第二次作业

1. 阅读 Sutton 书第三章

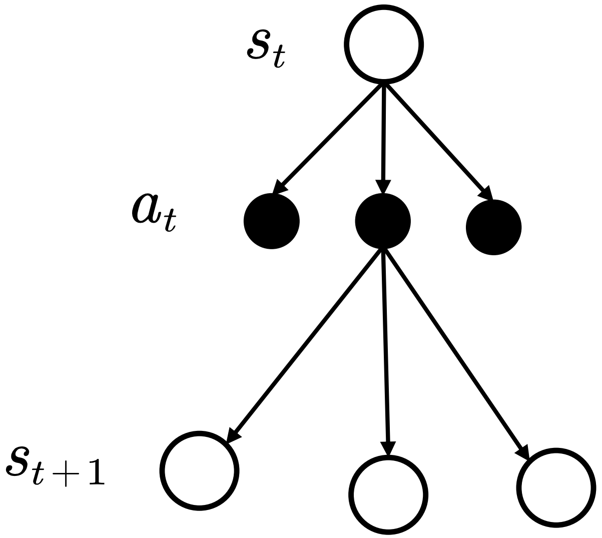
2. 安装 gym，阅读gym中的代码，写出gym中至少三款游戏的状态、动作、回报、状态转移概率如何设置。

解惑时间

1. 随机策略和状态转移概率什么区别?

策略的定义： $\pi(a|s)$

状态转移概率的定义： $P(s_{t+1}|s_t,a_t)$



第三讲：从动态规划到强化学习

郭宪

2019.10.13

人工智能学院
College of Artificial Intelligence
南開大學
Nankai University

数学规划：单阶段决策

数学规划：求一个数学函数的极值。

目标函数和约束条件均为线性函数，称为线性规划，否则称为非线性规划。

例 1.2.1　生产计划问题

设某工厂用 4 种资源生产 3 种产品，每单位第 j 种产品需要第 i 种资源的数量为 a_{ij}，可获利润为 c_j，第 i 种资源总消耗量不能超过 b_i，由于市场限制，第 j 种产品的产量不超过 d_j，试问如何安排生产才能使总利润最大？

$$
\begin{aligned}
\max \quad & \sum_{j=1}^{3} c_j x_j \\
\text{s. t.} \quad & \sum_{j=1}^{3} a_{ij} x_j \leqslant b_i, && i = 1, \cdots, 4, \\
& x_j \leqslant d_j, && j = 1, 2, 3, \\
& x_j \geqslant 0, && j = 1, 2, 3,
\end{aligned}
$$

陈宝林，　最优化：理论与算法（第二版），清华大学出版社，2005

动态规划：多阶段决策

最短路径问题：从 q 到 r 的最短路径

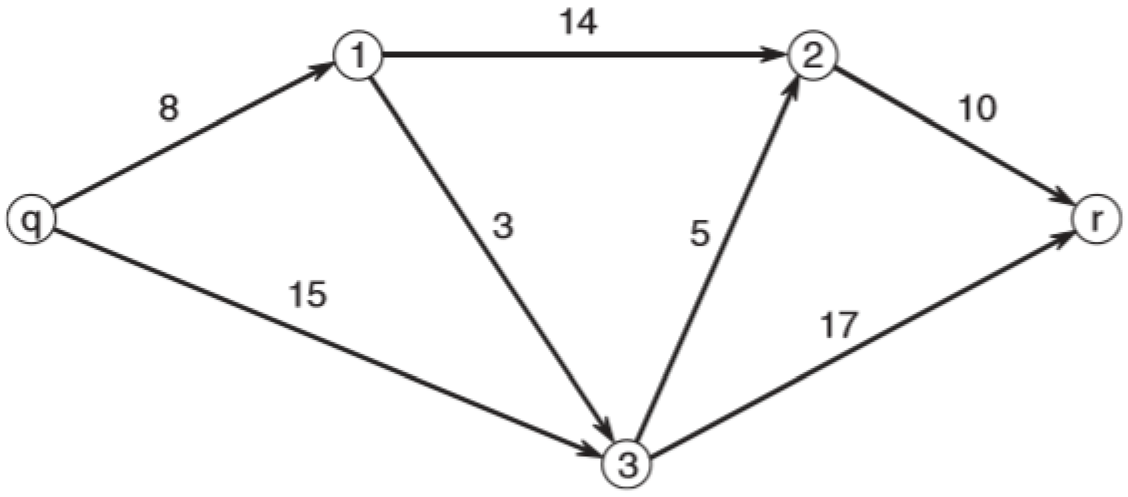


Table 1.1 **Path cost from each node to node *r* after each node has been visited**

