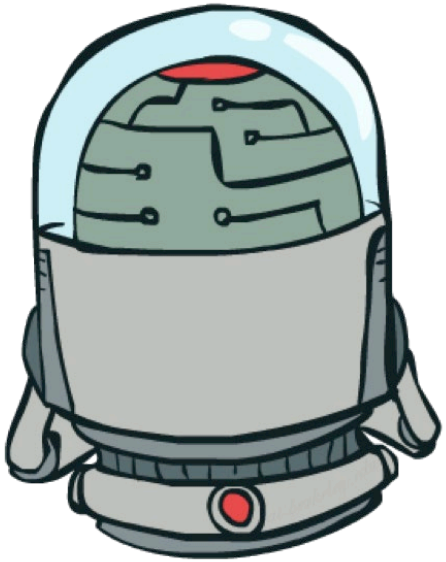
强化学习

Reinforcement Learning

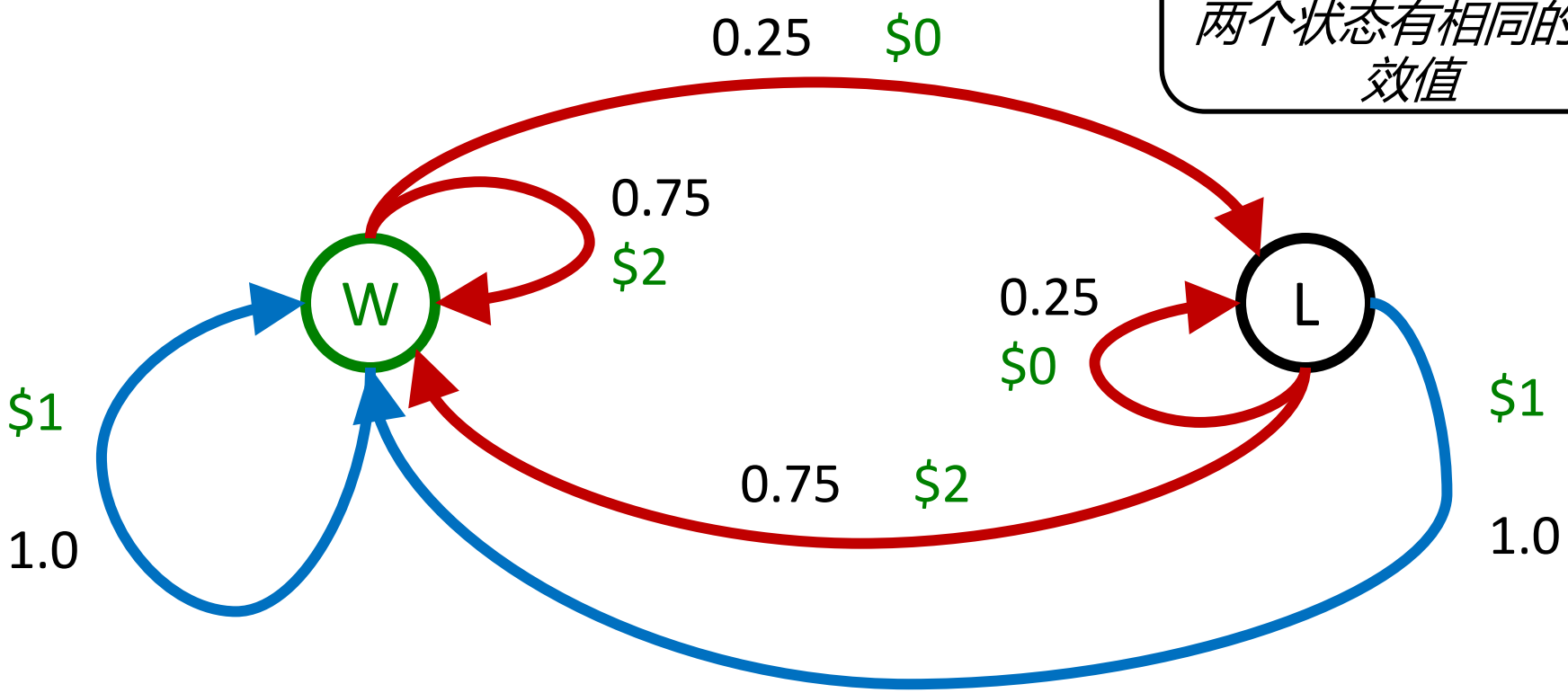
两个老虎机



两个老虎机 MDP

- 行动：*Blue*, *Red*
- 状态：Win, Lose

假设：奖赏没有折扣
玩 100 次
两个状态有相同的功效值

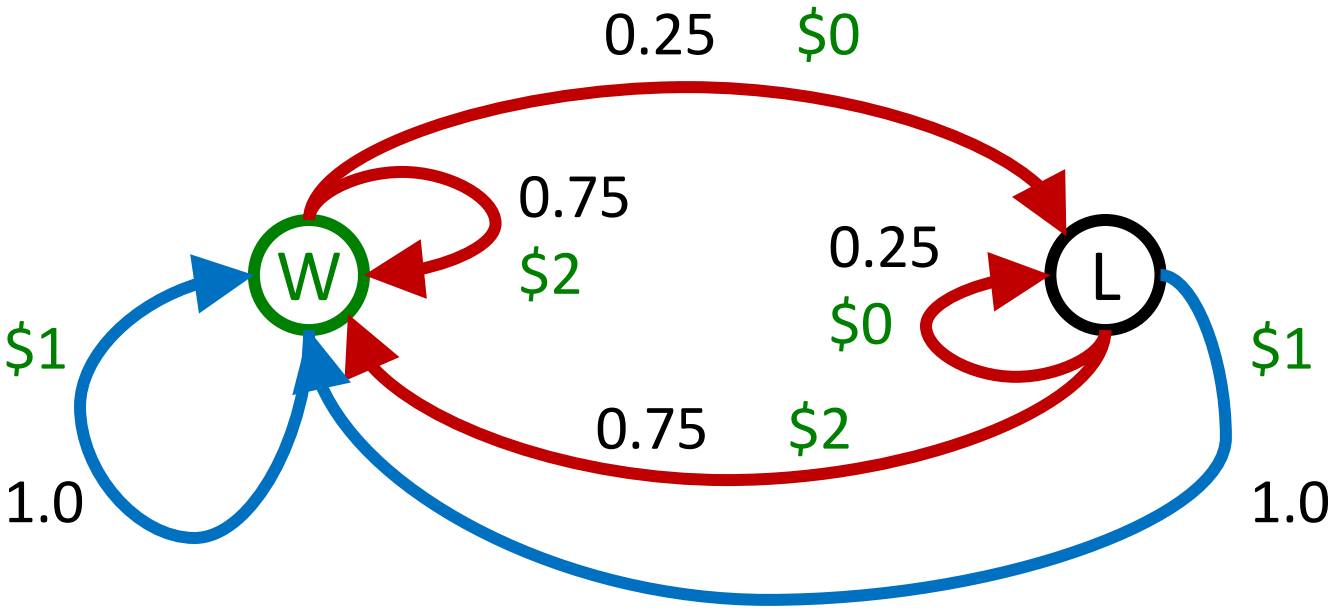


线下规划

- 求解 MDPs 是线下规划的过程
  - 通过计算确定所有的数量值
  - 需要知道MDP问题的细节
  - 实际上你并没有玩这个游戏!

