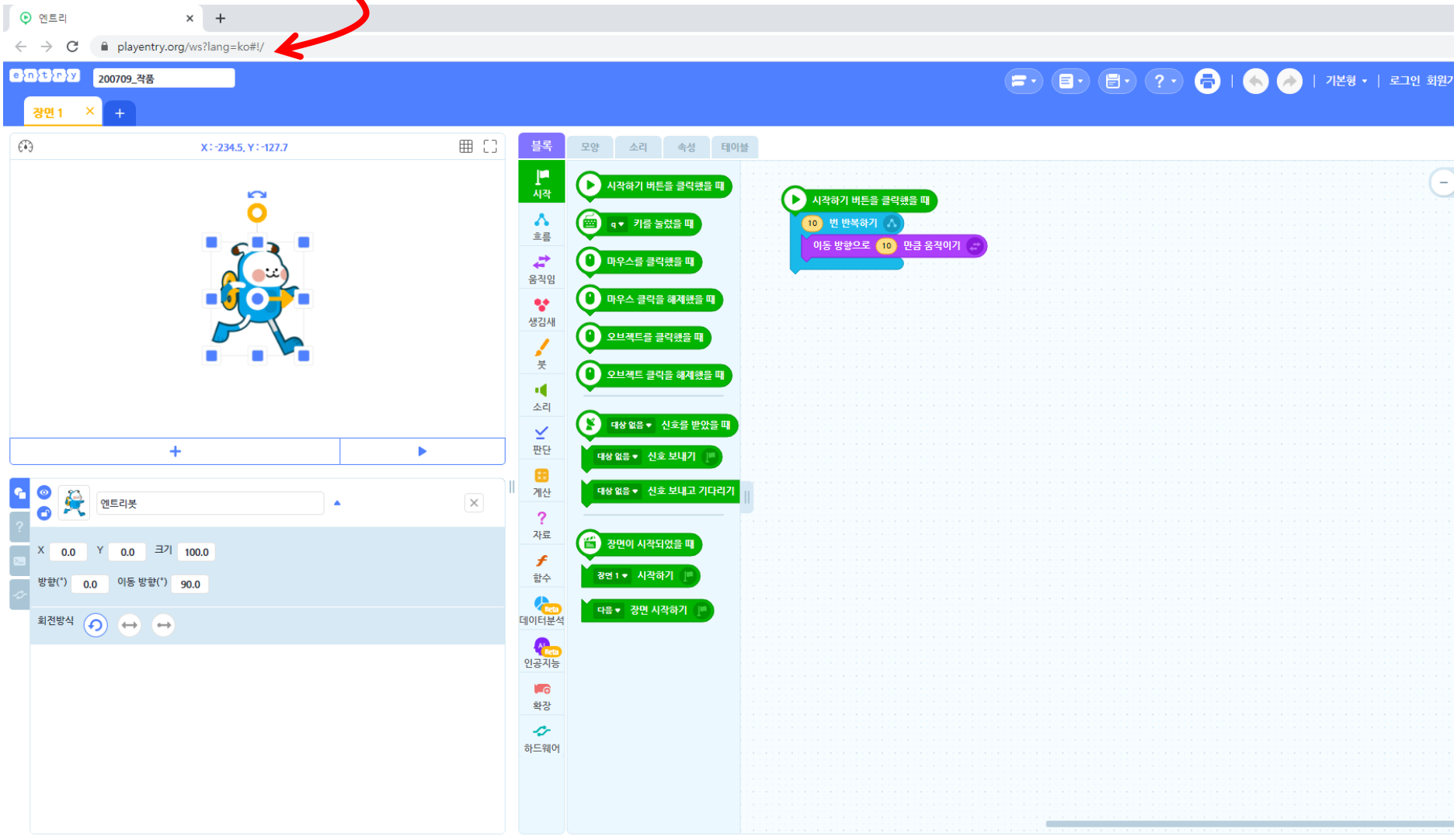
각 상태에서 Q값이 가장 큰 행동



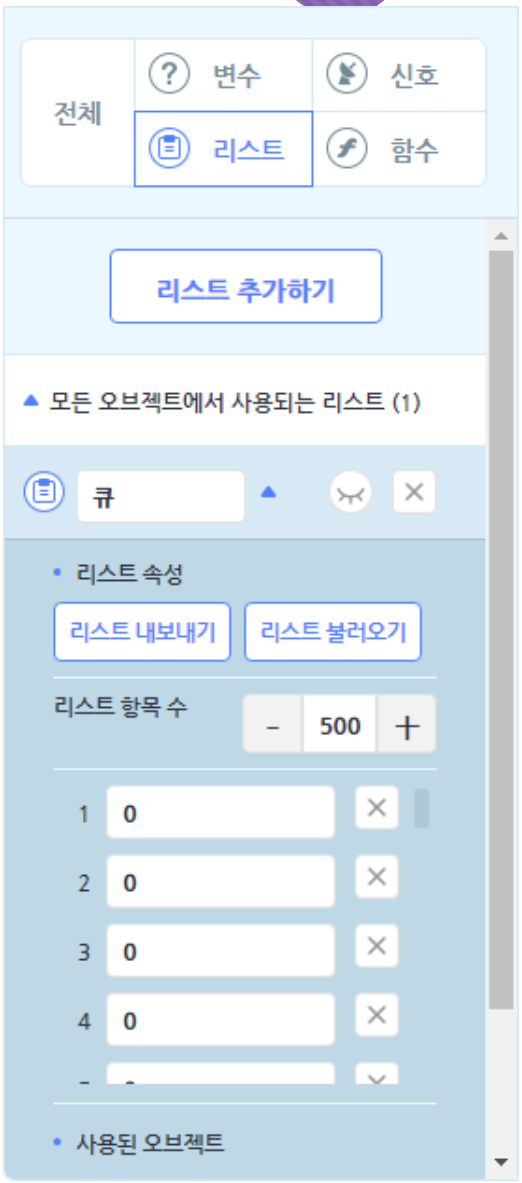
엔트리로 구현하기

템플릿 코드

http://naver.me/FoAixwi6



Q값을 저장할 공간의 인덱스



말판: 가로 4칸, 세로 4칸

각 칸의 행동: 4가지

- 왼쪽으로 이동하기
- 오른쪽으로 이동하기
- 위로 이동하기
- 아래로 이동하기

➔ 4 x 4 x 4 만큼의 저장 공간 필요

Q값을 저장할 공간의 인덱스

가로 방향 칸 번호 (로봇X)	세로 방향 칸 번호 (로봇Y)	행동 번호 (0: 위, 1: 오른쪽, 2: 아래, 3: 왼쪽)	리스트의 인덱스 (큐 번호)
1	1	0	21
1	1	1	22
1	1	2	23
1	1	3	24
1	2	0	25
1	2	1	26
1	2	2	27
1	2	3	28
1	3	0	29
1	3	1	30
⋮	⋮	⋮	⋮