	Cost from Node				
Iteration	q	1	2	3	r
	100	100	100	100	0
1	100	100	10	15	0
2	30	18	10	15	0
3	26	18	10	15	0
4	26	18	10	15	0

$$J^*[x(j),j]=\min_{\substack{u(j)\in U\\ x(j+1)\in X}}\left\{L(x(j),u(j),j)+J^*[x(j+1),j+1]\right\}$$

值函数的更新公式为：

$$\upsilon_i \leftarrow \min\left\{\upsilon_i,\min_{j\in I^+}\{c_{ij}+\upsilon_j\}\right\}$$

Warren B. Powell,
Approximate Dynamic Programming: Solving the Curses of Dimensionality

动态规划的本质

动态规划的本质是：将多阶段决策问题通过贝尔曼方程转化为多个单阶段的决策问题

离散贝尔曼方程：

$$J^*[x(j),j]=\min_{u(j)\in U}\min_{\{u(j+1),\cdots,u(N-1)\}\in U}\{L[x(j),u(j),j]+\sum_{k=j+1}^{N-1}L[x[k],u[k],k]\}$$

$$=\min_{\substack{u(j)\in U\\ x(j+1)\in X}}\left\{L(x(j),u(j),j)+J^*[x(j+1),j+1]\right\}$$

$$=\min_{\substack{u(j)\in U\\ x(j+1)\in X}}\left\{L(x(j),u(j),j)+J^*[f[x(j),u(j),j],j+1]\right\}$$

求出值函数后，通过贪婪策略重构出最优策略

最优控制中的动态规划

动态规划的本质是：将多阶段决策问题通过贝尔曼方程转化为多个单阶段的决策问题

连续贝尔曼方程：

$$J^*[x(t),t]=\min_{u[t,t+\Delta t]}\{\min_{u[t+\Delta t,t_f]}\left[\int_t^{t+\Delta t}L(x(\tau),u(\tau),\tau)\right]d\tau+\int_{t+\Delta t}^{t_f}L(x(\tau),u(\tau),\tau)d\tau+\varphi[x(t_f),t_f]\}$$

$$=\min_{\substack{u(\tau)\in U\\ t\le\tau\le t+dt}}\left\{\int_t^{t+dt}L[x(\tau),u(\tau),\tau]d\tau+J^*[x(t)+dx(t),t+dt]\right\} \qquad (1)$$

将 $J^*[x(t)+dx(t),t+dt]$ 进行泰勒展开有：

$$J^*[x(t)+dx(t),t+dt]=J^*[x(t),t]+\frac{\partial J^*[x(t),t]}{\partial x^T(t)}dx(t)+\frac{\partial J^*[x(t),t]}{\partial t}dt+\varepsilon[dx(t),dt] \qquad (2)$$

胡寿松等， 最优控制理论与系统，科学出版社，2005

将（2）带入（1），并令 $dt\to 0$ Hamilton-Jacobi-Bellman方程

$$-\frac{\partial J^*[x(t),t]}{\partial t}=\min_{u(t)\in U}\{L[x(t),u(t),t]+\frac{\partial J^*[x(t),t]}{\partial x^T(t)}f[x(t),u(t),t]\}\xRightarrow{\text{重构}}\min_{u(t)\in U}\left\{L[x(t),u(t),t]+\frac{\partial J^*[x(t),t]}{\partial x^T(t)}f[x(t),u(t),t]\right\}$$

