南京大学
人工智能学院
School of Artificial Intelligence, Nanjing University

第五部分：强化学习系统

章宗长

2021年5月19日

强化学习

- 强化学习
  - （状态转移和奖赏）模型未知
  - 从经验中学习如何行动
    - 通过观察行动的结果，选择最大化长期累积奖赏的行动
- 三方面的挑战
  - 探索与利用（exploration and exploitation）
    - 在探索环境和利用从经验中获得的知识之间保持平衡
  - 信度分配（credit assignment）
    - 奖赏具有延迟性，即Agent在做出重要决策的一段时间后才获得奖赏，需要把奖赏的信度赋给早些时候的决策
  - 泛化（generalization）
    - 从有限的经验中获得可泛化的策略

内容安排

- **探索与利用**
- 基于模型的方法
- 免模型的方法
- 泛化
- 策略梯度

探索与利用

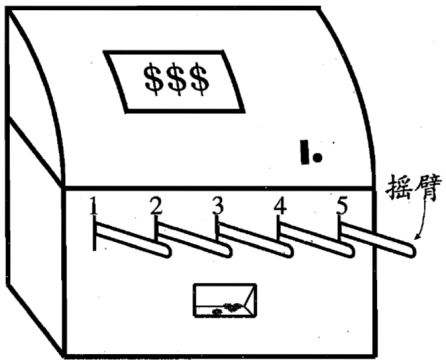
- 多摇臂赌博机问题、贝叶斯模型估计
- 探索策略
- 最优探索策略

n-摇臂赌博机

- 有n个摇臂，赌徒在投入一个硬币后可选择拉下其中一个摇臂
- 每个摇臂以一定的概率吐出硬币，但赌徒并不知道这个概率
- 总共能拉h次摇臂
- 目标：通过一定的策略最大化自己的奖赏，即获得最多的硬币

有限步数的MDP：1个状态，n个行动，步数为h，奖赏函数$R(s,a)$未知

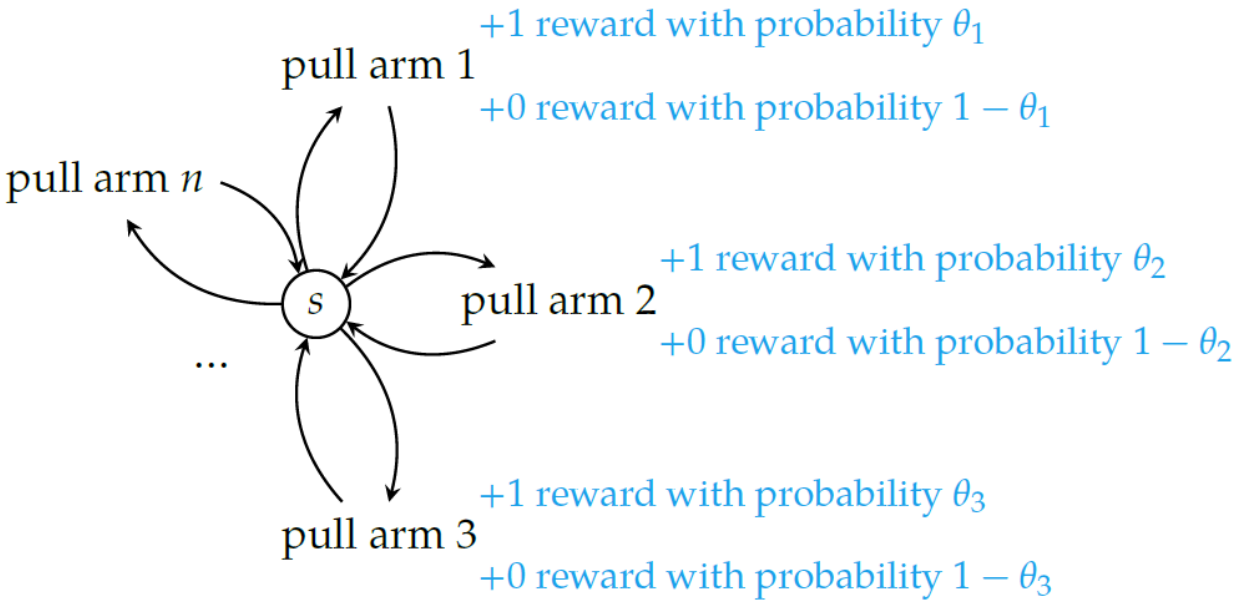
探索：估计摇臂的优劣
利用：选择当前最好的摇臂



应用：临床试验的分配和自适应网络路由等

简化的n-摇臂赌博机

- 有n个摇臂，赌徒总共能拉h次摇臂
- 摇臂i以θ_i的概率输出获胜（奖赏为1），以$1-\theta_i$的概率输出失败（奖赏为0），但赌徒并不知道θ_i
- 目标：通过一定的策略获得最多的奖赏



贝叶斯模型估计

- 使用均匀分布Beta(1,1)作为先验分布
- 对摇臂i，记录获胜的次数w_i和失败的次数ℓ_i
  - 使用贝塔分布表示摇臂i获胜概率θ_i的后验，则θ_i的后验为

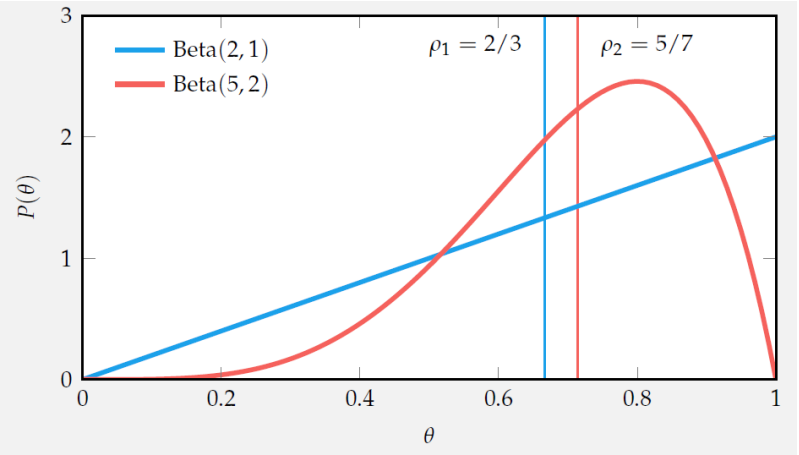
$$\text{Beta}(w_i + 1, \ell_i + 1)$$

- 计算获胜的后验概率：

$$\rho_i = P(\text{win}_i \mid w_i, \ell_i) = \int_0^1 \underbrace{\theta_i}_{P(\text{win}_i \mid \theta_i, w_i, l_i)} \times \underbrace{\text{Beta}(\theta_i \mid w_i + 1, \ell_i + 1)}_{P(\theta_i \mid w_i, l_i)} d\theta_i = \frac{w_i + 1}{w_i + \ell_i + 2}$$

极大似然估计 vs. 贝叶斯模型估计

- 假设有一个2摇臂赌博机，已经拉摇臂6次
  - 摇臂1获胜1次，失败0次
  - 摇臂2获胜4次，失败1次
  - 均匀先验



还可以拉摇臂1次，问：下一次选择拉哪个摇臂？

极大似然估计

- θ_1的极大似然估计为1
- θ_2的极大似然估计为$\frac{4}{5}$
- 下一次会拉摇臂1

贝叶斯模型估计

- θ_1的后验分布为Beta(2,1)
  - $\rho_1 = \frac{2}{3} \approx 0.67$
- θ_2的后验分布为Beta(5,2)
  - $\rho_2 = \frac{5}{7} \approx 0.71$
- 下一次会拉摇臂2

探索与利用

- 多摇臂赌博机问题、贝叶斯模型估计
- **探索策略**
- 最优探索策略

探索策略

- 无向探索策略
  - 不使用之前行动结果的信息来指导非贪心行动的选择
  - 如：ε −贪心、乐观初始化
- 有向探索策略
  - 使用之前行动结果的信息来指导非贪心行动的选择
  - 如：上置信界探索、随机梯度上升