假设：奖赏没有折扣
玩100 次
两个状态有相同的功效值

Value	
Play Red	150
Play Blue	100



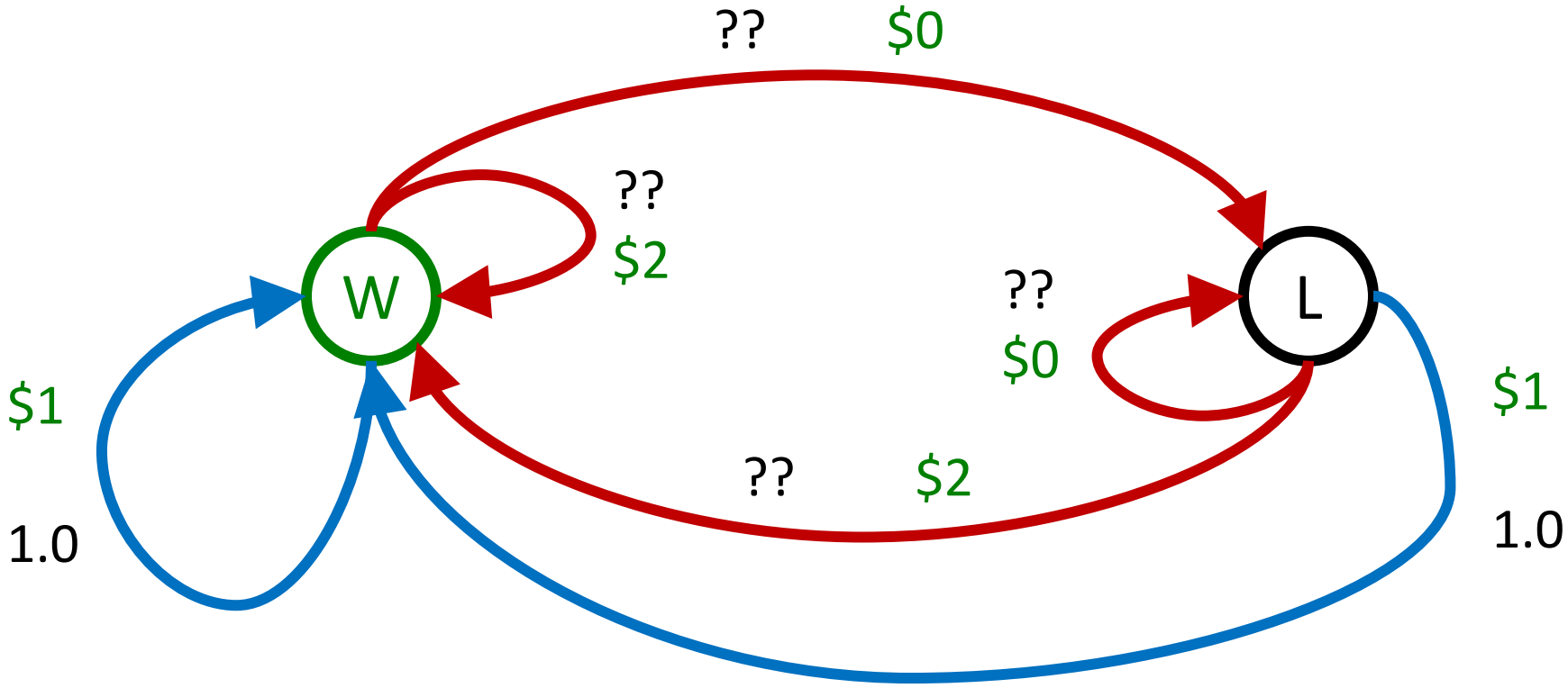
实际地玩一下！

$2 $2 $0 $2 $2

$2 $2 $0 $0 $0

在线规划

- 规则改变了！　红色机器的中奖概率和以前不一样了（未知的）

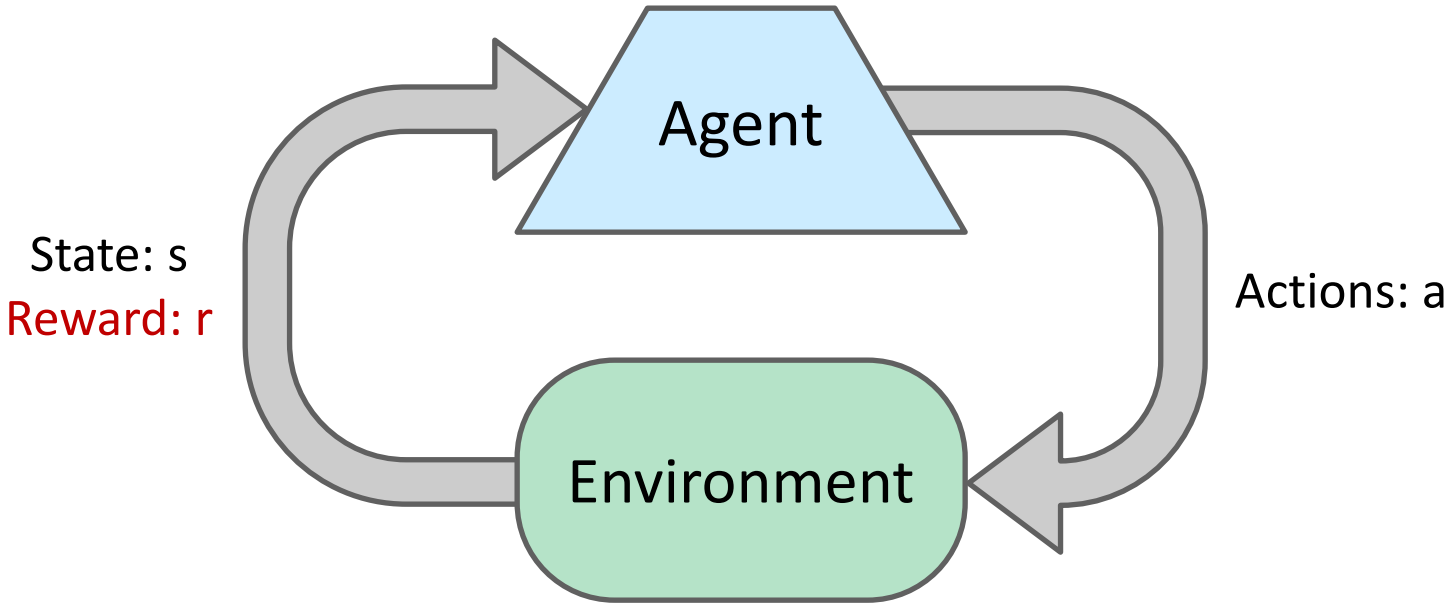


Let's Play!

$0 $0 $0 $2 $0

$2 $0 $0 $0 $0

强化学习



- 基本思想:
  - 通过 奖赏值 的形式来获得反馈
  - 智能体的功效值是由 奖赏函数(reward function)来定义的
  - 必须(通过学习) 行动，以获得最大化的期望奖赏值
  - 这里的学习是基于观察到的样本结果!

Example: Learning to Walk

Initial

A Learning Trial

After Learning [1K Trials]

[Kohl and Stone, ICRA 2004]

Example: Learning to Walk

Initial

[Kohl and Stone, ICRA 2004]

[Video: AIBO WALK – initial]

Example: Learning to Walk

Training

[Kohl and Stone, ICRA 2004]

[Video: AIBO WALK – training]

Example: Learning to Walk

Finished

[Kohl and Stone, ICRA 2004]

