十六、强化学习

主讲教师：俞扬

强化学习

从预测到决策：如何种出好瓜

种子 -> 💀

种子 -> 浇水 -> 浇水 -> 浇水 -> 💀

种子 -> 浇水 -> 施肥 -> 施肥 -> 施肥 -> 💀

种子 -> 浇水 -> 施肥 -> 浇水 -> 施肥 -> 施肥 -> 💀

种子 -> 浇水 -> 施肥 -> 浇水 -> 施肥 -> 浇水 -> 施肥 -> 💀

种子 -> 杀虫 -> 浇水 -> 施肥 -> 浇水 -> 施肥 -> 浇水 -> 施肥 -> 💀

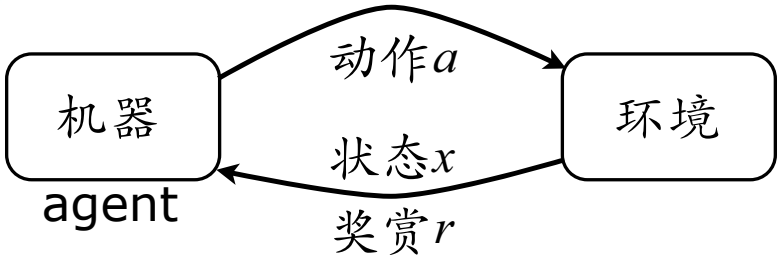
种子 -> 浇水 -> 施肥 -> 浇水 -> 施肥 -> 杀虫 -> 浇水 -> 施肥 -> 杀虫 -> 💀

种子 -> 除草 -> 浇水 -> 施肥 -> 浇水 -> 施肥 -> 杀虫 -> 浇水 -> 施肥 -> 杀虫 -> 💀

种子 -> 浇水 -> 施肥 -> 浇水 -> 施肥 -> 除草 -> 杀虫 -> 浇水 -> 施肥 -> 杀虫 -> 除草 -> 浇水 -> 施肥 -> 浇水 -> 施肥 -> 浇水 -> 施肥 -> 除草 -> 杀虫 -> 🍉

强化学习

关键要素：<*A*, *X*, *R*, *P*>



action space: A
state space: X
reward: $R : X \times A \times X \to \mathbb{R}$
transition: $P : X \times A \times X \to \mathbb{R}$

策略： $a = \pi(x)$

$$P(a|x) = \pi(x,a) \qquad \sum_{a \in A} \pi(x,a) = 1 \qquad \forall a \in A, \pi(x,a) \geq 0$$

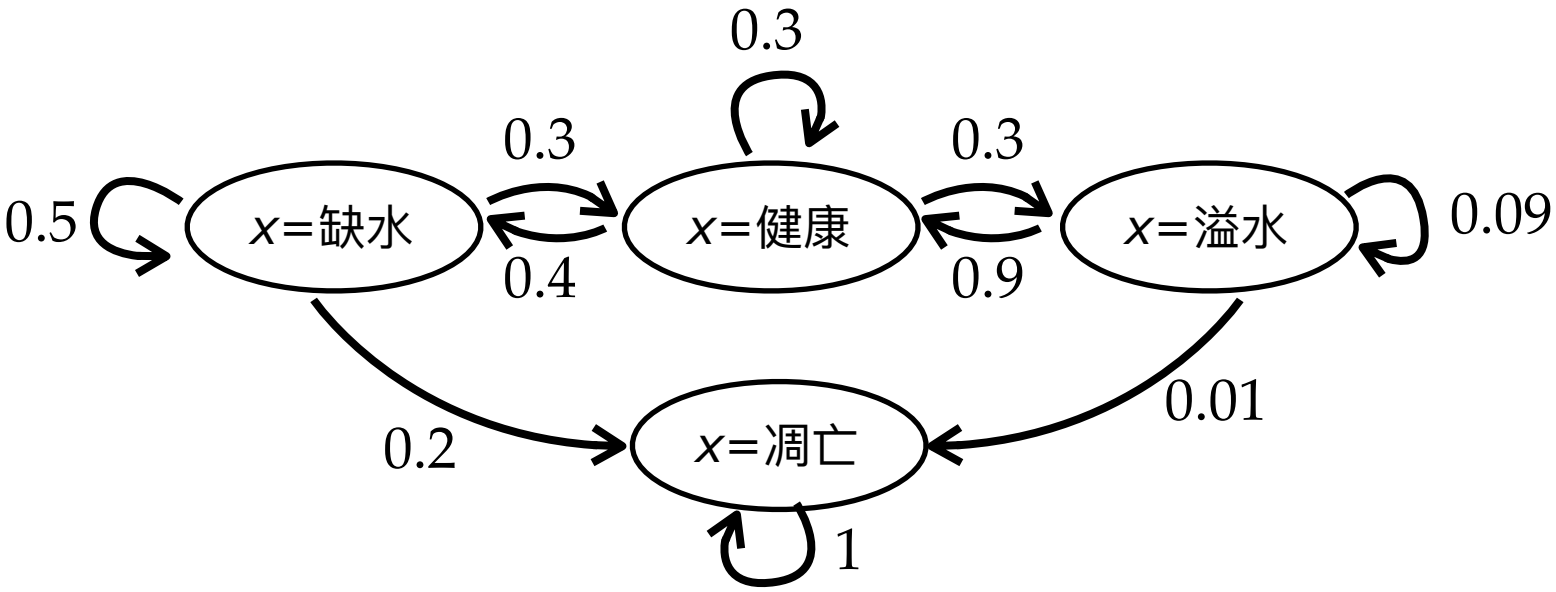
策略评价：累积回报

T-step: $\frac{1}{T}\sum_{t=1}^{T} r_t$ discounted: $\sum_{t=1}^{\infty} \gamma^t r_t$

