强化学习

高　阳，李文斌

http://cs.nju.edu.cn/rl

2023年12月25日

大 纲

起源

MDP模型

动态规划

强化学习

其他议题

大 纲

起源

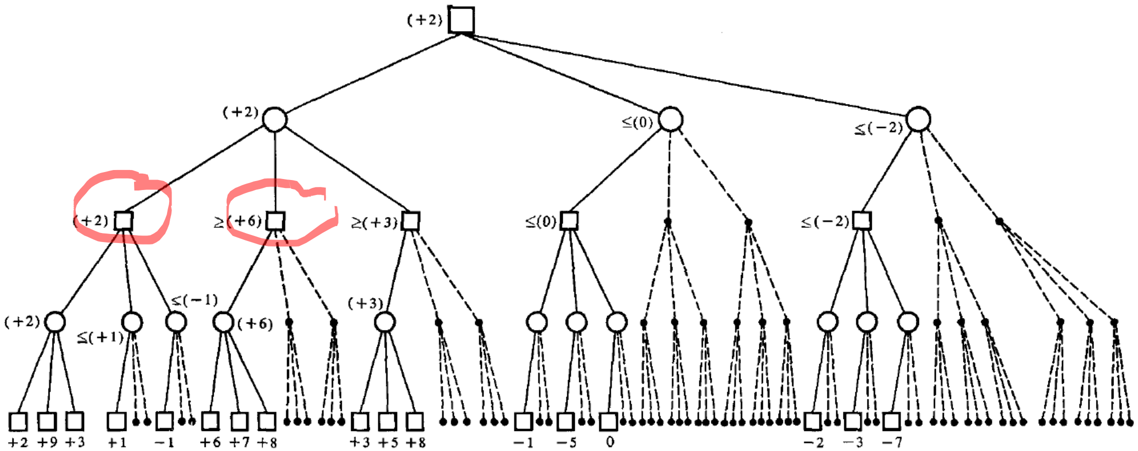
MDP模型

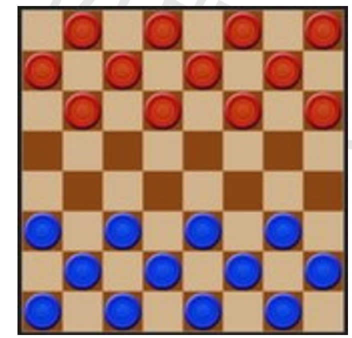
动态规划

强化学习

其他议题

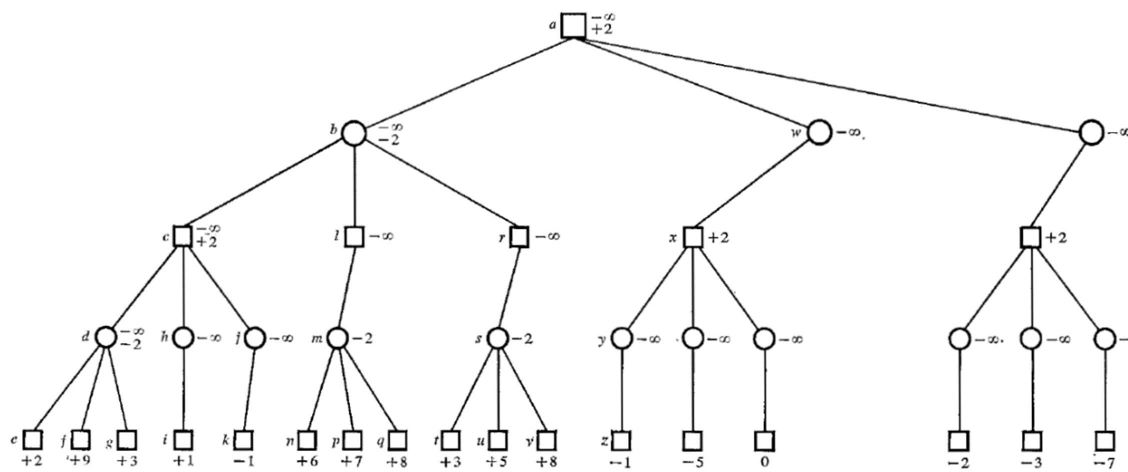
最早的“人机大战”





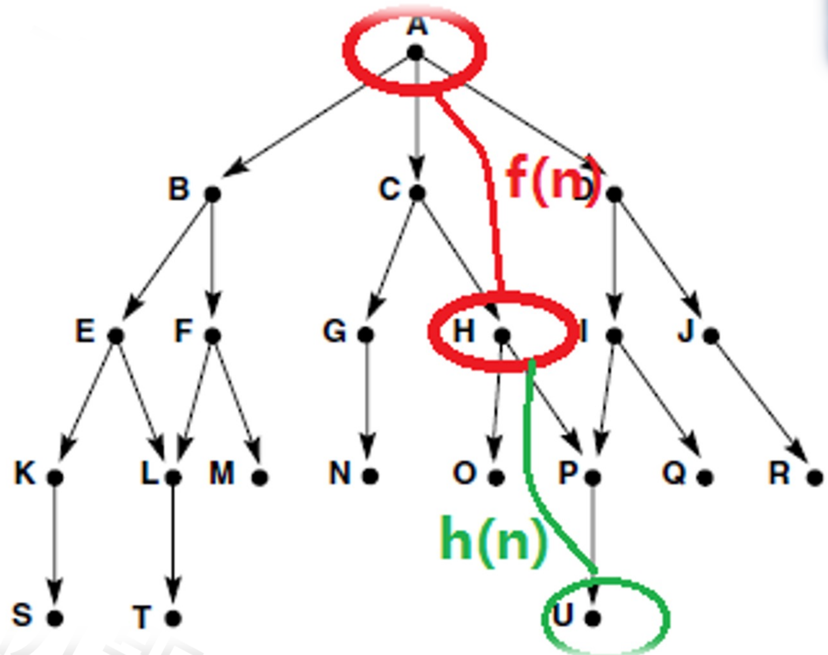
值函数

Alpha-beta剪枝



A. L. Samuel. Some studies in machine learning using the game of checkers. IBM Journal of R & D, 3:221-229, 1959

启发式搜索

A*

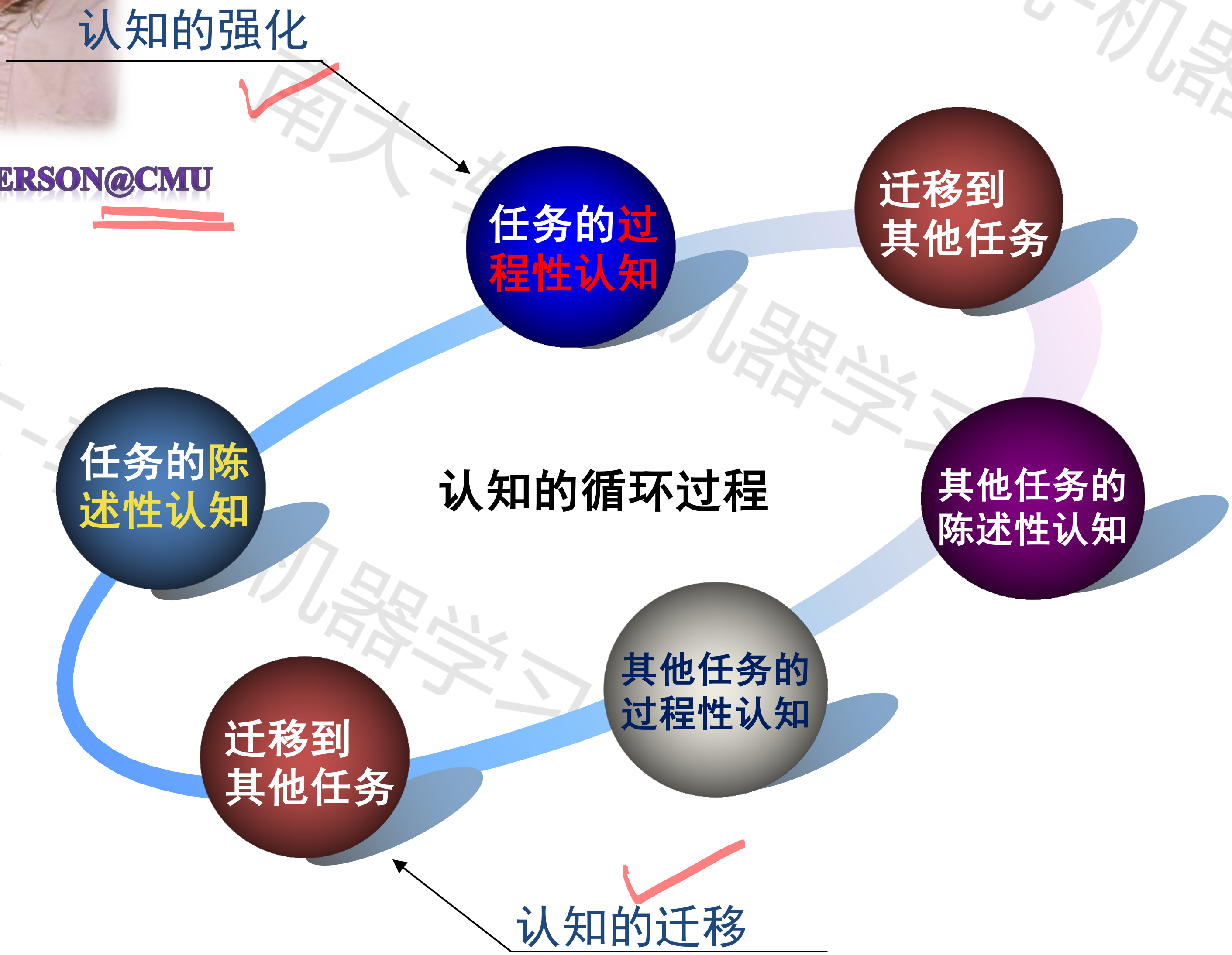
启发式估值的定义

$$f(n)=g(n)+h(n)$$

P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths in graphs. IEEE Trans. Syst. Sci. and Cybernetics, SSC-4(2):100-107, 1968