贪心

- 真实行动值$q^*(a)$：选择行动a的期望奖赏

$$q^*(a) \doteq \mathbb{E}[R_t \mid A_t = a]$$

- 行动值$Q_t(a)$： 在t时刻之前，选择行动a的平均奖赏

$$Q_t(a) \doteq \frac{\text{sum of rewards when } a \text{ taken prior to } t}{\text{number of times } a \text{ taken prior to } t} = \frac{\sum_{i=1}^{t-1} R_i \cdot \mathbb{1}_{A_i=a}}{\sum_{i=1}^{t-1} \mathbb{1}_{A_i=a}}$$

- 用行动值$Q_t(a)$作为真实行动值$q^*(a)$的估计
- 贪心行动选择方法：在t时刻，选择使得行动值$Q_t(a)$最大的行动：

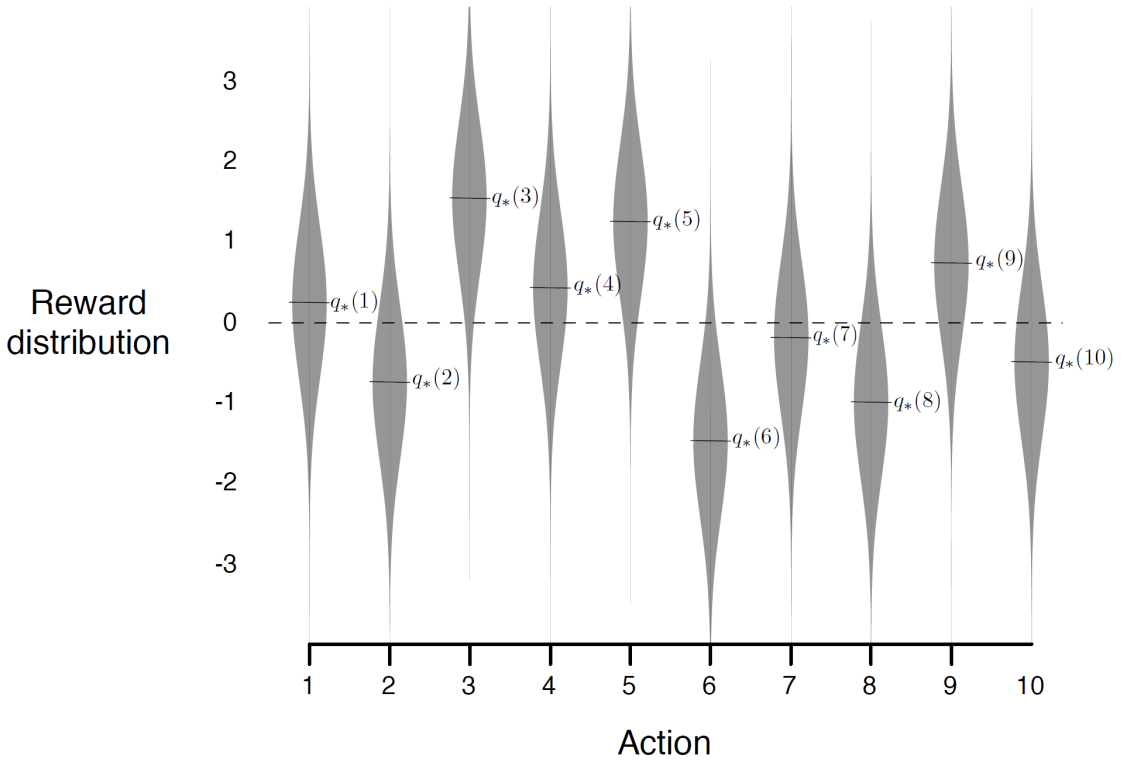
$$A_t \doteq \arg\max_a Q_t(a)$$

10-摇臂测试平台

- 由2000个随机生成的10-摇臂赌博机问题构成
- 每个10-摇臂赌博机问题的真实行动值$q^*(a)$：服从均值为0，方差为1的高斯分布

$$a = 1, \ldots, 10$$

- 拉真实行动值为$q^*(a)$的摇臂的奖赏：服从均值为$q^*(a)$，方差为1的高斯分布



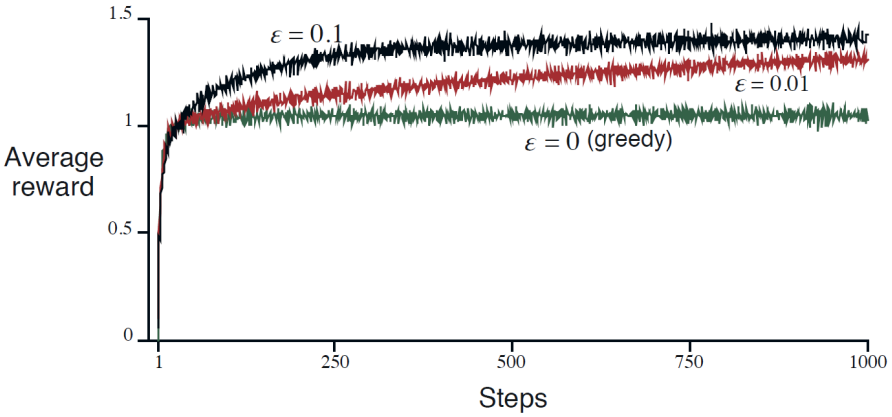
来自10-摇臂测试平台的一个10-摇臂赌博机问题

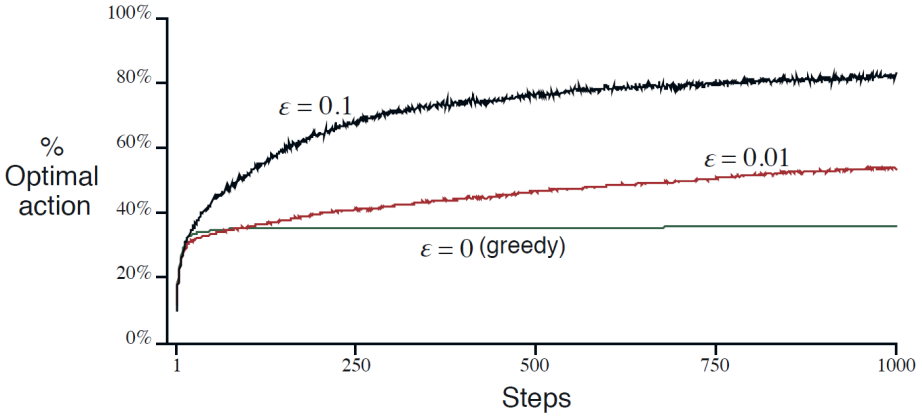
ε −贪心

- 基于概率值ε来对探索和利用进行折中
- 以ε的概率随机选择一个摇臂，以$1-\varepsilon$的概率选择摇臂

$$\arg\max_i Q_t(a)$$

- ε越大，越能快速鉴别最好的摇臂，但当拉摇臂的总次数h很大时，会浪费更多次数在次优摇臂上





增量实现

- R_i：第i次选择某一行动获得的奖赏
- Q_n：选择某一行动$n-1$次的行动值估计

$$Q_n \doteq \frac{R_1 + R_2 + \cdots + R_{n-1}}{n-1}$$

$$\begin{aligned} Q_{n+1} &= \frac{1}{n}\sum_{i=1}^{n} R_i \\ &= \frac{1}{n}\left(R_n + \sum_{i=1}^{n-1} R_i\right) \\ &= \frac{1}{n}\left(R_n + (n-1)\frac{1}{n-1}\sum_{i=1}^{n-1} R_i\right) \\ &= \frac{1}{n}\Big(R_n + (n-1)Q_n\Big) \\ &= \frac{1}{n}\Big(R_n + nQ_n - Q_n\Big) \\ &= Q_n + \frac{1}{n}\Big[R_n - Q_n\Big] \end{aligned}$$