微分动态规划方法

动态规划：

$$V(X,t)=\min_{u\in\Omega}\left\{\phi[X(t_f),t_f]+\int_{t_0}^{t_f}L(x(t),u(t),t)\right\}$$

$$=\min_{u\in\Omega}\left\{\int_{t_0}^{t_0+dt}L(x(\tau),u(\tau),\tau)d\tau+V(X+\Delta X,t+dt)\right\}$$

$$x_{k+1}=f(x_k,u_k)$$

$$V_k=\min_u\left[l(x_k,u_k)+V_{k+1}(x_{k+1})\right]$$

令：$Q(\delta x,\delta u)=V(x+\delta x)-V(x)$

将L和V在标称轨线 (x_k,u_k) 展开

$$Q(\delta x,\delta u)=V(x+\delta x)-V(x)$$

$$=l(x_k+\delta x_k,u_k+\delta u_k)+V_{k+1}(x_{k+1}+\delta x_{k+1})-(l(x_k,u_k)+V_{k+1}(x_{k+1}))$$

$$\approx\delta x_k^T l_{x_k}+\delta u_k^T l_{u_k}+\frac{1}{2}\left(\delta x_k^T l_{xx_k}\delta x_k^T+2\delta x_k^T l_{xu_k}\delta u_k+\delta u_k^T l_{uu_k}\delta u_k\right)+\delta x_{k+1}^T V_{x_{k+1}}+\delta x_{k+1}^T\frac{1}{2}V_{xx_{k+1}}\delta x_{k+1}$$

$$\delta x_{k+1}=\delta f(x_k,u_k)=f_{x_k}\delta x_k+f_{u_k}\delta u_k+\frac{1}{2}\left(\delta x_k^T f_{xx_k}\delta x_k+2\delta x_k^T f_{xu_k}\delta u_k+\delta u_k^T f_{uu_k}\delta u_k\right)$$

微分动态规划方法

微分动态规划：

$$Q(\delta x,\delta u)=\frac{1}{2}\begin{bmatrix}1\\ \delta x\\ \delta u\end{bmatrix}^T\begin{bmatrix}0 & Q_x^T & Q_u^T\\ Q_x & Q_{xx} & Q_{xu}\\ Q_u & Q_{ux} & Q_{uu}\end{bmatrix}\begin{bmatrix}1\\ \delta x\\ \delta u\end{bmatrix}$$

$$\Downarrow$$

$$Q(\delta x,\delta u)=\frac{1}{2}\left[\delta u^T Q_{uu}\delta u+\left(\delta u^T Q_{ux}\delta x+\delta x^T Q_{xu}\delta u\right)+Q_u^T\delta u+\delta u^T Q_u+\delta x^T Q_{xx}\delta x+\delta x^T Q_x+Q_x^T\delta x\right]$$

$$\delta u^*=\arg\min_{\delta u} Q(\delta x,\delta u)=-Q_{uu}^{-1}\left(Q_u+Q_{ux}\delta x\right)$$

$$\Delta V=-\frac{1}{2}Q_u Q_{uu}^{-1}Q_u$$

$$V_x=Q_x-Q_u Q_{uu}^{-1}Q_{ux}$$

$$V_{xx}=Q_{xx}-Q_{xu}Q_{uu}^{-1}Q_{ux}$$

$$Q_x=l_{x_k}+f_{x_k}^T V_{x_{k+1}}$$

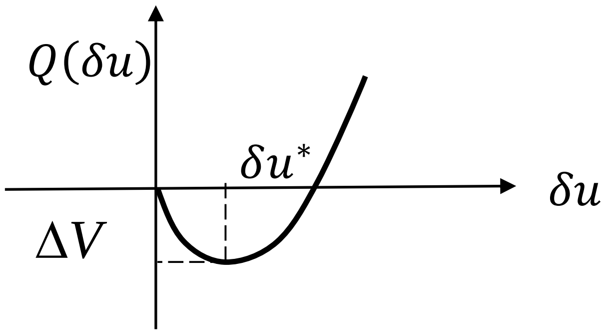
$$Q_u=l_{u_k}+f_{u_k}^T V_{x_{k+1}}$$

$$Q_{xx}=l_{xx_k}+f_{x_k}^T V_{xx_{k+1}} f_{x_k}+V_{x_{k+1}} f_{x_k x_k}$$

$$Q_{uu}=l_{uu_k}+f_{u_k}^T V_{xx_{k+1}} f_{u_k}+V_{x_{k+1}} f_{uu_k}$$

$$Q_{ux}=l_{ux_k}+f_{u_k}^T V_{xx_{k+1}} f_{x_k}+V_{x_{k+1}} f_{ux_k}$$

迭代计算



微分动态规划方法

微分动态规划：

1. 前向迭代：给定初始控制序列 $\bar{u}_k$ 正向迭代计算标称轨迹 $\bar{x}_{k+1} = f(\bar{x}_k, \bar{u}_k), l_{x_k}, f_{u_k}, l_{xx_k}, l_{xu_k}, l_{uu_k}$

