Policy Gradients : Introduction

Easwar Subramanian

TCS Innovation Labs, Hyderabad

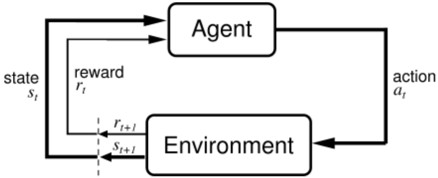
Email : easwar.subramanian@tcs.com / cs5500.2020@iith.ac.in

October 18, 2021

Overview

Introduction

Reinforcement Learning : Framework



A Markov decision process given by $< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma >$ is used as a framework to solve RL problems

Policy

Let π denote a policy that maps state space $\mathcal{S}$ to action space $\mathcal{A}$

Policy

- Deterministic policy: $a = \pi(s), s \in \mathcal{S}, a \in \mathcal{A}$
- Stochastic policy $\pi(a|s) = P[a_t = a|s_t = s]$

Policy Based Reinforcement Learning

- Last week, we parametrized value functions using parameter ϕ

$$V_\phi^\pi(s) = V^\pi(s)$$

$$Q_\phi^\pi(s,a) = Q^\pi(s,a)$$

- Policy was directly generated from value functions (greedy or ϵ greedy)

$$\pi_*(a|s) = \begin{cases} 1 & \text{if } a = \arg\max_{a \in \mathcal{A}} Q_*(s,a) \\ 0 & \text{Otherwise} \end{cases}$$

- In the next couple of lectures, we will directly parametrize the policy

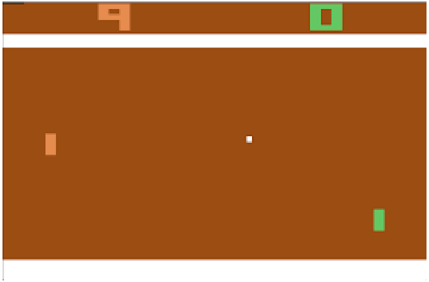
$$\pi_\theta(a|s) = P(a|s,\theta)$$

- We will consider model free control with parametrized policies

Policy Optimization Problem

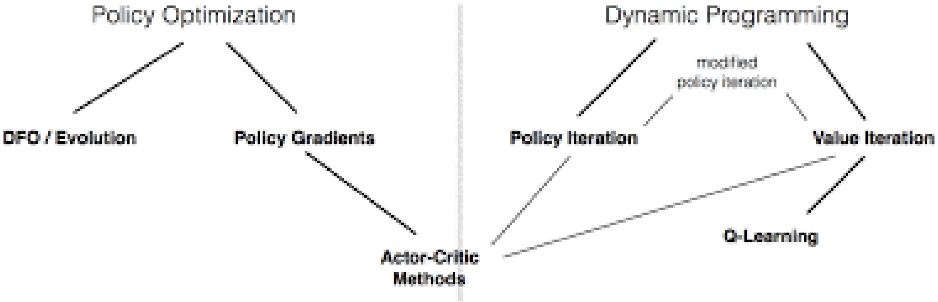
Why Policy Optimization ?

- Often policies (π) are simpler than value functions (V or Q)



- Computing optimal V is bit of problem (we did not see any control algorithms for V)
- With state-value functions Q, computing $\arg\max$ over actions gets tricky when action space is large or continuous
- Better convergence properties
- Can learn stochastic policies

RL Algorithms Landscape [2]



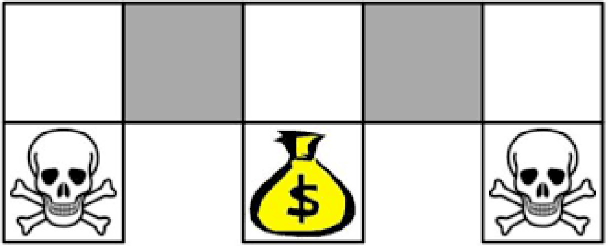
We will now actually look out for the optimal policies in the stochastic policy space !