[큐 번호] = 1 + [로봇X] x 16 +[(로봇Y] x 4 + [행동]

정의된 변수들

변수 이름	설명
로봇X	말판 위에서 로봇의 현재 위치의 x좌표 (1, 2, 3, 4)
로봇Y	말판 위에서 로봇의 현재 위치의 y좌표 (1, 2, 3, 4)
로봇 방향	말판 위에서 로봇의 현재 방향 (왼쪽, 오른쪽, 위쪽, 아래쪽)
최적의 행동	교차로의 네 방향에 서 있는 Q들에게 물어 보고 Q값이 가장 큰 행동을 선택한다고 하였는데, 현재 교차로(이동하기 전의 교차로)에서 유효한 행동 중 Q값이 가장 큰 행동
o키를 눌렀는가	키보드의 알파벳 o키를 눌렀으면 "참", 아니면 "거짓"
x키를 눌렀는가	키보드의 알파벳 x키를 눌렀으면 "참", 아니면 "거짓"
보상	현재 교차로에서 선택된 행동을 수행하였을 때의 보상 값
큐 최댓값	유효한 행동들에 대한 Q값의 최댓값
큐 번호	반복문 내에서 사용하는 임시 변수
수정할 큐 번호	현재 교차로(이동하기 전의 교차로)에서 선택된 행동에 대한 큐 리스트의 인덱스

