强化学习：

强化学习：

明天是打球还是学习呢？

强化学习：

+1（奖励）

-1（惩罚）

强化学习：

（奖励）

（惩罚）

强化学习：

学习系统没有像很多其它形式的机器学习方法一样被告知应该做出什么行为

必须在尝试了之后才能发现哪些行为会导致奖励的最大化

当前的行为可能不仅仅会影响即时奖励，还会影响下一步的奖励以及后续的所有奖励

强化学习：

智能体 (agent)

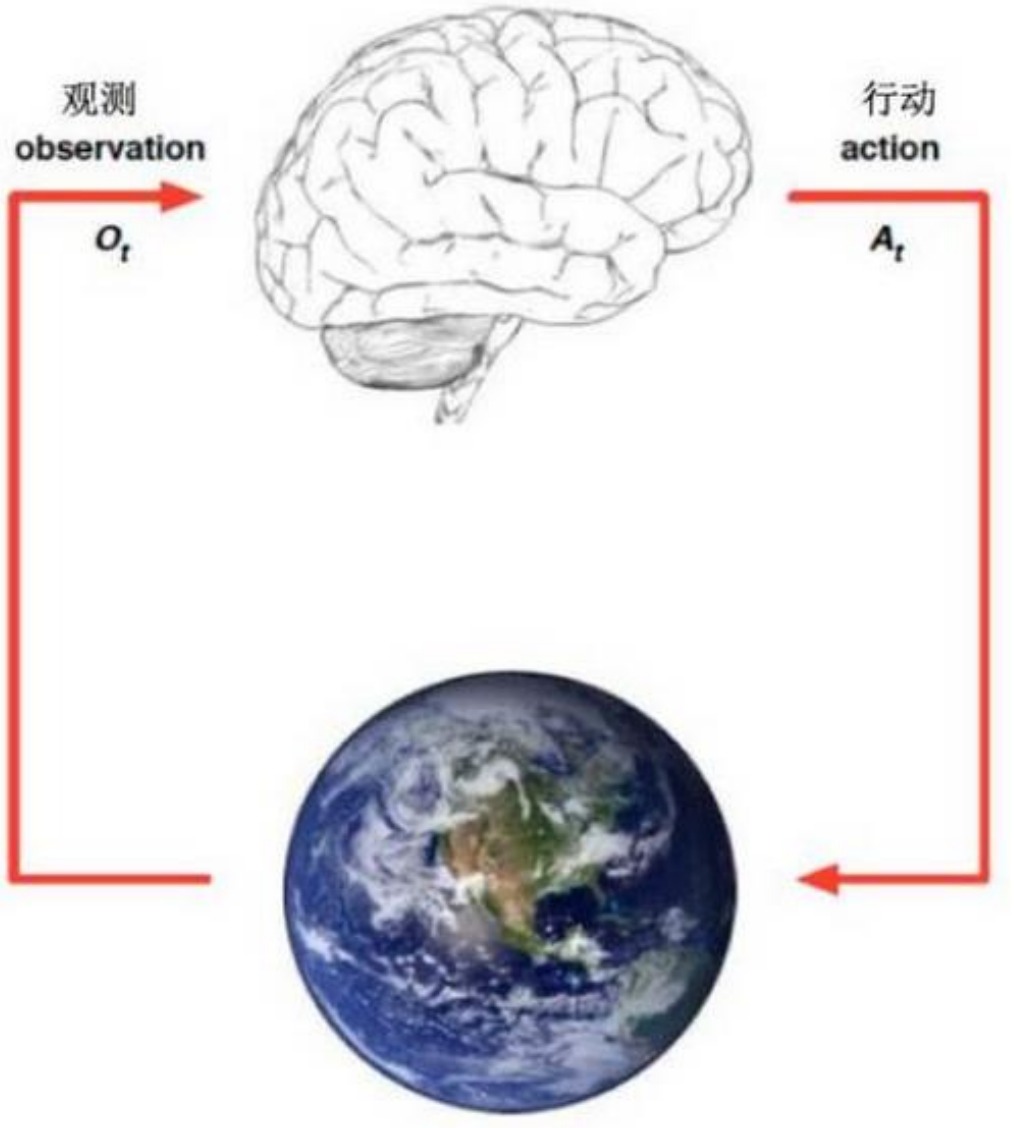
状态 (state)

行为 (action)

奖励 (reward)

策略 (policy)

强化学习：



先观察 再行动 再观测...