[2] Slide content from Schulman

Example : Rock-Paper-Scissors

- Two player game of rock-paper-scissors
  - ★ Scissors beats paper
  - ★ Rock beats scissors
  - ★ Paper beats rock
- Consider policies for iterated rock-paper-scissors
  - ★ A deterministic policy is easily exploited
  - ★ A uniform random policy is optimal (i.e. Nash equilibrium)

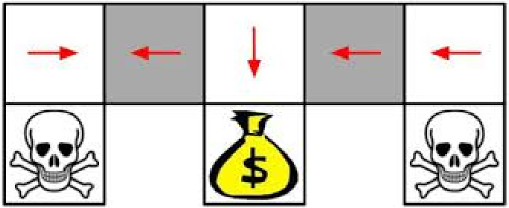
Example : Aliased Grid World



- The agent cannot differentiate the grey states
- For example, state could be represented by features of the following form

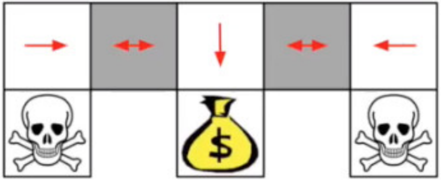
$$\psi(s,a) = 1(\text{wall to } \mathbf{S}, \text{ a=move } \mathbf{E})$$

Example : Aliased Grid World



- Under aliasing, an optimal deterministic policy will either
  - ★ move W in both grey states (shown as above)
  - ★ move E in both grey states
- Either way, it can get stuck and never reach the money
- Value based RL learns a near deterministic policy (greedy or ϵ greedy)
- Such a policy will go back and forth on the grid for a long time before hitting money

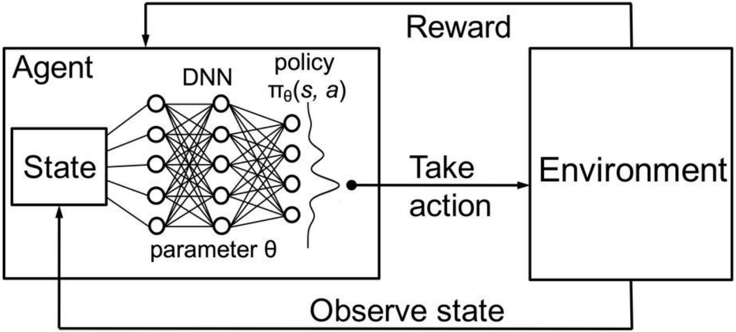
Example : Aliased Grid World



- An optimal stochastic policy will randomly move E or W in grey states
- It will reach the goal state in a few steps with high probability
- Policy-based RL can learn the optimal stochastic policy

Function Approximation

Policy Using Function Approximators



- If action space is discrete
  - ★ Network could output a vector of probabilities (softmax)
- If action space is continuous
  - ★ Network could output the parameters of a distribution (For e.g.. mean and variance of a Gaussian)

Continuous Action Space : Gaussian Policies

- Policy is Gaussian
- The mean (μ) of the Gaussian could be the output of the neural network
- The variance σ of the Gaussian could be constant or can be parametrized.
- One way to operate in continuous action space is to sample an action from the Gaussian distribution. i.e., $a \sim \mathcal{N}(\mu, \sigma)$
- Idea can be extended to any parametrized probability distribution (even multi-variable).

Policy Optimization

A policy $\pi(\cdot)$ is parametrized by parameter θ and denoted by π_θ

Performance of a policy π_θ is given by

$$J(\theta) = V^{\pi_\theta}(s) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} | s_0 = s\right]$$

Goal of RL is to find a policy

$$\pi_\theta^* = \arg\max_{\pi_\theta} V^{\pi_\theta}(s) = \arg\max_{\pi_\theta} \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} | s_0 = s\right]$$

We will look for π_θ^* in class of stochastic policies by finding θ that maximizes $J(\theta)$

Policy Gradient

- Let $J(\theta)$ be the policy objective function
- Policy gradient algorithms search for a local maximum in $J(\theta)$ by ascending the gradient of the policy, w.r.t. parameters θ

$$\Delta\theta = \alpha \nabla_\theta J(\theta)$$

- $\nabla_\theta J(\theta)$ is the policy gradient and
- α is the step size parameter