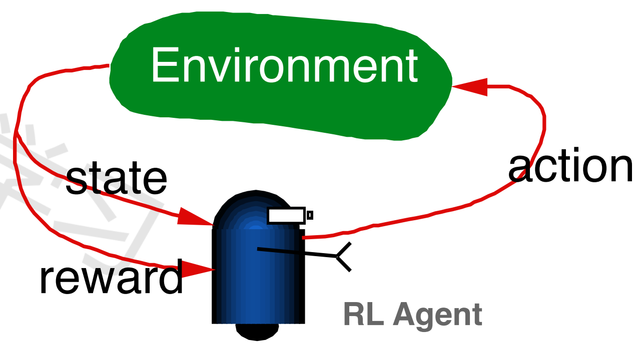
从认知的角度

JOHN R. ANDERSON@CMU



强化学习问题

具体事例



抽象模型

强化学习的本质：奖惩和试错(Trial and Error)

交互学习 VS 概念学习

□概念学习

- ✓ 给定正例/反例，学习目标概念(如监督学习)

□交互学习

- ✓ 通过交互学习一个任务(如走出迷宫)
  - ✓ 系统(或外部环境)存在若干个"状态"
  - ✓ 学习算法/动作会影响"状态"的分布
  - ✓ 潜在的Exploration和Exploitation折衷

挑 战

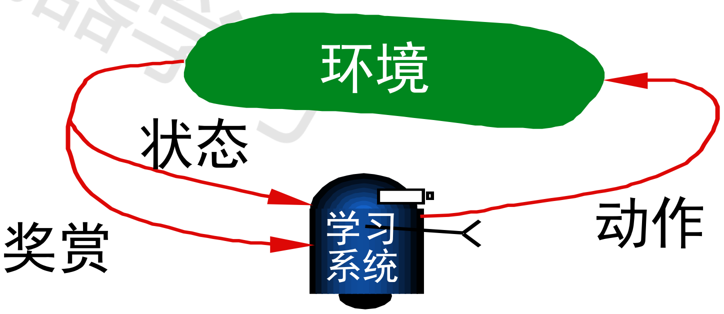
不确定性

- ✓ 环境、动作、反馈、模型

学习的目标

- ✓ 概念 → 决策
- ✓ 最大化长期奖赏

Markov **D**ecision **P**rocess



大 纲

起源

MDP模型

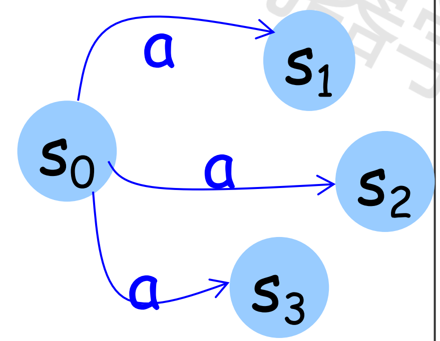
动态规划

强化学习

其他议题

数学模型 - MDP

Markov Decision Process



S- set of states， 状态集合

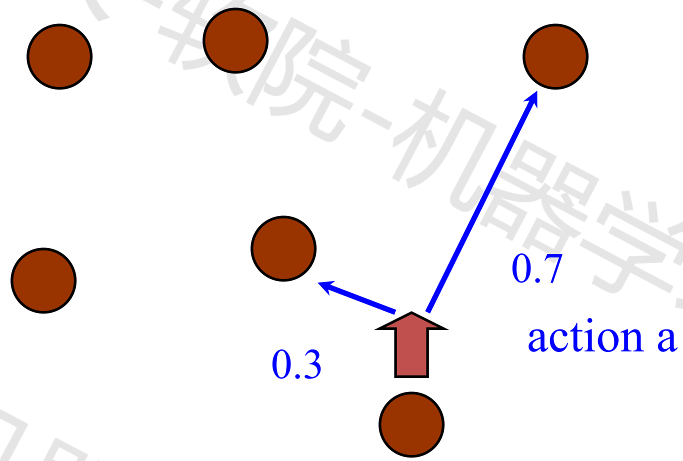
A- set of actions， 动作集合

δ - transition probability，状态转移概率

R – immediate reward function， 即时奖赏函数

MDP模型 – 状态和动作

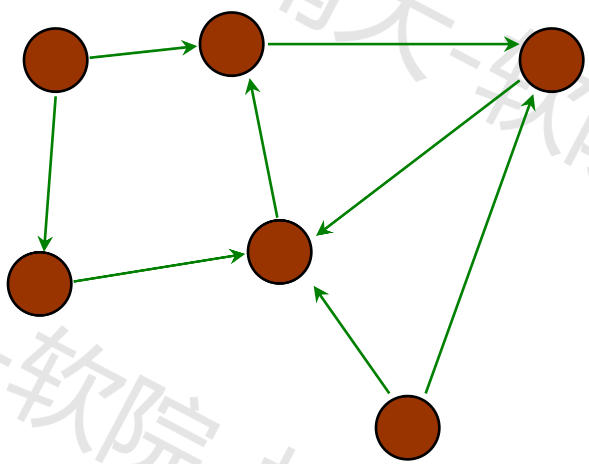
环境 = 状态集合



状态之间的转移　　$\delta(s,a,s')$

MDP模型 – 奖赏

R(s,a) = 在状态s，采用a动作获得的奖赏

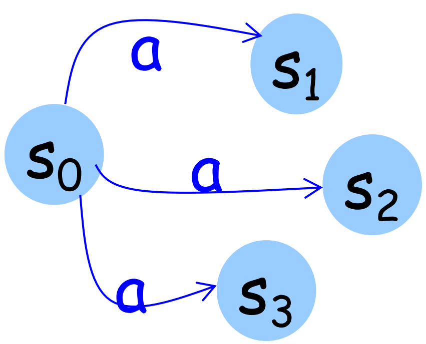


举例:

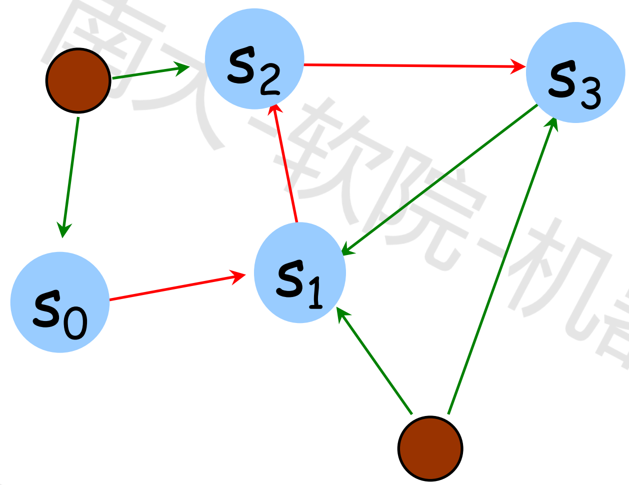
R(s,a) = -1 with probability 0.5

+10 with probability 0.35

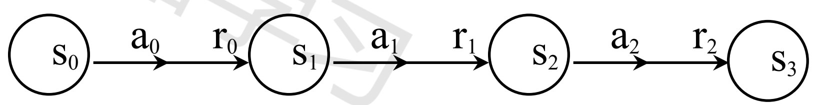
+20 with probability 0.15



MDP模型 – 轨迹



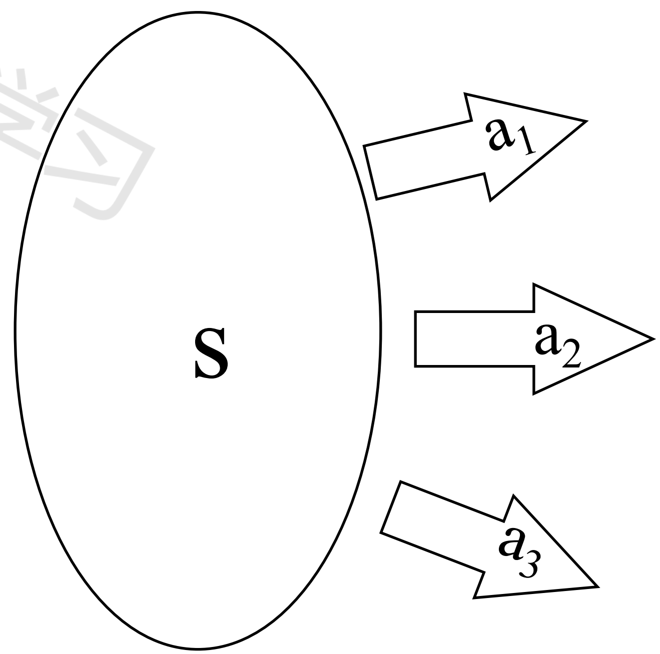
在一次Episode中，所获得的经验或轨迹(trajectory)



MDP模型 –动作选择

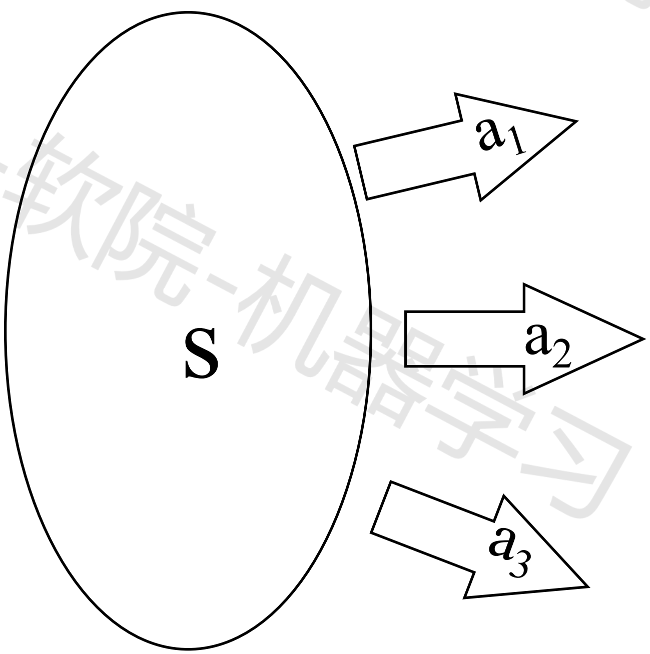
- 目标
  - ✓ 最大化期望奖赏(单状态下)
- 策略
  - ✓ 状态到动作的映射(π: S→A)

单状态学习问题



例：N-臂老虎机

单状态学习问题



目标：最大化期望即时奖赏

给定模型：采用贪心动作
(Greedy action)

困难：模型未知

MDP模型 – 返回函数

- 返回函数(面向多状态学习问题)
  - ✓ 将所有的即时奖赏组合成一个单一值
- Modeling Issues
  - ✓ 轨迹中早期的奖赏和晚期的奖赏相比，谁更重要?
  - ✓ 系统是持续的？还是有终止状态的?