$$Q_{n+1} \doteq Q_n + \alpha\Big[R_n - Q_n\Big]$$

- 增量更新形式

$$NewEstimate \leftarrow OldEstimate + StepSize\Big[Target - OldEstimate\Big]$$

ε −贪心算法

- 使用了增量行动值更新的ε −贪心算法

Initialize, for $a = 1$ to k:
$Q(a) \leftarrow 0$
$N(a) \leftarrow 0$

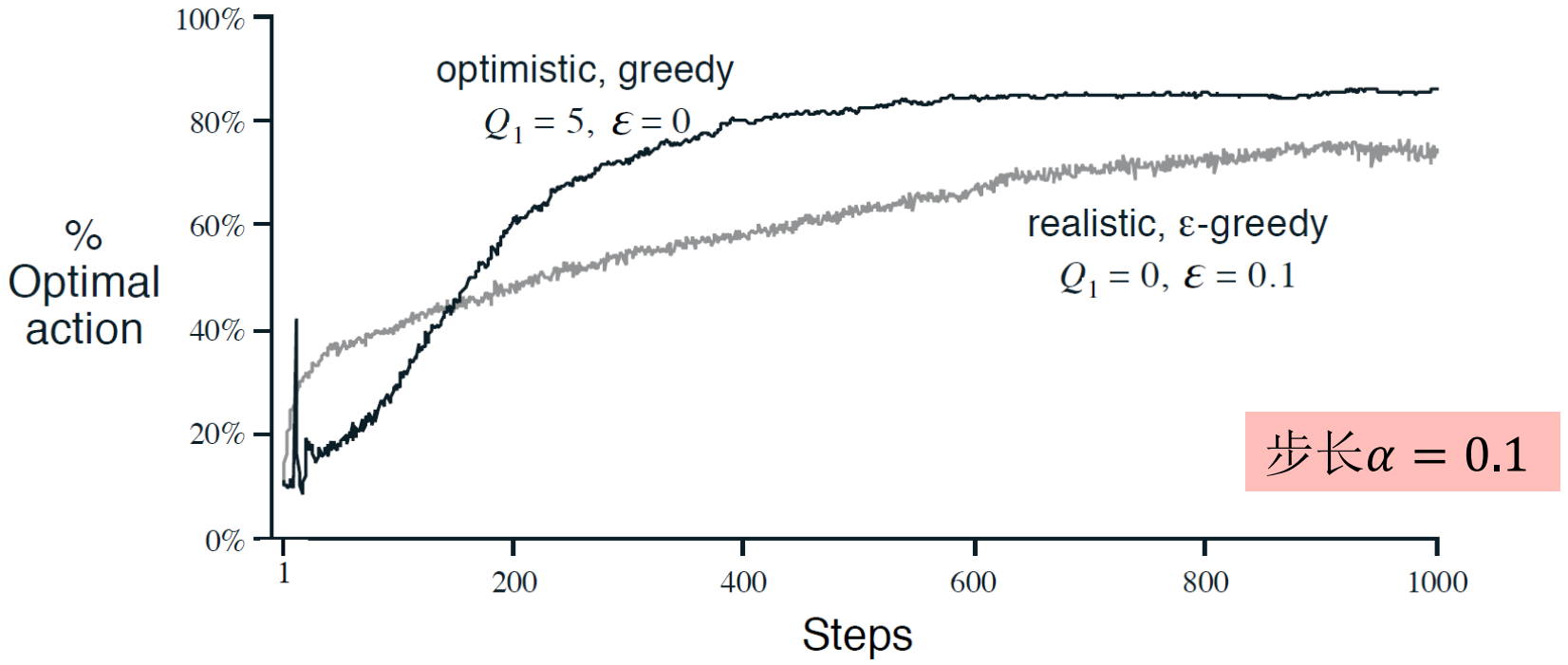
Repeat forever:
$$A \leftarrow \begin{cases} \arg\max_a Q(a) & \text{with probability } 1-\varepsilon \quad \text{(breaking ties randomly)} \\ \text{a random action} & \text{with probability } \varepsilon \end{cases}$$
$R \leftarrow bandit(A)$
$N(A) \leftarrow N(A) + 1$
$Q(A) \leftarrow Q(A) + \frac{1}{N(A)}\left[R - Q(A)\right]$

步长α

- 函数bandit(A)的作用：输入一个行动，返回一个奖赏

乐观初始化

- 思想：在不确定时保持乐观估计（Optimism Under Uncertainty）
- 乐观初始化：给所有初始行动值估计$Q_1(a)$一个乐观值
  - 鼓励对未探索过或者探索次数很少的行动进行探索



步长$\alpha = 0.1$

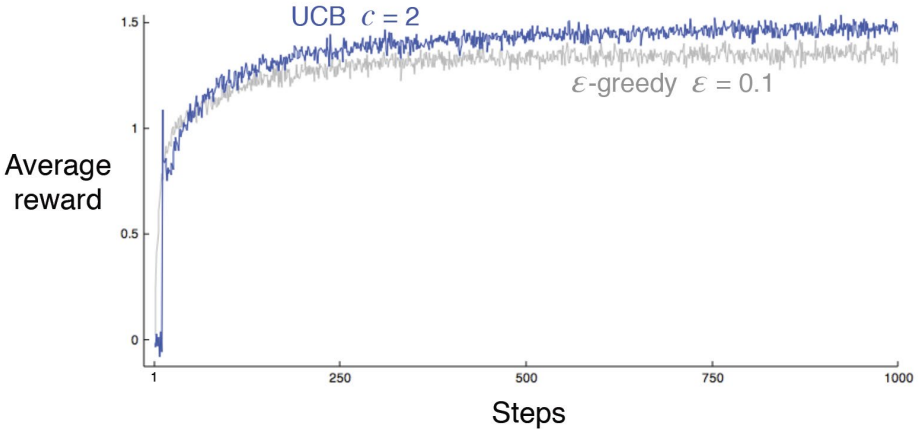
上置信界探索

- 区间探索：计算$q^*(a)$的α%置信区间，选择拉上置信界最大的摇臂

α越大，探索越多

- 上置信界（Upper Confidence Bound, UCB）行动选择：

$$A_t \doteq \arg\max_a \left[Q_t(a) + c\sqrt{\frac{\ln t}{N_t(a)}} \right]$$



参数c用来控制对探索的喜好程度

探索奖金（鼓励选择那些探索次数不多的行动）
若$N_t(a) = 0$，则奖金无穷大

随机梯度上升 Gradient Bandit

- $H_t(a)$：对行动a的偏好程度
- 根据软最大化（又称：吉布斯、玻尔兹曼）分布选择行动：

$$\Pr\{A_t = a\} \doteq \frac{e^{H_t(a)}}{\sum_{b=1}^{k} e^{H_t(b)}} \doteq \pi_t(a)$$