学习目标：学习最大回报策略

马尔可夫过程 $< X, P >$

状态图



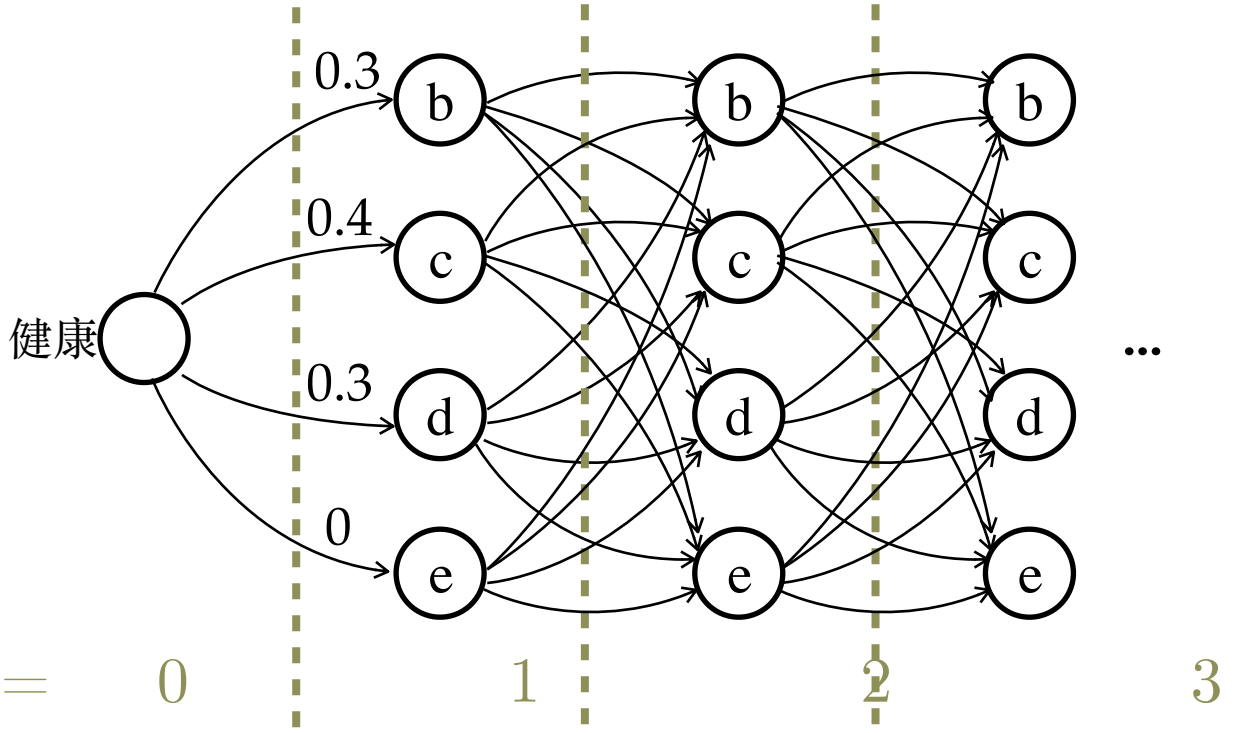
没有记忆的过程

$$P(x_{t+1}|x_t, \ldots, x_0) = P(x_{t+1}|x_t)$$

稳态分布 $\lim_{t\to\infty} P(x_{t+1}) - P(x_t) = 0$

马尔可夫过程 $< X, P >$

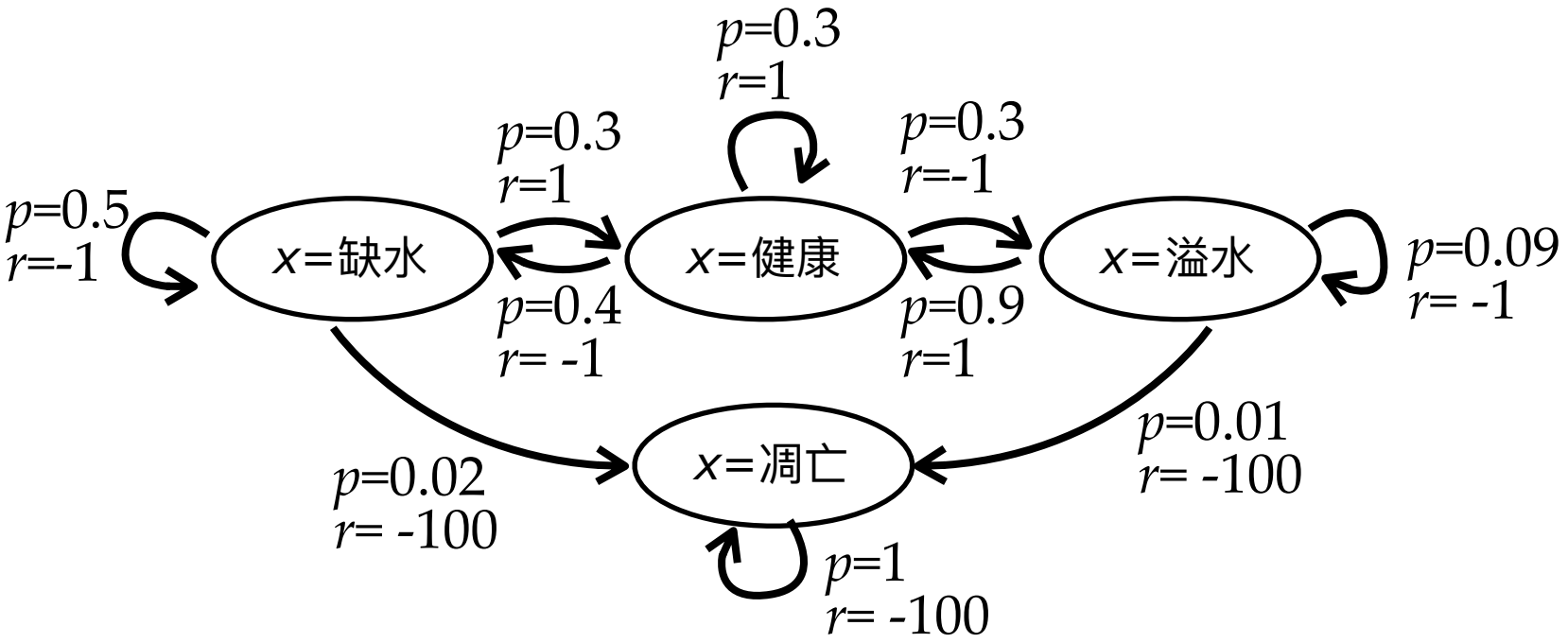
水平视角



采样：运行过程，得到过程采样

马尔可夫回报过程 MRP < X, R, P >

状态图

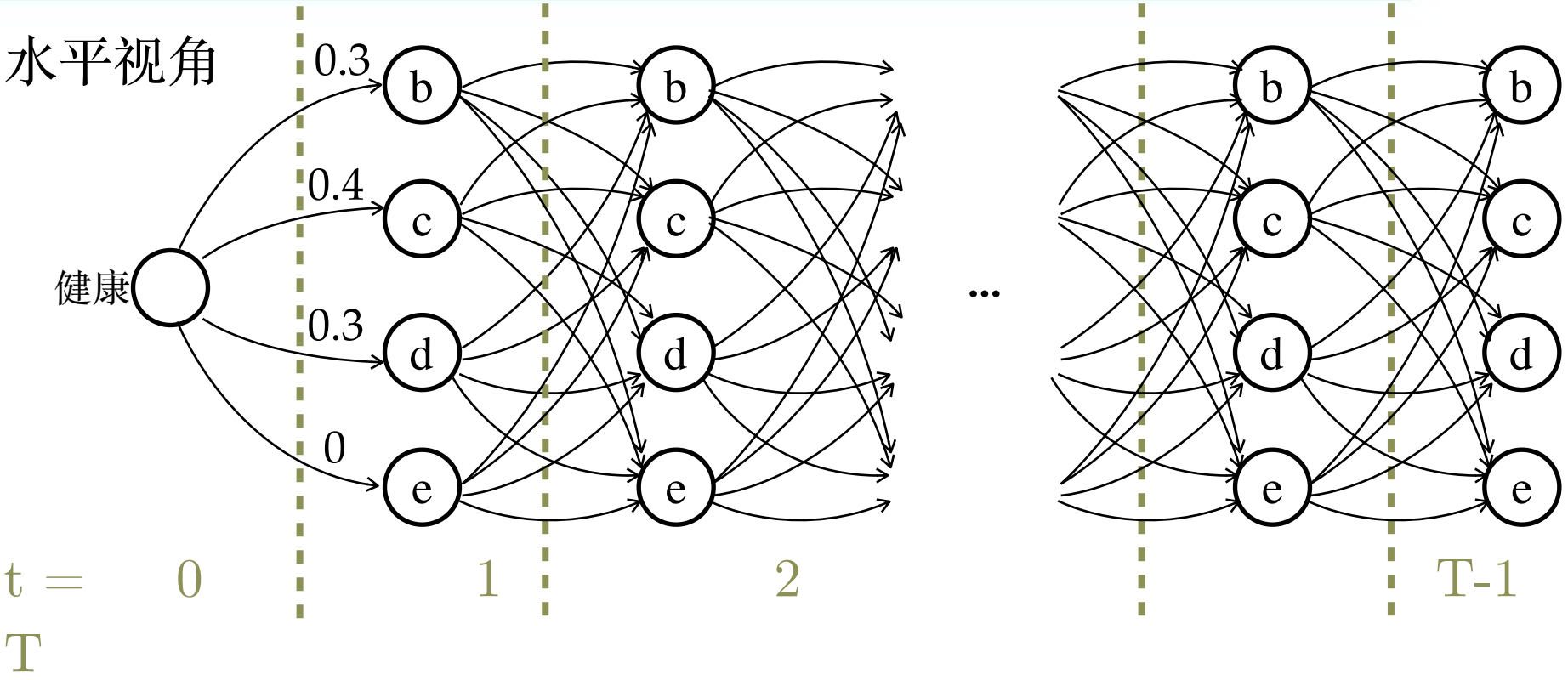


累积回报是多少?

$$V(b) = E[\sum_{t=1}^{T} r_t | x_0 = b]$$

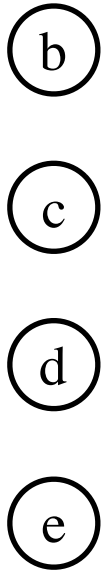
$$V(b) = E[\sum_{t=1}^{T} \gamma r_t | x_0 = b]$$

马尔可夫回报过程 MRP < X, R, P >



马尔可夫回报过程 MRP < X, R, P >

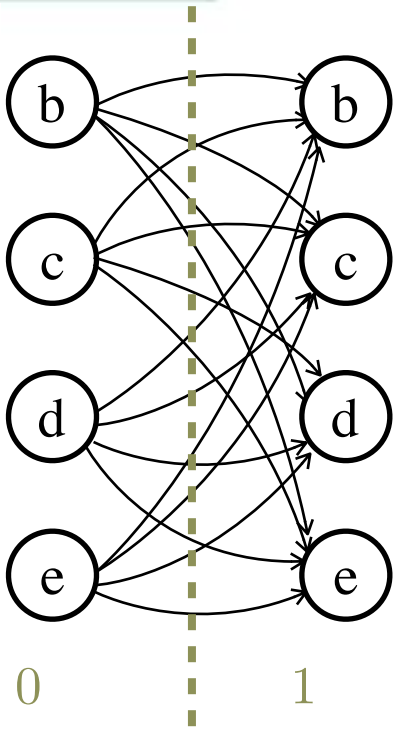
水平视角



$V(x) = 0$

马尔可夫回报过程 MRP < X, R, P >

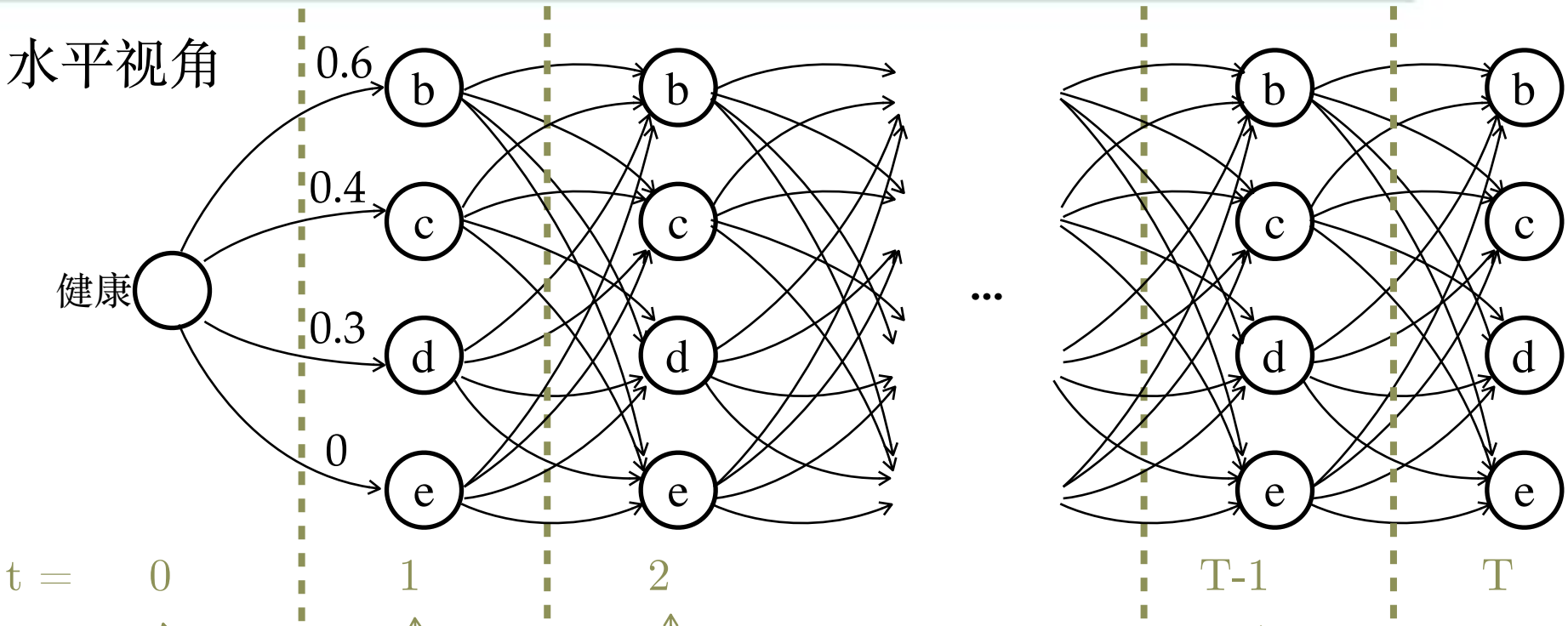
水平视角



$$V(x) = \sum_{x' \in X} P(x'|x) R(x', x) \quad V(x) = 0$$

$$V(x) = \sum_{x'} P(x'|x)(R(x', x) + V(x'))$$

马尔可夫回报过程 MRP < X, R, P >



$V(x) = 0$

迭代计算 $V(x) = \sum_{x'} P(x'|x)(R(x', x) + V(x'))$

马尔可夫回报过程 MRP < X, R, P >

迭代计算

$V(x) = 0$ for all x

loop

for each x: $V'(x) = \sum_{x'} P(x'|x)(R(x', x) + V(x'))$

V=V'

until T iterations

折扣回报

$V(x) = 0$ for all x

while true

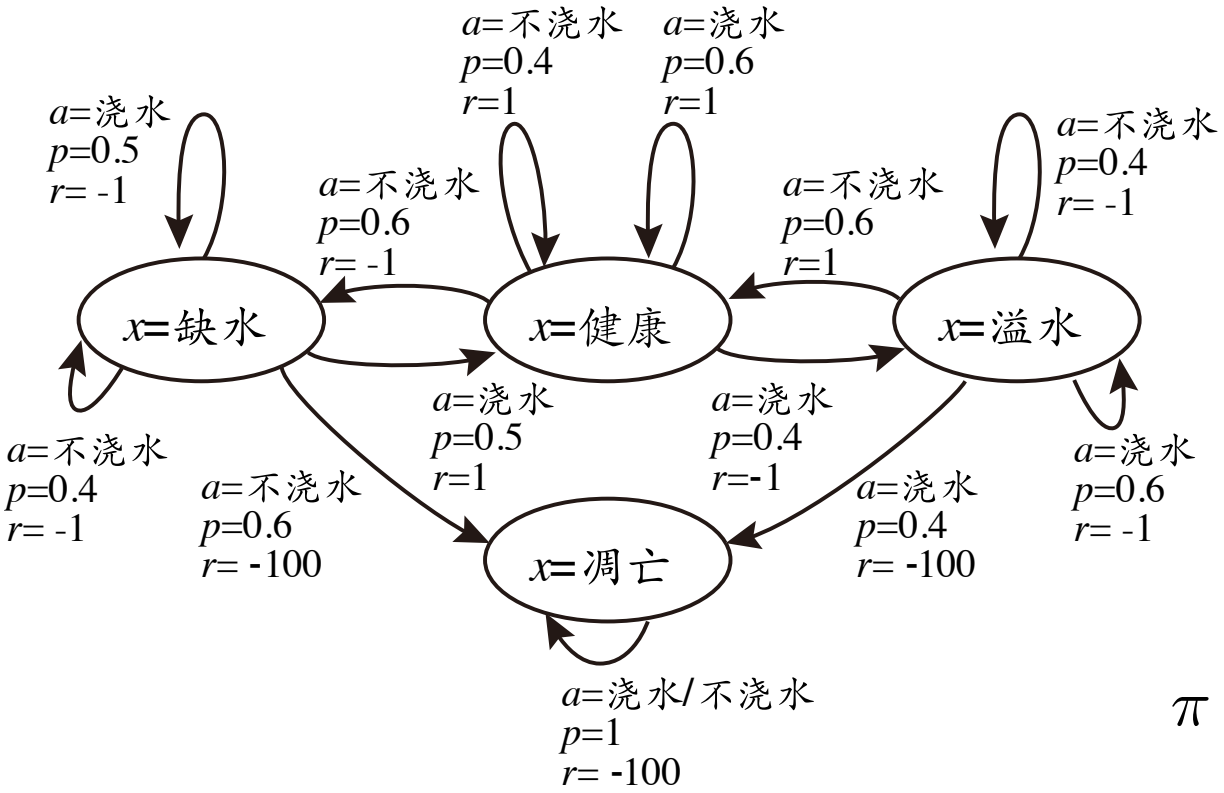
for each x: $V'(x) = \sum_{x'} P(x'|x)(R(x', x) + \gamma V(x'))$

break if $||V - V'|| < \theta$

V=V'