通常返回函数是即时奖赏值的线性组合

MDP模型 – 返回函数

- 有限窗口(Finite Horizon)

$$\text{return} = \sum_{1 \leqslant i \leqslant H} R(s_i, a_i)$$

- 无穷窗口(Infinite Horizon)
  - ✓ 有折扣 $\text{return} = \sum_{i=0}^{\infty} \gamma^i R(s_i, a_i)$
  - ✓ 无折扣 $\text{return} = \frac{1}{N} \sum_{i=0}^{N-1} R(s_i, a_i) \quad N \to \infty$

通常返回函数是即时奖赏值的线性组合

MDP模型 –动作选择

- 目标
  - ✓ 最大化期望返回(Return)
- 策略
  - ✓ 状态到动作的映射(π: S→A)
- 最优策略
  - ✓ 如果π是最优策略，则其从任一状态出发，均是最优的策略

定理：必然存在着一个确定性的最优策略

监督学习 VS 强化学习

□ 监督学习

✓ (正/反例)在样本上的分布是确定的*。

□ 强化学习

✓ (状态/奖赏)的分布是策略依赖的(Policy Dependent!!!)

✓ 策略上小的变化都会导致返回值的巨大改变.

*目前有迁移学习在考虑非独立同分布的学习任务

MDP模型 – 小结

状态集合, $|S|=n.$ $s \in S$

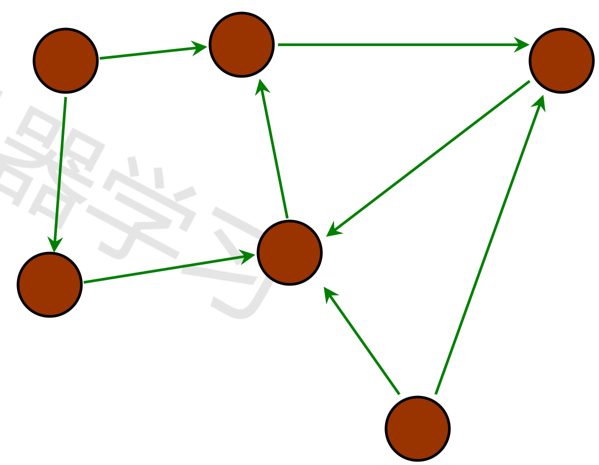
动作集合, $|A|=k.$ $a \in A$

转移函数 $\delta(s_1, a, s_2)$

即时奖赏函数 $R(s,a)$

策略 $\pi : S \rightarrow A$

折扣累计返回 $\sum_{i=0}^{\infty} \gamma^i r_i$



大 纲

起源

MDP模型

动态规划

强化学习

其他议题

动态规划

给定一个完全已知的MDP模型

- 策略评估(**P**olicy **E**valuation)
  - ✓ 给定一个策略π, 评估其返回值
- 最优控制(**O**ptimal **C**ontrol)
  - ✓ 寻找一个最优策略π^* (从任一状态出发，其返回值都为最大)

动态规划 – 值函数

- $V^{\pi}(s)$： 从s状态出发，采用π策略，所获得的期望返回值
- $Q^{\pi}(s,a)$：从s状态出发， 采用a动作，继而采用π策略，所获得的期望返回值
- 最优值函数$V^{*}(s)$ and $Q^{*}(s,a)$：采用最优策略π^{*}所获得的期望返回值

定理：策略π 为最优策略当且仅当，在每一个状态s

$$\boxed{V^{*}(s) = \max_{\pi} V^{\pi}(s)} \qquad \boxed{V^{\pi}(s) = \max_{a} Q^{\pi}(s,a)}$$

动态规划 – 策略评估

- Bellman等式(有折扣无限窗口)

  ✓ $V^{\pi}(s) = E_{s' \sim \pi(s)}\left[R\big(s, \pi(s)\big) + \gamma V^{\pi}(s')\right]$

- 重写

  ✓ $V^{\pi}(s) = E\left[R\big(s, \pi(s)\big)\right] + \gamma \sum_{s'} \delta(s, \pi(s), s')\, V^{\pi}(s')$

系统中所有值函数是以上公式
构成的公式组，需要进行线性规划求解

例 - 策略评估

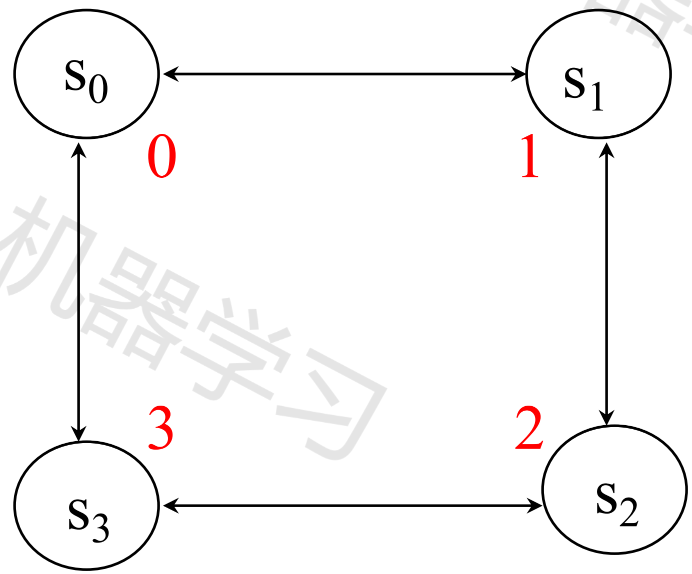
$A=\{+1, -1\}$

$\gamma = 1/2$

$\delta(s_i,a)= s_{i+a}$

π： 随机(一半概率选择+1或者-1动作)

$\forall a: R(s_i,a) = i$



$V^{\pi}(s_0) = 0 + \gamma\ [\pi(s_0,+1)V^{\pi}(s_1) + \pi(s_0,-1)\ V^{\pi}(s_3)]$

……

例 – 策略评估

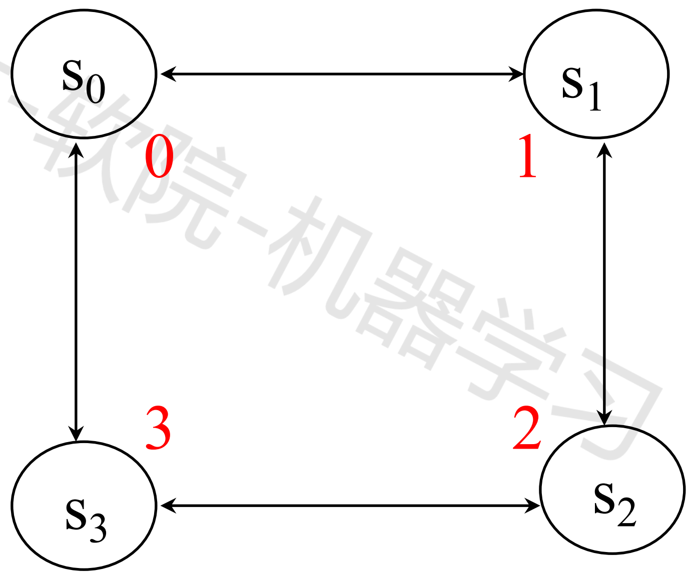
$A=\{+1,-1\}$

$\gamma = 1/2$

$\delta(s_i,a)= s_{i+a}$

π： 随机

$\forall a: R(s_i,a) = i$



$V^\pi(s_0) = 0 + (V^\pi(s_1) + V^\pi(s_3))/4$

……

$V^\pi(s_0) = 5/3$
$V^\pi(s_1) = 7/3$
$V^\pi(s_2) = 11/3$
$V^\pi(s_3) = 13/3$

动态规划 – 最优控制

- Bellman等式(有折扣无限窗口)
  - ✓ $V^{\pi}(s) = E_{s'\sim\pi(s)}\left[R\big(s,\pi(s)\big) + \gamma V^{\pi}(s')\right]$
- 重写
  - ✓ $V^{\pi}(s) = E\left[R\big(s,\pi(s)\big)\right] + \gamma\sum_{s'}\delta(s,\pi(s),s')\,V^{\pi}(s')$
- 状态-动作对值函数(对任意确定策略π)
  - ✓ $Q^{\pi}(s,a) = E[R(s,a)] + \gamma\sum_{s'}\delta(s,a,s')\,V^{\pi}(s')$
  - ✓ 其中，$V^{\pi}(s) = Q^{\pi}\big(s,\pi(s,a)\big)$

例 – 最优控制

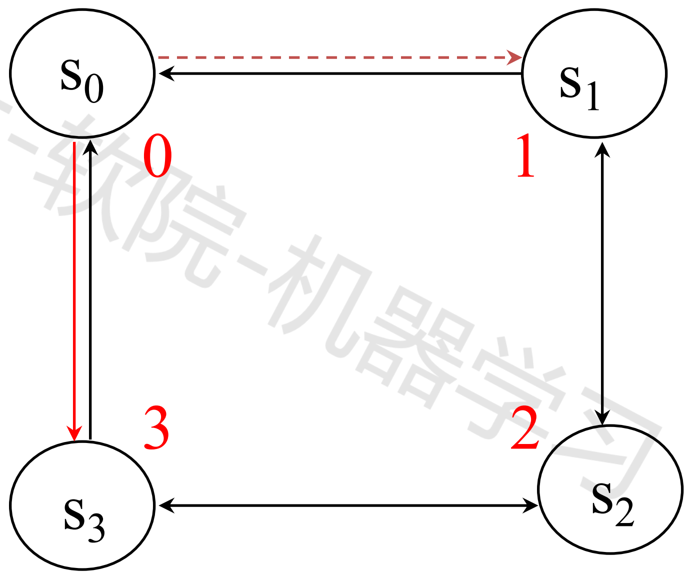
$A=\{+1,-1\}$

$\gamma = 1/2$

$\delta(s_i,a)= s_{i+a}$

π: 随机

$\forall a: R(s_i,a) = i$



$V^\pi(s_0) = 5/3$
$V^\pi(s_1) = 7/3$
$V^\pi(s_2) = 11/3$
$V^\pi(s_3) = 13/3$

$$\boxed{Q^\pi(s_0,+1) = 0 + \gamma V^\pi(s_1)}$$

$Q^\pi(s_0,+1) = 7/6$
$Q^\pi(s_0,-1) = 13/6$
$V^\pi(s_1) = 7/3$
$V^\pi(s_2) = 11/3$
$V^\pi(s_3) = 13/3$