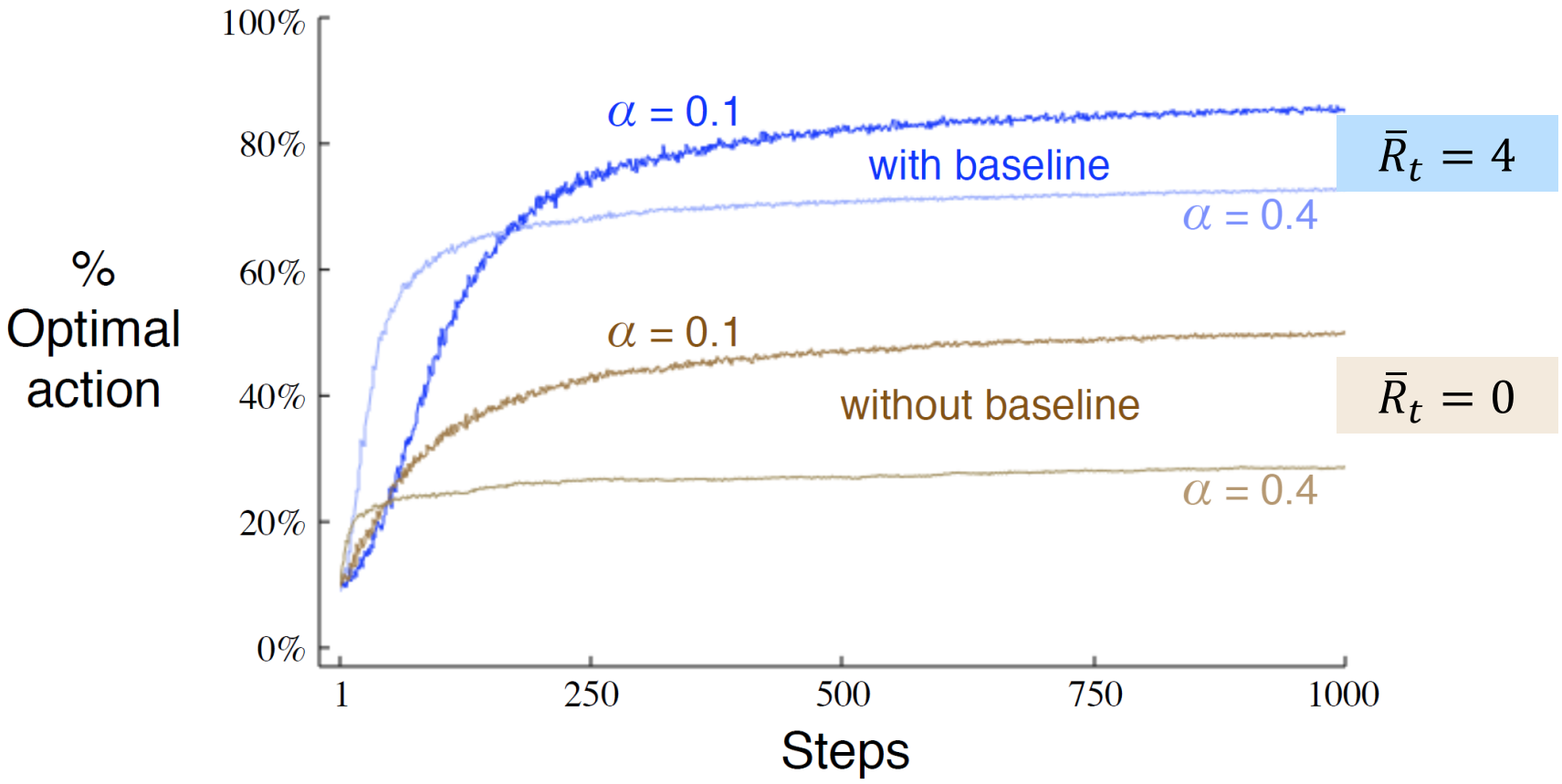
- 在每一步，采取了行动A_t，得到了奖赏R_t后，更新偏好：

$$H_{t+1}(A_t) \doteq H_t(A_t) + \alpha\big(R_t - \bar{R}_t\big)\big(1 - \pi_t(A_t)\big), \qquad \text{and}$$
$$H_{t+1}(a) \doteq H_t(a) - \alpha\big(R_t - \bar{R}_t\big)\pi_t(a), \qquad \text{for all } a \neq A_t$$

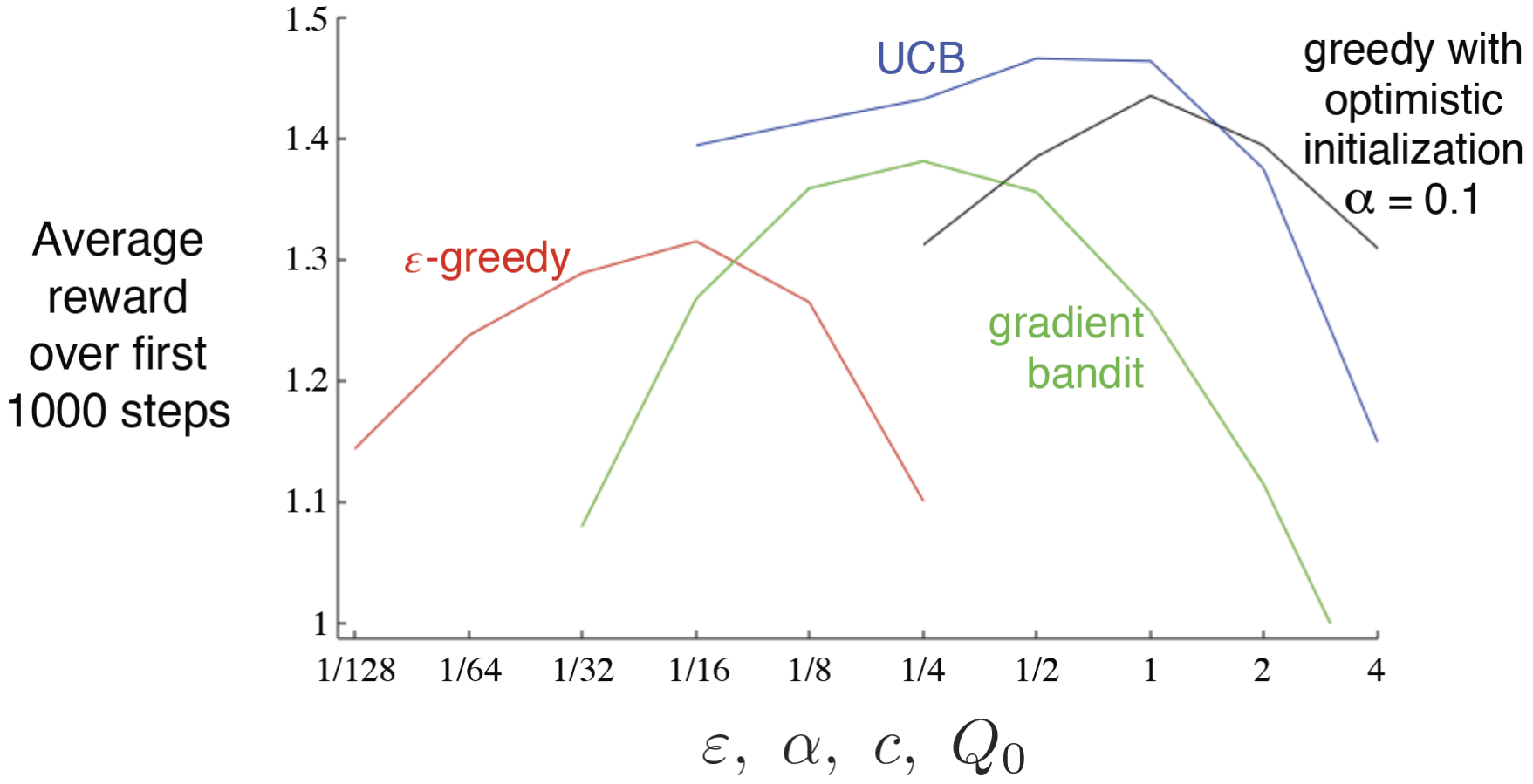
随机近似

精确梯度上升方法：$H_{t+1}(a) \doteq H_t(a) + \alpha \frac{\partial \mathbb{E}[R_t]}{\partial H_t(a)}$ $\mathbb{E}[R_t] = \sum_b \pi_t(b) q_*(b)$

随机梯度上升的性能


100%
80%
60%
40%
20%
0%
% Optimal action
1
250
500
750
1000
Steps
α = 0.1
with baseline
α = 0.4
α = 0.1
without baseline
α = 0.4
$\bar{R}_t = 4$
$\bar{R}_t = 0$

几种探索策略的性能对比



探索与利用

- 多摇臂赌博机问题、贝叶斯模型估计
- 探索策略
- **最优探索策略**

最优探索策略

ρ_i表示摇臂i获胜的后验概率

- 信念状态
  - 计数$w_1, \ell_1, \ldots, w_n, \ell_n$
  - 用作一个表示n-摇臂赌博机问题的MDP的状态

这$2n$个数表示 $\rho_{1:n}$可能取值的n个连续概率分布

- 下面用动态规划来确定一个最优策略π^*

输入：计数
输出：要拉的摇臂

- $Q^*(w_{1:n}, \ell_{1:n}, i)$：拉摇臂$i$，然后执行最优行动的期望回报
- 用Q^*表示的最优状态值函数和最优策略分别为：

$$U^*(w_1, \ell_1, \ldots, w_n, \ell_n) = \max_i Q^*(w_1, \ell_1, \ldots, w_n, \ell_n, i)$$

$$\pi^*(w_1, \ell_1, \ldots, w_n, \ell_n) = \arg\max_i Q^*(w_1, \ell_1, \ldots, w_n, \ell_n, i)$$

最优探索策略（续）

- 分解Q^*成两部分：

获胜的后验概率

$$Q^*(w_1,\ell_1,\ldots,w_n,\ell_n,i)=\frac{w_i+1}{w_i+\ell_i+2}\left(1+U^*(\ldots,w_i+1,\ell_i,\ldots)\right)$$
$$+\left(1-\frac{w_i+1}{w_i+\ell_i+2}\right)U^*(\ldots,w_i,\ell_i+1,\ldots)$$

