强化学习基础篇（六）动态规划之策略迭代（2）

1、策略改进（Policy improvement）的理论证明

考虑对一个确定性策略（Deterministic policy）$a = \pi(s)$，我们可以通过执行贪婪计算不断优化改进策略，即：

$$\pi'(s) = \underset{a \in A}{argmax}\, q_\pi(s, a)$$

在这个过程中每次只使用一次步骤改善状态s的动作值函数q。即：

$$q_\pi(s, \pi'(s)) = \max_{a \in A} q_\pi(s, a) \geq q_\pi(s, \pi(s)) = v_\pi(s)$$

如下将证明策略提升定理$v_\pi(s) \leq v_{\pi'}(s)$：

$$\begin{aligned}
v_\pi(s) &\leq q_\pi(s, \pi'(s)) && \text{已知条件} \\
&= \mathbb{E}_{\pi'}[R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s] && \text{在当前状态} S_t = s \text{时，采用动作} \pi'(s) \text{并跳转到下一个状态} S_{t+1} \\
&\leq \mathbb{E}_{\pi'}[R_{t+1} + \gamma q_\pi(S_{t+1}, \pi'(S_{t+1})) | S_t = s] && \text{根据} v_\pi(S_{t+1}) \leq q_\pi(S_{t+1}, \pi'(S_{t+1})) \\
&= \mathbb{E}_{\pi'}[R_{t+1} + \gamma \mathbb{E}_{\pi'}[R_{t+2} + \gamma v_\pi(S_{t+2}) | S_t = s]] && \text{跳转} S_{t+2} \\
&= \mathbb{E}_{\pi'}[R_{t+1} + \gamma R_{t+2} + \gamma^2 v_\pi(S_{t+2}) | S_t = s] && \text{对} \pi' \text{求两次期望还是等于对} \pi' \text{求期望} \\
&\leq \mathbb{E}_{\pi'}[R_{t+1} + \gamma R_{t+2} + \gamma^2 q_\pi(S_{t+2}, \pi'(S_{t+2})) | S_t = s] \\
&\leq \mathbb{E}_{\pi'}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \ldots | S_t = s] \\
&= v_{\pi'}(s)
\end{aligned}$$

如果q值不再改善，则在某一状态下，遵循当前策略采取的行为得到的q值将会是最优策略下所能得到的最大q值。

$$q_\pi(s, \pi'(s)) = \max_{a \in A} q_\pi(s, a) \geq q_\pi(s, \pi(s)) = v_\pi(s)$$

上述表示就满足了Bellman最优方程，说明当前策略下的状态价值就是最优状态价值：

$$\pi'(s) = \underset{a \in A}{argmax}\, q_\pi(s, a)$$

对于所有的$s \in S$，都满足$v_\pi(s) = v_*(s)$，因此π是最优策略。

2. 策略迭代算法伪代码

算法 2 策略迭代算法（利用迭代式策略评价）

1: 随机初始化 $V(s)$ 和 $\pi(s)$
2: **repeat**
3: 　对于当前策略 π，使用迭代式策略评价的算法估计 $v_\pi(s)$ 得到 $V(s)$
4: 　使用贪婪策略提升得到 $\pi'(s)$
5: **until** 策略保持不变 $\pi'(s) = \pi(s), \forall s$

PS. 算法中$V(s)$函数表示估计值，$v(s)$表示真实值。

3. 策略迭代的修改

策略迭代在每一个迭代步总是先对策略进行值函数估计，直至收敛，那我们能否在策略估计还未收敛时就进行策略改进呢?