马尔可夫决策过程 MDP < *A*, *X*, *R*, *P* >

状态图

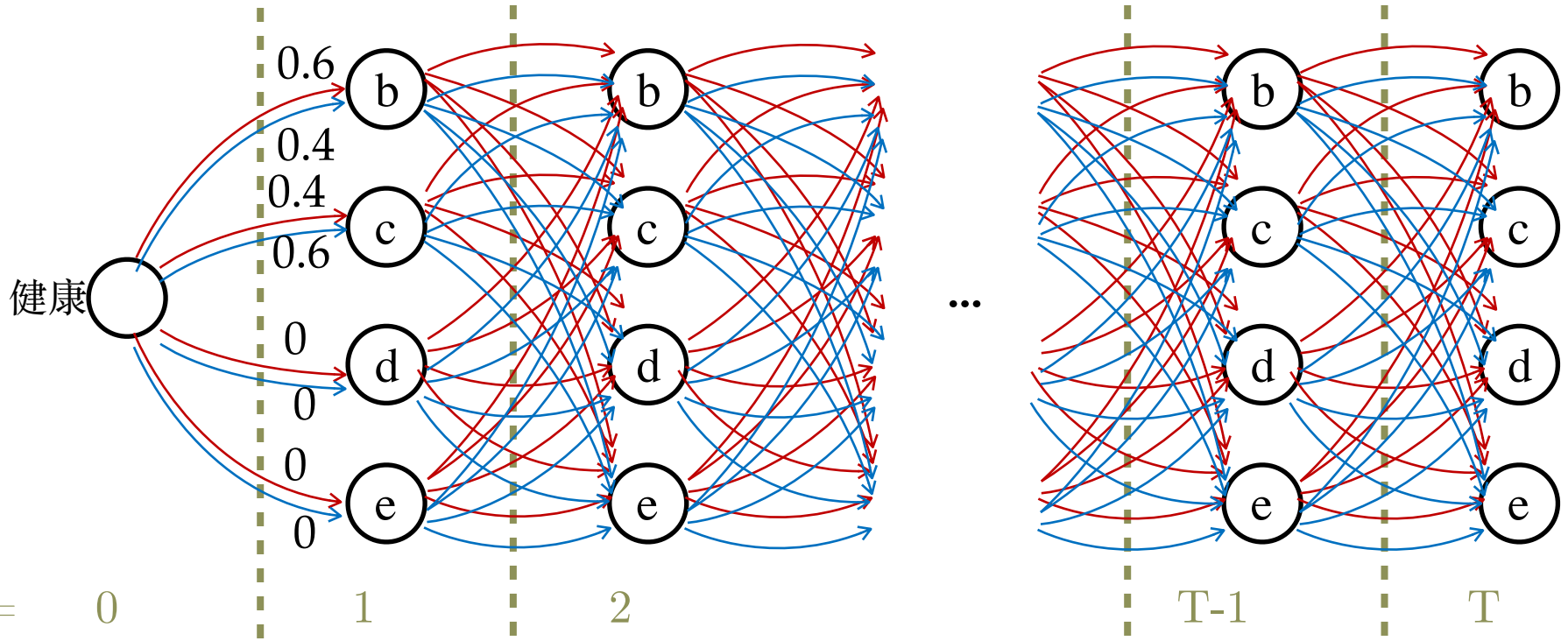


π =

表格策略表达

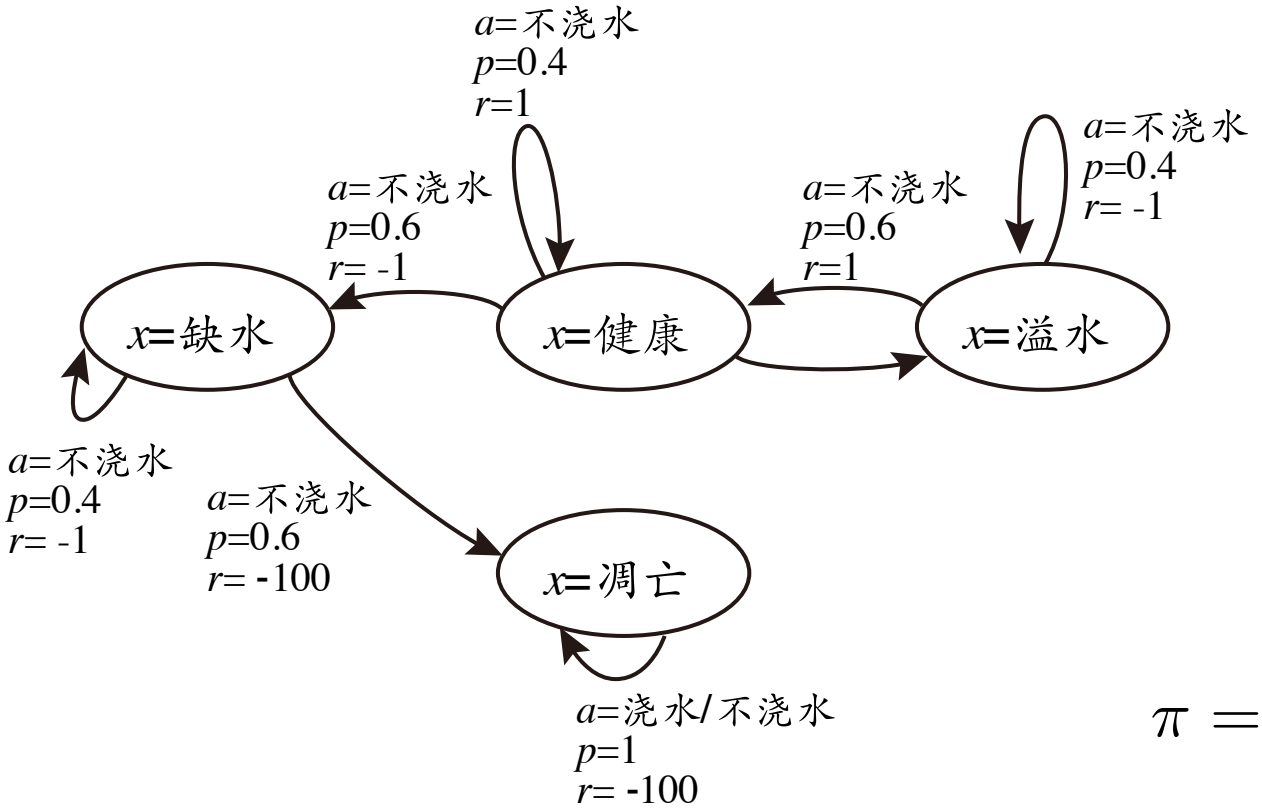
b	浇	0.3
	不	0.7
c	浇	0.6
	不	0.4
d	浇	0.1
	不	0.9
e	浇	0.5
	不	0.5

马尔可夫决策过程 MDP < *A*, *X*, *R*, *P* >



马尔可夫决策过程 MDP < *A*, *X*, *R*, *P* >

状态图



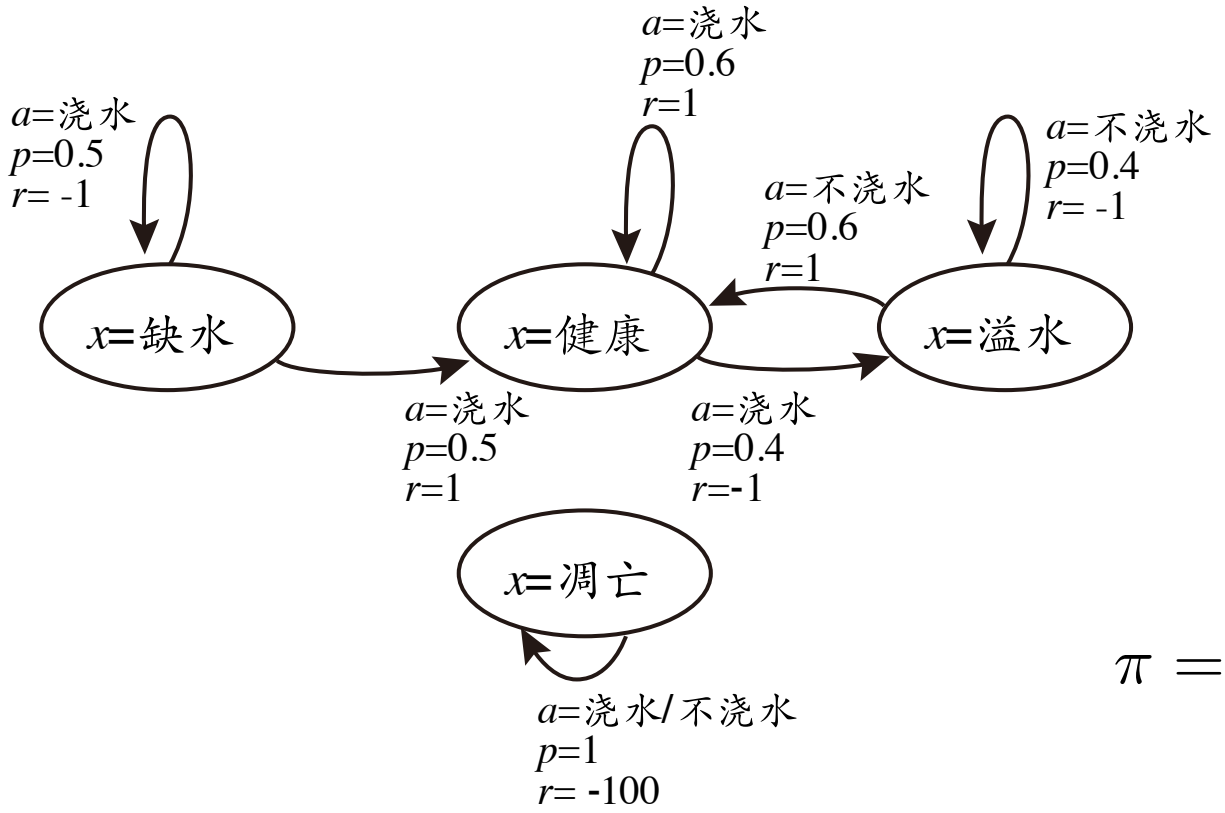
π =

表格策略表达

b	浇	0
	不	1
c	浇	0
	不	1
d	浇	0
	不	1
e	浇	0.5
	不	0.5

马尔可夫决策过程 MDP < *A*, *X*, *R*, *P* >

状态图



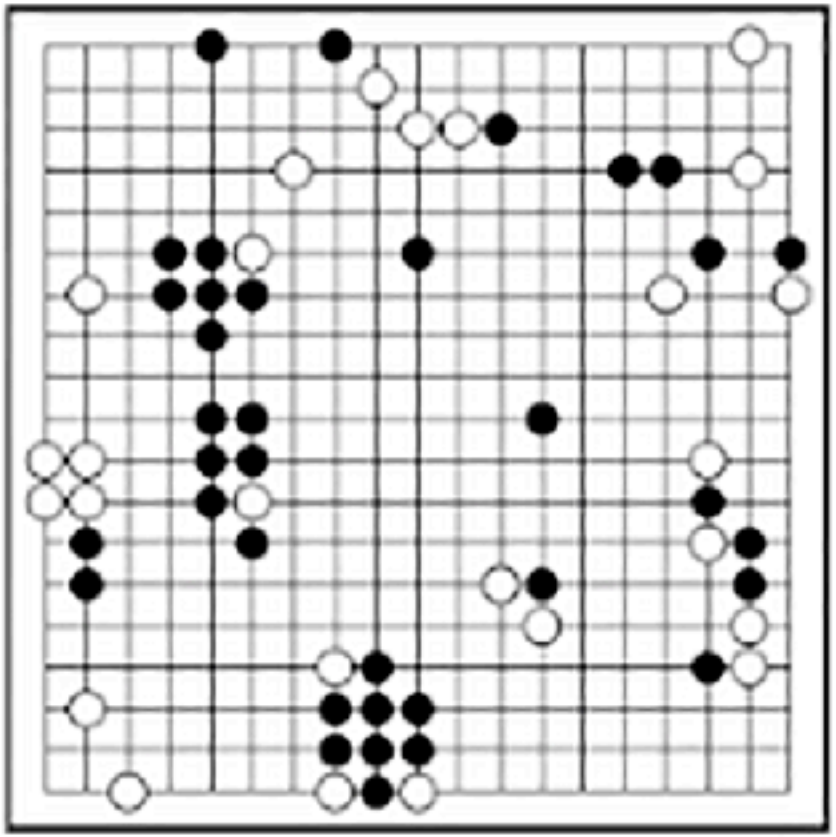
表格策略表达

π =

b	浇	1
	不	0
c	浇	0
	不	1
d	浇	1
	不	0
e	浇	0.5
	不	0.5

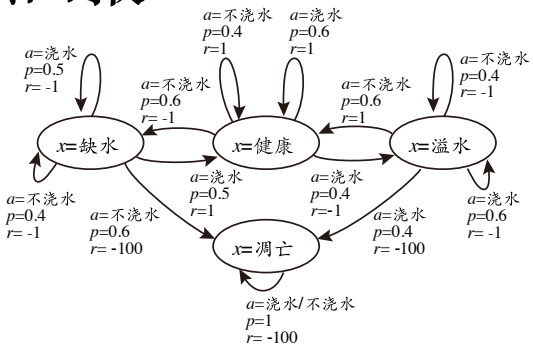
马尔可夫决策过程 MDP < A, X, R, P >

水平视角



策略回报

在MDP中，给定一个策略，计算策略的累积回报



MRP:

$$V(x) = \sum_{x' \in X} P(x'|x)\Big(R(x') + V(x')\Big)$$

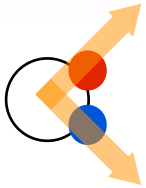
MDP:

$$V^{\pi}(x) = \sum_{a \in A} \pi(a|x) \sum_{x' \in X} P(x'|x,a)\Big(R(x,a,x') + V^{\pi}(x')\Big)$$

按照策略动作选择概率加和

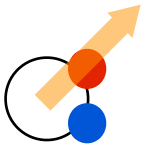
Q-function

状态值函数 state value function



$$V^{\pi}(x) = E[\sum_{t=1}^{T} r_t | x]$$

状态动作值函数 state-action value function



$$Q^{\pi}(x,a) = E[\sum_{t=1}^{T} r_t | x,a] = \sum_{x' \in X} P(x'|x,a)\Big(R(x,a,x') + V^{\pi}(x')\Big)$$