每一个动作（action）都能影响代理将来的状态（state）

通过一个标量的奖励（reward）信号来衡量成功

目标：选择一系列行动来最大化未来的奖励

强化学习：

AlphaGo如何确定每一次落子？

强化学习：

怎样才能接得住求呢？

强化学习：

状态(state)

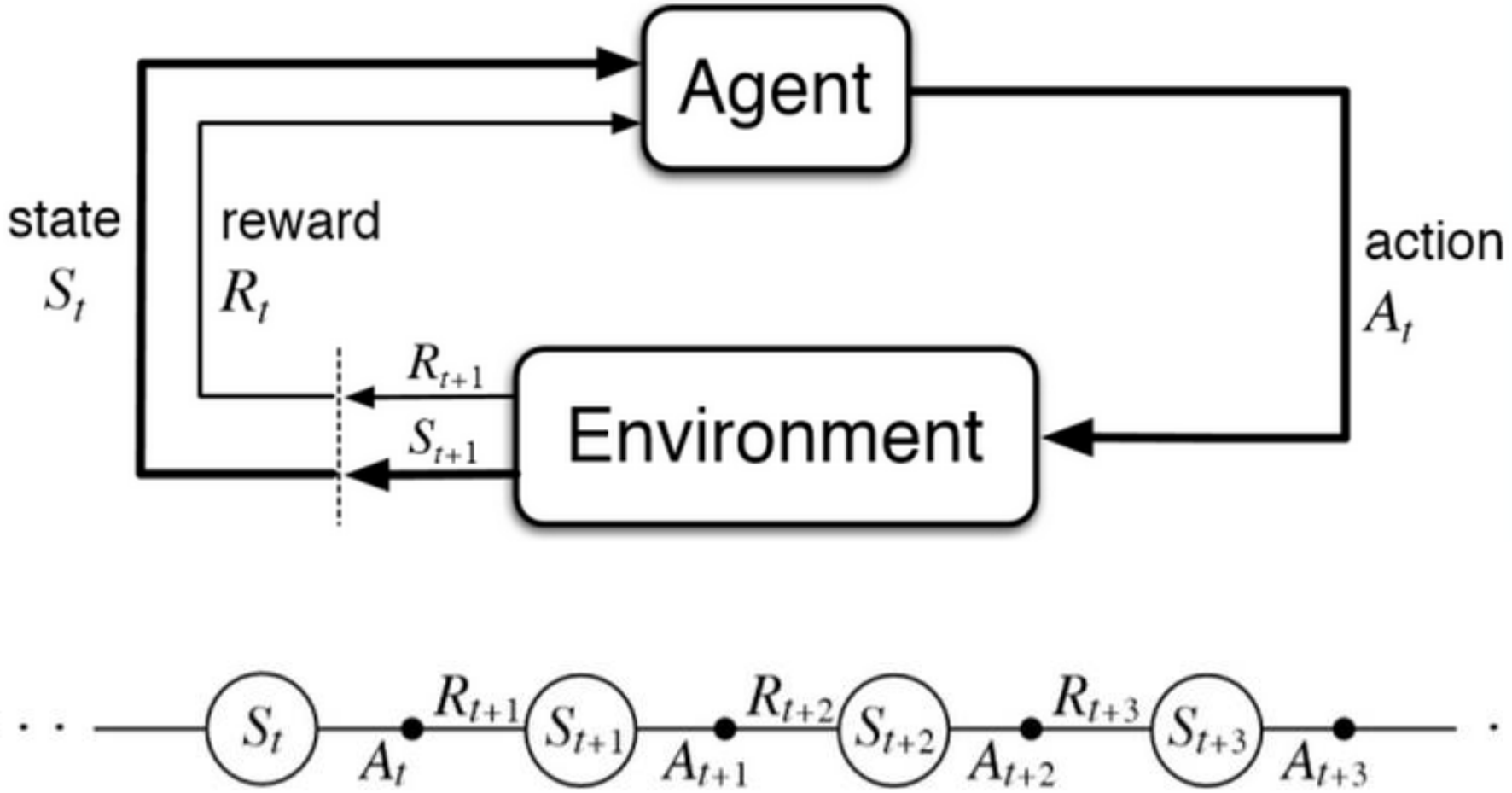
- Experience is a sequence of observations, actions, rewards

