# 强化学习（二）

关键词：
#Incremental_Implementation
#Tracking_a_nonstationary_problem
#Optimistic_Initial_Values
#Reinforcement_Comparison
#Pursuit_Methods
#Associative_Search

## Incremental Implementation

上一篇讲到，我们通过均值来评估某一个行为的价值。

$$Q_t(a) = \frac{r_1 + r_2 + r_3 + \ldots + r_{k_a}}{k_a}$$

为了使公式更加通用，我们改写上式

$$Q_t(a) = \sum_{i=1}^{t} \frac{r_i}{t}$$

如果在时间步 i，执行的动作不是 a，那么我们认为 $r_i = 0$。

为了能够尽可能地占用更少的内存空间，提高运行速度。实际上，我们并不会每次都按照上面的公式计算行为的价值，而是动态更新。

$$Q_t(a) = Q_{t-1}(a) + \frac{1}{t}[r_t - Q_{t-1}(a)]$$

上式中，$r_t$ 为时间步 t 获得的关于行为 a 的奖励信号。
这样的话，我们不需要存储每个时间步执行的行为和对应的奖励信号，只需要存储最新时间步各个行为对应的价值 $Q_t(a)$，以及最新时间步 t 即可推导出 $Q_{t+1}(a)$

我们可以将上面的更新策略总结为：

$$NewEstimate = OldEstimate + Stepsize[Target - OldEstimate]$$

$Target - OldEstimate$ 根据最新的奖励信号（$Target$）来促使 $OldEstimate$ 向着 $Target$ 步进。

## Tracking a Nonstationary Problem

我们之前探讨的大多是 stationary problem，即问题遵循假设

$$\text{行为的价值和时间步无关}$$

这个假设在实际问题中是很难成立的，因此我们接下来将讨论"行为的价值随时间步的变化而变化，而且价值随时间的变化比较缓慢"的情况。

只需要将

$$NewEstimate = OldEstimate + Stepsize[Target - OldEstimate]$$

中的 Stepsize 设置为一个常数（constant）即可一定程度上解决问题。

即我们的更新策略为：

$$Q_t(a) = Q_{t-1}(a) + c[r_t - Q_{t-1}(a)]$$

其中，$0 < c < 1$

根据这个更新策略化简，事实上我们可以得到下面这个式子

$$Q_t(a) = (1-c)^t Q_0(a) + \sum_{i=1}^{t} c(1-c)^{t-i} r_i$$

经过数学推导，也不难得到

$$(1-c)^t + \sum_{i=1}^{t} c(1-c)^{t-i} = 1$$

所以该更新策略本质上就是一种加权求和，而且越靠近当前时间步的奖励信号的权值也就越高。

正因为这个特点，该更新策略使得我们评估当前时间步某行为的价值时，着重考虑离现在最近的时间步，又由于行为的价值随时间的变化比较缓慢，我们得到的评估就会比较准确。

# Optimistic Initial Values

仔细观察我们两种更新策略的式子，针对

$$Q_t(a) = Q_{t-1}(a) + \frac{1}{t}[r_t - Q_{t-1}(a)]$$

我们会发现，只要一个行为被采取过，那么 $Q_0(a)$ 就不会对之后行为的价值产生任何影响。而

$$Q_t(a) = Q_{t-1}(a) + c[r_t - Q_{t-1}(a)]$$

中，$Q_0(a)$ 永远会对最后的结果产生影响。

因此，我们可以通过设置超高（Optimistic）的 $Q_0(a)$ 来鼓励系统多去探索不同的行为。

一旦我们设置了超高的 $Q_0(a)$ ，每次系统尝试一个行为后，产生的奖励信号都会抑制该行为，系统会对这个行为感到 ‘失望’，从而尝试其他行为。

但是这种方法，只能在刚开始的时候鼓励系统探索。随着时间步 t 的加大，$Q_0(a)$ 的权值近乎为 0，几乎不再影响行为的价值。

# Reinforcement Comparison

Reinforcement Comparison 是一种通过对比的方式更新的策略。

我们将每次执行行为后获得的奖励信号与一个参考奖励信号对比，如果大于参考值，我们认为这个行为是积极的、应该被执行的。反之，类似。

针对参考奖励信号，我们按照移动平均的方式更新

$$\bar{r}_{t+1} = \bar{r}_t + c[r_t - \bar{r}_t]$$

这个值描述了奖励信号大致的平均值。

我们不再记录每个行为的价值，而是记录系统对每个行为的偏好。
我们用 $P_t(a)$ 表示时间步为 t 时，系统对行为 a 的偏好；$\pi_t(a)$ 表示系统选择行为 a 的概率（通过softmax得到）。

更新偏好的策略如下：

$$P_{t+1}(a_t) = P_t(a_t) + \beta[r_t - \bar{r}_t]$$

Reinforcement Comparison 与之前方法的不同点在于 Reinforcement Comparison 是将奖励信号和平均奖励信号对比，来决定是 激活 还是 抑制，而之前讲述的方法，是将本次奖励信号和之前奖励信号对比，更新行为的价值后，再比较价值的大小决定行为的优劣。

# Pursuit Methods

这种方法既保留了对行为价值的评估，还借鉴了行为偏好的设定。
每次更新使得系统对 价值最高的行为 的偏好提升。就好像是在追赶 'greedy action'，这里我不再赘述具体实现，实际上，所有这些更新方法都大差不差，没有本质的提升（这只是笔者的个人观点，如果您有其他见解，欢迎讨论）。