两者关系

$$V^{\pi}(x) = \sum_{a \in A} \pi(a|x) Q^{\pi}(x,a)$$

Q值函数更方便获取策略

最优性质

表格策略表达

基于表格策略表达

存在最优策略 π^*

$$\pi =$$

b	浇	1
	不	0
c	浇	0
	不	1
d	浇	1
	不	0
e	浇	0.5
	不	0.5

$$\forall \pi, \forall x, V^{\pi^*}(x) \geq V^{\pi}(x)$$

最优值函数

$$\forall x, V^*(x) = V^{\pi^*}(x)$$

$$\forall x, \forall a, Q^*(x, a) = Q^{\pi^*}(x, a)$$

Bellman optimality equations

$$V^*(x) = \max_{a \in A} Q^*(x, a)$$

根据V和Q的关系

$$Q^*(x, a) = \sum_{x' \in X} P(x'|x, a)(R(x, a, x') + \gamma V^*(x'))$$

于是有

$$Q^*(x, a) = \sum_{x' \in X} P(x'|x, a)(R(x, a, x') + \gamma \max_{a \in A} Q^*(x', a))$$

$$V^*(x) = \max_{a \in A} \sum_{x' \in X} P(x'|x, a)(R(x, a, x') + \gamma V^*(x'))$$

唯一不动点即为最优值

求解MDP最优策略

主要思想:

获取策略性能评估 policy evaluation

改进策略 policy improvement

policy evaluation: 迭代计算

$$V^{\pi}(x) = \sum_{a \in A} \pi(a|x) \sum_{x' \in X} P(x'|x,a)(R(x,a,x') + \gamma V^{\pi}(x'))$$

policy improvement: 根据 Bellman optimality equation

$$\pi(x) = \arg\max_{a \in A} Q^{\pi}(x,a)$$

求解MDP最优策略

policy improvement: 根据 Bellman optimality equation

$$\pi'(x) = \arg\max_{a \in A} Q^{\pi}(x, a)$$

let π' be derived from this update

$$\begin{aligned} V^{\pi}(x) &\leq Q^{\pi}(x, \pi'(x)) \\ &= \sum_{x'} P(x'|x, \pi'(x))(R(x, \pi'(x), x') + \gamma V^{\pi}(x')) \\ &\leq \sum_{x'} P(x'|x, \pi'(x))(R(x, \pi'(x), x') + \gamma Q^{\pi}(x', \pi'(x))) \\ &= \dots \\ &= V^{\pi'} \end{aligned}$$

so the policy is improved

求解MDP最优策略

策略迭代算法:

loop until converges

policy evaluation: calculate V

policy improvement: choose the action greedily

$$\pi_{t+1}(x) = \arg\max_a Q^{\pi_t}(x,a)$$

收敛: $V^{\pi_{t+1}}(x) = V^{\pi_t}(x)$

$$Q^{\pi_{t+1}}(x,a) = \sum_{x'} P(x'|x,a)\left(R(x,a,x') + \gamma \max_a Q^{\pi_t}(x',a)\right)$$

回忆最优 Q 值函数表达式

求解MDP最优策略

将策略评估和改进整合

Value iteration algorithm:

$V_0 = 0$

for t=0, 1, ...

for all x <- synchronous v.s. asynchronous

$$V_{t+1}(x) = \max_a \sum_{x'} P(x'|x,a)\big(R(x,a,x') + \gamma V_t(x)\big)$$

end for

break if $||V_{t+1} - V_t||_\infty$ is small enough

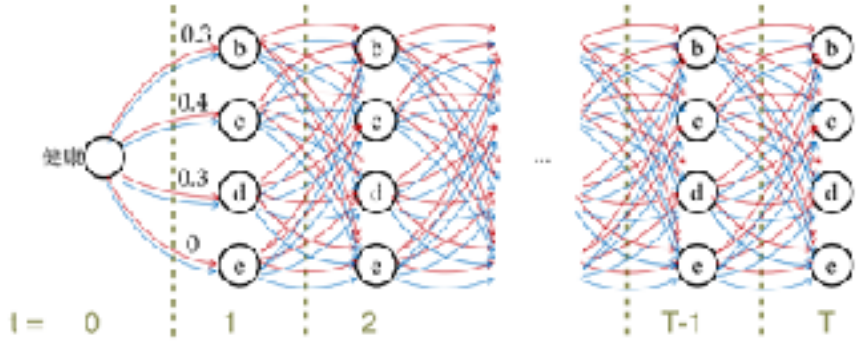
end for

回忆最优 V 值函数表达式

求解MDP最优策略

$$Q^{\pi_{t+1}}(x,a) = \sum_{s'} P(x'|x,a)\big(R(x,a,x') + \gamma \max_a Q^{\pi_t}(x',a)\big)$$

$$V_{t+1}(x) = \max_a \sum P(x'|x,a)\big(R(x,a,x') + \gamma V_t(x')\big)$$



R. E. Bellman
1920-1984

动态规划

复杂度

收敛需迭代 $\Theta(|X| \cdot |A|)$ 次，仅考虑确定性 MDP

[O. Madani. Polynomial Value Iteration Algorithms for Deterministic MDPs. UAI'02]

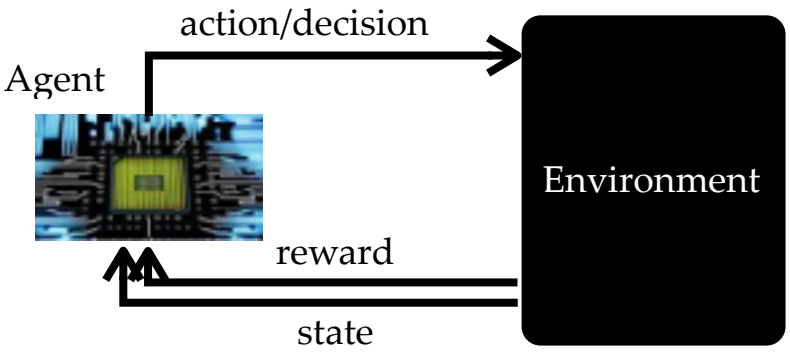
维度灾难：围棋棋盘19×19, $|S|=2.08\times10^{170}$

[https://github.com/tromp/golegal]

从MDP到强化学习

MDP 已知 <X,A,R,P> 所有信息

在强化学习中，R 和 P 往往未知



A: 还原 R 和 P,
然后求解MDP最优策略

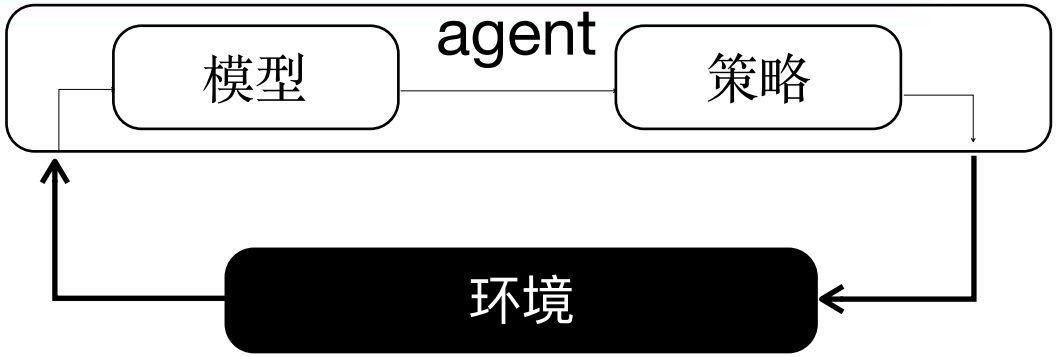
基于模型的RL

B: 不还原 R 和 P,
直接逼近最优策略

免模型

此处模型意为MDP

基于模型的 RL



基本思路

1. 探索环境
2. 从观测中学习环境
3. 通过VI/PI学习策略

关键问题:

如何有效探索和学习模型?

如何利用模型?

如何结合模型学习与策略学习?

...

还原MDP模型

随机游走，记录转移与回报信息

访问未充分探索的状态/动作

RMax algorithm:

[Bertsekas, Tsitsiklis. R-Max---A general polynomial time algorithm for near-optimal reinforcement learning. JMLR'02]

initialize $R(x)$=Rmax, P = self-trainsition
loop
    choose action a, observe state x' and reward r
    update transition count and reward count for x,a,x'
    if count of x,a >= m
        update reward and transition from estimations
    $x = x'$

样本复杂度：$\tilde{O}(|X|^2|A|V_{\max}^3/(\epsilon(1-\gamma))^3)$

[Strehl, et al. Reinforcement learning in finite MDPs: PAC analysis. JMLR'09]

免模型 RL

探索环境同时学习策略

Monte-Carlo 方法

时序差分方法

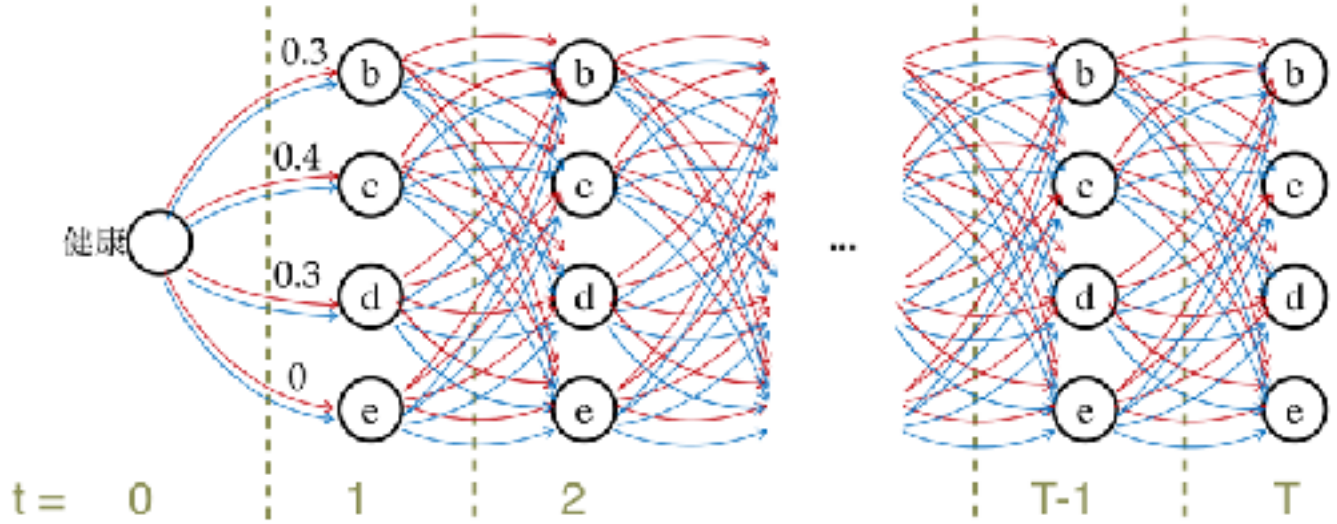
Monte Carlo RL - 评价

期望累积回报

$$Q^{\pi}(x,a) = E[\sum_{t=1}^{T} r_t | x, a]$$

Q, not V

采样逼近期望



采样 m 条轨迹,

$$Q^{\pi}(x,a) = \frac{1}{m}\sum_{i=1}^{m} R(\tau_i)$$

τ_i is sample by following π after *x,a*

Monte Carlo RL

$Q_0 = 0$

for i=0, 1, ..., m

generate trajectory $<x_0,\ a_0,\ r_1,\ x_1,\ ...,\ x_T>$

for t=0, 1, ..., T-1

R = sum of rewards from t to T

$Q(x_t, a_t) = (\mathrm{c}(x_t, a_t)\, Q(x_t, a_t) + \mathrm{R}) / (\mathrm{c}(x_t, a_t) + 1)$

$\mathrm{c}(x_t, a_t)$++

end for

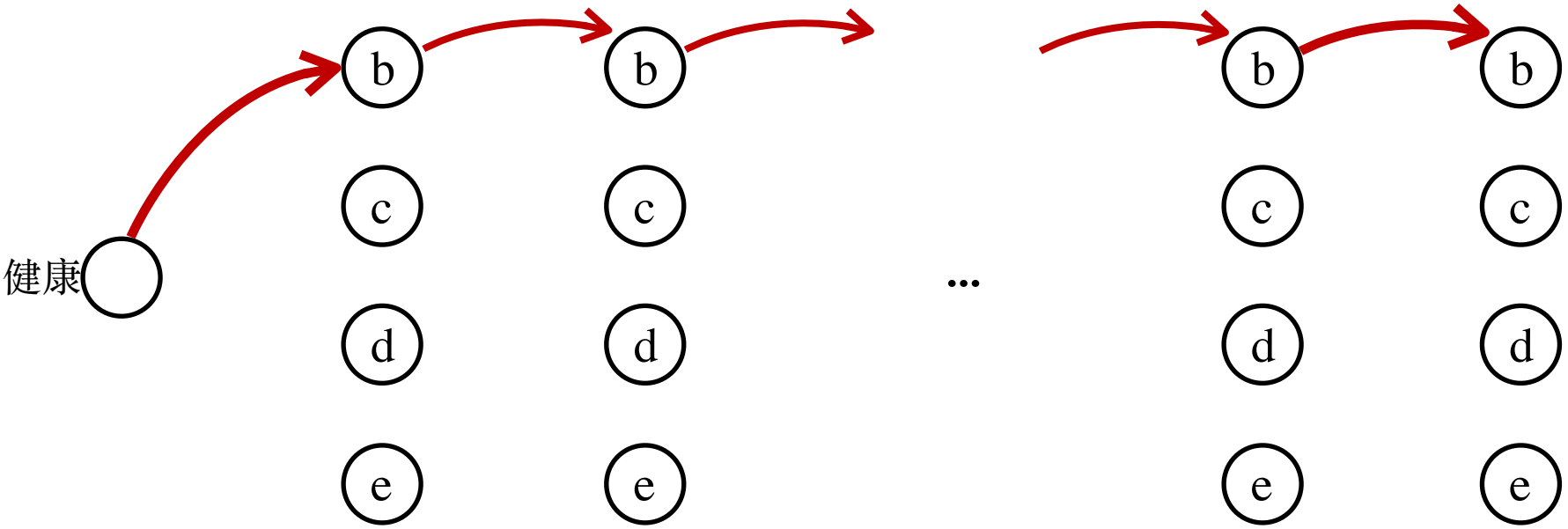
update policy $\pi(x) = \arg\max_a Q(x, a)$

end for

improvement ?