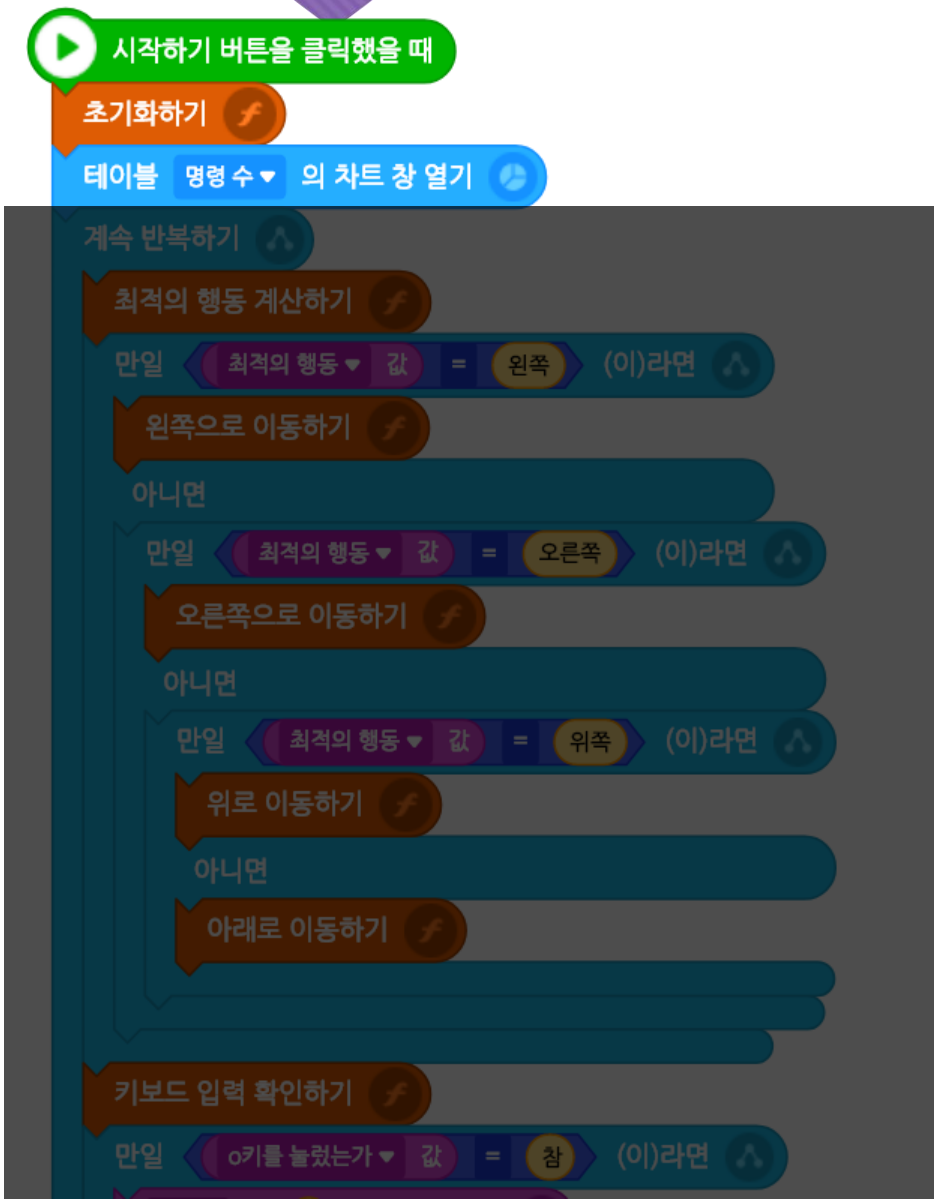
정의된 함수들

함수	설명
초기화하기	로봇의 위치를 출발 위치 (1, 1)로 설정하고, 로봇의 방향을 오른쪽으로 설정합니다.
왼쪽으로 이동하기	말판 위에서 왼쪽으로 한 칸 이동합니다.
오른쪽으로 이동하기	말판 위에서 오른쪽으로 한 칸 이동합니다.
위로 이동하기	말판 위에서 위로 한 칸 이동합니다.
아래로 이동하기	말판 위에서 아래로 한 칸 이동합니다.
최적의 행동 계산하기	교차로의 네 방향에 서 있는 Q들에게 물어 보고 Q값이 가장 큰 행동을 선택한다고 하였는데, 현재 교차로(이동하기 전의 교차로)에서 유효한 행동 중 Q값이 가장 큰 행동을 찾아서 변수 "최적의 행동"에 대입합니다. 