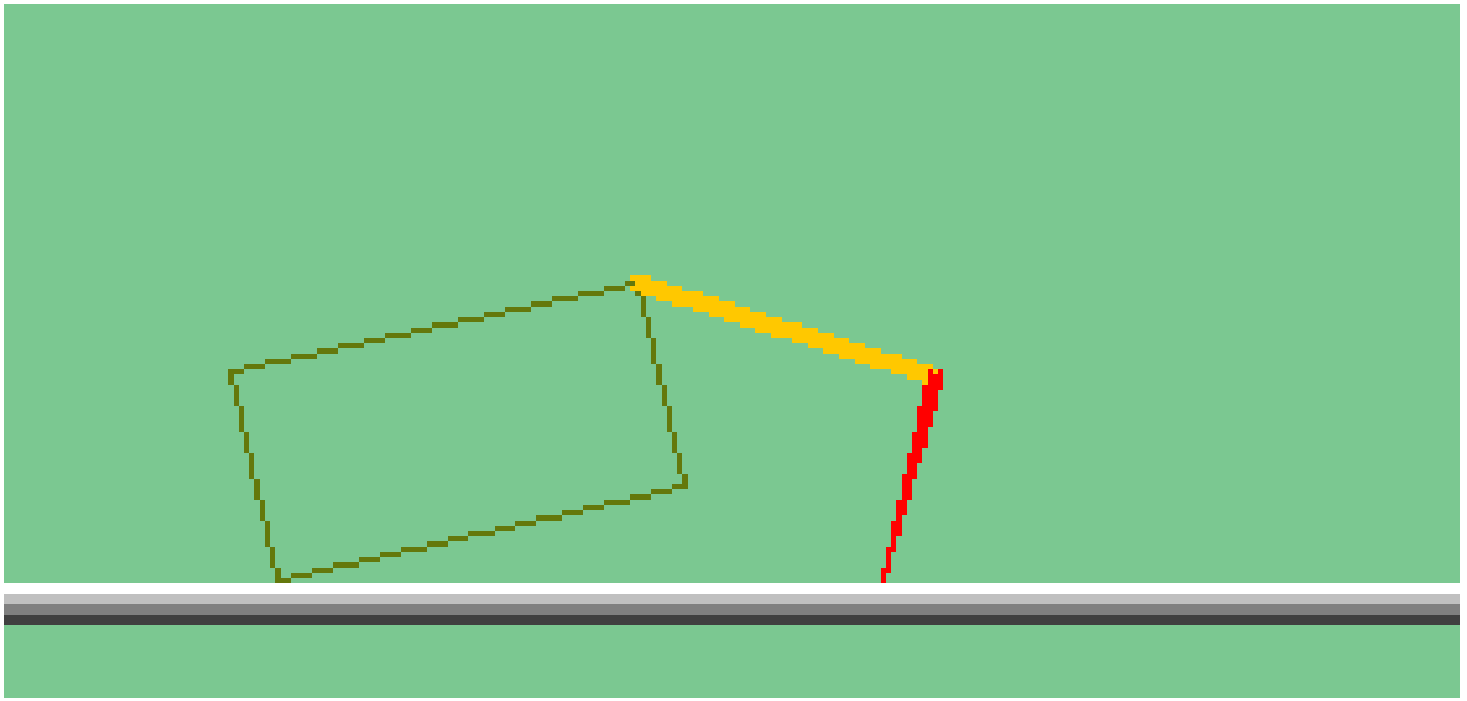
[Video: AIBO WALK – finished]

Example: Toddler Robot

[Tedrake, Zhang and Seung, 2005]

[Video: TODDLER – 40s]

The Crawler!



[Demo: Crawler Bot (L10D1)] [You, in Project 3]

Video of Demo Crawler Bot

小结

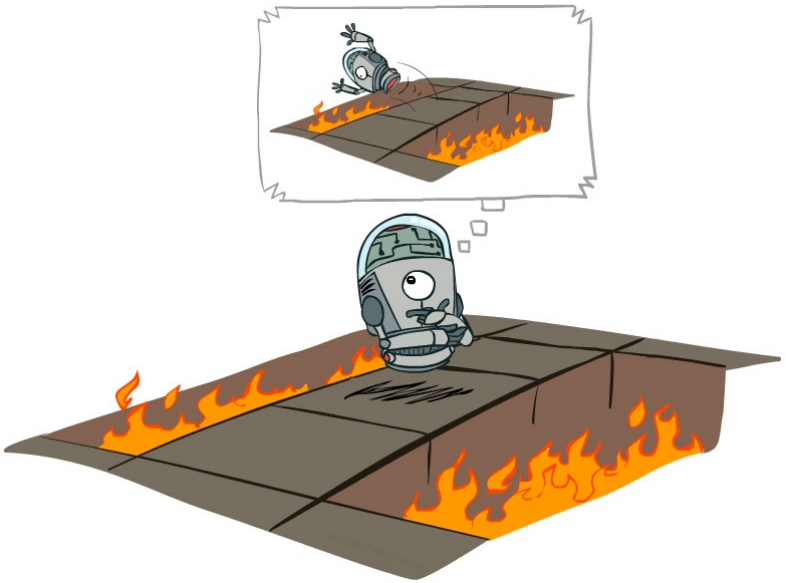
- 刚才做的不是事先规划，而是在实践中学习！
  - 这实际上就是 强化学习 (reinforcement learning)
  - 有这样一个 MDP问题，但是你不知道它的细节所以不能预先计算求解
  - 需要实际去操作从而搞明白它是如何工作的
- 强化学习的重要思想：
  - 探索：尝试不同的行动，从而获得相关的信息
  - 利用：最终，你需要利用所获得的知识
  - 遗憾误差(Regret)：即使很聪明地学习，也会犯错误
  - 采样：由于存在概率因素，需要重复的尝试
  - 难点：这种学习比求解一个已知的MDP还要难

强化学习

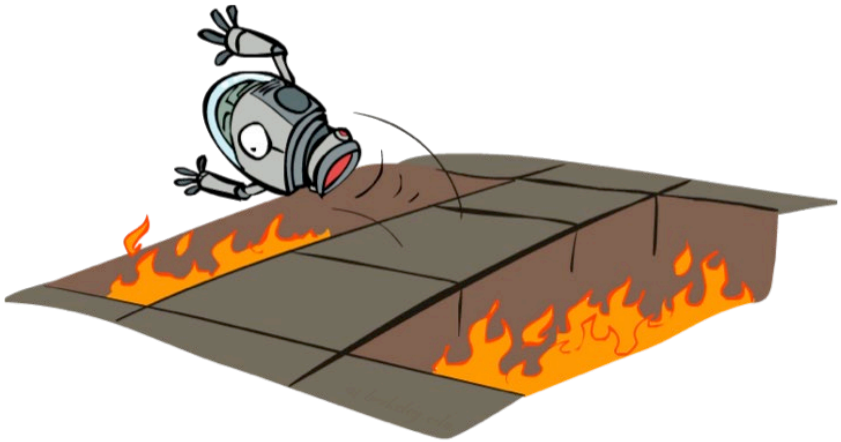
- 依然假设是一个马可夫决策过程(MDP):
  - 一个 状态集合 s ∈ S
  - 一个 行动集合 A
  - 一个 转移模型 T(s,a,s')
  - 一个 奖赏函数 R(s,a,s')
- 仍旧是搜寻一个策略 π(s)

- 新的地方: 不知道 T 或 R
  - 不知道哪些状态是好的，或那些行动会带来什么
  - 必须在实践中尝试行动和状态，并从中进行学习

Offline (MDPs) vs. Online (RL)