2. 反向迭代：由代价函数边界条件 V_{x_N}, V_{xx_N}反向迭代计算（1），（2），(3) 得到k_k, K_k序列

3. 正向迭代新的控制序列：

$$k_k, K_k$$

Yuval Tassa, et.al. Synthesis and Stabilization of Complex Behaviors through Online Trajectory Optimization, 2012

$$\begin{aligned}
Q_x &= l_{x_k} + f_{x_k}^T V_{x_{k+1}} \\
Q_u &= l_{u_k} + f_{u_k}^T V_{x_{k+1}} \\
Q_{xx} &= l_{xx_k} + f_{x_k}^T V_{xx_{k+1}} f_{x_k} + V_{x_{k+1}} f_{x_k x_k} \\
Q_{uu} &= l_{uu_k} + f_{u_k}^T V_{xx_{k+1}} f_{u_k} + V_{x_{k+1}} f_{uu_k} \\
Q_{ux} &= l_{ux_k} + f_{u_k}^T V_{xx_{k+1}} f_{x_k} + V_{x_{k+1}} f_{ux_k}
\end{aligned} \quad (1)$$

$$\begin{aligned}
\delta u^* &= \arg\min_{\delta u} Q(\delta x, \delta u) = k + K\delta x \\
k &= -Q_{uu}^{-1} Q_u, \quad K = -Q_{uu}^{-1} Q_{ux}
\end{aligned} \quad (2)$$

$$\begin{aligned}
\Delta V &= -\frac{1}{2} Q_u Q_{uu}^{-1} Q_u \\
V_{x_k} &= Q_x - Q_u Q_{uu}^{-1} Q_{ux} \\
V_{xx_k} &= Q_{xx} - Q_{xu} Q_{uu}^{-1} Q_{ux}
\end{aligned} \quad (3)$$

从动态规划到强化学习

控制领域集中于连续高维问题

最优控制：模型已知，立即回报解析地给出，状态空间小。

近似动态规划：利用机器学习方法学习值函数，解决维数灾难

计算机领域：集中于离散大规模问题

以马尔科夫决策过程为基础，　强化学习　　无模型，回报函数不解，状态空间无穷

动态规划：策略评估（估计值函数），策略改进

所有强化学习算法都可以看成是动态规划算法，只是用更少的计算，并且没有假设完美模型

值函数（续）

MDP

动作

	1	2	3
4	5	6	7
8	9	10	11
12	13	14	

状态空间：S={1,2..14}

动作空间:{东，南，西，北}

回报函数：−1，直到终止状态

均匀随机策略：

$\pi(东|\cdot)$=0.25， $\pi(南|\cdot)$=0.25， $\pi(西|\cdot)$=0.25， $\pi(北|\cdot)$=0.25

当智能体采用策略 π 时，累积回报服从一个概率分布，累积回报在状态s处的期望值定义为值函数：

$$v_\pi(s) = E_\pi[G_t|S_t = s] = E_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1}|S_t = s\right]$$

