Policy Gradient

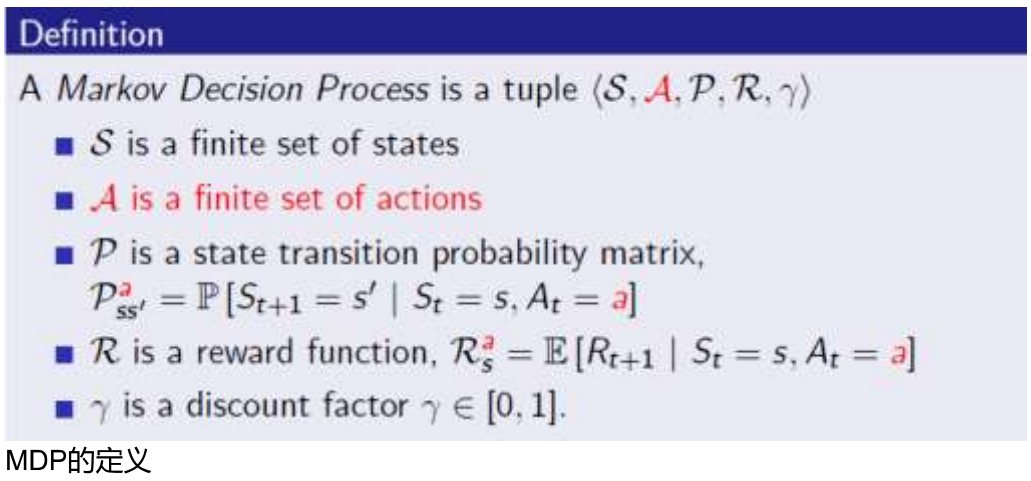
Policy Gradient

https://www.jianshu.com/p/af668c5d783d (https://www.jianshu.com/p/af668c5d783d)

虽然前段时间稍微了解过Policy Gradient，但后来发现自己对其原理的理解还有诸多模糊之处，于是希望重新梳理一番。
Policy Gradient的基础是强化学习理论，同时我也发现，由于强化学习的术语众多，杂乱的符号容易让我迷失方向，所以对我自己而言，很有必要重新确立一套统一的符号使用习惯。UCL的David Silver可谓是强化学习领域数一数二的专家（AlphaGo首席研究员），他的课程在网上也大受欢迎，因此我接下来用于讨论问题的符号体系就以他的课件为准。

Markov Decision Process (MDP)

在概率论和统计学中，**Markov Decision Processes (MDP)** 提供了一个数学架构模型，刻画的是“**如何在部分随机，部分可由决策者控制的状态下进行决策**”的过程。强化学习的体系正是构建在MDP之上的。



MDP的定义

有了这样的定义，自然引申出**policy**和**reward**的概念：

policy的定义

Definition

A *policy* π is a distribution over actions given states,

$$\pi(a|s) = \mathbb{P}[A_t = a \mid S_t = s]$$

reward的定义

Definition

The *return* G_t is the total discounted reward from time-step t.

$$G_t = R_{t+1} + \gamma R_{t+2} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

Value function

Value function也是MDP中一个非常重要的概念，衡量的是从某个状态开始计算的reward期望值，但容易令初学者混淆的是，value function一般有两种定义方式。
一种叫**state-value function**：

Definition

The *state-value function* $v_\pi(s)$ of an MDP is the expected return starting from state s, and then following policy π

$$v_\pi(s) = \mathbb{E}_\pi[G_t \mid S_t = s]$$

另一种叫**action-value function**，会**显式地将当前采取的动作**纳入考量之中：

Definition

The *action-value function* $q_\pi(s, a)$ is the expected return starting from state s, taking action a, and then following policy π

$$q_\pi(s, a) = \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a]$$

从定义上看，两者显然可以互相转换：$$v_\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s) q_\pi(s, a)$$

$$q_\pi(s, a) = \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_\pi(s')$$

另外，如果仔细观察**reward**的定义

$$G_t = R_{t+1} + \gamma R_{t+2} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

会发现这两种**value function**其实都可以写成递归的形式：

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

$$q_\pi(s, a) = \mathbb{E}_\pi[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1}) \mid S_t = s, A_t = a]$$

这又被称为**Bellman Equation**，把**value function**分解成了**immediate reward**加上**后续**状态的**discounted value**。

Policy Gradient

强化学习的一类求解算法是**直接优化policy**，而**Policy Gradient**就是其中的典型代表。
首先需要讨论一下policy的目标函数。一般而言，policy的目标函数主要有三种形式：

- 在**episodic**环境（**有终止状态**，从起始到终止的模拟过程称为一个episode，系统通过一次次地模拟episode进行学习）中，**衡量从起始状态开始计算的value**：

$$J_1(\theta) = V^{\pi_\theta}(s_1) = \mathbb{E}_{\pi_\theta}[v_1]$$

- 在**continuing**环境（**没有终止状态**，是一个无限的过程）中，**衡量value均值**：

$$J_{avV}(\theta) = \sum_s d^{\pi_\theta}(s) V^{\pi_\theta}(s)$$

- 不管在哪个环境中，**只关注immediate reward**，**衡量的是每个时刻的平均reward**：

$$J_{avR}(\theta) = \sum_s d^{\pi_\theta}(s) \sum_a \pi_\theta(s,a) \mathcal{R}_s^a$$

以上的

$d^{\pi_\theta}(s)$ 是指**状态的概率分布**，与policy有关，并且是**stationary distribution of Markov chain**，意思是这个**概率分布不会随着MDP的时间推进而变化。**虽然这三种**目标函数形式不同**，但最后分析得到的**梯度表达式都是一样**的。