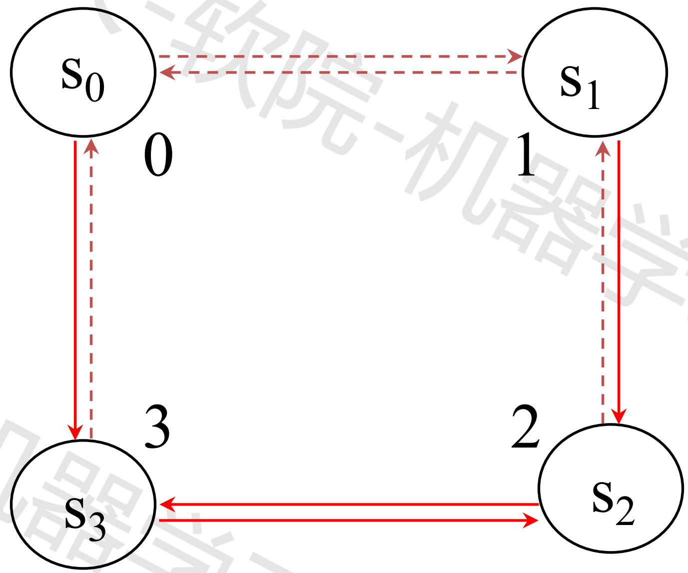
例 - 最优控制

$A=\{+1,-1\}$

$\gamma = 1/2$

$\delta(s_i,a)= s_{i+a}$

$R(s_i,a) = i$



π：根据状态-动作值函数进行修改

动态规划 - 最优控制

贪心策略

- ✓ $\pi(s) = argmax_a Q^{\pi}(s,a)$

ε-贪心策略

- ✓ 以1- ε概率选择， $\pi(s) = argmax_a Q^{\pi}(s,a)$
- ✓ 以ε概率选择其他动作

动态规划 - 计算最优策略

1. 线性规划
2. 值迭代方法

$$V^{i+1}(s) \leftarrow \max_a \{R(s,a) + \gamma \textstyle\sum_{s'} \delta(s,a,s')\, V^i(s')\}$$

3. 策略迭代方法

$$\pi_i(s) = \arg\max_a \{Q^{\pi_{i-1}}(s,a)\}$$

$$\pi_0 \rightarrow V^{\pi_0} \rightarrow \pi_1 \rightarrow V^{\pi_1} \rightarrow \cdots \pi^* \rightarrow V^* \rightarrow \pi^*$$

策略评估　　　　最优控制

大 纲

起源

MDP模型

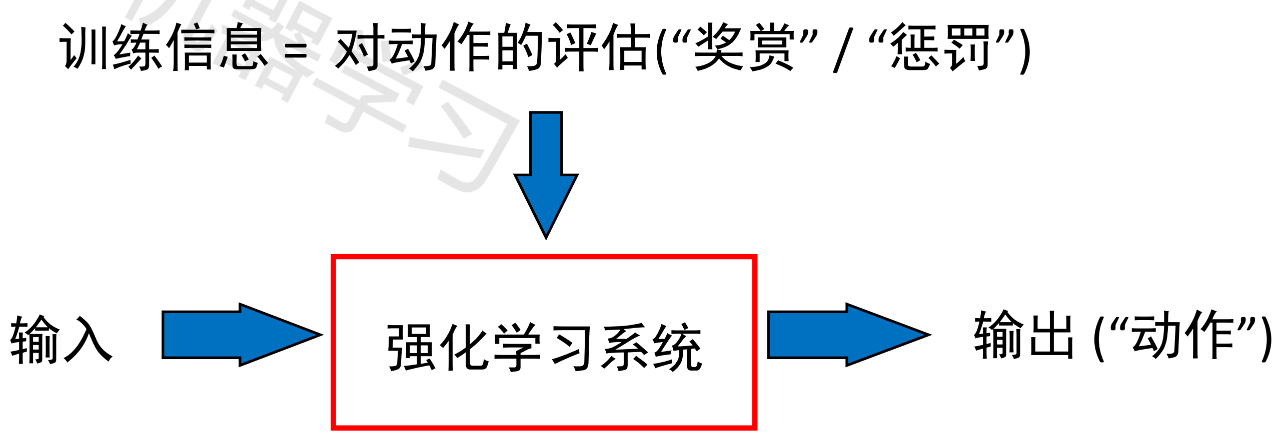
动态规划

强化学习

其他议题

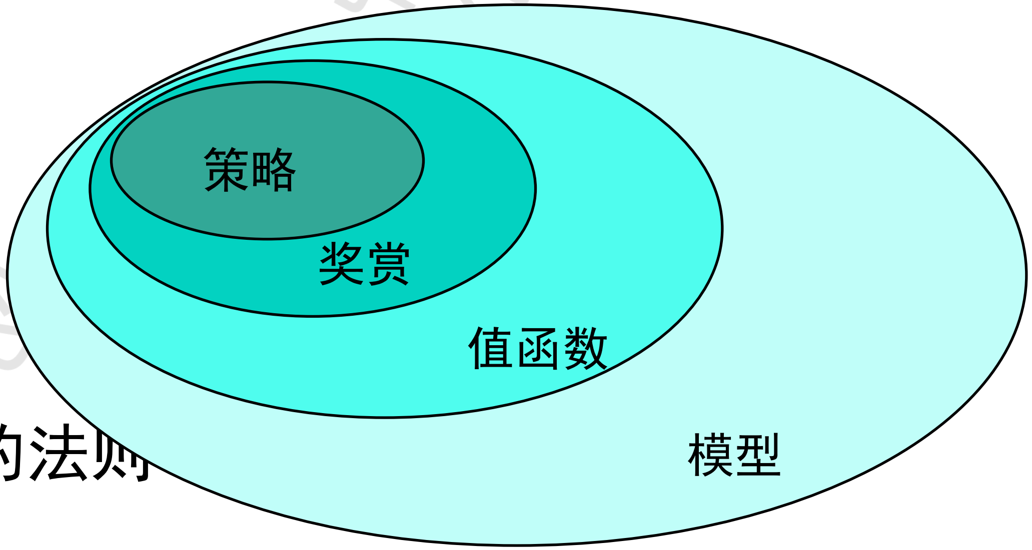
监督学习 VS 强化学习

- **监督学习**
  - ✓ (正/反例)在样本上的分布是确定的
- **强化学习**
  - ✓ (状态/奖赏)的分布是策略依赖的(policy dependent!!!)
  - ✓ 策略上小的变化都会导致返回值的巨大改变



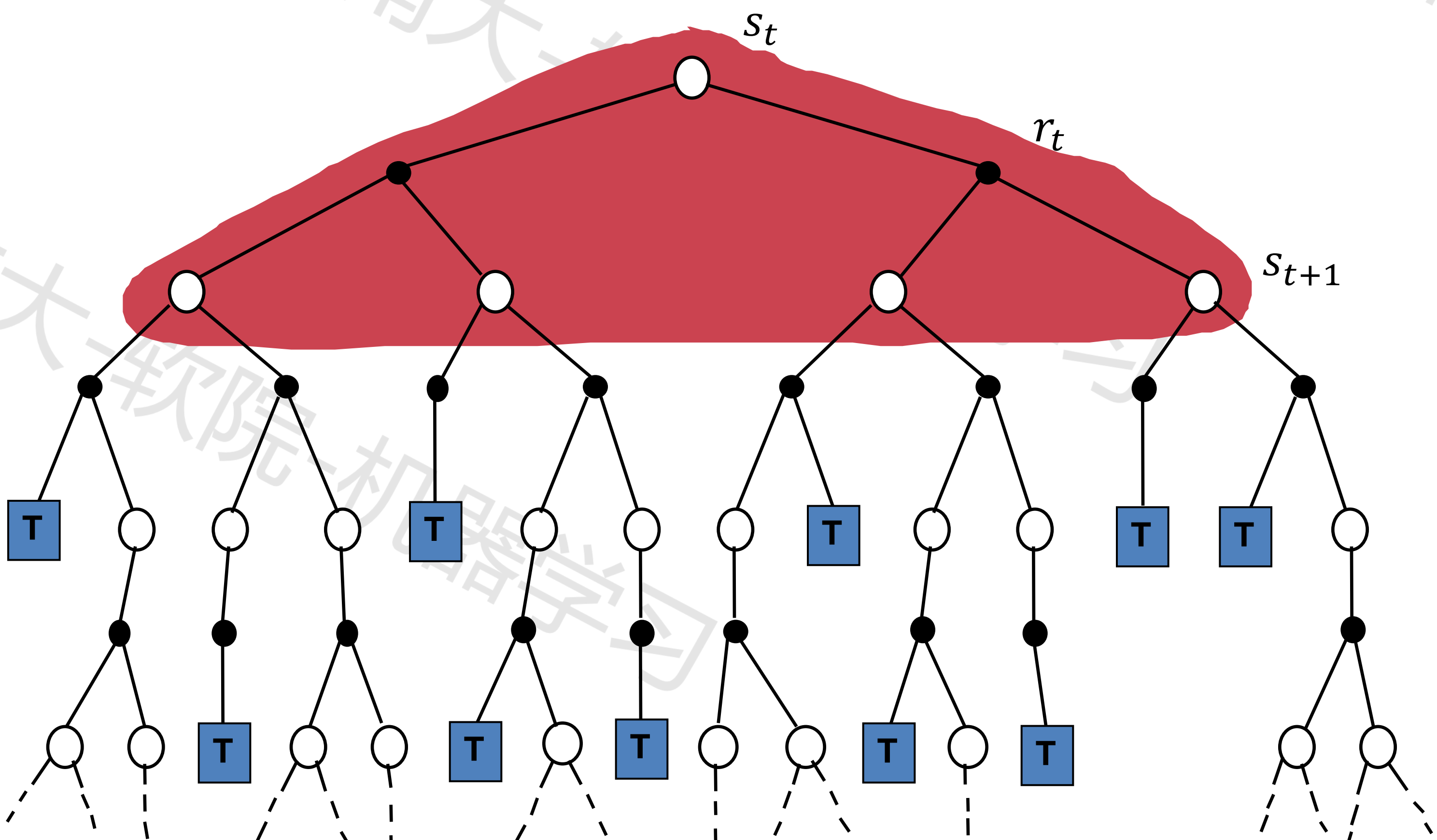
强化学习要素

- □ 策略
  - ✓ 选择动作的(确定/不确定)规则
- □ 奖赏/返回
  - ✓ 学习系统试图最大化的函数
- □ 值函数
  - ✓ 评估策略好坏的函数
- □ 模型
  - ✓ 环境(问题)演变遵循的法则