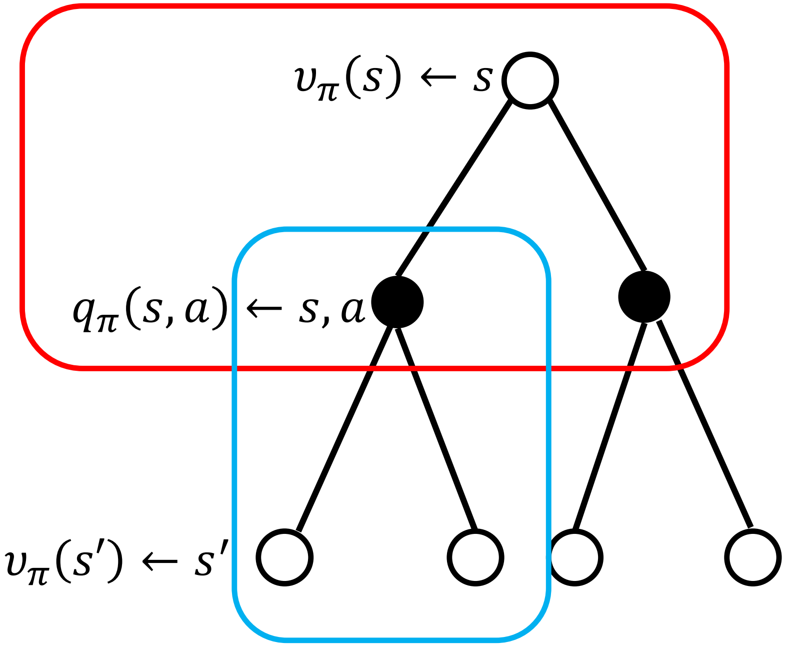
状态−行为值函数：

$$q_\pi(s,a) = E_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1}|S_t = s, A_t = a\right]$$

0.0	-14	-20	-22
-14	-18	-20	-20
-20	-20	-18	-14
-22	-20	-14	0.0

策略评估(policy evaluation)

给定策略 π 构造值函数：



$$v_\pi(s) = \sum_{a \in A} \pi(a|s)\, q_\pi(s,a)$$

$$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a v_\pi(s')$$

$$\Downarrow$$

$$\boxed{v_\pi(s) = \sum_{a \in A} \pi(a|s) \left(R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a v_\pi(s') \right)}$$

模型已知，方程组中只有值函数是未知数，方程组是线性方程组。未知数的数目等于状态的数目。

$$\Downarrow$$

采用数值迭代算法

策略评估(policy evaluation)

$$v_\pi(s) = \sum_{a\in A} \pi(a|s)\left(R_s^a + \gamma \sum_{s'\in S} P_{ss'}^a v_\pi(s')\right)$$

$\Downarrow$ 高斯-赛德尔迭代

$$v_{k+1}(s) = \sum_{a\in A} \pi(a|s)\left(R_s^a + \gamma \sum_{s'\in S} P_{ss'}^a v_k(s')\right)$$

策略评估算法

[1] 输入：需要评估的策略 π 状态转移概率 $P_{ss'}^a$,回报函数 R_s^a ，折扣因子 γ

[2] 初始化值函数：$V(s) = 0$

一次状态扫描

[3] Repeat k=0,1,…

[4] for every s do

[5] $$v_{k+1}(s) = \sum_{a\in A} \pi(a|s)\left(R_s^a + \gamma \sum_{s'\in S} P_{ss'}^a v_k(s')\right)$$

[6] end for

[7] Until $v_{k+1} = v_k$

[8] 输出： $v(s)$

高斯-赛德尔迭代

线性方程组的数值求解算法：

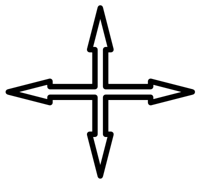
$$AX = b$$

高斯-赛德尔方法：

$$\begin{cases} x_1^{(k+1)} = \dfrac{1}{a_{11}}(-a_{12}x_2^{(k)} - a_{13}x_3^{(k)} - \cdots - a_{1n}x_n^{(k)} + b_1) \\ x_2^{(k+1)} = \dfrac{1}{a_{22}}(-a_{21}x_1^{(k+1)} - a_{23}x_3^{(k)} - \cdots - a_{2n}x_n^{(k)} + b_2) \\ \cdots \\ x_n^{(k+1)} = \dfrac{1}{a_{nn}}(-a_{n1}x_1^{(k+1)} - a_{n2}x_2^{(k+1)} - \cdots - a_{n,n-1}x_{n-1}^{(k+1)} + b_n) \end{cases}$$

策略评估(policy evaluation)

MDP



动作

	1	2	3
4	5	6	7
8	9	10	11
12	13	14	

状态空间：S={1,2..14}

动作空间:{东，南，西，北}

回报函数：−1，直到终止状态

均匀随机策略：

π(东| ·)=0.25， π(南| ·)=0.25， π(西| ·)=0.25，
π(北| ·)=0.25

策略评估算法

输入：需要评估的策略 π 状态转移概率 $P_{ss'}^{a}$,回报函数 R_s^a ，折扣因子 γ

初始化值函数：$V(s)=0$

一次状态扫描

Repeat k=0,1,…

for every s do

$$v_{k+1}(s)=\sum_{a\in A}\pi(a|s)\left(R_s^a+\gamma\sum_{s'\in S}P_{ss'}^{a}v_k(s')\right)$$

end for

Until $v_{k+1}=v_k$

输出： $v(s)$

策略评估(policy evaluation)