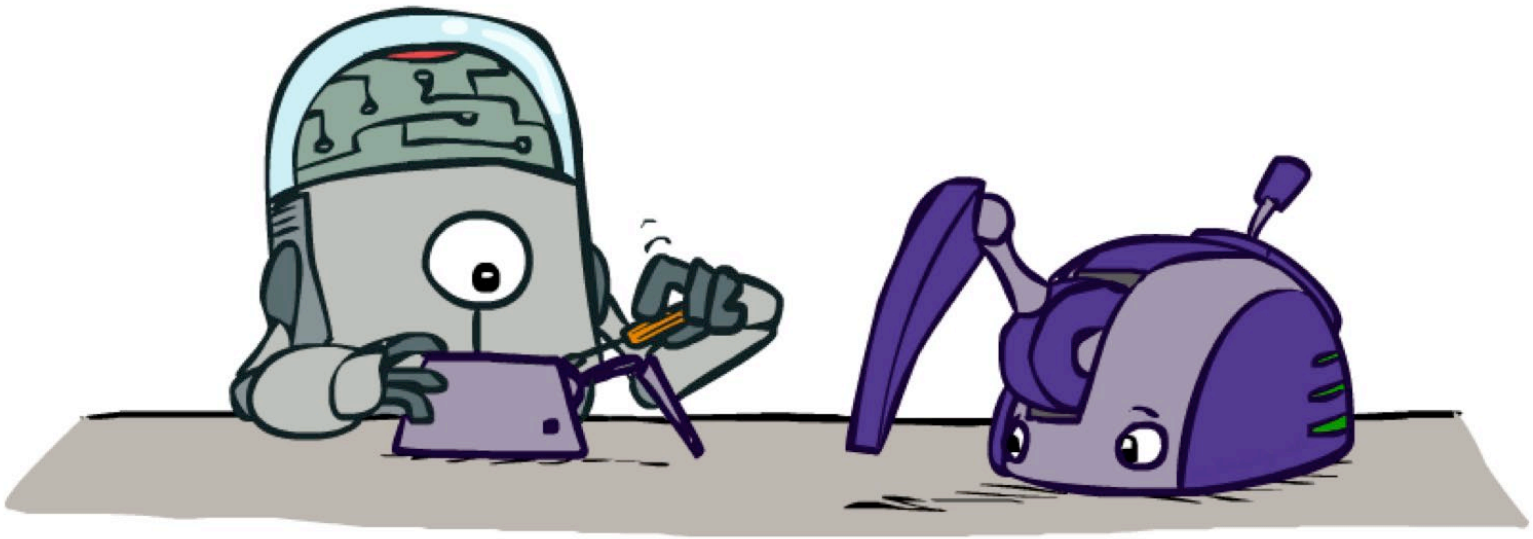


Offline Solution



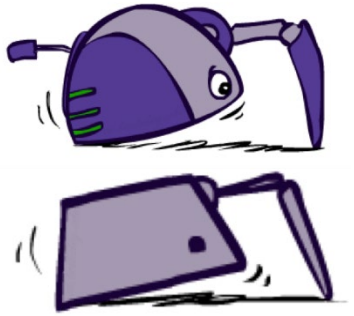
Online Learning

基于模型的学习
Model-Based Learning



基于模型的学习
Model-Based Learning

- 思想:
  - 基于经验学习一个近似的模型
  - 利用学好的模型去求解当前的MDP问题(可以应用状态值迭代方法等)

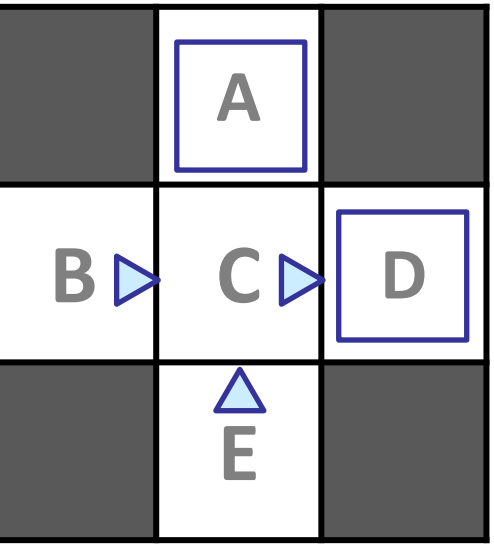


- 步骤 1: 学习经验化的 MDP 模型
  - 对于每对 s,a，数出结果是 s’的数量
  - 归一化后给出一个估计 $\hat{T}(s,a,s')$
  - 当我们经历（s, a, s’）后，发现奖赏值 $\hat{R}(s,a,s')$

- 步骤 2: 求解基于学到的模型的 MDP
  - 例如，使用状态值迭代法，就像之前一样进行线下求解

举例：基于模型的学习

输入策略 π



Assume: γ = 1

观察到的(训练过程)

Episode 1

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 2

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 3

E, north, C, -1
C, east, D, -1
D, exit, x, +10

Episode 4

E, north, C, -1
C, east, A, -1
A, exit, x, -10

学到的模型

$\hat{T}(s, a, s')$

T(B, east, C) = 1.00
T(C, east, D) = 0.75
T(C, east, A) = 0.25
…

$\hat{R}(s, a, s')$

R(B, east, C) = -1
R(C, east, D) = -1
R(D, exit, x) = +10
…

举例：年龄期望值

目标：计算班上同学的年龄期值

Known P(A)

$$E[A] = \sum_a P(a) \cdot a \qquad = 0.35 \times 20 + \ldots$$

如果事先没有 P(A)，那么可以进行采样 $[a_1, a_2, \cdots a_N]$

Unknown P(A): "Model Based"

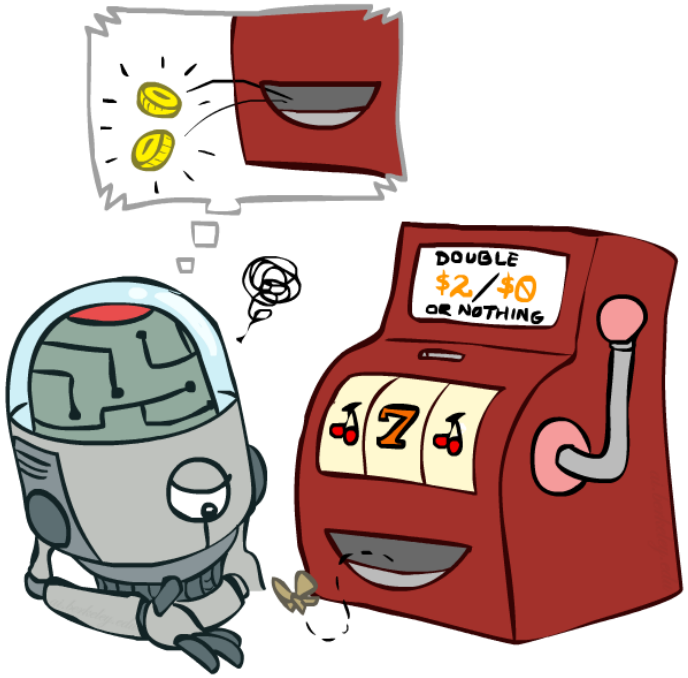
$$\hat{P}(a) = \frac{\text{num}(a)}{N}$$