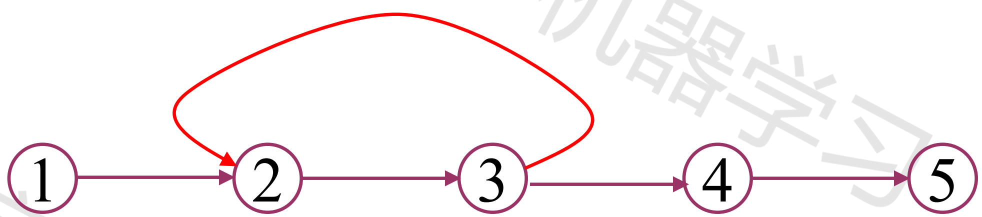
动态规划方法

$$V(s_t) \leftarrow E_\pi\{r_t + \gamma V(s_{t+1})\}$$



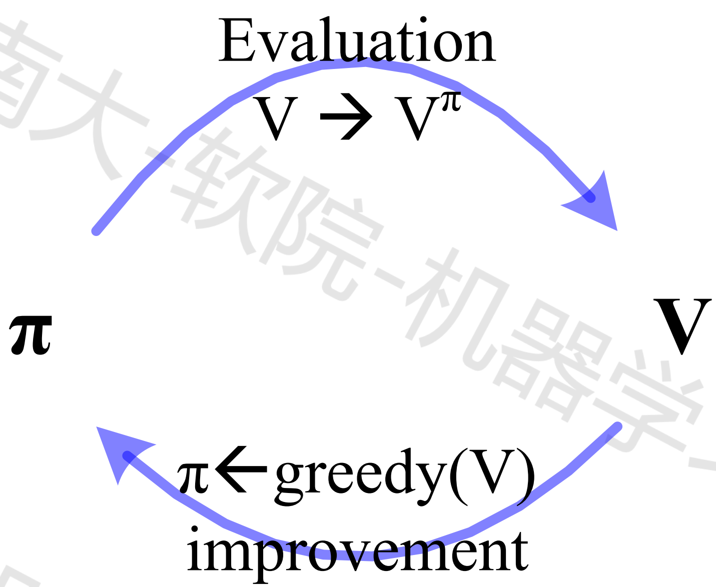
Monte Carlo策略评价

- **目标**：学习$V^{\pi}(s)$；
- **给定**：在访问状态s，采用策略下π，获得的若干经验；
- **思路**：在访问状态s 后，对所获得的返回，进行平均。



- Every-Visit MC:
  - ✓ 在一次经验中，对每次访问到的s都进行平均
- First-visit MC
  - ✓ 在一次经验中，只对首次访问到的s进行平均