失败的后验概率

- 假设步数为h，可以计算整个信念状态空间的Q^*：
  - $\sum_i(w_i+\ell_i)=h$的信念状态，满足$U^*(w_1,\ell_1,\ldots,w_n,\ell_n)=0$
  - 然后，用Q^*的分解式计算$\sum_i(w_i+\ell_i)=h-1$的信念状态的U^*
  - 如此进行，直至计算完$\sum_i(w_i+\ell_i)=0$的信念状态的U^*

信念状态的总数为：$O(h^{2n})$

最优探索策略（续）

证明：信念状态的总数为：$O(h^{2n})$

- $\sum_i(w_i+\ell_i)=h$的信念状态数等于

$$w_1+\ell_1+\cdots+w_n+\ell_n=h,\ \ w_i,\ell_i\geq 0$$

的非负整数解的个数：

$$C_{h+2n-1}^{h}=\frac{(h+2n-1)!}{(2n-1)!\,h!}$$

$$=\frac{(h+2n-1)(h+2n-2)\ldots(h+1)}{(2n-1)!}$$

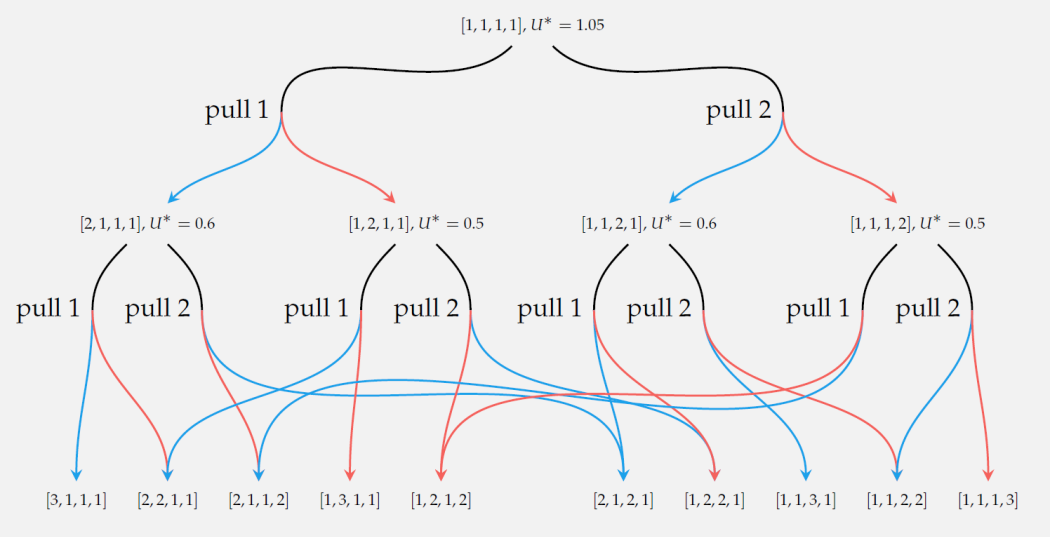
$$=O\left(h^{(2n-1)}\right)$$

- 类似地，$\sum_i(w_i+\ell_i)=h-1$的信念状态数为$O\left((h-1)^{(2n-1)}\right)$
- 信念状态的总数量：

$$O\left(h^{(2n-1)}+(h-1)^{(2n-1)}+\cdots+1^{(2n-1)}\right)\leq O\left(h*h^{(2n-1)}\right)=O\left(h^{2n}\right)$$

最优探索策略（续）

- 2-摇臂赌博机问题的深度为2的状态-行动树
  - 状态向量：$[w_1, \ell_1, w_2, \ell_2]$
  - 蓝色箭头：获胜
  - 红色箭头：失败



- 摇臂1和摇臂2的策略是对称的

- 最优策略：拉获胜的摇臂2次，不拉失败的摇臂2次

$$Q^*([2,1,1,1],1) = \frac{3}{5}(1+0) + \frac{2}{5}(0) = 0.6$$

$$Q^*([2,1,1,1],2) = \frac{2}{4}(1+0) + \frac{2}{4}(0) = 0.5$$

$$Q^*([1,2,1,1],1) = \frac{2}{5}(1+0) + \frac{3}{5}(0) = 0.4$$

$$Q^*([1,2,1,1],2) = \frac{2}{4}(1+0) + \frac{2}{4}(0) = 0.5$$

$$Q^*([1,1,1,1],1) = \frac{2}{4}(1+0.6) + \frac{2}{4}(0.5) = 1.05$$

小结：探索与利用

- n-摇臂赌博机问题
  - 探索：估计摇臂的优劣
  - 利用：选择当前最好的摇臂
- 极大似然估计 vs. 贝叶斯模型估计
- 探索策略
  - 无向：ε－贪心、乐观初始化
  - 有向：上置信界探索、随机梯度上升
- 最优探索策略

内容安排

- 探索与利用
- **基于模型的方法**
- 免模型的方法
- 泛化
- 策略梯度

基于模型的方法

- 基于极大似然模型的方法
- 基于贝叶斯模型的方法

模型的极大似然估计

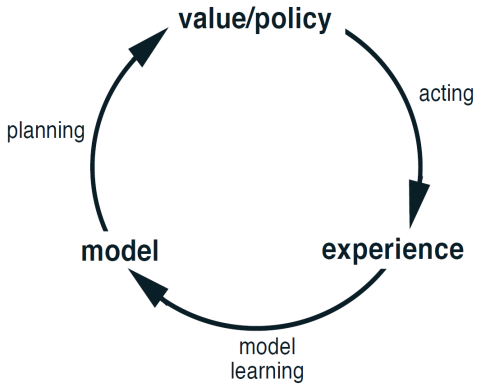
- 考虑有多个状态的强化学习问题
  - 比n-摇臂赌博机问题更有挑战性
  - 要考虑行动的延迟效果，即行动不仅有立即奖赏，还会影响后续状态
- 极大似然方法：直接从经验中估计转移模型和奖赏模型
- 记录转移数量$N(s,a,s')$和奖赏之和$\rho(s,a)$
- 转移模型和奖赏模型的极大似然估计如下：

$$N(s,a)=\sum_{s'}N(s,a,s')$$

$$T(s'\mid s,a)=N(s,a,s')/N(s,a)$$

$$R(s,a)=\rho(s,a)/N(s,a)$$

基于极大似然模型的方法



- 算法5.1：基于极大似然模型的强化学习

Algorithm 5.1 Maximum likelihood model-based reinforcement learning

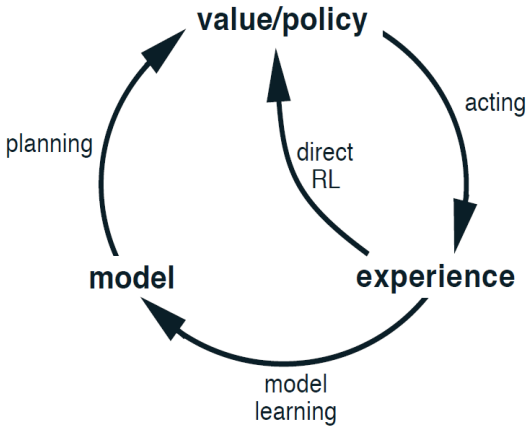
```
function MaximumLikelihoodModelBasedReinforcementLearning
    t ← 0
    s_0 ← initial state
    Initialize N, ρ, and Q
    loop
        Choose action a_t based on some exploration strategy
        Observe new state s_{t+1} and reward r_t
        N(s_t, a_t, s_{t+1}) ← N(s_t, a_t, s_{t+1}) + 1
        ρ(s_t, a_t) ← ρ(s_t, a_t) + r_t
        Update Q based on revised estimate of T and R
        t ← t + 1
```

若有先验知识，可初始化为非零值

结合某一探索策略，以逐渐收敛至一个最优策略

基于估计的模型，求解MDP问题，得到新的Q值

随机化的更新



- 算法5.1的第10行
  - 可以用动态规划算法来更新Q值，但这种计算花销通常是不必要的
- Dyna：避免在每一时间步求解整个MDP

1. 在当前状态执行如下更新：

$$Q(s,a) \leftarrow Q(s,a) + \alpha\left(r + \gamma \max_{a'} Q(s',a') - Q(s,a)\right)$$

Q值增量更新：不需要知道模型

由下式推导得到：

$$Q(s,a) \leftarrow R(s,a) + \gamma \sum_{s'} T(s' \mid s,a) \max_{a'} Q(s',a')$$