0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0

K=0

0.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	0.0

K=1

0.0	-1.7	-2.0	-2.0
-1.7	-2.0	-2.0	-2.0
-2.0	-2.0	-2.0	-1.7
-2.0	-2.0	-1.7	0.0

K=2

0.0	-14	-20	-22
-14	-18	-20	-20
-20	-20	-18	-14
-22	-20	-14	0.0

$K=\infty$

算法2：策略评估算法

输入：需要评估的策略 π 状态转移概率 $P_{ss'}^a$，回报函数 R_s^a，折扣因子 γ

初始化值函数：$V(s)=0$

一次状态扫描

Repeat k=0,1,…

for every s do

$$v_{k+1}(s)=\sum_{a\in A}\pi(a|s)\left(R_s^a+\gamma\sum_{s'\in S}P_{ss'}^a v_k(s')\right)$$

end for

Until $v_{k+1}=v_k$

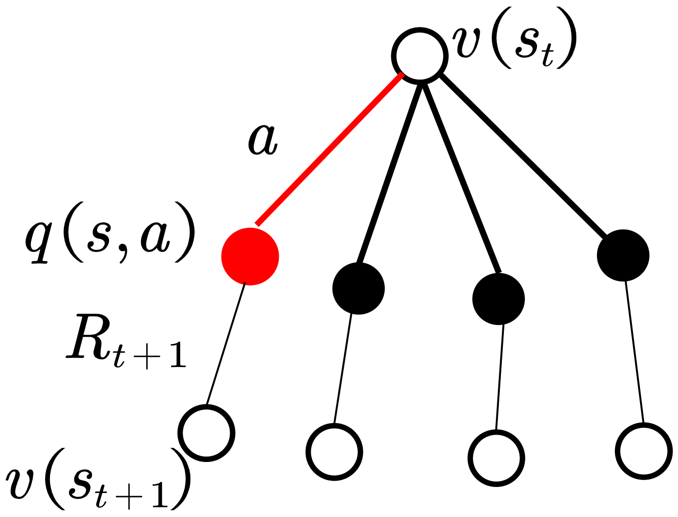
输出：$v(s)$

策略改进(policy improvement)

0.0	-14	-20	-22
-14	-18	-20	-20
-20	-20	-18	-14
-22	-20	-14	0.0

π_0 均匀策略：

$\xrightarrow{?}$ 更好的策略 π_1



$$q_\pi(s,a) = \mathbb{E}[R_{t+1} + \gamma v_\pi(s_{t+1}) | s_t = s, a_t = a]$$

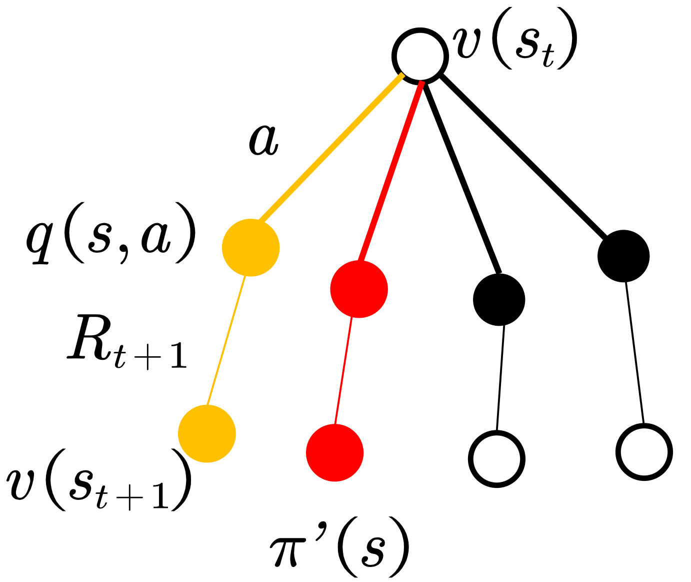
策略改善理论：

如果： $q_\pi(s, \pi'(s)) \geqslant v_\pi(s)$

那么：π' 不比 π 差，甚至比之还要好

策略改进证明(policy improvement)