Monte Carlo RL

如果仅仅通过执行当前策略收集样本



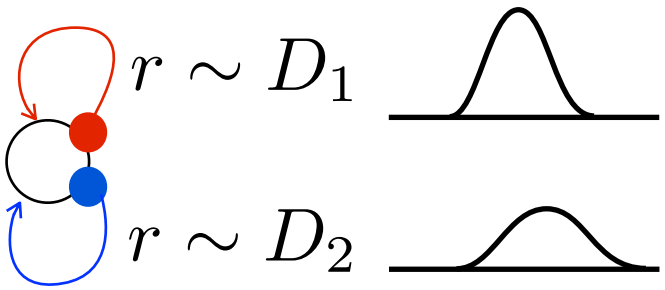
数据局限，无法知晓其他路径的好坏，无法提高策略

需要探索环境!

环境探索方法

考虑一个状态 MDP:

即“多臂老虎机模型”



（同样）最大化长期累积奖赏

- 只探索: 平均分配尝试次数

  浪费大量尝试

- 只利用: 尝试每个摇臂一次, 根据反馈结果，只选择“最好”的摇臂

  选择存在很大误差

平衡探索与利用

环境探索方法

ϵ-greedy:

以概率 1-ϵ 选取当下最好的决策
以概率 ϵ 随机选取决策

ϵ should decrease along time

softmax:

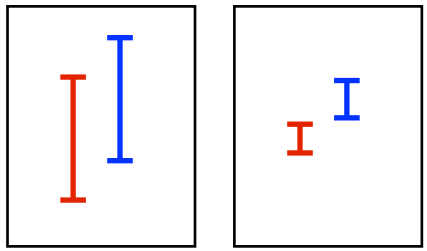
以以下概率选取决策 （风险：概率坍缩）

$$P(k) = e^{Q(k)/\theta} / \sum_{i=1}^{K} e^{Q(i)/\theta}$$

upper confidence bound (UCB):

以值 + 置信度选取决策

$$Q(k) + \sqrt{2 \ln n / n_k}$$



RL：动作层面探索

ϵ-greedy 策略:

给定策略 π

$$\pi_\epsilon(x) = \begin{cases} \pi(x), \text{with prob. } 1-\epsilon \\ \text{randomly chosen action, with prob. } \epsilon \end{cases}$$

每个状态被访问概率 > 0

也存在在其他层面探索的方法

Monte Carlo RL

$Q_0 = 0$

for i=0, 1, ..., m

generate trajectory $<x_0,\ a_0,\ r_1,\ x_1,\ ...,\ x_T>$ by π_ϵ

for t=0, 1, ..., T-1

R = sum of rewards from t to T

$Q(x_t, a_t) = (\mathrm{c}(x_t, a_t)\, Q(x_t, a_t) + \mathrm{R}) / (\mathrm{c}(x_t, a_t) + 1)$

$\mathrm{c}(x_t, a_t)$++

end for

update policy $\pi(x) = \arg\max_a Q(x, a)$

end for

Monte Carlo RL - 同/异策略（on/off-policy）

Monte Carlo RL 算法评估对象：π_ϵ ！ 同策略

如果需要求解 π？ 异策略

重要性采样：

$$E[f] = \int_x p(x) f(x) \mathrm{d}x = \int_x q(x) \frac{p(x)}{q(x)} f(x) \mathrm{d}x$$

sample from p | sample from q

$$\frac{1}{m} \sum_{i=1}^{m} f(x) \qquad \frac{1}{m} \sum_{i=1}^{m} \frac{p(x)}{q(x)} f(x)$$

Monte Carlo RL -- 异策略

$Q_0 = 0$

for i=0, 1, ..., m

generate trajectory $<s_0, a_0, r_1, s_1, ..., s_T>$ by π_ϵ

for t=0, 1, ..., T-1

R = sum of rewards from t to $T \times \prod_{i=t+1}^{T-1} \frac{\pi(x_i, a_i)}{p_i}$

$Q(s_t, a_t) = (\mathrm{c}(s_t, a_t) Q(s_t, a_t) + \mathrm{R}) / (\mathrm{c}(s_t, a_t) + 1)$

$\mathrm{c}(s_t, a_t)$++

end for

update policy $\pi(s) = \arg\max_a Q(s, a)$

end for

$$p_i = \begin{cases} 1 - \epsilon + \epsilon/|A|, a_i = \pi(s_i), \\ \epsilon/|A|, a_i \neq \pi(s_i) \end{cases}$$

Monte Carlo RL

小结

Monte Carlo 评估方法:
均值替代期望

动作层面探索

同策略/异策略：重要性采样

Monte Carlo RL:
评估 + 动作层面探索 + 策略改进（同/异策略）

增量式求和

$$Q(x_t,a_t) = (\mathrm{c}(x_t,a_t)\,Q(x_t,a_t)+\mathrm{R})/(\mathrm{c}(x_t,a_t)+1)$$

$$\mu_t = \frac{1}{t}\sum_{i=1}^{t} x_i = \frac{1}{t}(x_t + \sum_{i=1}^{t-1} x_i) = \frac{1}{t}(x_t + (t-1)\mu_{t-1})$$

$$= \mu_{t-1} + \frac{1}{t}(x_t - \mu_{t-1})$$

一般的，有 $\mu_t = \mu_{t-1} + \alpha(x_t - \mu_{t-1})$

Monte-Carlo 更新:

$$Q(x_t,a_t) \Leftarrow Q(x_t,a_t) + \alpha(\underbrace{R - Q(x_t,a_t)}_{\text{MC 误差}})$$

时序差分学习 - 评估

在线评估策略

时序差分(TD)评估

Monte-Carlo 更新:

$$Q(x_t, a_t) \Leftarrow Q(x_t, a_t) + \alpha(\underbrace{R - Q(x_t, a_t)}_{\text{MC 误差}})$$

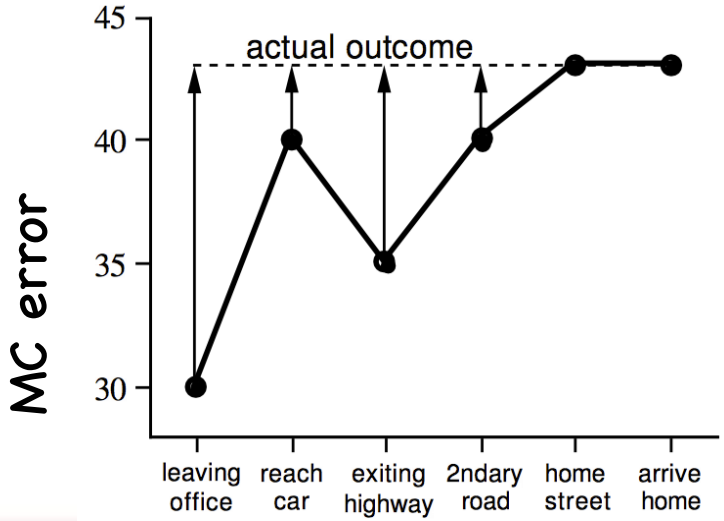
TD 更新:

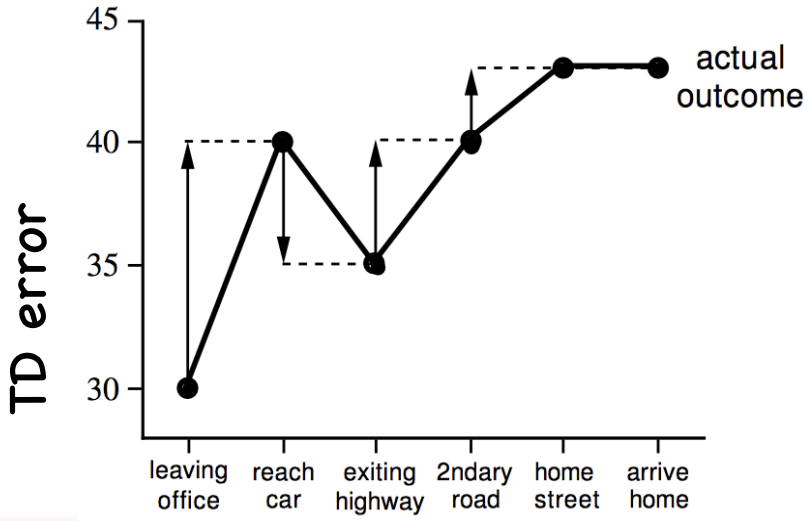
$$Q(x_t, a_t)$$

$$\Leftarrow Q(x_t, a_t) + \alpha(\underbrace{r_{t+1} + \gamma Q(x_{t+1}, a_{t+1}) - Q(x_t, a_t)}_{\text{TD 误差}})$$

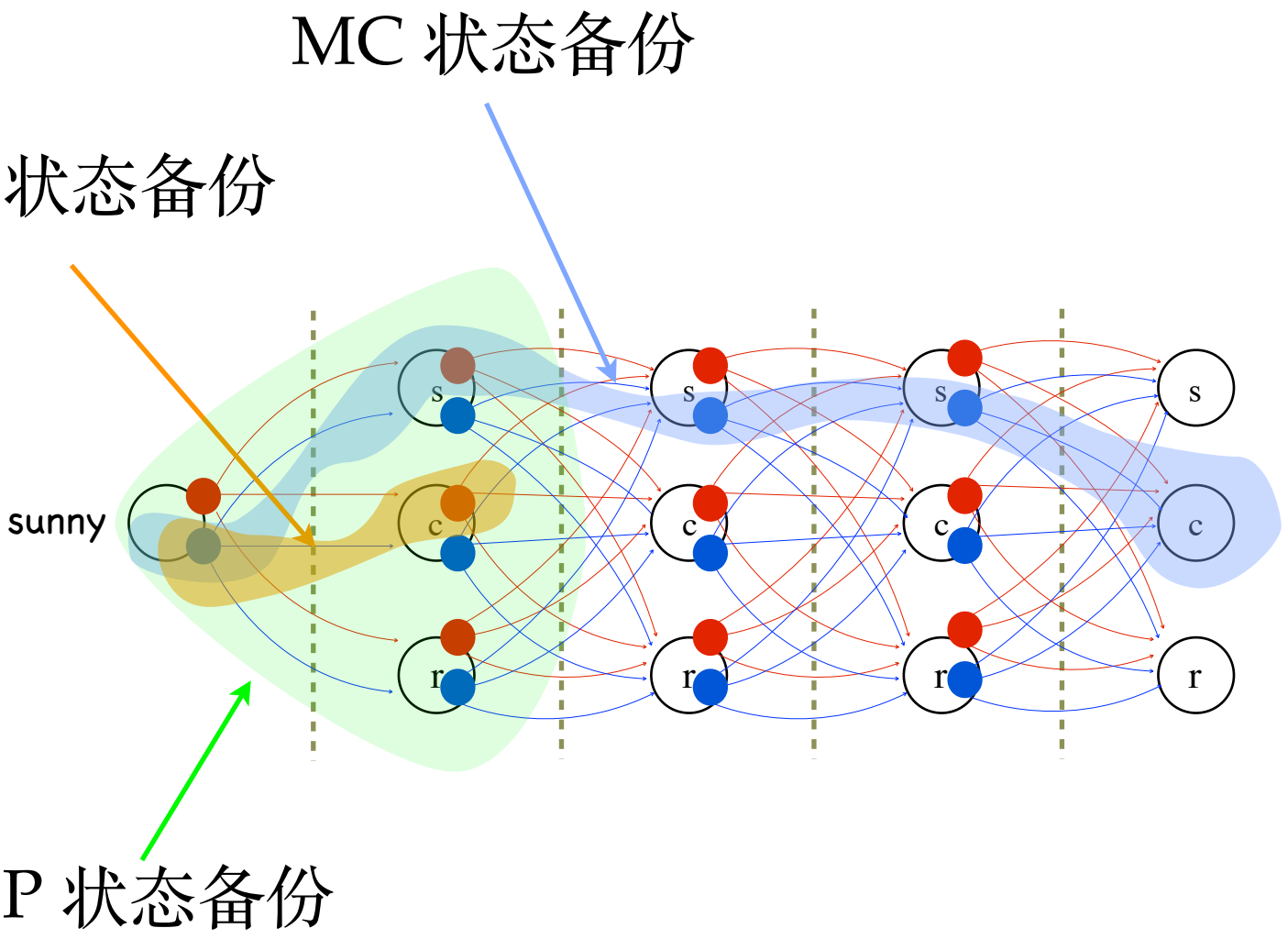
时序差分学习 - 直观解释

state	elapsed time	predicted remaining time	predicted total time
leaving office	0	30	30
reach car, raining	5	35	40
exit highway	20	15	35
behind truck	30	10	40
home street	40	3	43
arrive home	43	0	43





时序差分学习 - 状态备份



SARSA

同策略 TD 控制

$Q_0 = 0$, initial state
for i=0, 1, ...
$a = \pi_\epsilon(x)$
x', r = do action a
a' $= \pi_\epsilon(x')$
$Q(x,a)$+= $\alpha(r + \gamma Q(x', a') - Q(x,a))$
$\pi(x) = \arg\max_a Q(x,a)$
$x = x$'
end for

Q-learning

异策略 TD 控制

$Q_0 = 0$, initial state

for i=0, 1, ...