$$E[A] \approx \sum_a \hat{P}(a) \cdot a$$

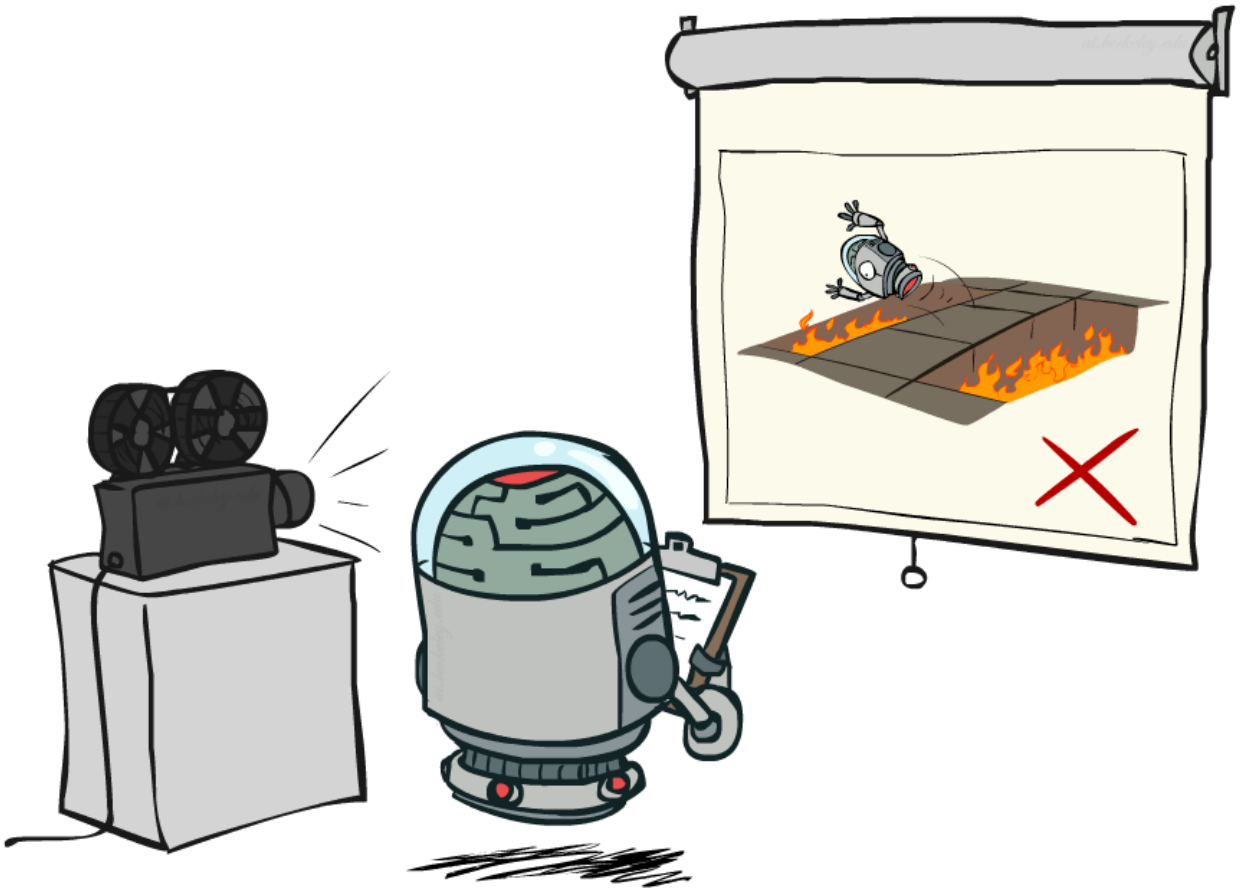
Unknown P(A): "Model Free"

$$E[A] \approx \frac{1}{N} \sum_i a_i$$

不基于模型的学习
Model-Free Learning

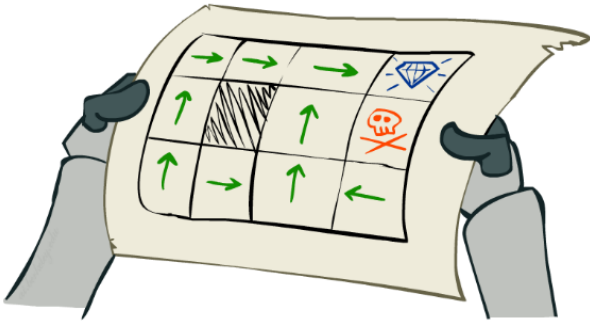


被动强化学习
Passive Reinforcement Learning



被动(passive)强化学习

- 简单的任务：策略评价 policy evaluation
  - 输入：一个跟定的策略 π(s)
  - 你不知道转移概率 T(s,a,s’)
  - 你不知道奖赏函数 R(s,a,s’)
  - 目标：计算出状态值 (state values)



- 在这种情况下：
  - Learner is “along for the ride”
  - 智能体自身没有行动的选择权
  - 只执行策略，并从经验中学习
  - 这并不是线下规划！ You actually take actions in the world.

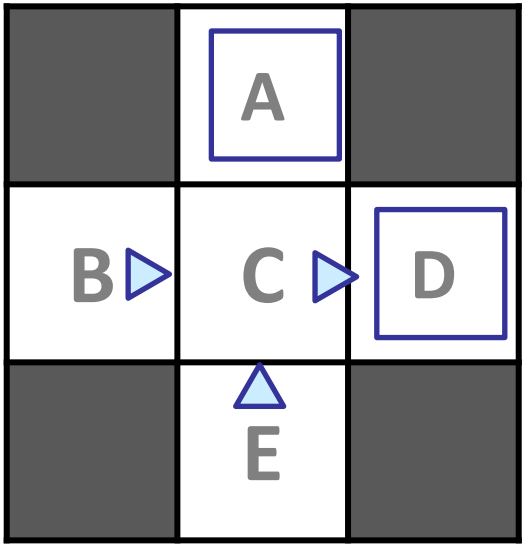
直接估值

- 目标：在策略π下，计算每个状态的值
- 思想：从每个样本进行估值，然后再平均化所有样本的值
  - 按照策略 π 进行行动
  - 每次你访问到一个状态，记录下沿路折扣后的奖赏值之和
  - 再平均化那些样本估值
- 这就叫做直接估值

举例：直接估值

观察到的样本（训练）

输入策略 π



Episode 1

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 2

B, east, C, -1
C, east, D, -1
D, exit, x, +10

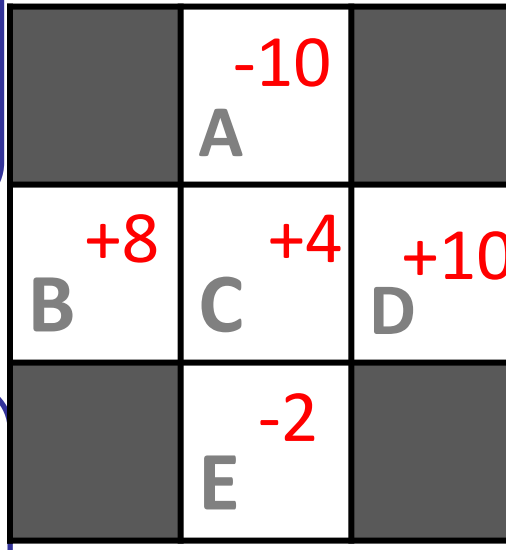
Episode 3

E, north, C, -1
C, east, D, -1
D, exit, x, +10

Episode 4

E, north, C, -1
C, east, A, -1
A, exit, x, -10

输出的状态值



Assume: $\gamma = 1$

直接估值的问题所在

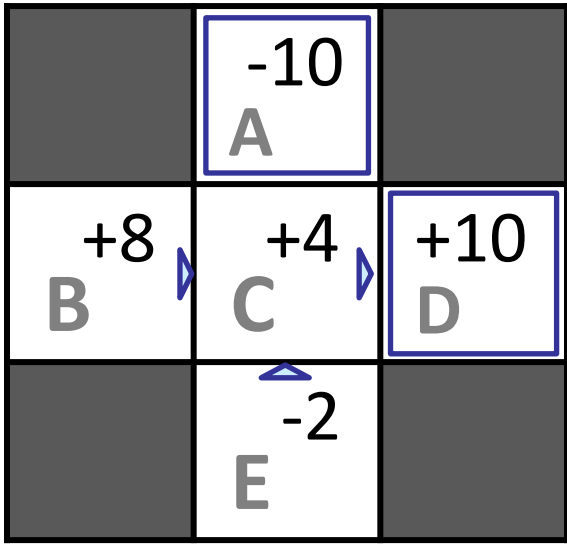
其优势?

- 容易理解
- 不需要知道 T，R
- 最终计算正确的状态值，通过大量样本

其劣势?

- 浪费了一些关于状态之间连接的信息
- 每个状态值必须分开单独学习
- 所以，将花费很长的时间去学习

输出值



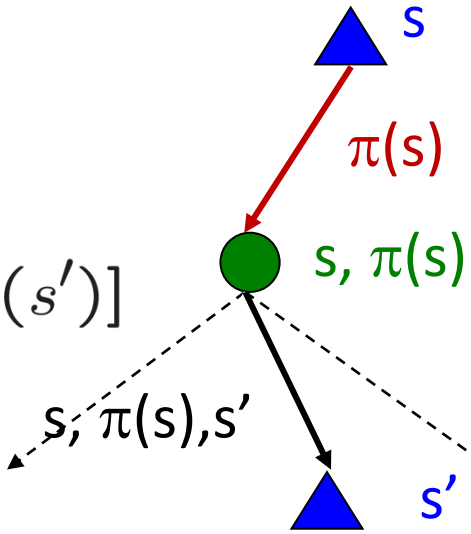
为什么不能直接用策略评价（Policy Evaluation)?

