PPO

201810808 정민지

PPO란?

- Proximal Policy Optimization
- TRPO를 실용적으로 수정한 것

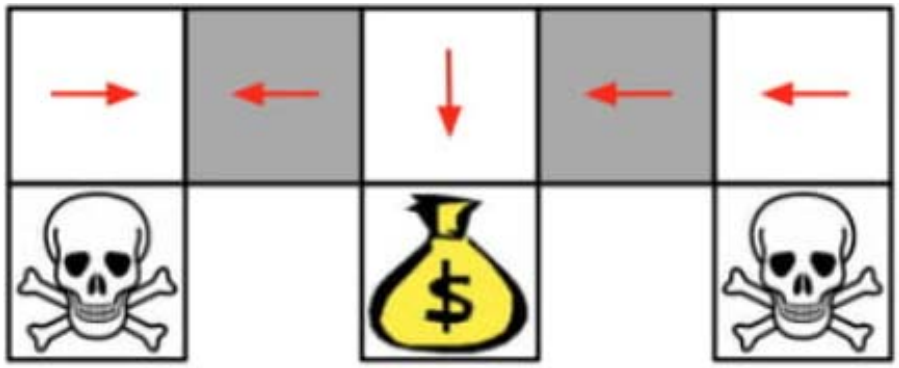
TRPO란?

- Trust Region Policy Optimization
- Policy gradient의 기법 중 하나
  - Policy gradient: 정책 함수를 gradient descent(경사 하강법)을 이용하여 찾는 방법

TRPO란?

- 지금까지 우리가 한 Value-based RL이 아닌 Policy-based RL
  - Value based RL: Q-Learning통하여 action을 선택함
  - Policy based RL: Explicit policy, value function 없이 policy만을 학습함

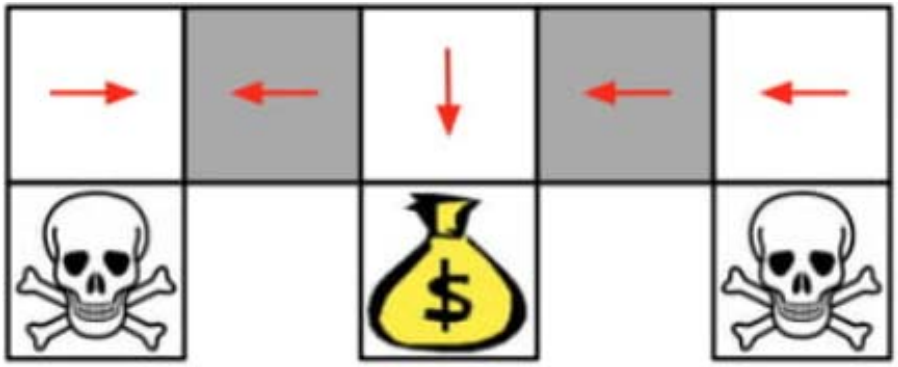
- Policy based RL이 필요한 이유



TRPO란?

- 지금까지 우리가 한 Value-based RL이 아닌 Policy-based RL
  - Value based RL: Q-Learning통하여 action을 선택함
  - Policy based RL: Explicit policy, value function 없이 policy만을 학습함

- Policy based RL이 필요한 이유

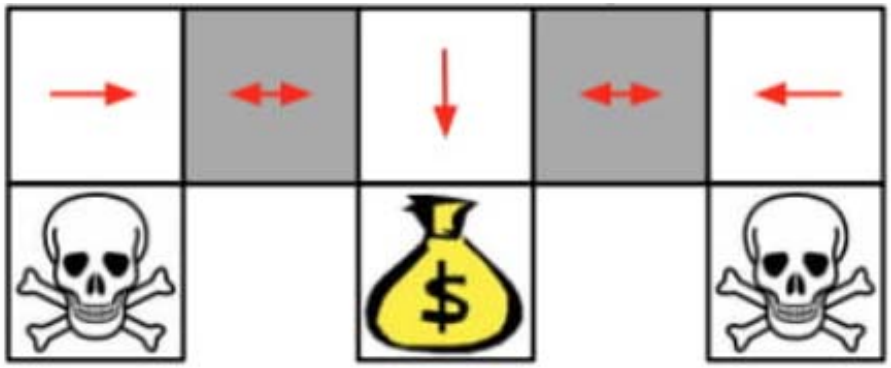


- 흰 부분의 경우 어디로 가야할지 명확하지만, 회색 부분의 경우 같은 상황인데 어디로 가야할지 알 수 없다.

TRPO란?

- 지금까지 우리가 한 Value-based RL이 아닌 Policy-based RL
  - Value based RL: Q-Learning통하여 action을 선택함
  - Policy based RL: Explicit policy, value function 없이 policy만을 학습함

- Policy based RL이 필요한 이유



PPO1, PPO2 차이

PPO2는 GPU용으로 만들어져서 PPO1과 다중 처리시 차이가 있다.

- PPO1: MPI 사용시 단일 환경에서 학습됨
- PPO2: 일괄 처리를 하기 때문에 Vectorize된 환경 필요, MPI도 사용할 수 있지만 병렬 처리를 더 효율적으로 한다.

PPO OpenAI (CartPole)

```python
import gym

from stable_baselines.common.policies import MlpPolicy
from stable_baselines.common.vec_env import DummyVecEnv
from stable_baselines import PPO2

frames=[] #gif 만들기 위함

# Create the environment
env = gym.make('CartPole-v1')
#env.reset()

#ppo2는 vectorized된 환경 필요
env = DummyVecEnv([lambda: env])  # The algorithms require a vectorized environment to run

# Define the model
model = PPO2(MlpPolicy, env, verbose=1)

# 훈련 얼마나 시킬건지?? -> 보통 오래할수록 잘됨
model.learn(total_timesteps=30000)

# 움직임 확인하기
obs = env.reset()
for i in range(10000):
    action, _states = model.predict(obs)
    obs, rewards, dones, info = env.step(action)
    frames.append(env.render(mode = 'rgb_array'))
    env.render() #창띄우기
env.close()
```

↙ PPO2는 Vectorized 된 환경 필요

PPO OpenAI (CartPole)