对目标函数求梯度会用到一个很重要的trick，叫**likelihood ratios**：

$$\begin{aligned}\nabla_\theta \pi_\theta(s,a) &= \pi_\theta(s,a) \frac{\nabla_\theta \pi_\theta(s,a)}{\pi_\theta(s,a)} \\ &= \pi_\theta(s,a) \nabla_\theta \log \pi_\theta(s,a)\end{aligned}$$

对**目标函数求梯度**最终都是要转化为**对policy求梯度**，而这个转化的作用是为了凑出

$\pi_\theta(s,a)$

项，便于后续化简出**期望项。** 一个简单的例子是考虑最基本的情况——单步的MDP，在执行了一个时间单位之后就终止，所得的reward就等于这个时刻的**immediate reward**，记为

$r = \mathcal{R}_{s,a}$

目标函数就采用上述第三种的形式：

$$\begin{aligned}J(\theta) &= \mathbb{E}_{\pi_\theta}[r] \\ &= \sum_{s \in \mathcal{S}} d(s) \sum_{a \in \mathcal{A}} \pi_\theta(s,a) \mathcal{R}_{s,a}\end{aligned}$$

利用likelihood ratios推导出梯度是：

$$\begin{aligned}\nabla_\theta J(\theta) &= \sum_{s \in \mathcal{S}} d(s) \sum_{a \in \mathcal{A}} \pi_\theta(s,a) \nabla_\theta \log \pi_\theta(s,a) \mathcal{R}_{s,a} \\ &= \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a) r]\end{aligned}$$

有个叫**Policy Gradient Theorem**的理论表明，无论采用上述哪种目标函数，在多步的MDP下，都有：

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a)\ Q^{\pi_\theta}(s,a)]$$

在实际的优化中，采用stochastic gradient ascent算法，对

$Q^{\pi_\theta}(s_t, a_t)$

进行无偏采样，记为

v_t

，因此可以把期望项去掉，参数更新的公式为：

$$\theta \leftarrow \theta + \alpha \nabla_\theta \log \pi_\theta(s_t, a_t) v_t$$

作者：冯乌尔里希
链接：https://www.jianshu.com/p/af668c5d783d (https://www.jianshu.com/p/af668c5d783d)
來源：简书

---

edit (https://blogs.cuit.columbia.edu/zp2130/wp-admin/post.php?post=4507&action=edit)

Author: Z Pei (https://blogs.cuit.columbia.edu/zp2130/author/zp2130/) on February 17, 2019

Categories: AI (https://blogs.cuit.columbia.edu/zp2130/category/ai/), Policy Gradient Methods (https://blogs.cuit.columbia.edu/zp2130/category/policy-gradient-methods/), Reinforcement Learning (https://blogs.cuit.columbia.edu/zp2130/category/reinforcement-learning/), RL (https://blogs.cuit.columbia.edu/zp2130/category/rl/)

Tags: Policy Gradient Methods (https://blogs.cuit.columbia.edu/zp2130/tag/policy-gradient-methods/), Reinforcement Learning (https://blogs.cuit.columbia.edu/zp2130/tag/reinforcement-learning/), RL (https://blogs.cuit.columbia.edu/zp2130/tag/rl/)

Other posts

Actor-Critic Algorithms (https://blogs.cuit.columbia.edu/zp2130/actor-critic_algorithms/) «» Policy Gradient and Q-learning (https://blogs.cuit.columbia.edu/zp2130/policy_gradient_q-learning/)

Last posts

- Symbolic Netlist to Innovus-friendly Netlist (https://blogs.cuit.columbia.edu/zp2130/symbolic_netlist_to_innovus-friendly_netlist/)
- Finite-Sample Convergence Rates for Q-Learning and Indirect Algorithms (https://blogs.cuit.columbia.edu/zp2130/finite-sample_convergence_rates_for_q-learning_and_indirect_algorithms/)
- Solving H-horizon, Stationary Markov Decision Problems In Time Proportional To Log(H) (https://blogs.cuit.columbia.edu/zp2130/paul_tseng_1990/)
- Randomized Linear Programming Solves the Discounted Markov Decision Problem In Nearly-Linear (Sometimes Sublinear) Run Time (https://blogs.cuit.columbia.edu/zp2130/randomized_linear_programming_solves_the_discounted_markov_decision_problem_in_nearly-linear_sometimes_sublinear_run_time/)
- KL Divergence (https://blogs.cuit.columbia.edu/zp2130/kl_divergence/)
- The Asymptotic Convergence-Rate of Q-learning (https://blogs.cuit.columbia.edu/zp2130/the_asymptotic_convergence-rate_of_q-learning/)
- Hierarchical Apprenticeship Learning, with Application to Quadruped Locomotion (https://blogs.cuit.columbia.edu/zp2130/hierarchical_apprenticeship_learning_with_application_to_quadruped_locomotion/)
- Policy Gradient Methods (https://blogs.cuit.columbia.edu/zp2130/policy_gradient_methods/)
- Actor-Critic Algorithms for Hierarchical Markov Decision Processes (https://blogs.cuit.columbia.edu/zp2130/actor-critic_algorithms_for_hierarchical_markov_decision_processes/)
- Hierarchical Deep Reinforcement Learning: Integrating Temporal Abstraction and Intrinsic Motivation (https://blogs.cuit.columbia.edu/zp2130/hierarchical_deep_reinforcement_learning_integrating_temporal_abstraction_and_intrinsic_motivation/)