给定一个策略，根据Bellman更新公式计算 V-值:

- 每一轮,用向前一步更新计算结果替换当前的 V

$$V_0^\pi(s) = 0$$

$$V_{k+1}^\pi(s) \leftarrow \sum_{s'} T(s, \pi(s), s')[R(s, \pi(s), s') + \gamma V_k^\pi(s')]$$



- 这种方法充分利用了状态节点间的连接关系
- 但是，我们需要 T 和 R 才能计算!

关键问题：如何在不知道T和R的情况下，更新 V-值?

- 换句话说，如何在不知道权值的情况下，估计一个加权均值？

能否用基于样本的策略评价?

- 我们想改进对 V-值得估计，通过计算下面这些加权均值:

$$V_{k+1}^{\pi}(s) \leftarrow \sum_{s'} T(s, \pi(s), s')[R(s, \pi(s), s') + \gamma V_k^{\pi}(s')]$$

- 思想：通过行动进行采样，得到结果状态 s’，然后进行均值

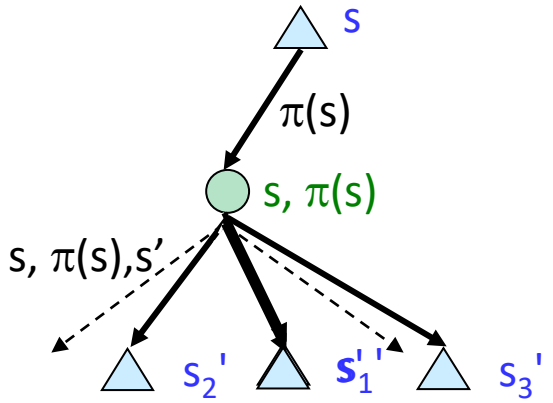
$$sample_1 = R(s, \pi(s), s'_1) + \gamma V_k^{\pi}(s'_1)$$

$$sample_2 = R(s, \pi(s), s'_2) + \gamma V_k^{\pi}(s'_2)$$

$$\cdots$$

$$sample_n = R(s, \pi(s), s'_n) + \gamma V_k^{\pi}(s'_n)$$

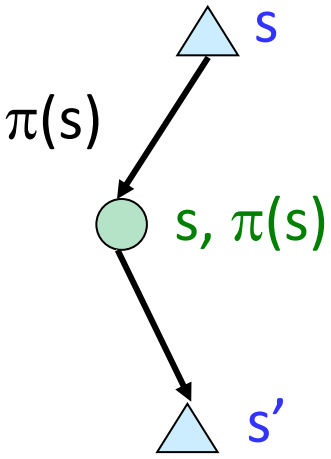
$$V_{k+1}^{\pi}(s) \leftarrow \frac{1}{n} \sum_i sample_i$$



时间差分学习
Temporal Difference Learning

基本思想：从每一次经验中学习！

- 每经历一次状态过渡（s，a，s’，r），就更新一下 V(s)
- 逐渐地，结果状态 s’将会越来越多地参与到更新过程中



V-值的时间差分学习

- 固定的策略，还是在做策略评价！
- 把V值逐渐地向最终结果的值方向移动：逐渐在计算均值

Sample of V(s): $sample = R(s, \pi(s), s') + \gamma V^{\pi}(s')$

Update to V(s): $V^{\pi}(s) \leftarrow (1-\alpha)V^{\pi}(s) + (\alpha)sample$

Same update: $V^{\pi}(s) \leftarrow V^{\pi}(s) + \alpha(sample - V^{\pi}(s))$

指数移动平均法
Exponential Moving Average

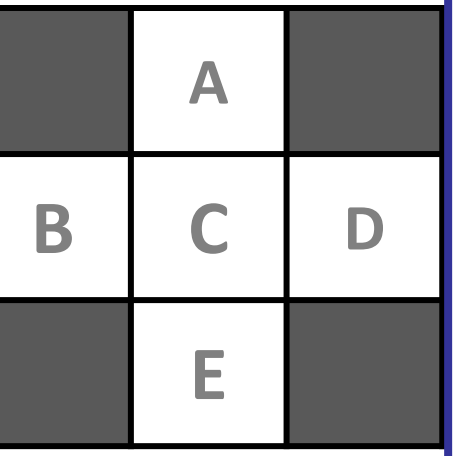
- 指数移动平均
  - 插值更新法: $\bar{x}_n = (1-\alpha)\cdot\bar{x}_{n-1} + \alpha\cdot x_n$
  - 使最近的样本更重要
  - 逐渐遗忘过去的值(distant past values were wrong anyway)
- （逐渐）降低学习参数(alpha)能够使该均值收敛

$$V_k^\pi(s) \leftarrow (1-\alpha)V_{k-1}^\pi(s) + \alpha \cdot sample_k$$

$$\begin{aligned}
V_k^\pi(s) &\leftarrow (1-\alpha)V_{k-1}^\pi(s) + \alpha \cdot sample_k \\
V_k^\pi(s) &\leftarrow (1-\alpha)[(1-\alpha)V_{k-2}^\pi(s) + \alpha \cdot sample_{k-1}] + \alpha \cdot sample_k \\
V_k^\pi(s) &\leftarrow (1-\alpha)^2 V_{k-2}^\pi(s) + (1-\alpha) \cdot \alpha \cdot sample_{k-1} + \alpha \cdot sample_k \\
&\vdots \\
V_k^\pi(s) &\leftarrow (1-\alpha)^k V_0^\pi(s) + \alpha \cdot [(1-\alpha)^{k-1} \cdot sample_1 + \ldots + (1-\alpha) \cdot sample_{k-1} + sample_k] \\
V_k^\pi(s) &\leftarrow \alpha \cdot [(1-\alpha)^{k-1} \cdot sample_1 + \ldots + (1-\alpha) \cdot sample_{k-1} + sample_k]
\end{aligned}$$