$a = \pi_\epsilon(x)$

x', r = do action a

a' = $\pi(x')$

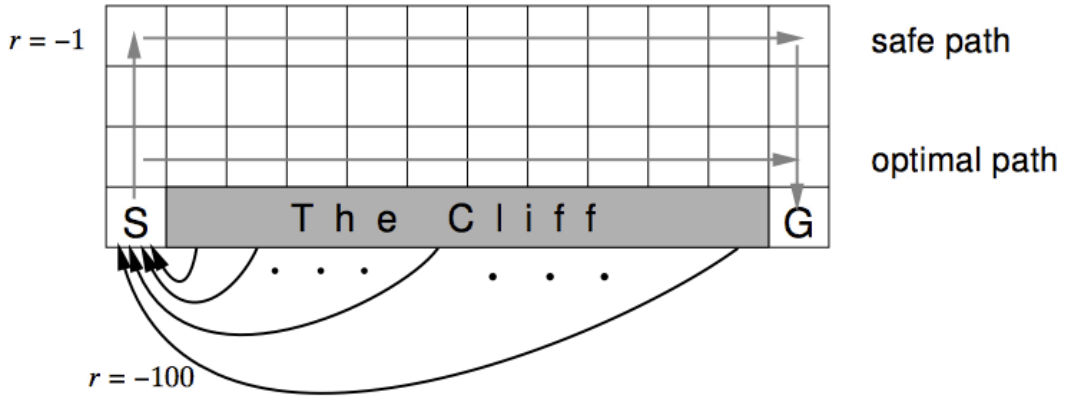
$Q(x,a)$+= $\alpha(r + \gamma Q(x', a') - Q(x,a))$

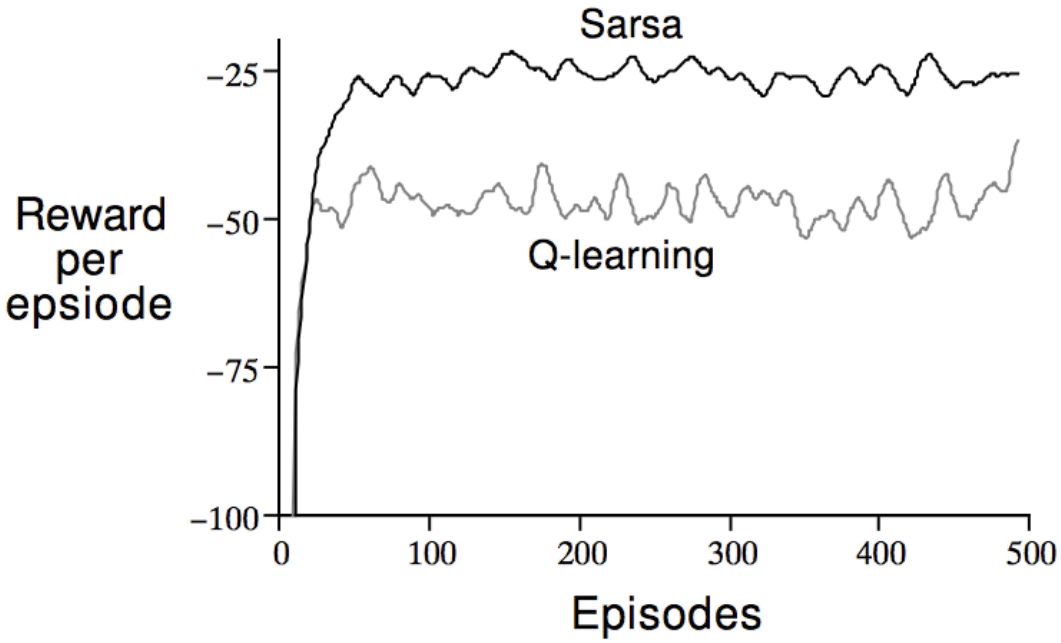
$\pi(x) = \arg\max_a Q(x,a)$

$x = x$'

end for

SARSA v.s. Q-learning

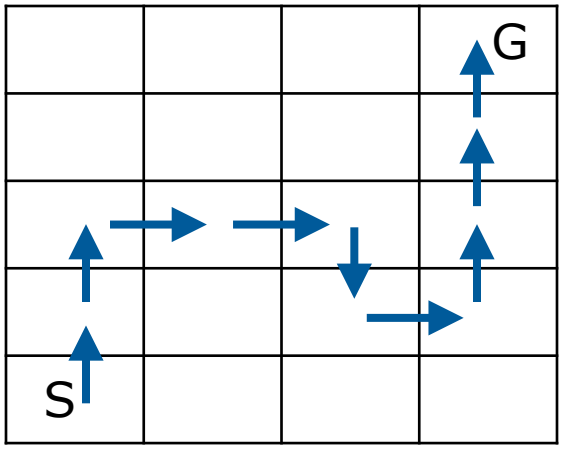




单步TD控制的局限

值函数更新太慢

奖赏 = 到达G:1; 其他: 0



初始V

0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

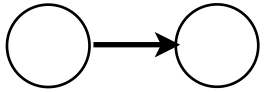
走到一次G后

0	0	0	0
0	0	0	0.99
0	0	0	0
0	0	0	0
0	0	0	0

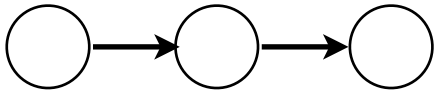
λ-return

介于 TD 和 MC 评估: n-步评估

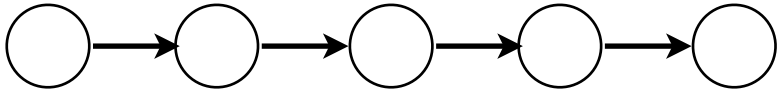
n-step return

TD(1-step) 

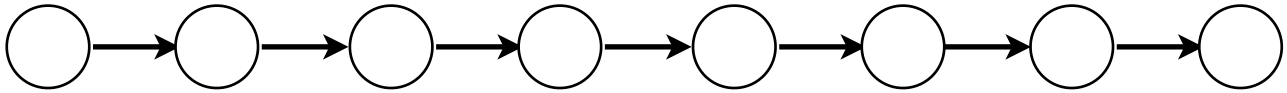
$$R^{(1)} = r_{t+1} + \gamma Q(x_{t+1}, a_{t+1})$$

TD(2-step) 

$$R^{(2)} = r_{t+1} + \gamma r_{t+2} + \gamma^2 Q(x_{t+2}, a_{t+2})$$

TD(n-step) 

$$R^{(n)} = \sum_{i=1}^{n} \gamma^{i-1} r_{t+i} + \gamma^n Q(x_{t+n}, a_{t+n})$$

MC 

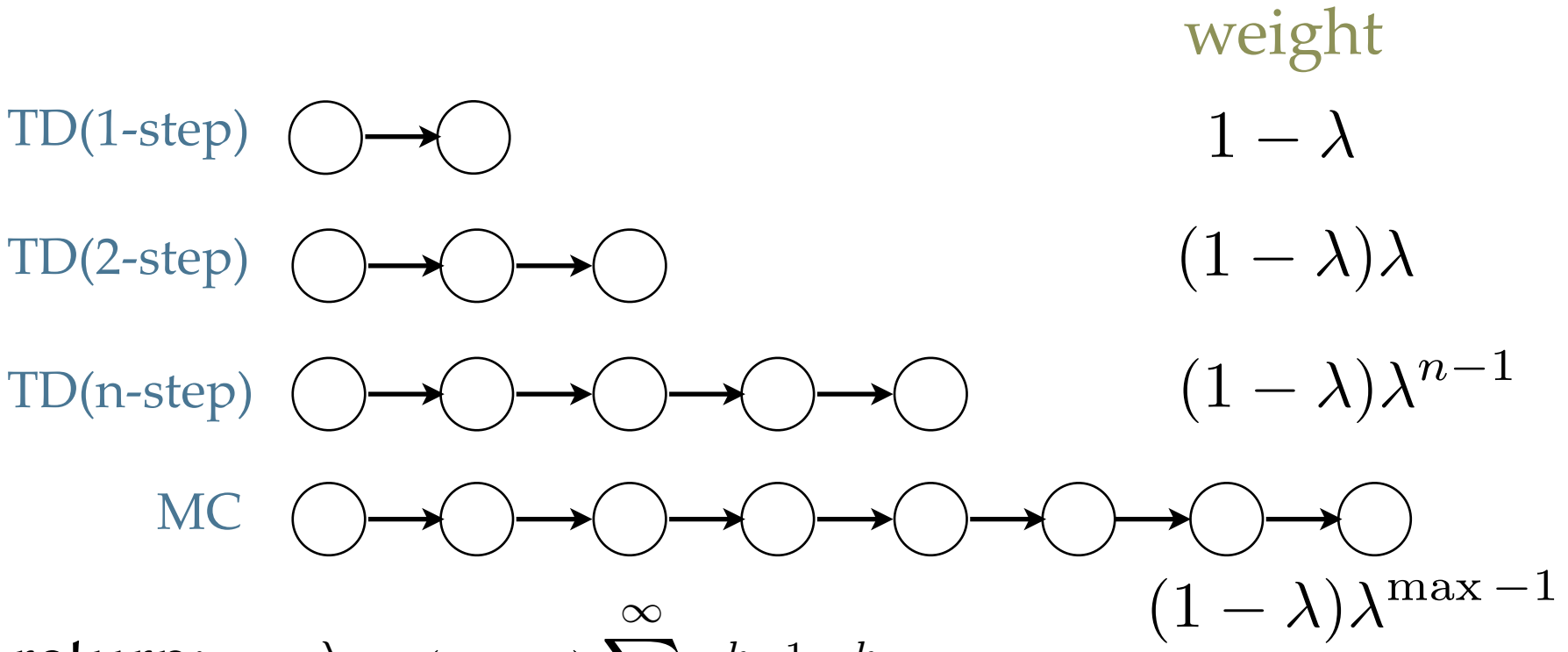
k-step TD:

$$Q(x_t, a_t) = Q(x_t, a_t) + \alpha(R^{(k)} - Q(x_t, a_t))$$

$$R^{(\max)} = \sum_{i=1}^{T} \gamma^{i-1} r_{t+i}$$

λ-return

加权平均 k-step returns，参数 λ



λ-return: $R^\lambda = (1-\lambda)\sum_{k=1}^{\infty}\lambda^{k-1}R^k$

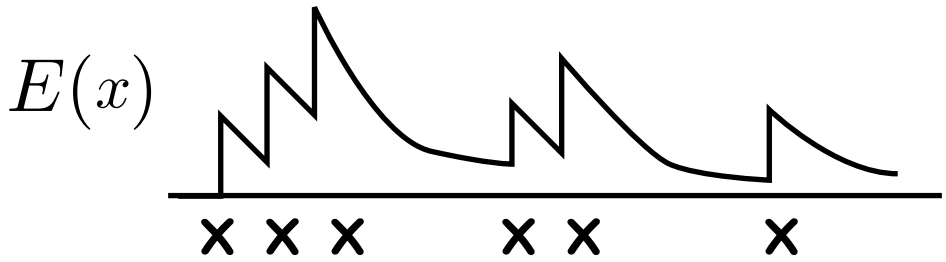
TD(λ): $Q(x_t,a_t) = Q(x_t,a_t) + \alpha(R^\lambda - Q(x_t,a_t))$

实现: 资格迹（eligibility traces）

维护状态上次访问时间记忆 **E(s)**

$$E_0(x,a) = 0$$

$$E_t(x,a) = \gamma\lambda E_{t-1}(x,a) + I(x_t = x, a_t = a)$$



TD(λ)

TD 误差:

$$\delta_t = r_{t+1} + \gamma Q(x_{t+1}, a_{t+1}) - Q(x_t, a_t)$$

对所有状态-动作，更新Q:

$$Q(x,a) \Leftarrow Q(x,a) + \alpha\delta_t E_t(x,a)$$

SARSA(λ)

$Q_0 = 0$, initial state

for i=0, 1, ...

x', r = do action from policy π_ϵ

a' $= \pi_\epsilon(x')$

$\delta = r + \gamma Q(x', a') - Q(x, a)$

$E(x, a) + +$

for all $x,\ a$

$Q(x, a) = Q(x, a) + \alpha\delta E_t(x, a)$

$E(x, a) = \gamma E(x, a)$

end for

$x = x$', $a =$ a', $\pi(x) = \arg\max_a Q(x, a)$

end for

从“表格表示”到“值函数近似”

表格表达

$\pi =$

s	0	0.3
	1	0.7
c	0	0.6
	1	0.4
r	0	0.1
	1	0.9

能表达所有可能的策略

线性函数近似

$$\hat{V}(x) = w^\top \phi(x)$$
$$\hat{Q}(x, a) = w^\top \phi(x, a)$$
$$\hat{Q}(x, a_i) = w_i^\top \phi(x)$$

ϕ 是特征映射

w 参数向量

无法表达所有可能的策略！

值函数近似

近似表达趋于真实值

最小二乘回归

$$J(w) = E_{x\sim\pi}[\left(Q^{\pi}(x,a) - \hat{Q}(x,a)\right)^2]$$

在线环境：随机梯度

$$\Delta w_t = \theta(Q^{\pi}(x_t,a_t) - \hat{Q}(x_t,a_t))\nabla_w \hat{Q}(x_t,a_t)$$

替代

回顾误差:

MC 误差: $Q(x_t,a_t) += \alpha(R - Q(x_t,a_t))$

TD 误差: $Q(x_t,a_t) += \alpha(r_{t+1} + \gamma Q(x_{t+1},a_{t+1}) - Q(x_t,a_t))$

目标值　　预估值

值函数近似

MC 误差:

$$\Delta w_t = \theta(R - \hat{Q}(x_t, a_t))\nabla_w \hat{Q}(x_t, a_t)$$

TD 误差:

$$\Delta w_t = \theta(r_{t+1} + \gamma \hat{Q}(x_{t+1}, a_{t+1}) - \hat{Q}(x_t, a_t))\nabla_w \hat{Q}(x_t, a_t)$$

资格迹

$$E_t = \gamma\lambda E_{t-1} + \nabla_w \hat{Q}(x_t, a_t)$$

值函数近似 Q-learning

$w = 0$, initial state

for i=0, 1, ...