즉, 이 함수를 호출하면 변수 "최적의 행동"에 "왼쪽", "오른쪽", "위쪽", "아래쪽" 중 하나가 대입됩니다.

정의된 함수들

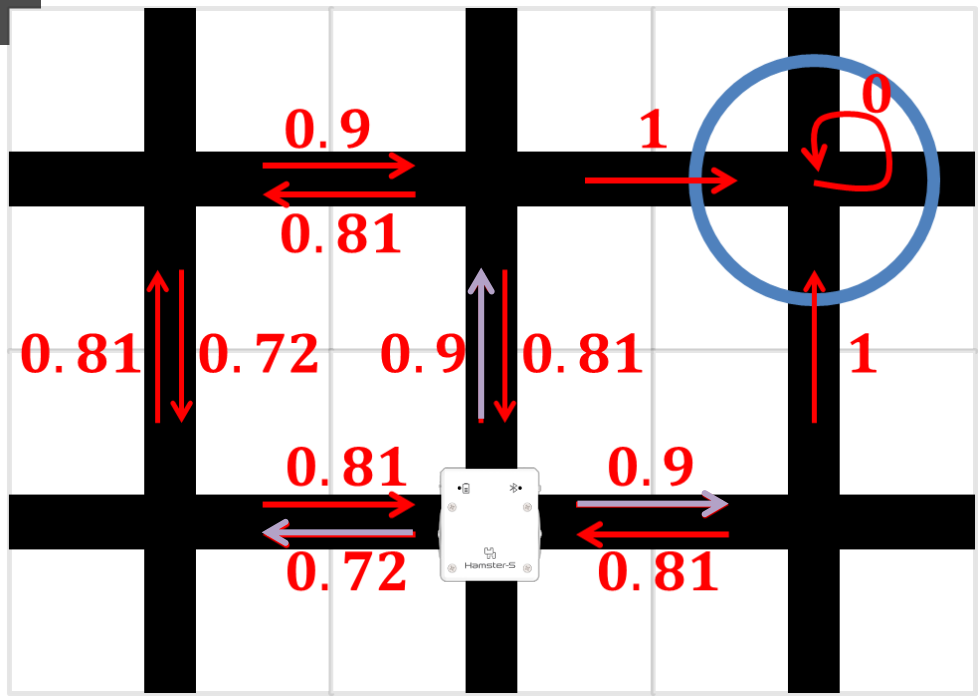
함수	설명
결과 표현하기 ƒ	키보드의 알파벳 o키를 눌렀을 때 또는 x키를 눌렀을 때 각각에 대한 행동을 소리와 LED로 표현하고 출발 위치로 이동합니다.
키보드 입력 확인하기 ƒ	키보드의 스페이스 키, 알파벳 o키, 알파벳 x키를 누를 때까지 기다립니다.
큐 번호 계산하기 ƒ	로봇의 현재 위치(로봇X, 로봇Y)에 대한 큐 리스트의 인덱스를 계산하여 변수 "큐 번호"에 대입합니다. 즉 (큐 번호) = 1 + (로봇X) × 16 + (로봇Y) × 4를 계산합니다.