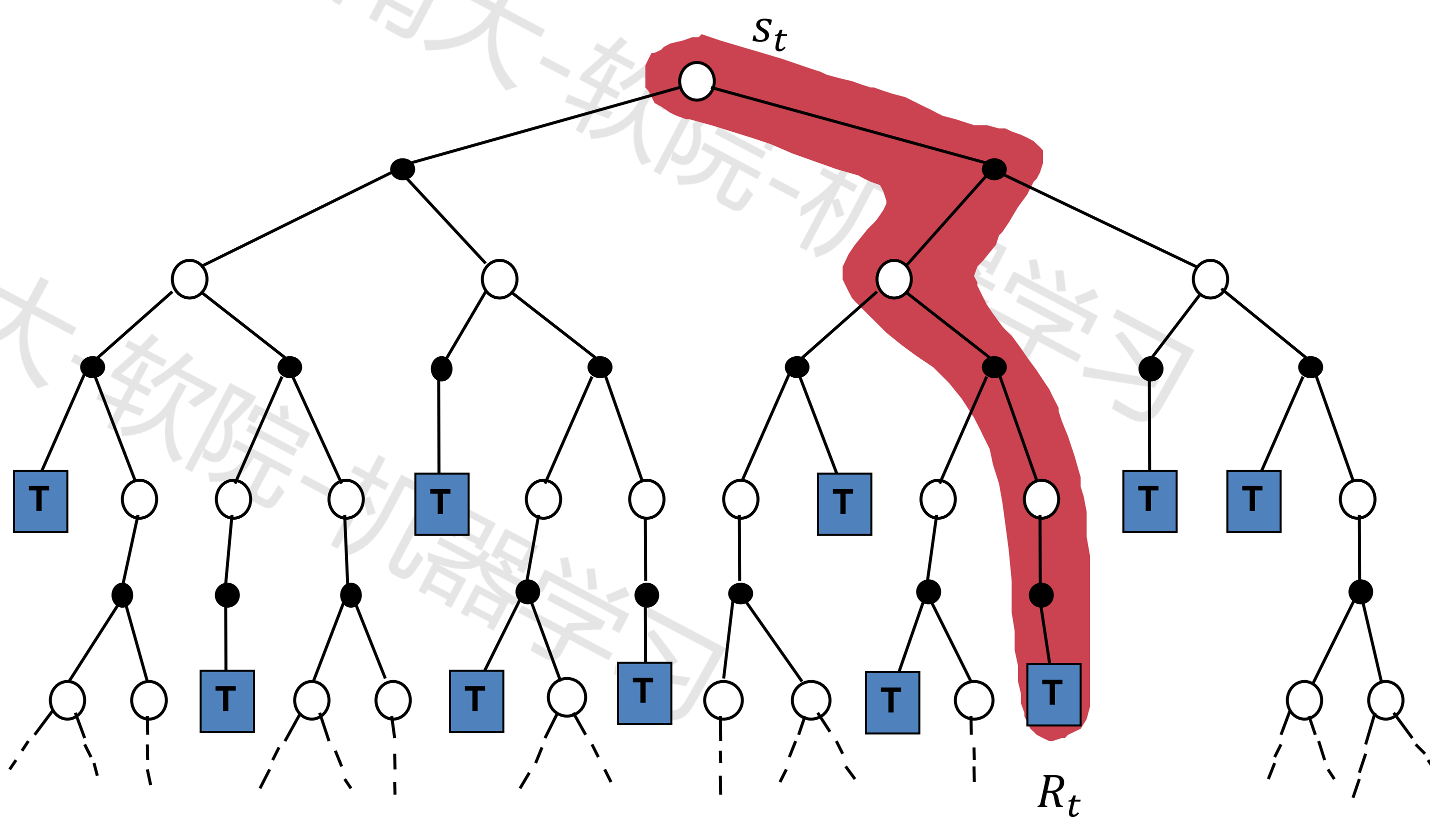
Monte Carlo最优控制



- MC策略迭代：使用MC方法对策略进行评估，计算值函数；
- MC策略修正：根据值函数(或者状态-动作对值函数)，采用贪心策略进行策略修正；

Monte Carlo方法

$$V(s_t) \leftarrow V(s_t) + \alpha\left[R_t - V(s_t)\right]$$

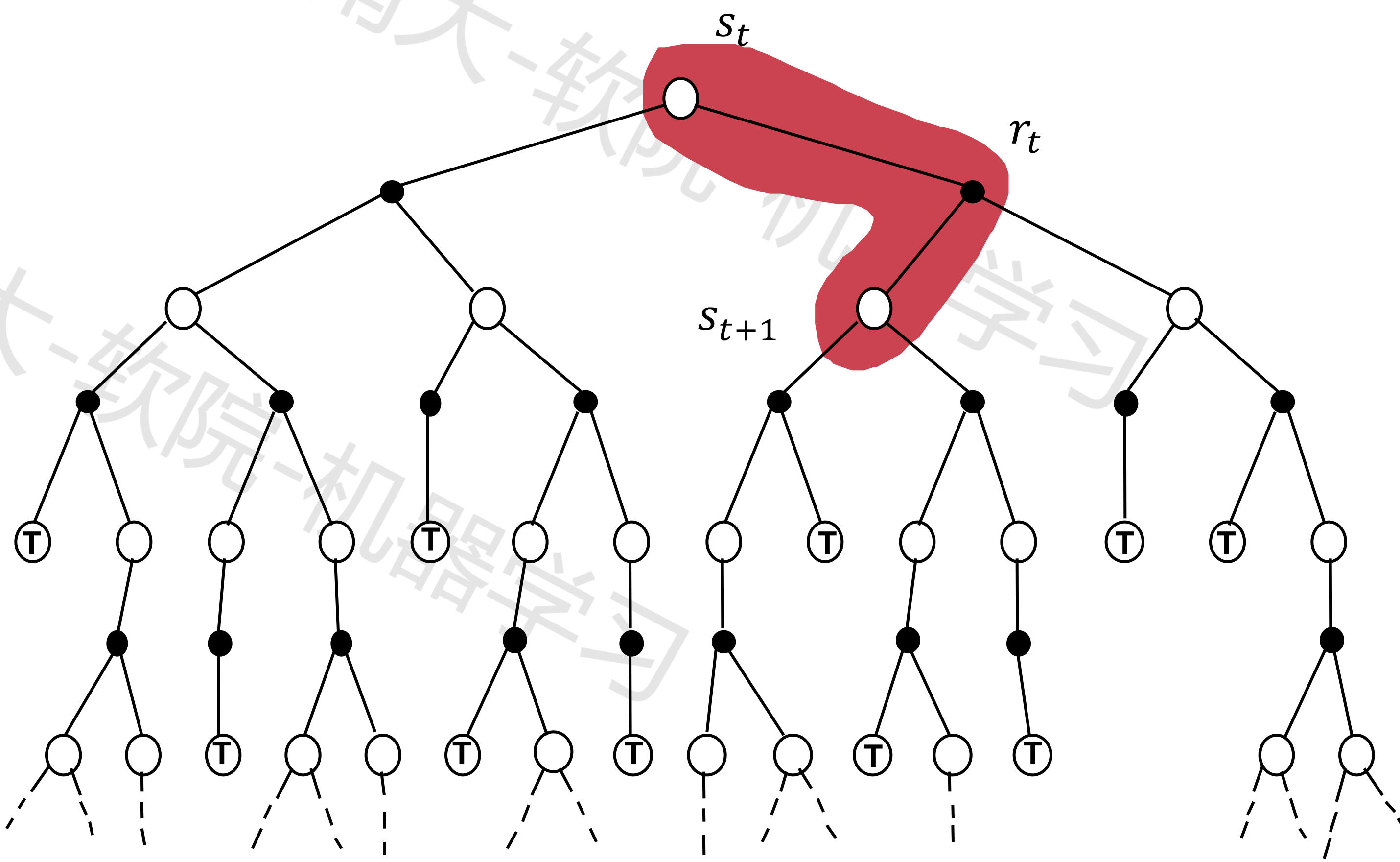


时差学习

- Every-Visit MC:
  - ✓ $V(s_t) = V(s_t) + \alpha[R_t - V(s_t)]$
  - ✓ 目标：在经过若干次平均后，得到真实的返回值
- 最简单的时间差分方法(Temporal **D**ifference)
  - ✓ $V(s_t) = V(s_t) + \alpha[r_t + \gamma V(s_{t+1}) - V(s_t)]$
  - ✓ 目标：在每一次经验后，都对返回值进行估计

时差方法

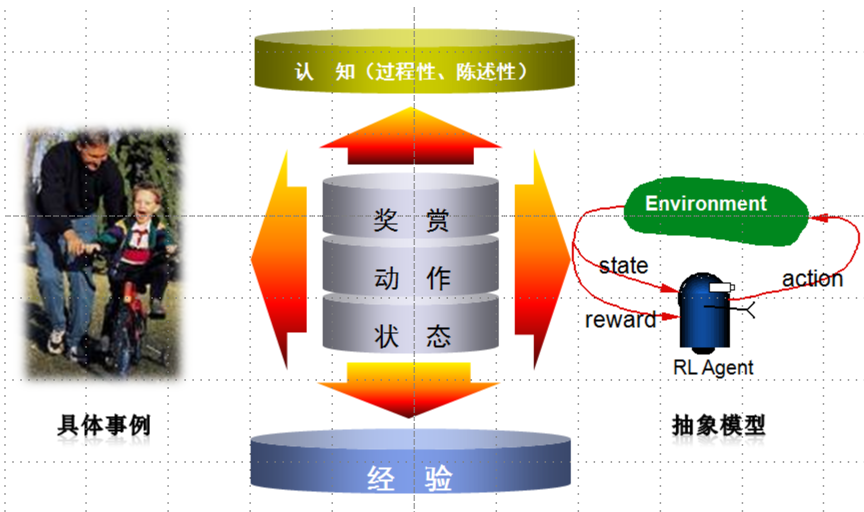
$$V(s_t) \leftarrow V(s_t) + \alpha\left[r_t + \gamma V(s_{t+1}) - V(s_t)\right]$$

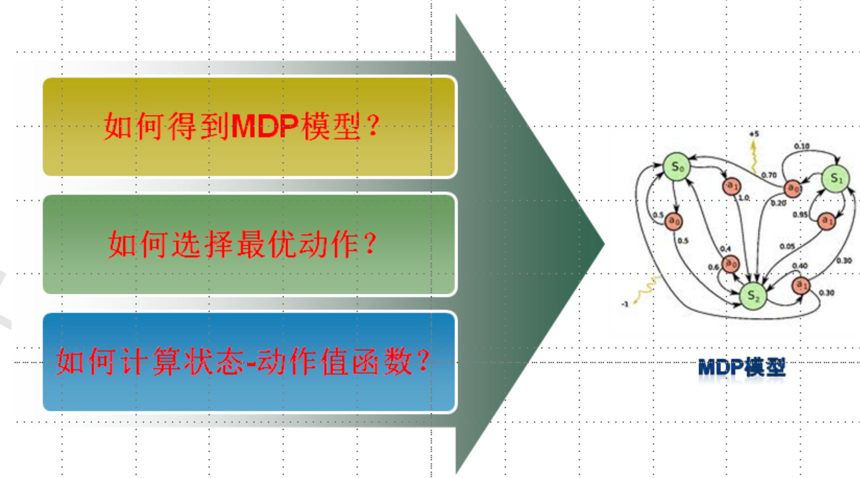


Bootstraps和Sampling

- Bootstraps
  - ✓ 通过一个估计值进行更新
  - ✓ 动态规划/时差学习中采用
  - ✓ 蒙特卡罗方法不采用
- 采样
  - ✓ 不通过估计值进行更新，而根据经验进行更新
  - ✓ 蒙特卡罗方法/时差学习中采用
  - ✓ 动态规划中不采用

强化学习算法





算法构造思路

- ✓ 根据先验得到初始认知（值函数）
- ✓ 根据认知选择动作（伴随一定的随机性）
- ✓ 获得经验
- ✓ 根据反馈，修改认知
- ✓ 根据延迟的反馈，回退修改历史认知

离策略 VS. 在策略

离策略(off-policy)和在策略(on-policy)

- Q-学习算法

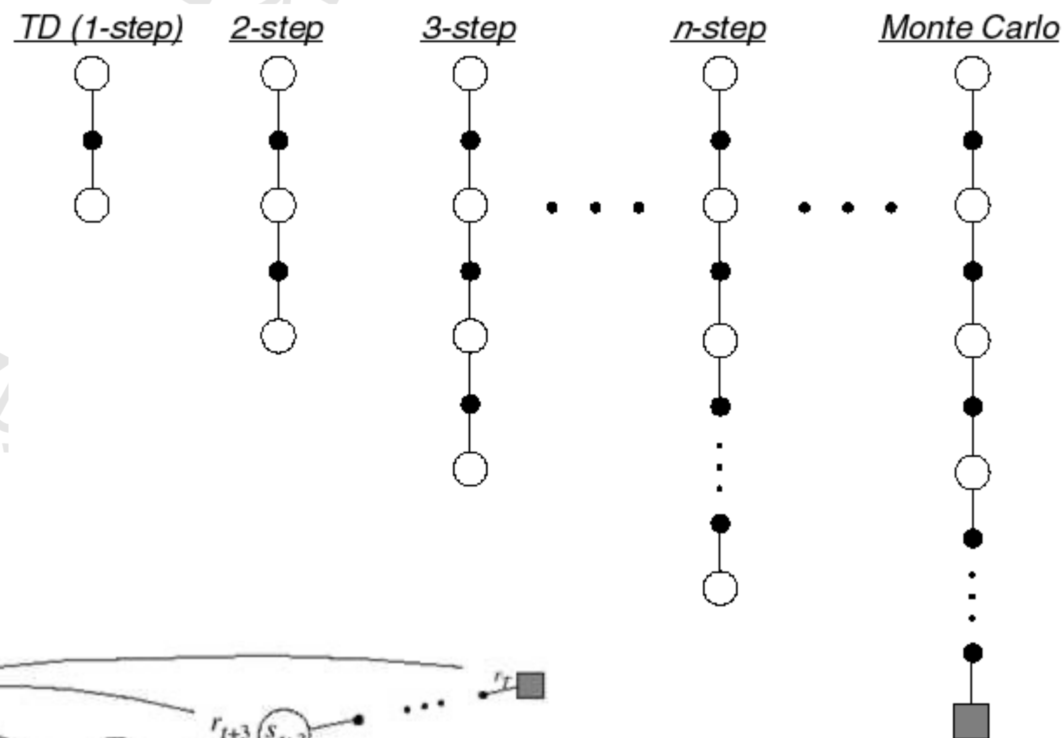
$$Q(s,a) \leftarrow Q(s,a) + \mu\left(r + \gamma \max_{a'} Q(s',a') - Q(s,a)\right)$$

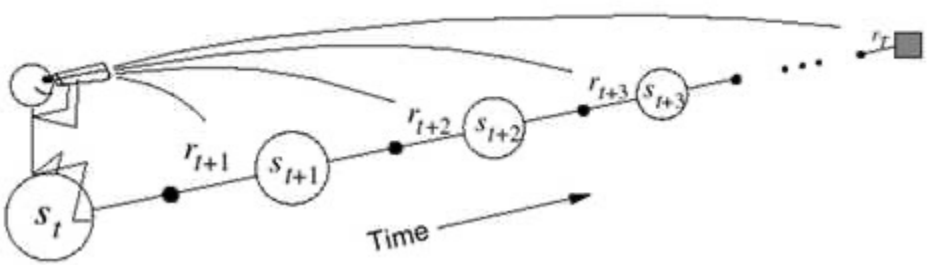
- SARSA算法

$$Q(s,a) \leftarrow Q(s,a) + \mu\left(r + \gamma Q(s',a') - Q(s,a)\right)$$

N步TD预测

- 思路：当做TD回退时，可以看到“更远的未来”；





N步TD预测

- Every-Visit MC:
  - ✓ $V(s_t) = V(s_t) + \alpha[R_t - V(s_t)]$
  - ✓ $R_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots + \gamma^T r_{t+T}$
- TD(0)
  - ✓ $V(s_t) = V(s_t) + \alpha[\underbrace{r_t + \gamma V(s_{t+1})}_{\widehat{R_t}} - V(s_t)]$
- TD(n)
  - ✓ 2步：$R^{(2)}{}_t = r_t + \gamma r_{t+1} + \gamma^2 V(s_{t+2})$
  - ✓ N步：$R^{(n)}{}_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots + \gamma^{n-1} r_{t+n-1} + \gamma^n V(s_{t+n})$