$a = \pi_\epsilon(x)$

x', $r =$ do action a

$a' = \pi(x')$

$w+ = \theta(r + \gamma\hat{Q}(x, a) - \hat{Q}(x, a))\nabla_w \hat{Q}(x_t, a_t)$

$\pi(x) = \arg\max_a \hat{Q}(x, a)$

$x = x$'

end for

近似模型

线性模型

$$\hat{Q}(x,a) = w^\top \phi(x,a)$$

$$\nabla_w \hat{Q}(x,a) = \phi(x,a)$$

原始特征

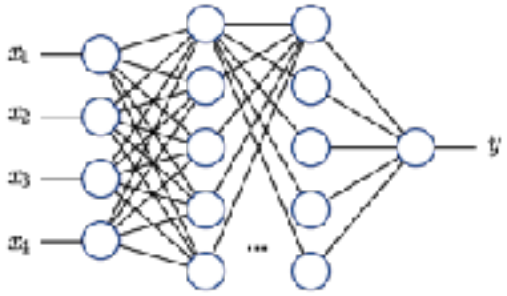
离散化：离散网格作为特征

核化：$\hat{Q}(x,a) = \sum_{i=1}^{m} w_i K((x,a),(x_i,a_i))$

(x_i, a_i) 初始随机采样

近似模型

非线性模型 $\hat{Q}(x,a) = f(x,a)$



神经网络：可微模型

回顾参数更新规则:

$$\Delta w_t = \theta(r_{t+1} + \gamma \hat{Q}(x_{t+1}, a_{t+1}) - \hat{Q}(x_t, a_t))\nabla_w \hat{Q}(x_t, a_t)$$

梯度传导给神经网络

近似模型 — 批量方法

在单个样本上求梯度导致方差极大

批量评估:

收集历史数据

$$D = \{(x_1, V_1^\pi), (x_2, V_2^\pi), \ldots, (x_m, V_m^\pi)\}$$

求解回归模型

$$J(w) = E_D[\left(V^\pi - \hat{V}(x)\right)^2]$$

线性函数: 闭式解

神经网络: batch update

LSMC, LSTD, LSTD(λ)

近似模型 — 批量方法

批量策略迭代

$Q_0 = 0$, initial state

for i=0, 1, ...

collect data D

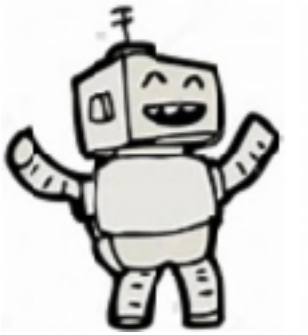
$$w = \arg\min_w \sum_{(s,a)\in D} (r + \gamma \hat{Q}(x, \pi(x)) - \hat{Q}(x, a))\phi(x, a)$$

$$\forall x, \pi(x) = \arg\max_a Q(x, a)$$

end for

深度强化学习

function approximation by
deep neural networks

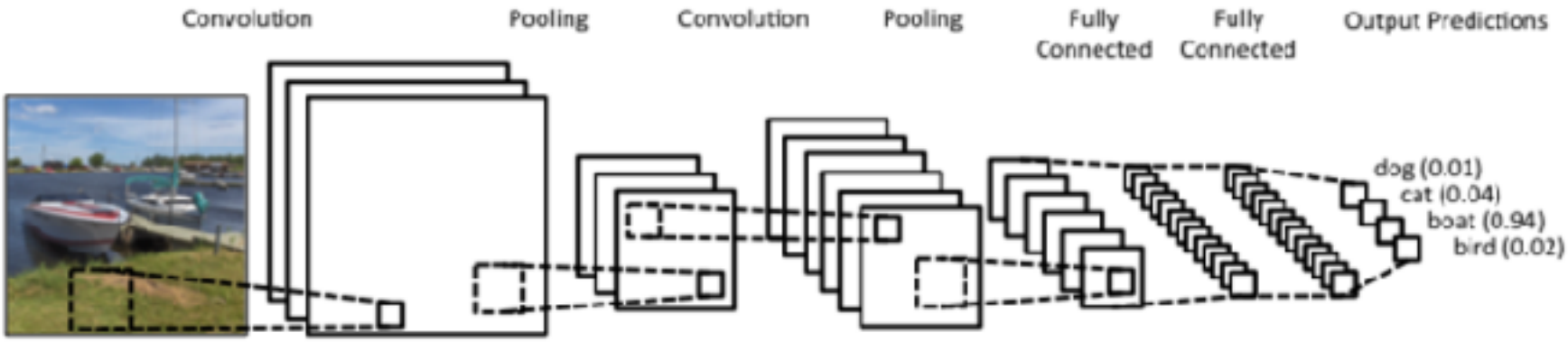


Convolutional neural networks

a powerful neural network architecture for image analysis

differentiable

require a lot of samples to train



Deep Q-Network

DQN

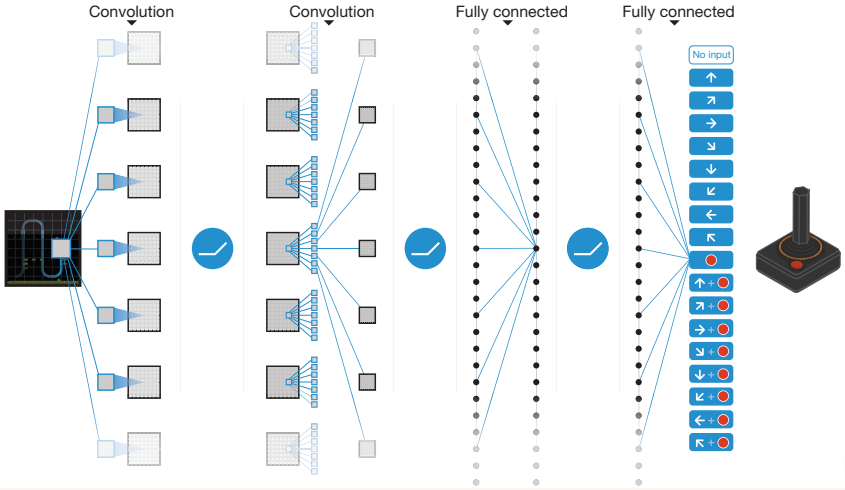
- using ϵ-greedy policy
- store 1million recent history (x,a,r,x′) in replay memory D
- sample a mini-batch (32) from D
- calculate Q-learning target $\tilde{Q}$
- update CNN by minimizing the Bellman error (delayed update)

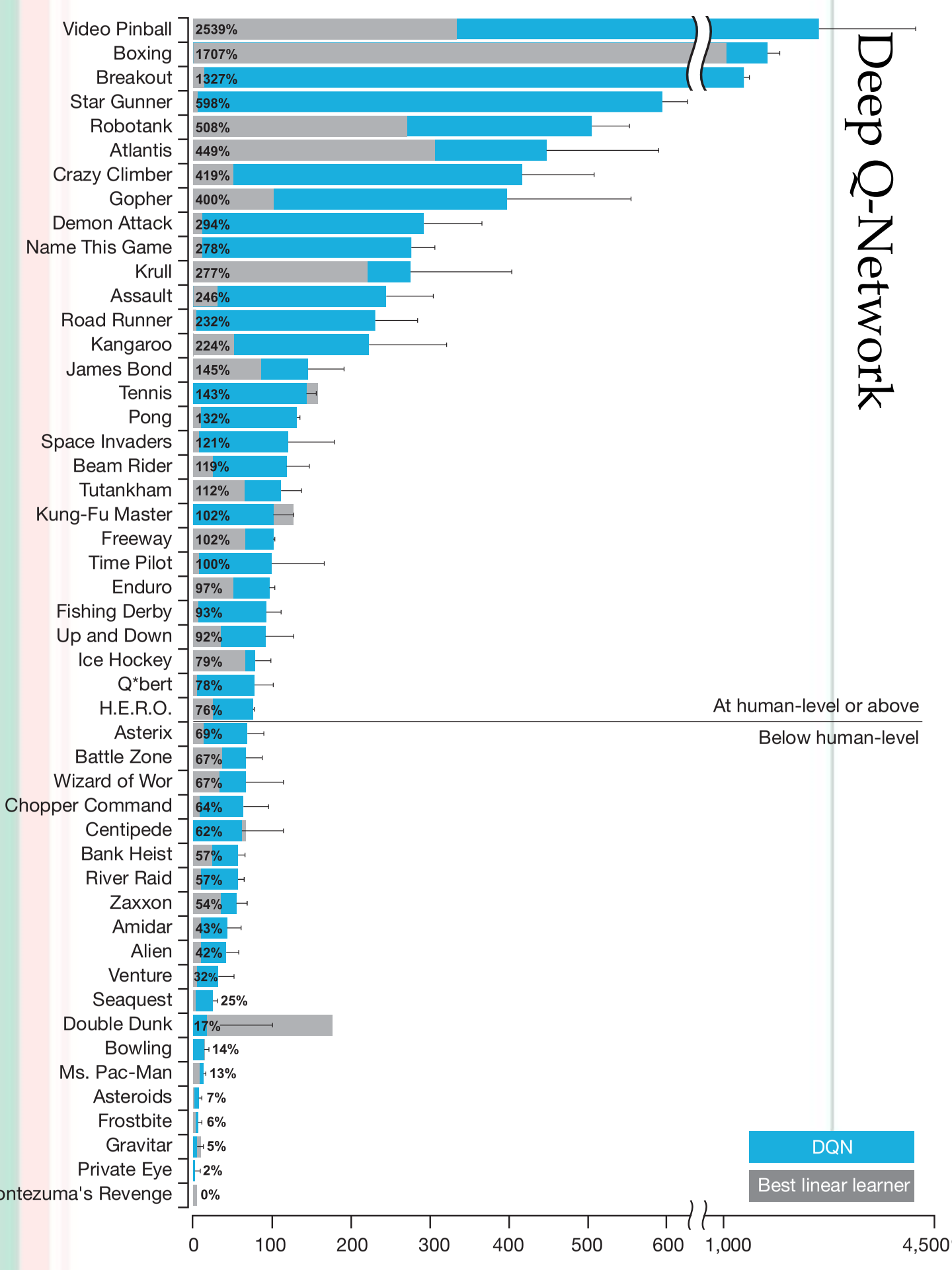
$$\sum (r + \gamma \max_{a'} \tilde{Q}(x', a') - Q_w(x, a))^2$$

DQN on Atari

learn to play from pixels

4 frames




Deep Q-Network
Video Pinball 2539%
Boxing 1707%
Breakout 1327%
Star Gunner 598%
Robotank 508%
Atlantis 449%
Crazy Climber 419%
Gopher 400%
Demon Attack 294%
Name This Game 278%
Krull 277%
Assault 246%
Road Runner 232%
Kangaroo 224%
James Bond 145%
Tennis 143%
Pong 132%
Space Invaders 121%
Beam Rider 119%
Tutankham 112%
Kung-Fu Master 102%
Freeway 102%
Time Pilot 100%
Enduro 97%
Fishing Derby 93%
Up and Down 92%
Ice Hockey 79%
Q*bert 78%
H.E.R.O. 76%
At human-level or above
Asterix 69%
Battle Zone 67%
Wizard of Wor 67%
Chopper Command 64%
Centipede 62%
Bank Heist 57%
River Raid 57%
Zaxxon 54%
Amidar 43%
Alien 42%
Venture 32%
Seaquest 25%
Double Dunk 17%
Bowling 14%
Ms. Pac-Man 13%
Asteroids 7%
Frostbite 6%
Gravitar 5%
Private Eye 2%
ontezuma's Revenge 0%
Below human-level
DQN
Best linear learner
0
100
200
300
400
500
600
1,000
4,500