可能有如下几种思路：

- 引入epsilon收敛
- 简单地在对策略估计迭代k次之后就进行策略改进。
- 在迭代$k=1$次就进行策略改进，迭代$k=1$次就等同于值迭代(value iteration)。

4. 广义策略迭代（Generalized Policy Iteration)

策略迭代包括两个同时进行的交互过程，一个使得值函数（value function）与当前策略一致（策略评价 policy evaluation），另一个使得策略相对于当前值函数较贪婪（策略提升 policy improvement）。

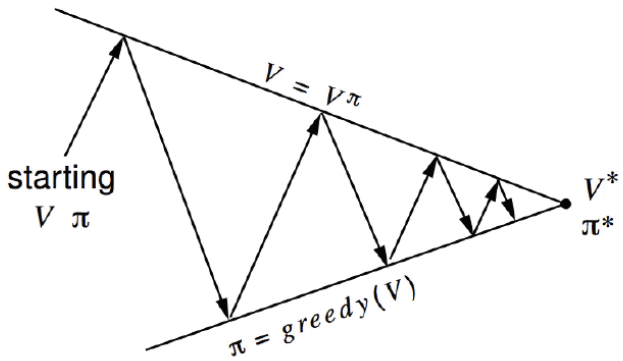
在策略迭代中，这两个过程交替进行，每个过程在另一个过程开始之前完成，但这显然不是必需的。 例如，值迭代（value iteration）中，在每个策略提升（policy improvement）之间仅执行一次策略评估（policy evaluation）迭代。 在异步（asynchronous）动态规划时，评价和提升过程则以更精细的方式交错。 只要两个过程都持续更新所有的状态，那么最终结果通常是相同的，即收敛到最优值函数和最优策略。

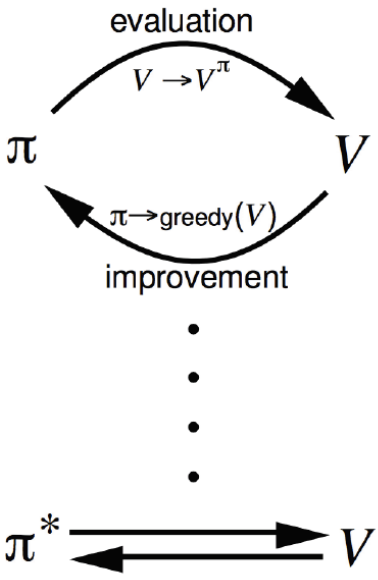
使用术语——广义策略迭代（Generalized Policy iteration，GPI）来指代让策略评价和策略提升交互的一般概念，而不依赖于两个过程的粒度（granularity）和其他细节。

几乎所有强化学习方法都可以被很好地描述为GPI。也就是说，它们都具有可识别的策略 π (identifiable policy）和值函数V，策略π 总是相对于值函数V被改善，并且值函数V总是趋向策略π下的值函数V^{π} 。

> the policy always being improved with respect to the value function.
>
> the value function always being driven toward the value function for the policy.

他们的交互过程如下所示：





Policy evaluation Estimate v_{π}
Any policy evaluation algorithm
Policy improvement Generate $\pi' \geq \pi$
Any policy improvement algorithm

如果评价过程和提升过程都稳定下来，即不再发生变化，那么值函数和策略必须都是最优的。这意味着贝尔曼最优方程成立。

$$
\begin{aligned}
v_*(s) &= \max_a \mathbb{E}[R_{t+1} + \gamma v_*(S_{t+1}|S_t = s, A_t = a)] \\
&= \max_a \sum_{s',r} p(s', r|s, a)[r + \gamma v^*(s')]
\end{aligned}
$$

还可以用两个目标来考虑GPI中评价和提升过程的相互作用，如上图所示，上面的线代代表目标 $v = v^{\pi}$ ，下面的线代表目标$\pi = greedy(V)$。目标会发生相互作用，因为两条线不是平行的。从一个策略π 和一个价值函数v开始，每一次箭头向上代表着利用当前策略进行值函数的更新，每一次箭头向下代表着根据更新的值函数贪婪地选择新的策略，说它是贪婪的，是因为每次都采取转移到可能的、状态函数最高的新状态的行为。最终将收敛至最优策略和最优值函数。