엔트리 코드


시작하기 버튼을 클릭했을 때
초기화하기
테이블 명령 수 의 차트 창 열기
계속 반복하기
최적의 행동 계산하기
만일 최적의 행동 값 = 왼쪽 (이)라면
왼쪽으로 이동하기
아니면
만일 최적의 행동 값 = 오른쪽 (이)라면
오른쪽으로 이동하기
아니면
만일 최적의 행동 값 = 위쪽 (이)라면
위로 이동하기
아니면
아래로 이동하기
키보드 입력 확인하기
만일 o키를 눌렀는가 값 = 참 (이)라면

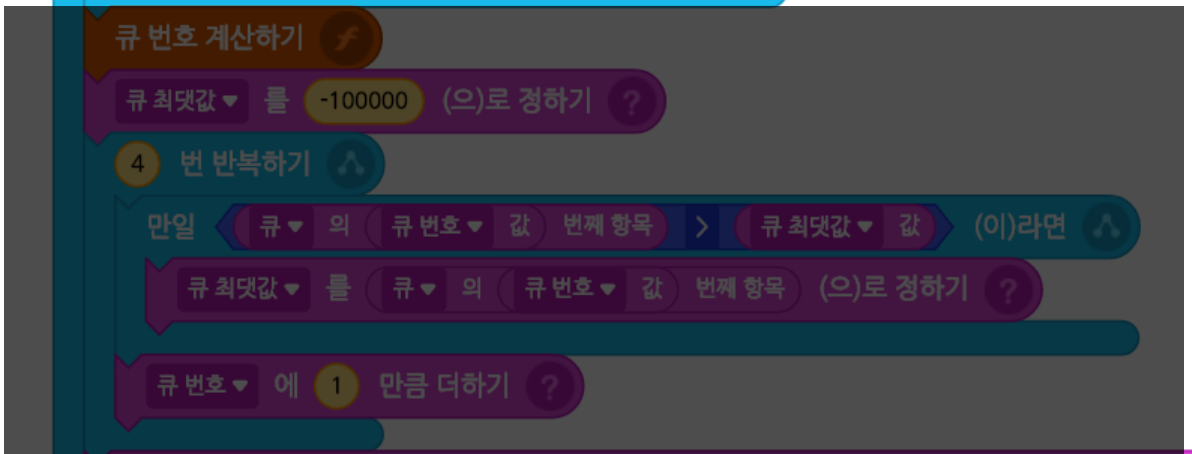