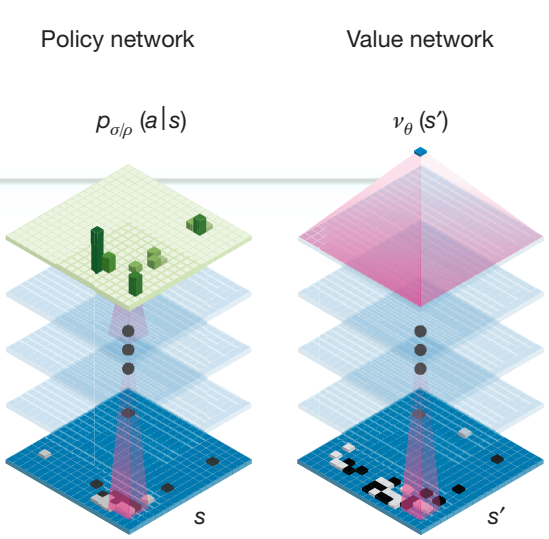
Deep Q-Network

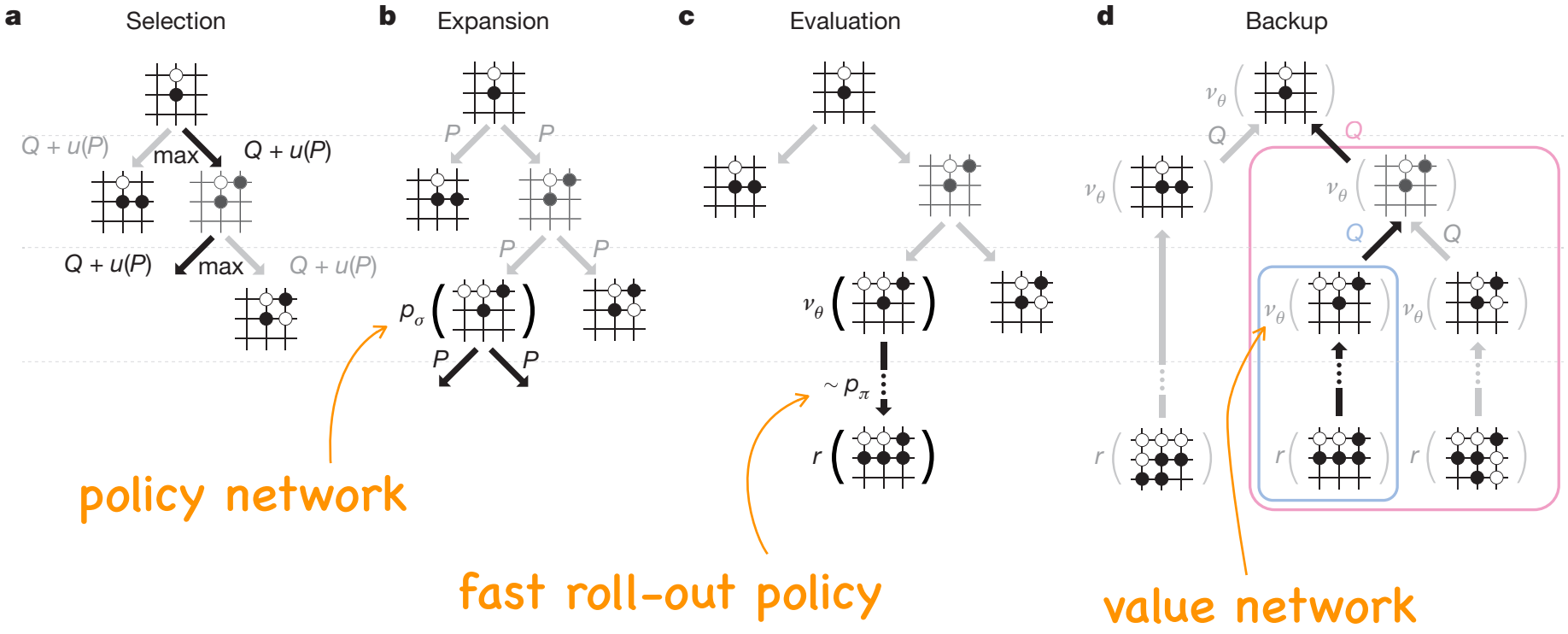
分解调查

Game	With replay, with target Q	With replay, without target Q	Without replay, with target Q	Without replay, without target Q
Breakout	316.8	240.7	10.2	3.2
Enduro	1006.3	831.4	141.9	29.1
River Raid	7446.6	4102.8	2867.7	1453.0
Seaquest	2894.4	822.6	1003.0	275.8
Space Invaders	1088.9	826.3	373.2	302.0

AlphaGo

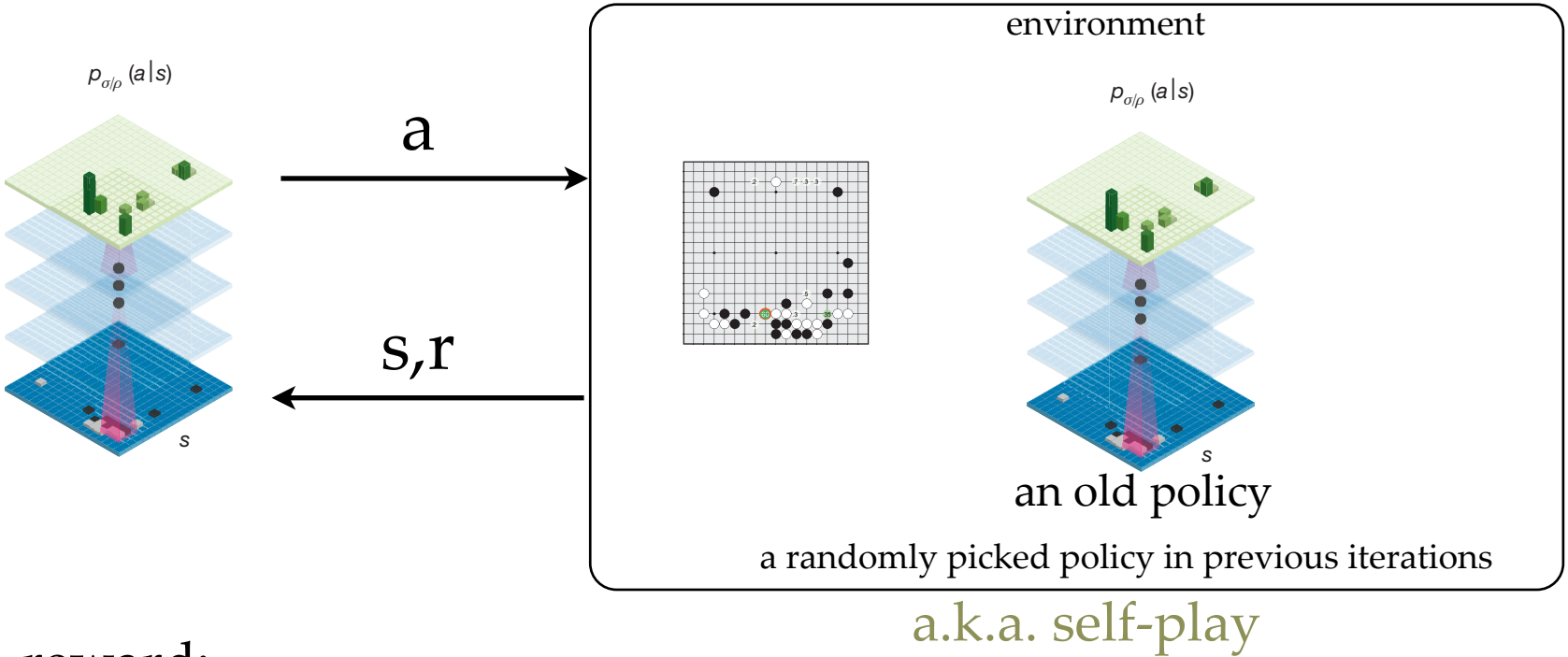
结合模特卡洛搜索树、
监督学习、强化学习





AlphaGo

策略网络（即策略）学习：自博弈



a.k.a. self-play

reward:
+1 -- win at terminate state
-1 -- loss at terminate state

IntelAct

Visual Doom AI Competition, September 2016
First place, “Full Deathmatch”

Alexey Dosovitskiy and Vladlen Koltun

Intel Labs

模仿学习：从教师数据中学习

强化学习的困难：反馈滞后、搜索空间巨大

模仿学习：引入教师

演示数据：

$(x_0, a_0, x_1, a_1, \ldots, x_n, a_n)$

行为克隆

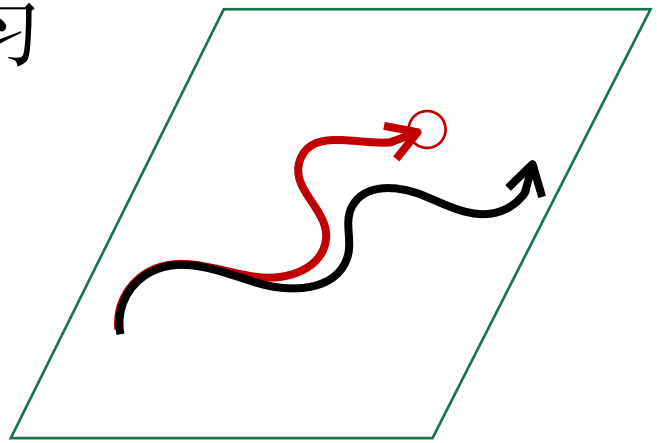
演示数据：

$$(x_0, a_0, x_1, a_1, \ldots, x_n, a_n)$$

$$(x_0, a_0), (x_1, a_1), \ldots, (x_n, a_n)$$

拆分为标记数据，使用监督学习

- 累积误差
- 策略初始化

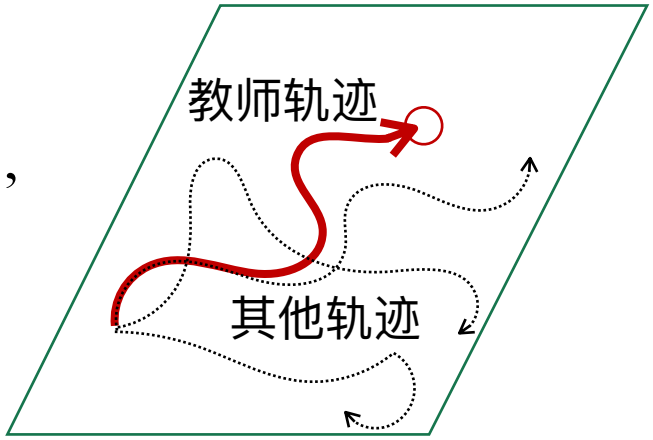


逆强化学习

学习教师意图（奖赏函数）

教师演示轨迹应当来自最优策略，对应最大回报

$$\forall \pi \quad R^{\pi^*} \geq R^{\pi}$$



考虑线性奖赏函数

$$R(x) = w^\top x \qquad R(x,a) = w^\top (x,a)$$

轨迹累积奖赏：$w^\top x_1 + w^\top x_2 + \ldots + w^\top x_T$

$$= w^\top \sum_{t=1}^{T} x_t = w^\top \bar{x}$$

逆强化学习目标：$\forall \bar{x} \quad w^\top (\bar{x}^* - \bar{x}) \geq 0$

逆强化学习

基于线性奖赏函数的逆强化学习

输入: 环境 E;
状态空间 X;
动作空间 A;
范例轨迹数据集 $D = \{\tau_1, \tau_2, \ldots, \tau_m\}$.

过程:

1: $\bar{\boldsymbol{x}}^* =$ 从范例轨迹中算出状态加权和的均值向量;
2: $\pi =$ 随机策略;
3: **for** $t = 1, 2, \ldots$ **do**
4: $\quad \bar{\boldsymbol{x}}_t^{\pi} =$ 从 π 的采样轨迹算出状态加权和的均值向量;
5: $\quad$ 求解 $\boldsymbol{w}^* = \arg\max_{\boldsymbol{w}} \min_{i=1}^{t} \boldsymbol{w}^{\mathrm{T}}(\bar{\boldsymbol{x}}^* - \bar{\boldsymbol{x}}_i^{\pi}) \quad \text{s.t.} \quad \|\boldsymbol{w}\| \leq 1$;
6: $\quad \pi =$ 在环境 $\langle X, A, R(\boldsymbol{x}) = {\boldsymbol{w}^*}^{\mathrm{T}} \boldsymbol{x} \rangle$ 中求解最优策略;
7: **end for**

输出: 奖赏函数 $R(\boldsymbol{x}) = {\boldsymbol{w}^*}^{\mathrm{T}} \boldsymbol{x}$ 与策略 π

前往……