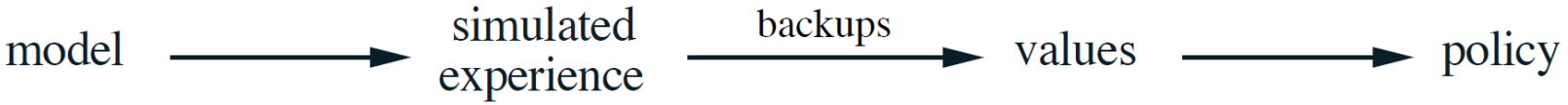
需要用到样本模型和规划来实现

2. 然后随机选取一些状态和行动，更新这些状态-行动对上的Q值
3. 更新后，使用某一探索策略（如软最大化）来选择要执行的行动

模型和规划

- 环境的**模型**：可以用于预测环境如何对Agent的行动做出响应
  - 给定一个状态和一个行动，一个模型能产生对下一个状态和立即奖赏的预测
- 分布模型：能产生下一个状态和立即奖赏的概率分布
- 样本模型：能从下一个状态和立即奖赏的概率分布中产生一个样本状态和一个样本奖赏
- **规划**：以一个模型为输入，输出一个策略的计算过程

- **状态空间规划**：



Q规划算法

- 特点：需要给定一个样本模型作为输入

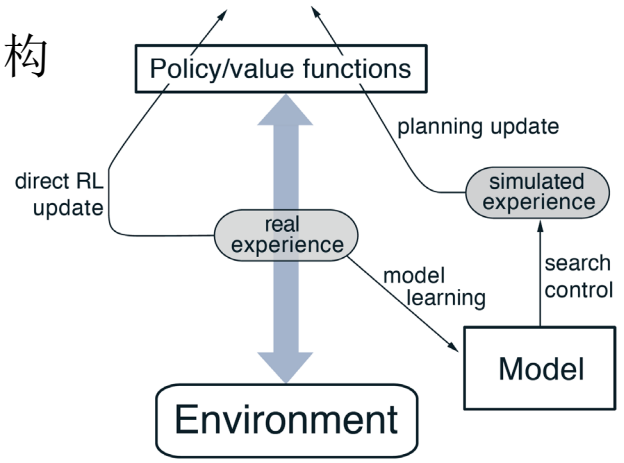
Random-sample one-step tabular Q-planning

Loop forever:

1. Select a state, $S \in \mathcal{S}$, and an action, $A \in \mathcal{A}(S)$, at random
2. Send S, A to a sample model, and obtain
   a sample next reward, R, and a sample next state, S'
3. Apply one-step tabular Q-learning to S, A, R, S':
   $Q(S,A) \leftarrow Q(S,A) + \alpha\left[R + \gamma \max_a Q(S',a) - Q(S,A)\right]$

Dyna-Q算法

Dyna的架构

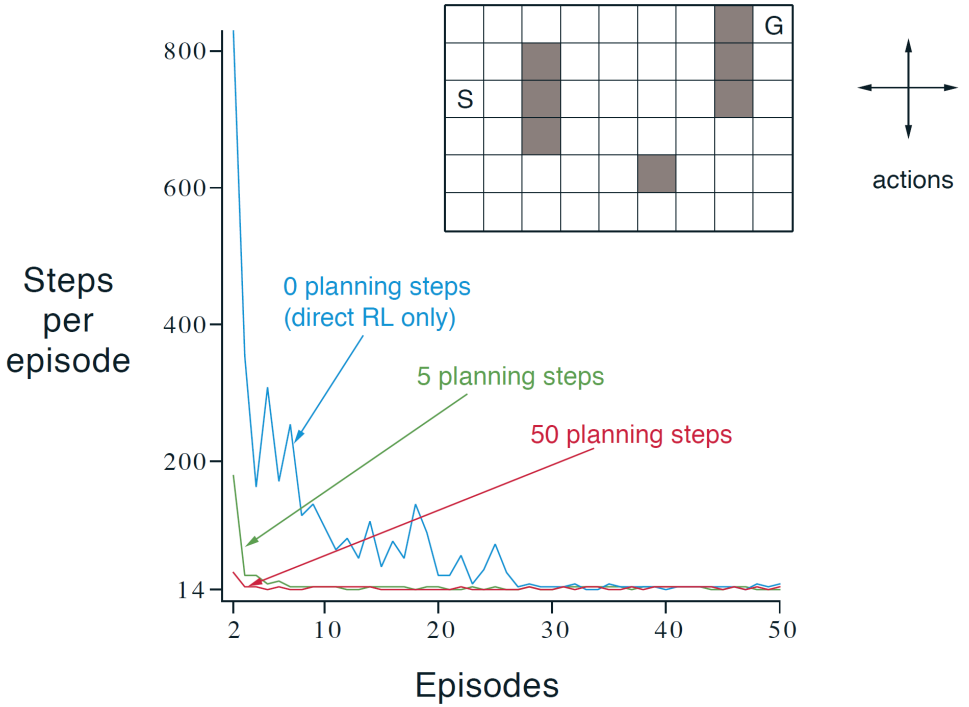


- 结合了Dyna和Q规划
- 假设环境的模型是确定性的

Initialize $Q(s,a)$ and $Model(s,a)$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$
Loop forever:
(a) $S \leftarrow$ current (nonterminal) state
(b) $A \leftarrow \varepsilon$-greedy(S,Q)
(c) Take action A; observe resultant reward, R, and state, S'
(d) $Q(S,A) \leftarrow Q(S,A) + \alpha\left[R + \gamma \max_a Q(S',a) - Q(S,A)\right]$
(e) $Model(S,A) \leftarrow R, S'$ (assuming deterministic environment)
(f) Loop repeat n times:
$S \leftarrow$ random previously observed state
$A \leftarrow$ random action previously taken in S
$R, S' \leftarrow Model(S,A)$
$Q(S,A) \leftarrow Q(S,A) + \alpha\left[R + \gamma \max_a Q(S',a) - Q(S,A)\right]$

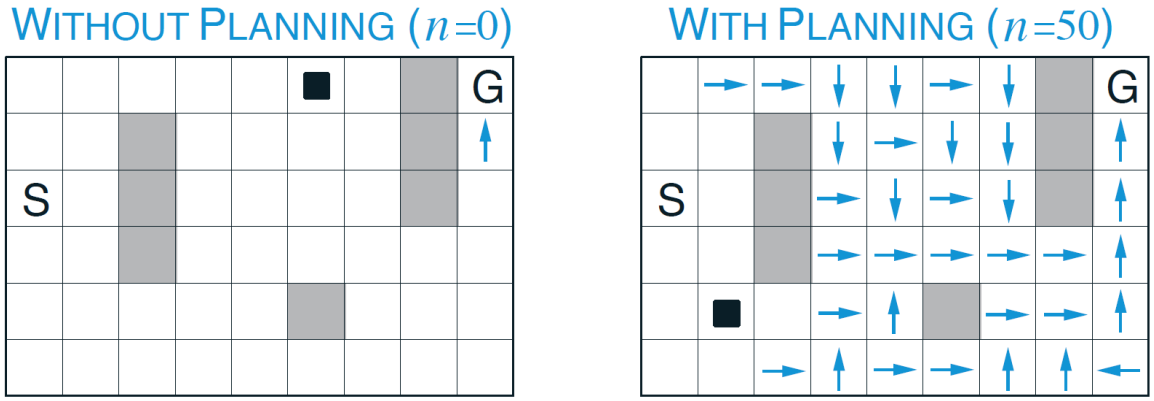
迭代次数为n

- Dyna迷宫
  - 47个状态
  - 4个行动
  - 行动的结果是确定的
  - 进入目标状态G的立即奖赏为1，否则为0
  - 到达G后，重置Agent到起始状态S，进入下一个情节