엔트리 코드

$\hat{Q}(s,a)$

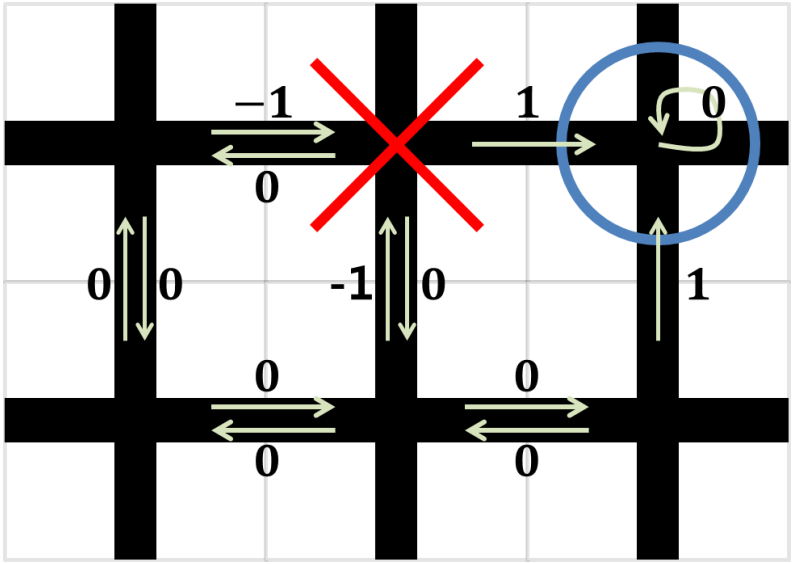


엔트리 코드

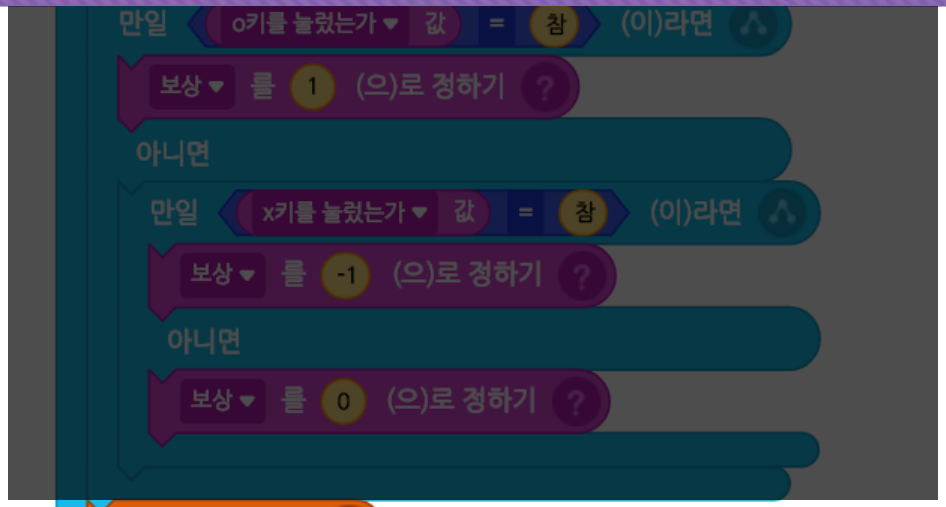

아니면
아래로 이동하기
키보드 입력 확인하기
만일 o키를 눌렀는가 값 = 참 (이)라면
보상 를 1 (으)로 정하기
아니면
만일 x키를 눌렀는가 값 = 참 (이)라면
보상 를 -1 (으)로 정하기
아니면
보상 를 0 (으)로 정하기