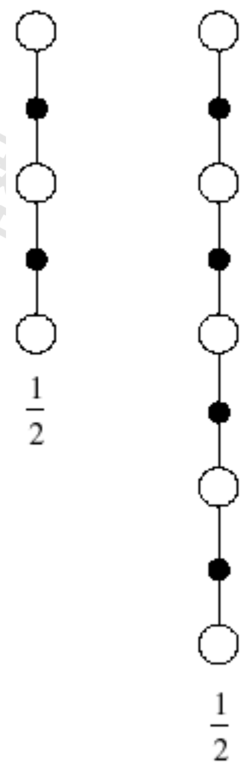
N步回退学习

N步回退

- ✓ $\Delta V(s_t) = \alpha\left[R^{(n)}{}_t - V(s_t)\right]$

- ✓ $R^{avg}{}_t = \frac{1}{2}R^{(n)}{}_2 + \frac{1}{2}R^{(n)}{}_4$

两次多步回退的平均

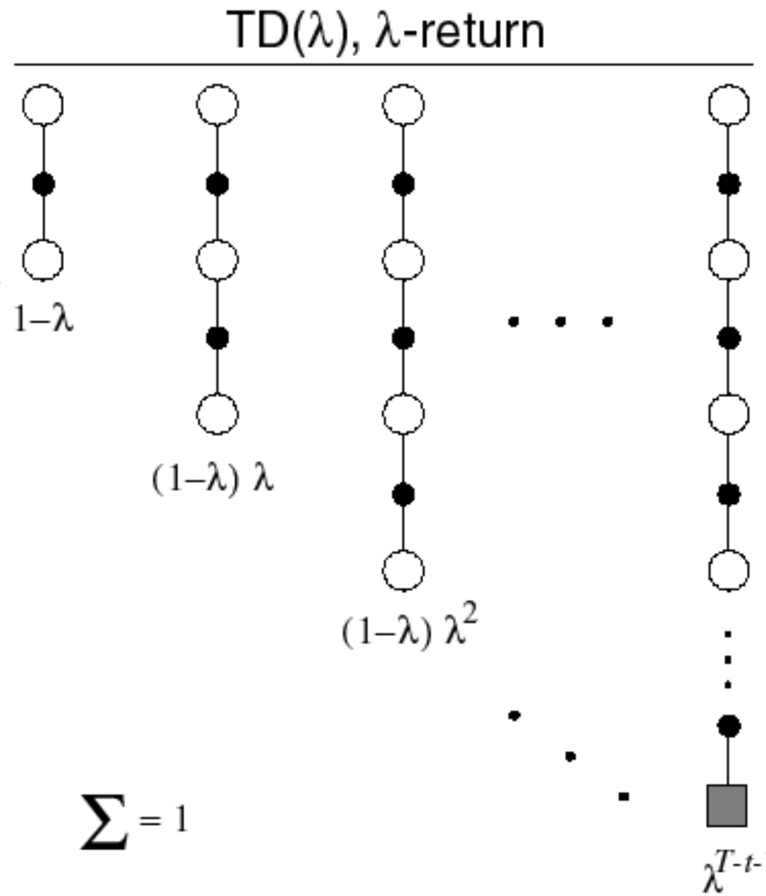


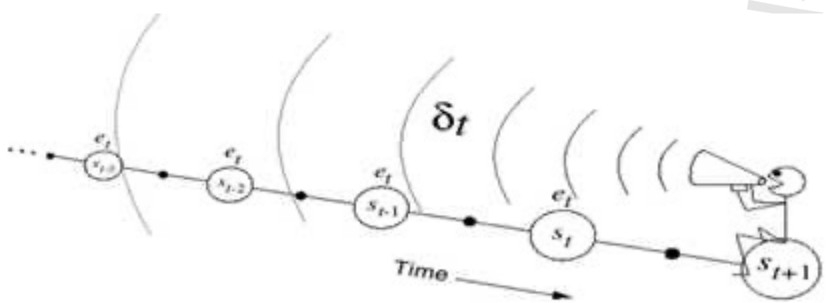
N步回退学习

λ-返回

- ✓ $R^{\lambda}{}_{t} = (1-\lambda)\sum_{n=1}^{\infty}\lambda^{n-1} R^{(n)}{}_{t}$

- ✓ $\Delta V(s_t) = \alpha\left[R^{(\lambda)}{}_{t} - V(s_t)\right]$





TD(λ)算法

1. 初始化V(s), e(s)=0
2. 对每一个episode，重复

   初始化s

   对episode中的每一步

      根据ε-贪心策略选择动作a

      执行动作a，获得r和s'

      $\Delta \leftarrow r + \gamma V(s') - V(s)$

      $e(s) \leftarrow e(s)$+1

      对于所有s

         $V(s) \leftarrow V(s)$+$\alpha \Delta e(s)$

         $e(s) \leftarrow \gamma\lambda e(s)$

     $s \leftarrow s'$

   直到s为终止状态

大 纲

起源

MDP模型

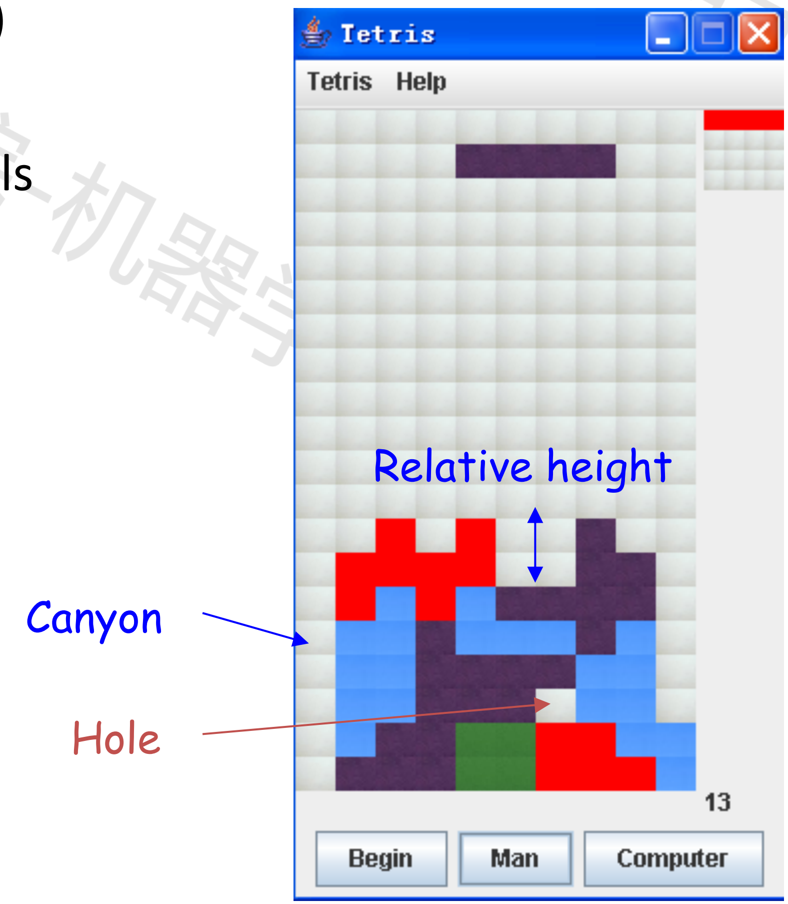
动态规划

强化学习

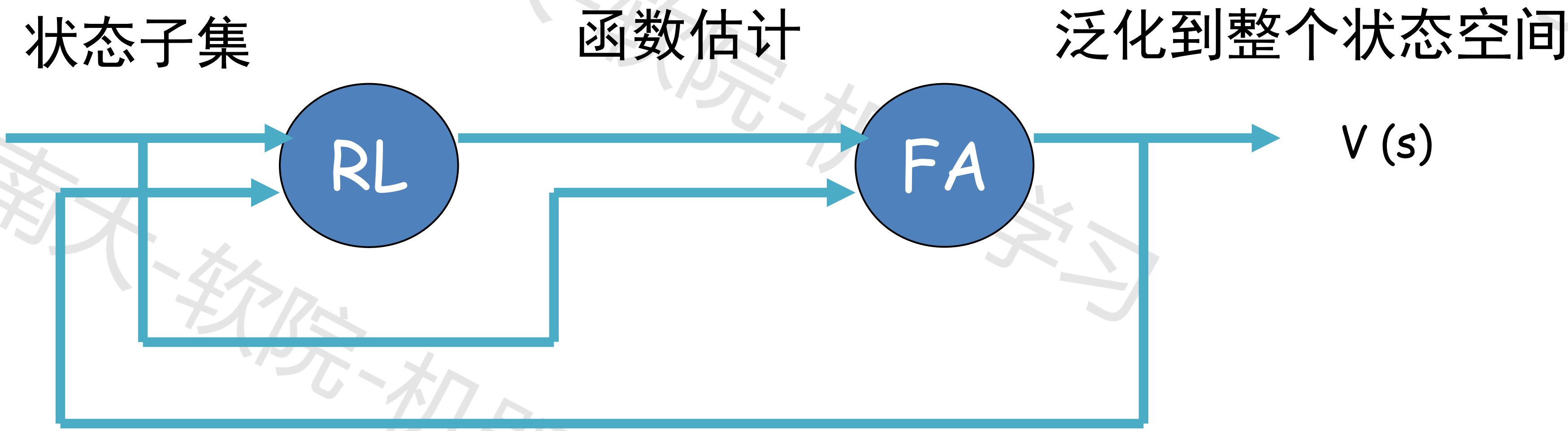
其他议题

关系强化学习

- 属性
  - Height of wall (max, avg, min)
  - Number of Holes
  - Height difference adjacent cols
  - Canyon (width, height)
  - ...
- 宏动作
  - Fits
  - Increasesheight, ...
  - Number of deleted lines
  - ...