- Dyna-Q的性能对比：n取不同值

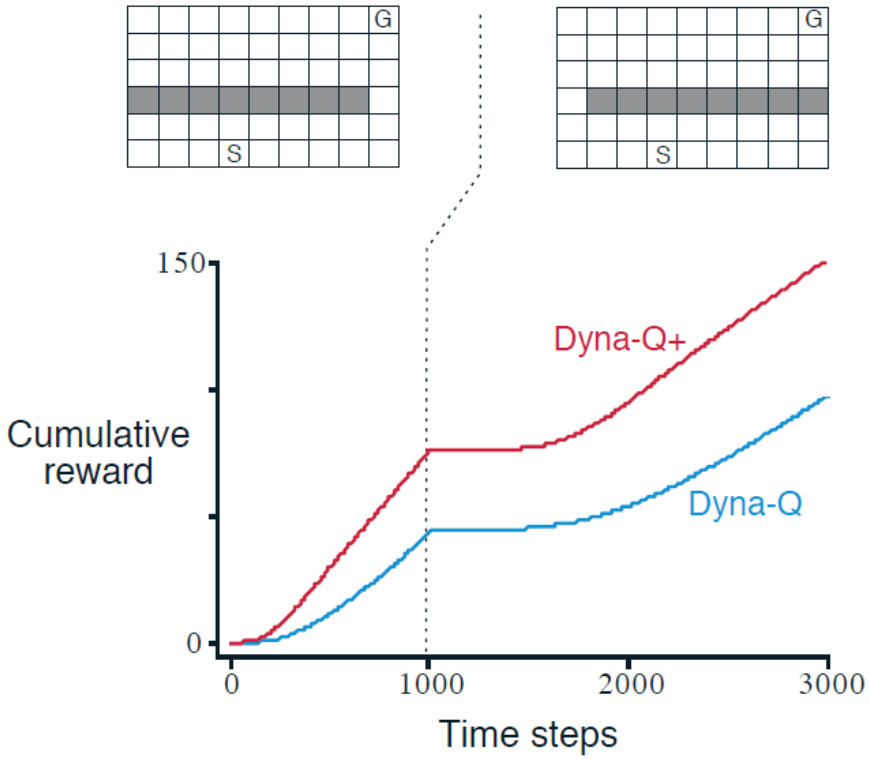
在第二个情节开始了一段时间后，两个算法找到的策略对比

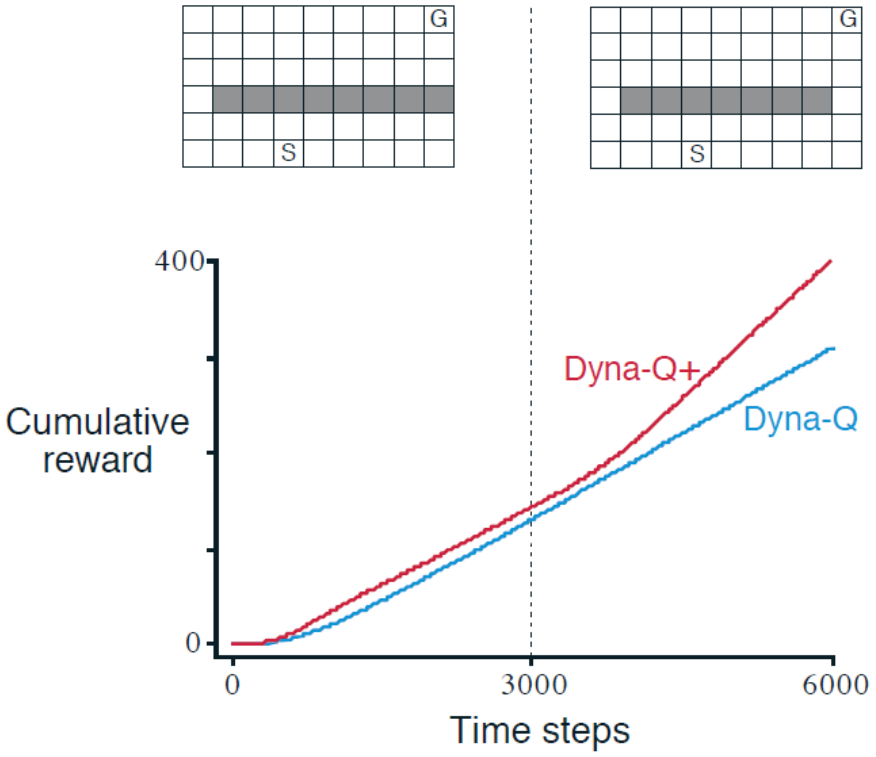


Dyna-Q vs. Dyna-Q+

τ：与最后一次探索某转移的时间间隔

- Dyna-Q+：使用了探索奖金以鼓励探索 $r + \kappa\sqrt{\tau}$
- 阻塞迷宫
  - 1000时间步后，阻塞捷径
- 捷径迷宫
  - 1000时间步后，开通捷径





优先级更新

- 优先级扫描
  - 使用一个优先级队列来帮助鉴别最需要更新U值的状态

Algorithm 5.2 Prioritized sweeping

```
1: function PRIORITIZEDSWEEPING(s)
2:     Increase the priority of s to ∞
3:     while priority queue is not empty
4:         s ← highest priority state
5:         UPDATE(s)
6: function UPDATE(s)
7:     u ← U(s)
8:     U(s) ← max_a [R(s,a) + γ Σ_{s'} T(s' | s,a) U(s')]
9:     for (s',a') ∈ pred(s)
10:        p ← T(s | s',a') × |U(s) − u|
11:        Increase priority of s' to p
```