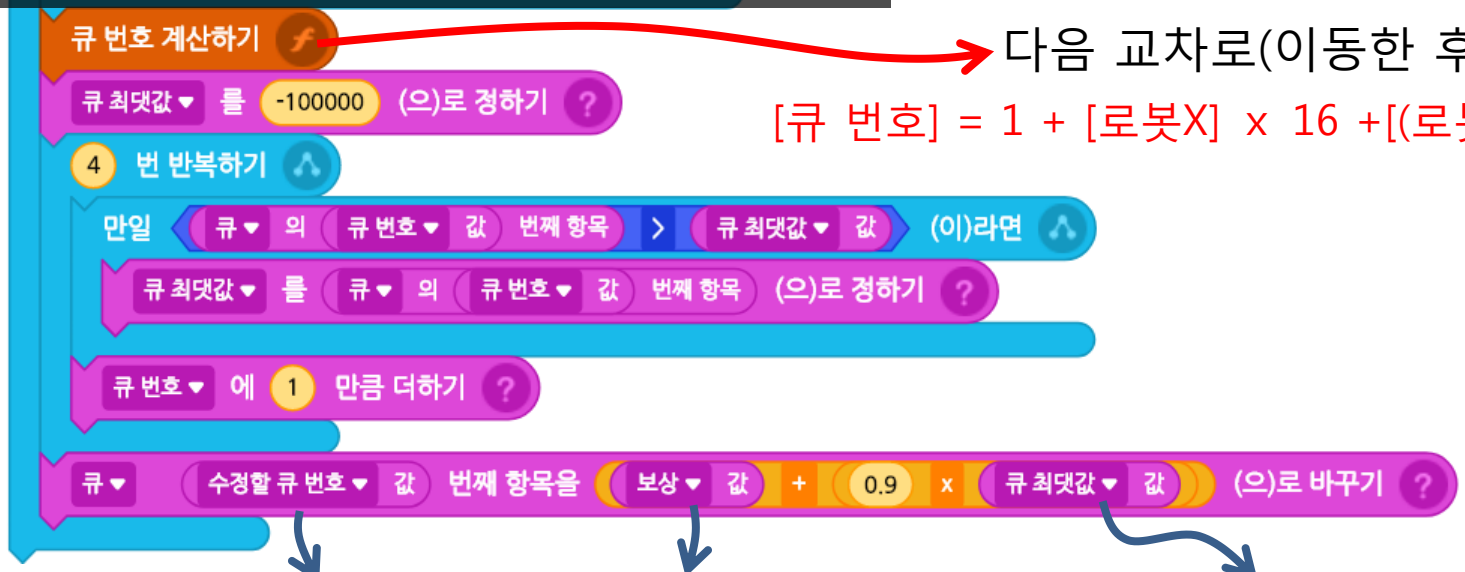

큐 번호 계산하기
큐 최댓값 를 -100000 (으)로 정하기
4 번 반복하기
만일 큐 의 큐 번호 값 번째 항목 > 큐 최댓값 값 (이)라면
큐 최댓값 를 큐 의 큐 번호 값 번째 항목 (으)로 정하기
큐 번호 에 1 만큼 더하기

보상 r


−1
0
1
0
0
0
-1
0
1
0
0
0
0

엔트리 코드





다음 교차로(이동한 후)의 큐 번호

[큐 번호] = 1 + [로봇X] x 16 +[(로봇Y] x 4 + [행동]

[현재 교차로에서 Qa 값] ← [점수 r] + 0.9 x [다음 교차로에서 Qa 값들 중 최댓값]

심화

ε-그리디(greedy)


시작하기 버튼을 클릭했을 때
초기화하기
테이블 명령 수 의 차트 창 열기
계속 반복하기
최적의 행동 계산하기
만일 0 부터 100 사이의 무작위 수 < 10 (이)라면
무작위로 움직이기
아니면
만일 최적의 행동 값 = 왼쪽 (이)라면
왼쪽으로 이동하기
아니면
만일 최적의 행동 값 = 오른쪽 (이)라면
오른쪽으로 이동하기
아니면
만일 최적의 행동 값 = 위쪽 (이)라면
위로 이동하기
아니면
아래로 이동하기
키보드 입력 확인하기

보상을 다르게 주면 어떻게 될까?

변수 추가

숙제

어느 날 로봇이 도착해야 하는 목표 지점이 제일 오른쪽에 있다는 정보를 입수하였습니다.
제일 오른쪽에서 정확히 어느 곳에 있는지는 알 수 없지만 오른쪽으로 이동하는 것이 유리할 것 같습니다.
로봇이 오른쪽으로 이동하는 것을 선호하도록 보상을 주는 방법을 변경하고 다시 게임을 수행하여 로봇의 행동에 어떤 변화가 있는지 서술해 봅시다.
경우에 따라서는 기대했던 대로 로봇이 행동하지 않는 경우도 있는데 게임을 처음부터 다시 여러 번 해보면서 전체적으로 어떻게 행동하는지 관찰해 봅시다.

감사합니다