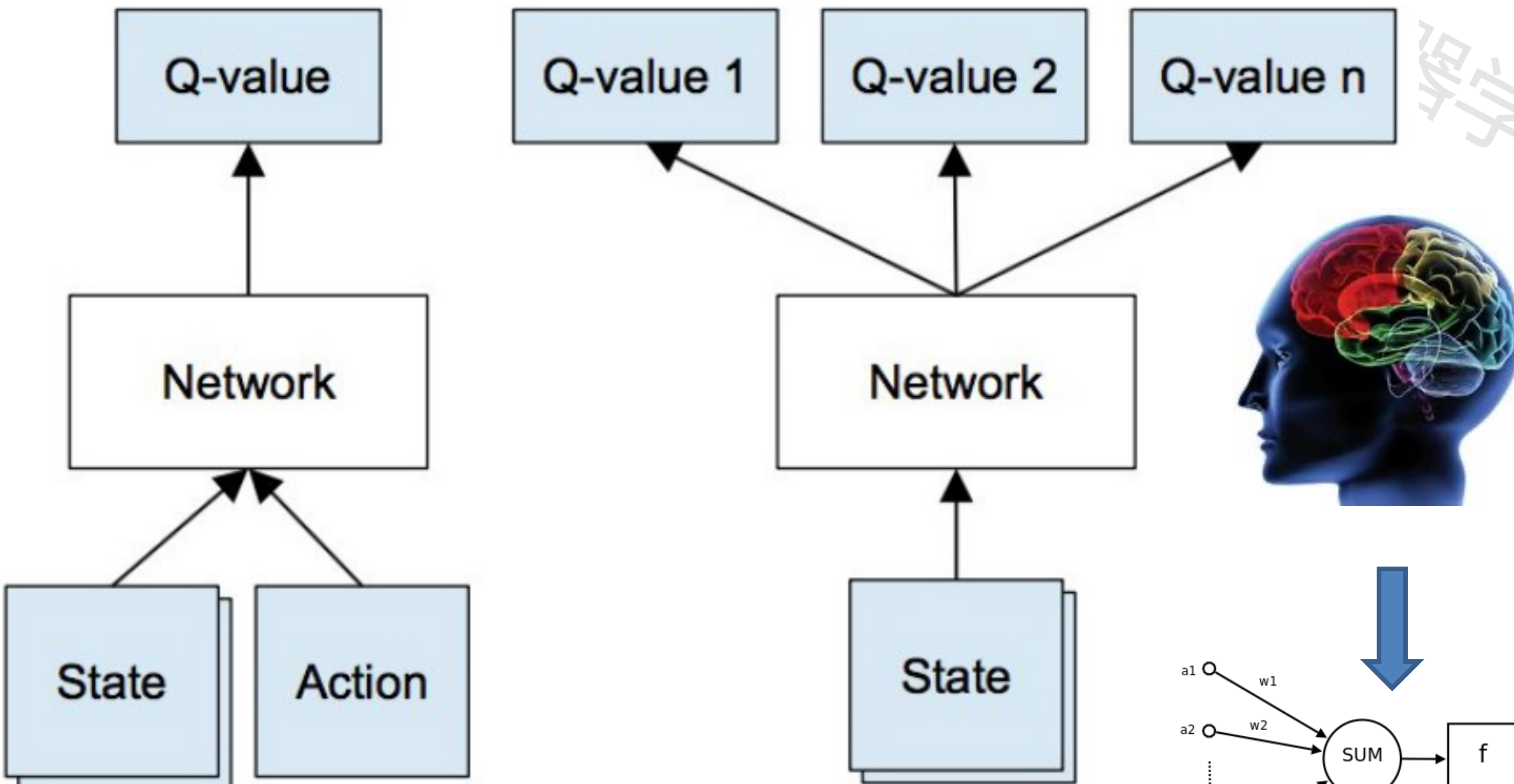


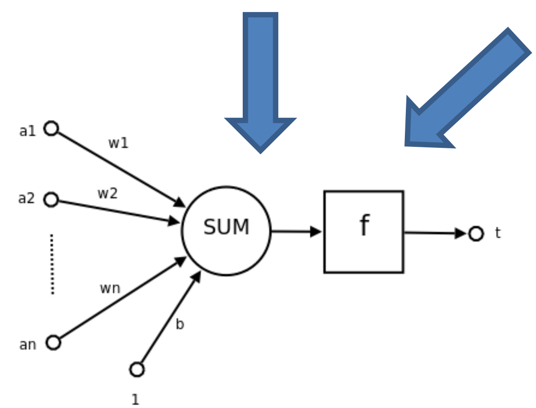
强化学习中的值函数估计



$$V_0, M(V_0), \Gamma(M(V_0)), M\left(\Gamma(M(V_0))\right), \Gamma\left(M\left(\Gamma(M(V_0))\right)\right), \ldots$$

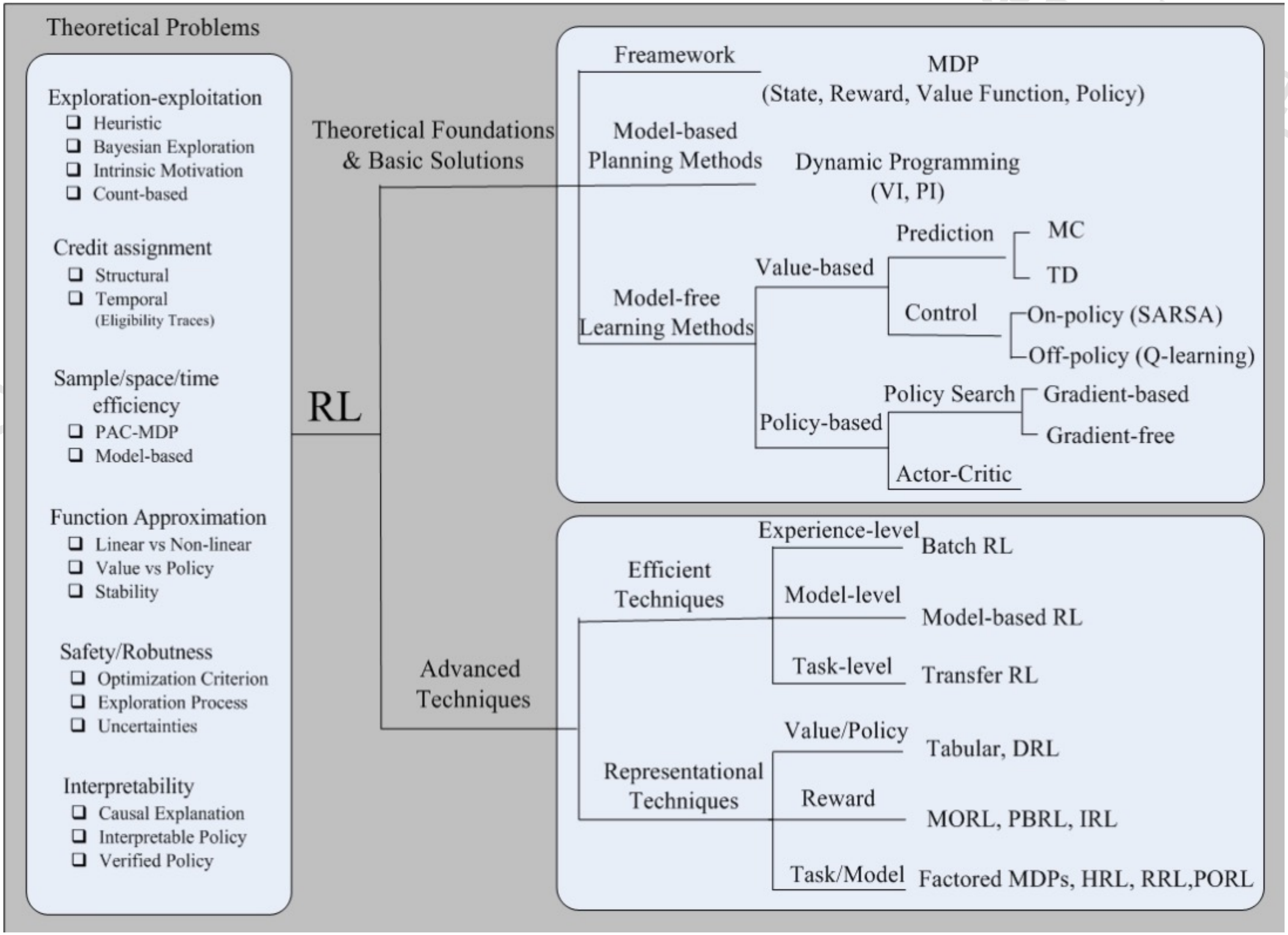
深度强化学习





DQN结构示意图

强化学习总结


Theoretical Problems
Exploration-exploitation
Heuristic
Bayesian Exploration
Intrinsic Motivation
Count-based
Credit assignment
Structural
Temporal
(Eligibility Traces)
Sample/space/time efficiency
PAC-MDP
Model-based
Function Approximation
Linear vs Non-linear
Value vs Policy
Stability
Safety/Robustness
Optimization Criterion
Exploration Process
Uncertainties
Interpretability
Causal Explanation
Interpretable Policy
Verified Policy
RL
Theoretical Foundations & Basic Solutions
Freamework
MDP
(State, Reward, Value Function, Policy)
Model-based Planning Methods
Dynamic Programming
(VI, PI)
Model-free Learning Methods
Value-based
Prediction
MC
TD
Control
On-policy (SARSA)
Off-policy (Q-learning)
Policy-based
Policy Search
Gradient-based
Gradient-free
Actor-Critic
Advanced Techniques
Efficient Techniques
Experience-level
Batch RL
Model-level
Model-based RL
Task-level
Transfer RL
Representational Techniques
Value/Policy
Tabular, DRL
Reward
MORL, PBRL, IRL
Task/Model
Factored MDPs, HRL, RRL,PORL

思考和讨论

1. 学习Q, SARSA, TD算法
2. 回报函数和值函数
3. 动态规划和蒙特卡罗采样的区别
4. 学习DQN

谢 谢 ！