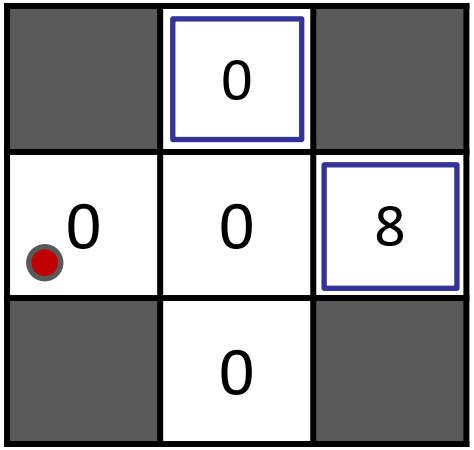
举例：时间差分学习法

States

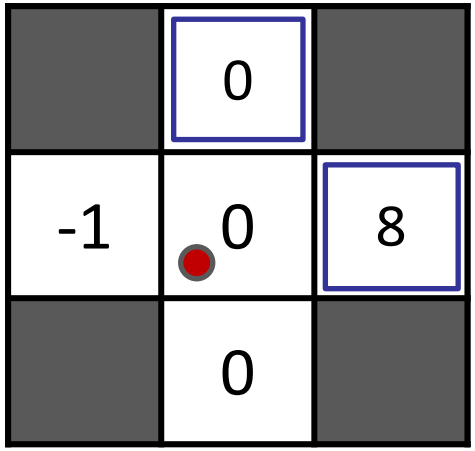


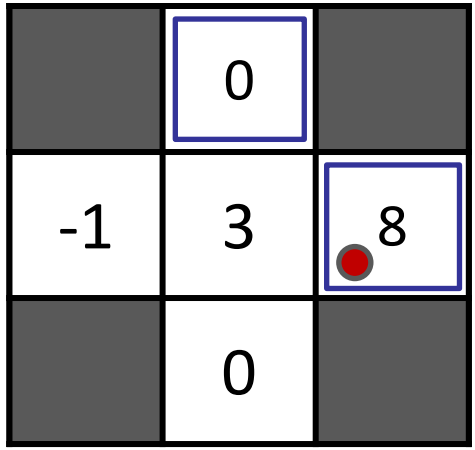
Observed Transitions

B, east, C, -2



C, east, D, -2





Assume: $\gamma = 1$, $\alpha = 1/2$

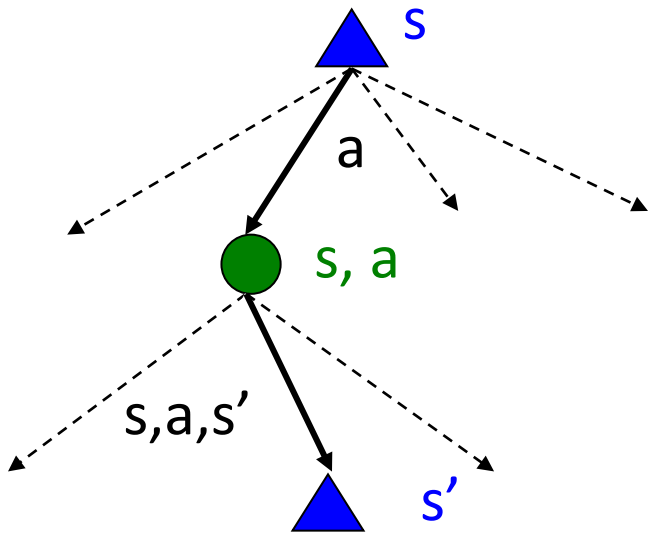
$$V^{\pi}(s) \leftarrow (1-\alpha)V^{\pi}(s) + \alpha\left[R(s,\pi(s),s') + \gamma V^{\pi}(s')\right]$$

通过时间差分法学习到状态值以后怎么办？

- 时间差分状态值学习 是一种model-free方法进行策略评价. 通过计算样本均值, 模仿Bellman的更新，
- 然而，如果想要把状态值转换为(新的)策略，就变得很难了：

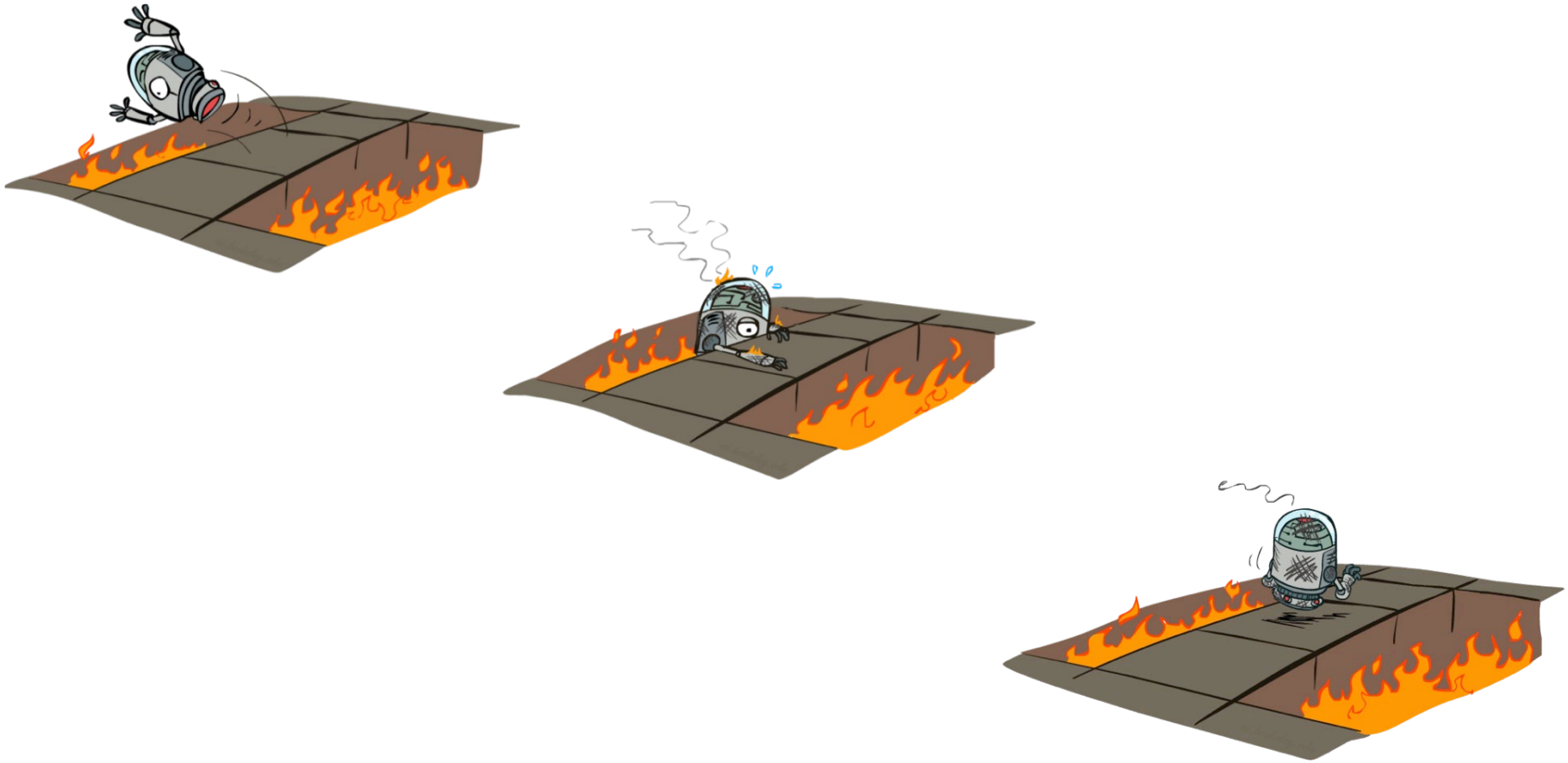
$$\pi(s) = \arg\max_a Q(s,a)$$

$$Q(s,a) = \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma V(s') \right]$$



- 想法：学习 Q-值，而不是V-值
- 从而使行动的选择变得也是model-free!

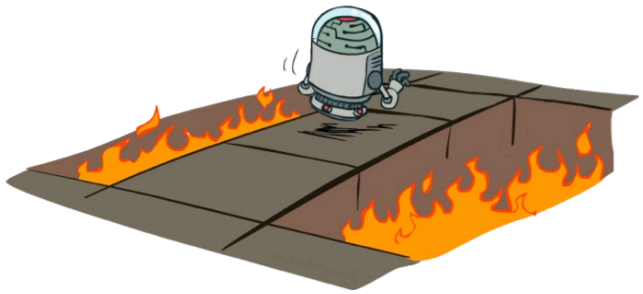
主动强化学习
Active Reinforcement Learning



主动强化学习

完整的强化学习：找到最优策略

- 你不知道转移概率 T(s,a,s’)
- 你不知道奖赏函数 R(s,a,s’)
- 你现在可以**自主选择行动**
- 目标：学习最优策略 / 状态值



在这种情况下：

- 智能体决定做什么(行动)!
- 基本上的平衡制约：探索 vs. 利用
- 这并不是线下规划！ You actually take actions in the world and find out what happens…

Q值迭代
先介绍一下：Q-Value Iteration

- 状态值迭代：找到后继（有限深度的）值
  - 初始化 $V_0(s) = 0$, which we know is right
  - 给定 V_k，计算深度为 k+1 的值 for all states:

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma V_k(s') \right]$$

- 但是 Q-值 更有用，所以转而计算它们
  - 初始化 $Q_0(s,a) = 0$, which we know is right
  - 给定 Q_k，计算深度为 k+1 的 q值 for all q-states:

$$Q_{k+1}(s,a) \leftarrow \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma \max_{a'} Q_k(s',a') \right]$$

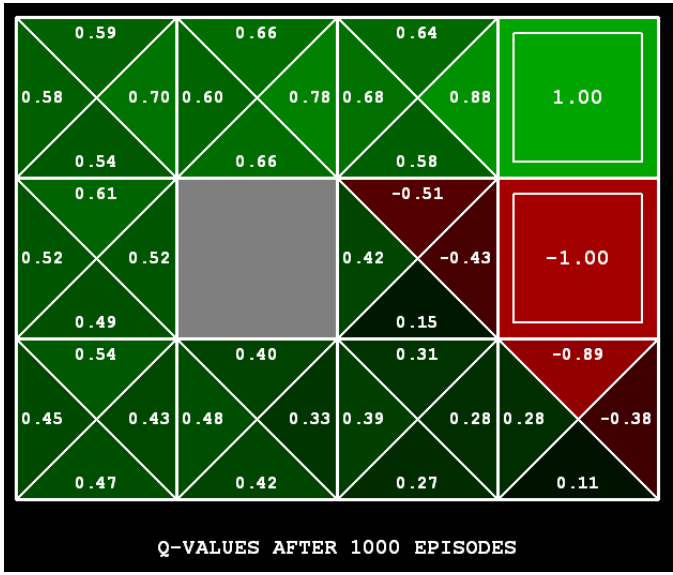
Q-值学习
Q-Learning

- Q-Learning：基于样本的 Q-值 迭代

$$Q_{k+1}(s,a) \leftarrow \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma \max_{a'} Q_k(s',a') \right]$$

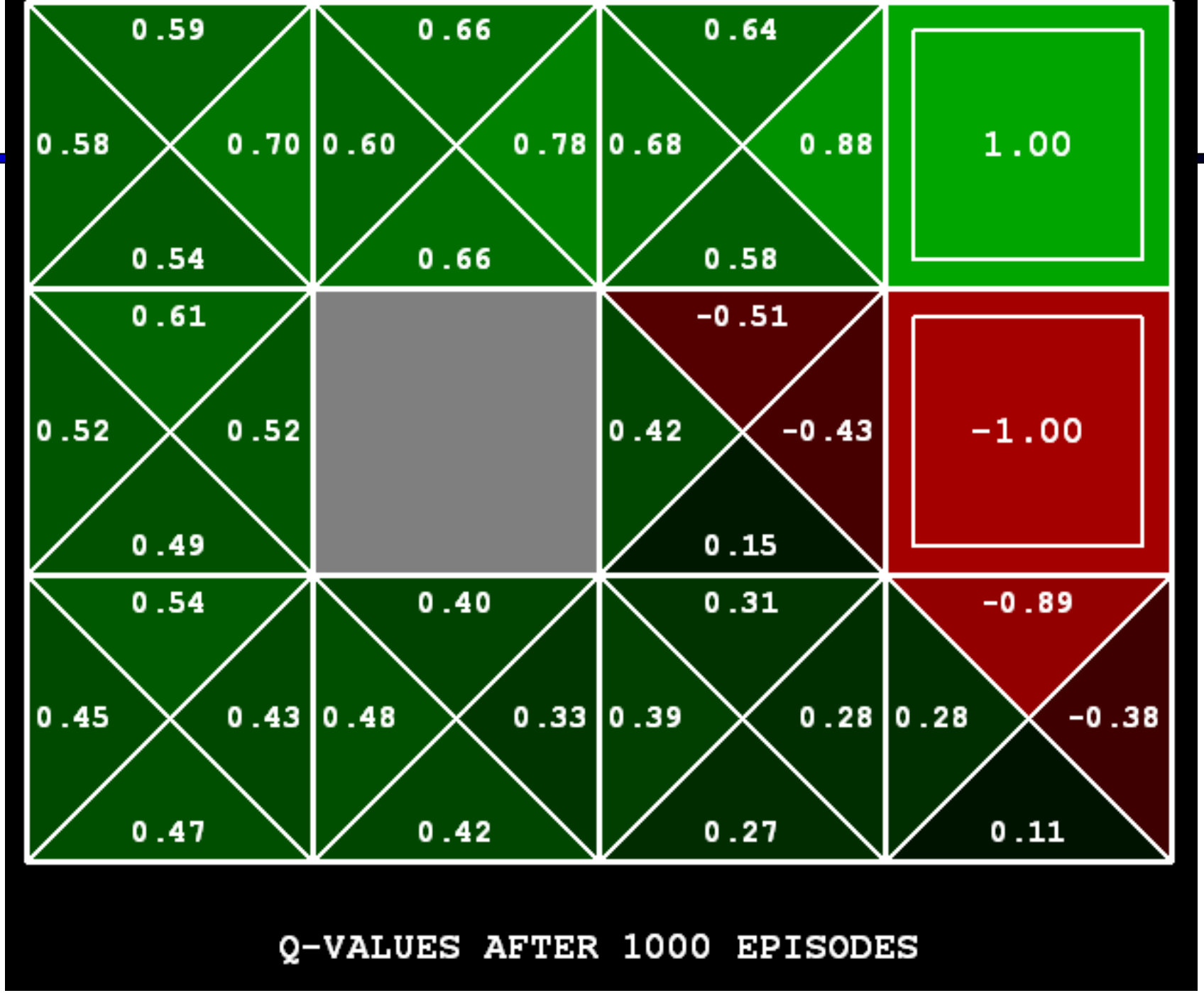
- 学习 Q(s, a) 的值，在实践过程中
  - 获得一个样本（s, a, s’, r）
  - 考虑你旧的估值：$Q(s,a)$
  - 考虑你新的估值：

$$sample = R(s,a,s') + \gamma \max_{a'} Q(s',a')$$

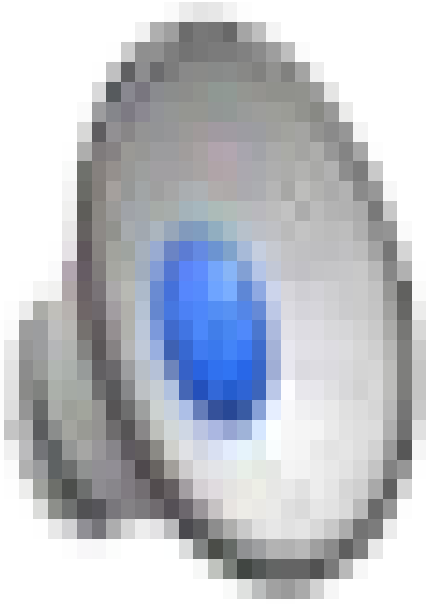


- 把新的估值结合进来，计算实时均值更新：

$$Q(s,a) \leftarrow (1-\alpha)Q(s,a) + (\alpha)\,[sample]$$


0.59
0.58
0.70
0.54
0.66
0.60
0.78
0.66
0.64
0.68
0.88
0.58
1.00
0.61
0.52
0.52
0.49
-0.51
0.42
-0.43
0.15
-1.00
0.54
0.45
0.43
0.47
0.40
0.48
0.33
0.42
0.31
0.39
0.28
0.27
-0.89
0.28
-0.38
0.11
Q-VALUES AFTER 1000 EPISODES

视频演示 Q-Learning -- Gridworld



视频演示 Q-Learning -- Crawler

Q-Learning 性质

- 好结果：Q-learning 收敛于最优策略 - 即便你在实践采样过程中行动是次优化的!
- 这叫做 没有策略的学习 off-policy learning
- 需要注意的：
  - 你不得不要探索充分
  - 你最终要调整learning rate 到足够小
  - … 但也不能减小太快
  - 最终来讲，学习过程中你如何选择行动无关紧要(!)