$$
\begin{aligned}
& v_\pi(s) \leqslant q_\pi(s, \pi'(s)) \\
& = \mathbb{E}[R_{t+1} + \gamma v_\pi(s_{t+1}) | s_t = s,\ a_t = \pi'(s)] \\
& = \mathbb{E}_{\pi'}[R_{t+1} + \gamma v_\pi(s_{t+1}) | s_t = s] \\
& \leqslant \mathbb{E}_{\pi'}[R_{t+1} + \gamma q_\pi(s_{t+1}, \pi'(s_{t+1})) | s_t = s] \\
& = \mathbb{E}_{\pi'}[R_{t+1} + \gamma \mathbb{E}_{\pi'}[R_{t+2} + \gamma v_\pi(s_{t+1}) | s_{t+1}, a_{t+1} = \pi'(s_{t+1})] | s_t = s] \\
& = \mathbb{E}_{\pi'}[R_{t+1} + \gamma R_{t+2} + \gamma^2 v_\pi(s_{t+2}) | s_t = s] \\
& \leqslant \mathbb{E}_{\pi'}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 v_\pi(s_{t+2}) | s_t = s] \\
& \vdots \\
& \leqslant \mathbb{E}_{\pi'}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \cdots | s_t = s] \\
& = v_{\pi'}(s)
\end{aligned}
$$



最直观的一个改进策略是什么？ 贪婪策略！

策略改进(policy improvement)

计算策略值的目的是为了帮助找到更好的策略，在每个状态采用贪婪策略。

$$\pi_{l+1}(s) \in \arg\max_{a} q^{\pi_l}(s, a)$$

π_0 均匀策略：↑↓←→

π_1 贪婪策略：

K=10

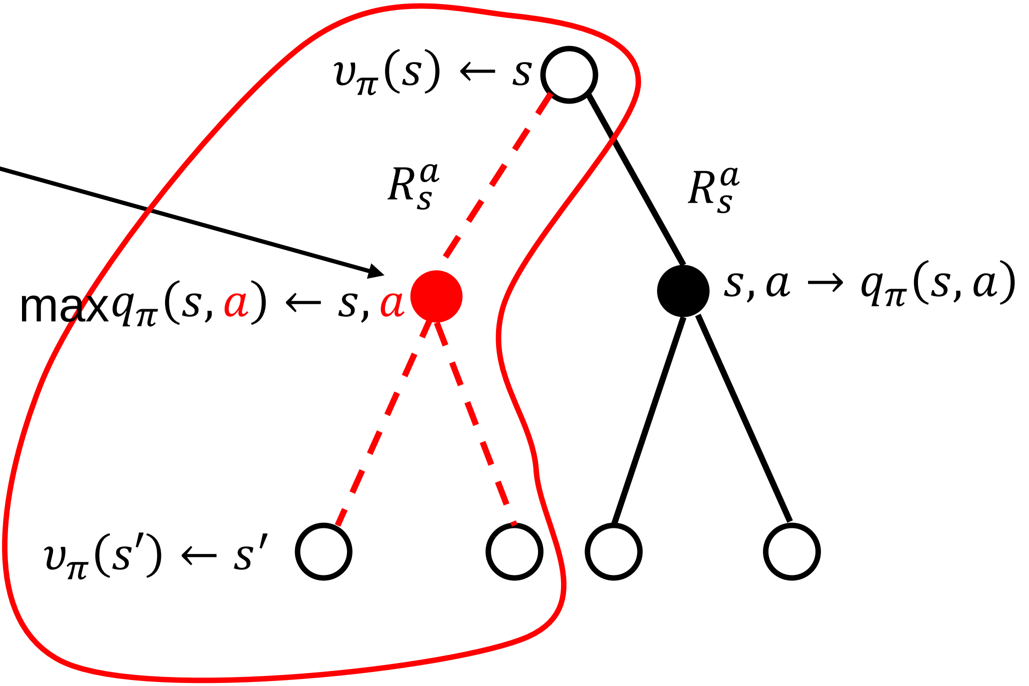
0.0	-6.1	-8.4	-9.0
-6.1	-7.7	-8.4	-8.4
-8.4	-8.4	-7.7	-6.1
-9.0	-8.4	-6.1	0.0