持续更新队列中有最高优先级的状态，直至队列为空

s的前辈集合：$\text{pred}(s) = \{(s', a') \mid T(s \mid s', a') > 0\}$

如果从s转移到s'，则基于转移和奖赏模型来更新$U(s)$

把s'的优先级增至p，u是$U(s)$在更新前的值

使用了优先级扫描Q规划的Dyna算法

- 假设环境的模型是确定性的

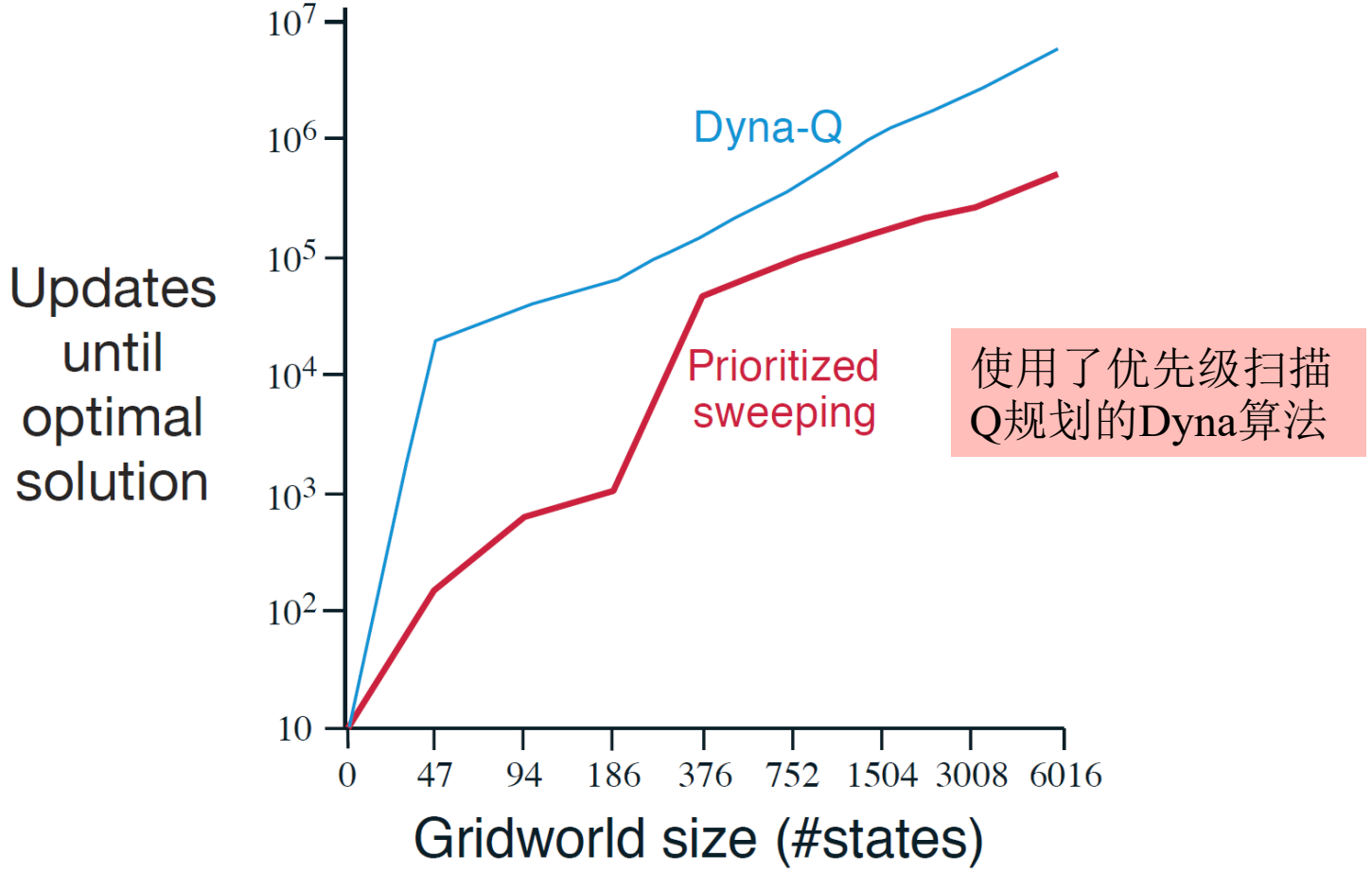
Dyna-Q

Loop repeat n times:
$S \leftarrow$ random previously observed state
$A \leftarrow$ random action previously taken in S
$R, S' \leftarrow Model(S, A)$
$Q(S, A) \leftarrow Q(S, A) + \alpha\left[R + \gamma \max_a Q(S', a) - Q(S, A)\right]$

使用了优先级扫描Q规划的Dyna

Loop repeat n times, while $PQueue$ is not empty:
$S, A \leftarrow first(PQueue)$
$R, S' \leftarrow Model(S, A)$
$Q(S, A) \leftarrow Q(S, A) + \alpha\left[R + \gamma \max_a Q(S', a) - Q(S, A)\right]$
Loop for all $\bar{S}, \bar{A}$ predicted to lead to S:
$\bar{R} \leftarrow$ predicted reward for $\bar{S}, \bar{A}, S$
$P \leftarrow |\bar{R} + \gamma \max_a Q(S, a) - Q(\bar{S}, \bar{A})|$.
if $P > \theta$ then insert $\bar{S}, \bar{A}$ into $PQueue$ with priority P

Dyna-Q w./w.o. 优先级扫描



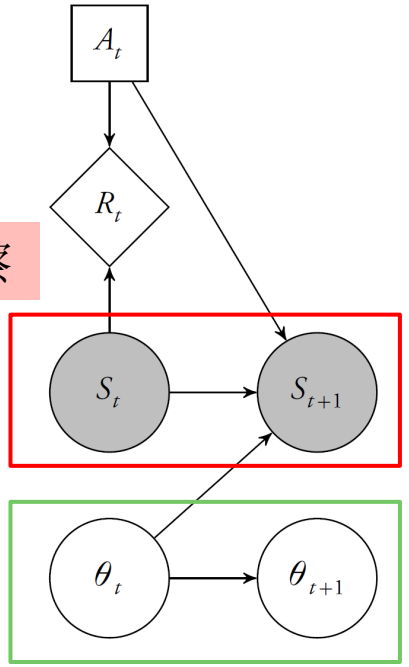
使用了优先级扫描Q规划的Dyna算法

基于模型的方法

- 基于极大似然模型的方法
- **基于贝叶斯模型的方法**

基于贝叶斯模型的方法

用动态决策网络表示问题的结构



状态：可观察

- 贝叶斯方法
  - 不依赖于启发式探索策略
  - 能在探索和利用之间最优平衡

事实上，模型参数还包括立即奖赏分布的参数

- 模型参数
  - 这里只考虑状态转移概率的参数
  - 参数（向量）θ由$|S|^2|\mathcal{A}|$个元素构成
  - $\theta_{(s,a,s')}$：与转移概率$T(s' \mid s,a)$对应的元素
  - 需要指定模型参数θ的一个先验分布

模型参数：不可观察

一般假设模型参数是时间不变的，即$\theta_{t+1} = \theta_t$

θ的信念状态会随着转移到新的状态发生变化

模型参数的信念状态

- θ的先验信念状态
  - 对于离散状态空间，用狄利克雷分布的乘积来表示

$$b_0(\theta) = \prod_s \prod_a \mathrm{Dir}(\theta_{(s,a)} \mid \alpha_{(s,a)})$$

$\theta_{(s,a)}$是一个$|\mathcal{S}|$维向量，服从先验分布$\mathrm{Dir}(\theta_{(s,a)} \mid \alpha_{(s,a)})$

  - $\mathrm{Dir}(\theta_{(s,a)} \mid \alpha_{(s,a)})$由$\alpha_{(s,a)}$中的$|\mathcal{S}|$个参数控制
  - 均匀先验分布： $\alpha_{(s,a)}$中的所有参数均设为1
  - 如果有转移模型的先验知识，则可以把这些参数设为不同值

- b_t：在t步后，θ的后验信念状态

用贝叶斯规则算出

$$b_t(\theta) = \prod_s \prod_a \mathrm{Dir}(\theta_{(s,a)} \mid \alpha_{(s,a)} + \mathbf{m}_{(s,a)})$$

$\mathbf{m}_{(s,a)}$：一个$|\mathcal{S}|$维向量，其中$m_{(s,a,s')}$表示在前t步，观察到从s采取行动a转移到s'的次数

贝叶斯自适应MDPs

基础MDP

模型未知的MDP

形式化

贝叶斯自适应MDP

模型已知的、更高维的MDP

- 贝叶斯自适应MDP
  - 状态：$(s, b) \in \mathcal{S} \times \mathcal{B}$
    - s是基础MDP的状态，b是信念状态
    - $\mathcal{B}$是模型参数θ的所有可能信念状态构成的空间
  - 行动空间、奖赏函数与基础MDP完全相同
  - 转移函数：$T(s', b' \mid s, b, a) = \delta_{\tau(s,b,a,s')}(b') P(s' \mid s, b, a)$

$$P(s' \mid s, b, a) = \int_\theta b(\theta) P(s' \mid s, \theta, a)\, d\theta = \int_\theta b(\theta) \theta_{(s,a,s')}\, d\theta$$

克罗内克（Kronecker）delta函数

$$\delta_x(y) = \begin{cases} 1 & \text{if } x = y \\ 0 & \text{otherwise} \end{cases}$$

$\tau(s, b, a, s')$ 是s, b, a, s'的确定性函数，由贝叶斯规则算出

求解方法

- 把Bellman最优方程从模型已知的MDPs泛化至模型未知的情形：

$$U^*(s,b)=\max_a\left(R(s,a)+\gamma\sum_{s'}P(s'\mid s,b,a)U^*(s',\tau(s,b,a,s'))\right)$$

  - 因为b是连续的，不能直接用第4部分的值迭代、策略迭代算法求解
  - 可以用第4部分的近似方法和在线方法，更好的方法在第6部分

- 汤普森采样
  - 从当前信念状态b_t中抽取一个样本θ
  - 假设θ是真实的模型，使用动态规划来求解最好的行动
  - 在下一个时间步，更新信念状态，抽取一个新样本，重新求解MDP

缺点：过度探索，计算复杂度高（每一步都要求解一个MDP）

小结：基于模型的方法

- 基于极大似然模型的方法
  - 随机化的更新：Dyna方法
  - 优先级更新：优先级扫描方法

- 基于贝叶斯模型的方法
  - 模型参数的信念状态
  - 贝叶斯自适应MDPs
  - 求解方法：汤普森采样方法等

课后练习5.1

- 为什么探索与利用的概念在强化学习中如此重要？
- 什么是n-摇臂赌博机问题？
- 假想你有一个2-摇臂赌博机，已经知道拉其中1个摇臂会有0.9的概率输出$1，但你还没有拉过另外一个摇臂，不能确定拉它是否有回报。试说明这个问题中的探索与利用。

课后练习5.2

- 假设我们有一个2-摇臂赌博机。我们估计第1个摇臂的回报为0.7，第2个摇臂的回报为0.6，即$\rho_1 = 0.7$，$\rho_2 = 0.6$。θ_1和θ_2的95%置信区间为$(0.6, 0.8)$，$(0.3, 0.9)$。问：θ_i和ρ_i之间的不同是什么？假设你使用了一个$\varepsilon-$贪心策略，其中$\varepsilon = 0.5$。你如何决定拉哪个摇臂？假设你使用了一个95%置信区间的区间探索策略，你会拉哪个摇臂？