0.0	←	←	←↓
↑	←↑	←↓	↓
↑	↑→	↓→	↓
↑→	→	→	0.0

$K = \infty$

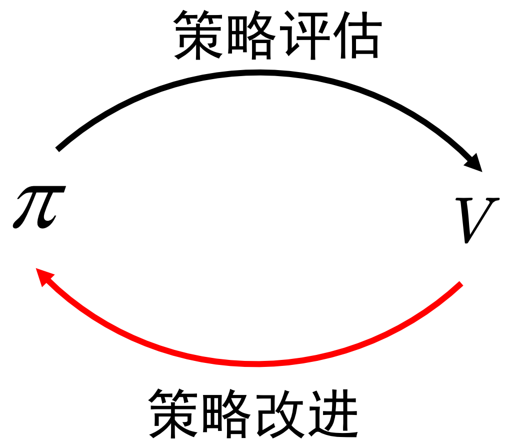
0.0	-14	-20	-22
-14	-18	-20	-20
-20	-20	-18	-14
-22	-20	-14	0.0

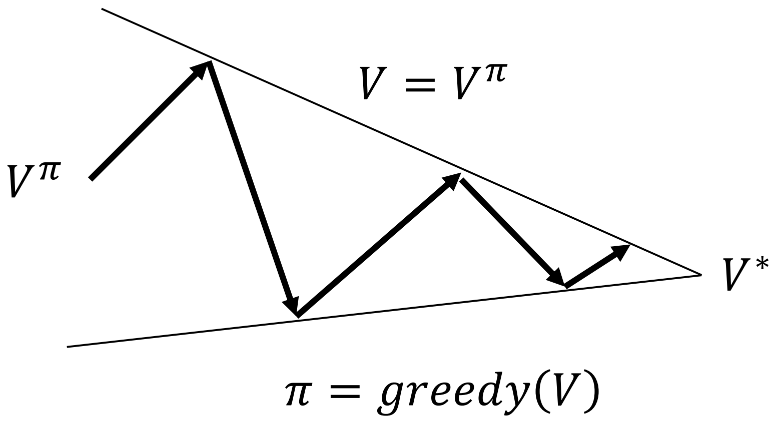
0.0	←	←	←↓
↑	←↑	←↓	↓
↑	↑→	↓→	↓
↑→	→	→	0.0



$$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a v_\pi(s')$$

策略迭代(policy iteration)





算法1：策略迭代算法

[1] 输入：状态转移概率 $P_{ss'}^{a}$,回报函数R_s^a ，折扣因子

初始化值函数：$V(s) = 0$ 初始化策略 π_0

[2] Repeat l=0,1,…

[3] find V^{π_l} Policy evaluation

[4] $\pi_{l+1}(s) \in \arg\max_a q^{\pi_l}(s,a)$ Policy improvement

[5] Until $\pi_{l+1} = \pi_l$

[6] 输出：$\pi^* = \pi_l$

值函数迭代

策略改进一定要等到值函数收敛吗？

π_0 均匀策略：↑↓←→ $\qquad$ π_1 贪婪策略：

K=10

0.0	-6.1	-8.4	-9.0
-6.1	-7.7	-8.4	-8.4
-8.4	-8.4	-7.7	-6.1
-9.0	-8.4	-6.1	0.0

0.0	←	←	←↓
↑	←↑	←↓	↓
↑	↑→	↓→	↓
↑→	→	→	0.0

$K = \infty$

0.0	-14	-20	-22
-14	-18	-20	-20
-20	-20	-18	-14
-22	-20	-14	0.0

0.0	←	←	←↓
↑	←↑	←↓	↓
↑	↑→	↓→	↓
↑→	→	→	0.0

当K=1时便进行策略改进，得到值函数迭代算法

$$v^*(s) = \max_a R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a v_*(s')$$

[1] 输入：状态转移概率 $P_{ss'}^a$,回报函数R_s^a ，折扣因子γ

初始化值函数:$v(s) = 0$ 初始化策略 π_0

[2] Repeat l=0,1,…

[3] for every s do

[4] $$v_{l+1}(s) = \max_a R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a v_l(s')$$

[5] Until $v_{l+1} = v_l$

[6] 输出： $\pi(s) = \operatorname{argmax}_a R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a v_l(s')$

第三次作业

1.阅读《Reinforcement Learning： An Introduction》第四章

2. 利用策略迭代和值迭代解决鸳鸯找朋友的问题

3. 利用策略迭代和值迭代解决gym中离散的问题

4. 注册华为云并实名认证