$$o_1, r_1, a_1, \ldots, a_{t-1}, o_t, r_t$$

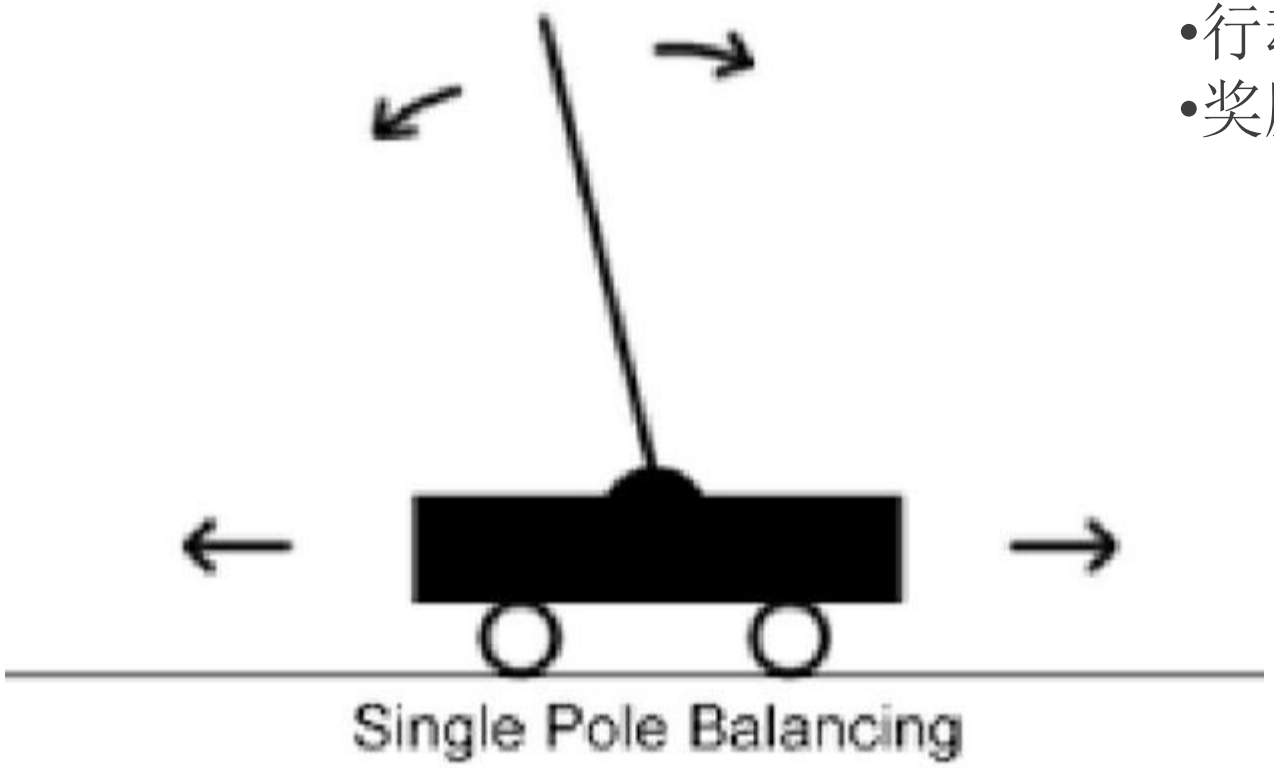
- The state is a summary of experience

$$s_t = f(o_1, r_1, a_1, \ldots, a_{t-1}, o_t, r_t)$$

强化学习：


Agent
state
S_t
reward
R_t
action
A_t
R_{t+1}
S_{t+1}
Environment
S_t
A_t
R_{t+1}
S_{t+1}
A_{t+1}
R_{t+2}
S_{t+2}
A_{t+2}
R_{t+3}
S_{t+3}
A_{t+3}

强化学习：



- 状态（state）：杆的角度和速度
- 行动（action）：小车进行左或右方向移动
- 奖励（reward）：杆不倾斜 1
  杆倾斜 0

强化学习：

马尔科夫决策要求：

1.能够检测到理想的状态

2.可以多次尝试

3.系统的下个状态只与当前状态信息有关，而与更早之前的状态无关
在决策过程中还和当前采取的动作有关

强化学习：

马尔科夫决策过程由5个元素构成：

S:表示状态集（states）

A:表示一组动作（actions）

P:表示状态转移概率 P_{sa}表示在当前s ∈ S状态下，经过a ∈ A作用后，会转移到的其他状态的概率分布情况在状态s下执行动作a，转移到s’的概率可以表示为p(s’|s,a)

R: 奖励函数（reward function）表示 agent 采取某个动作后的即时奖励

y：折扣系数意味着当下的 reward 比未来反馈的 reward 更重要 $\sum_{t=0}^{\infty}\gamma^t R(s_t) \quad 0 \le \gamma < 1$

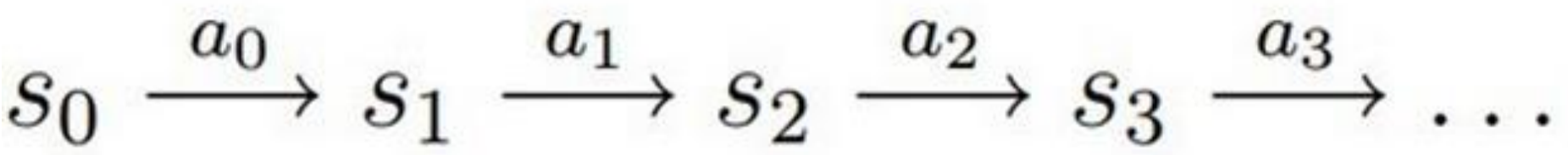
强化学习：

1.智能体初始状态为S0

2.选择一个动作a0

3.按概率转移矩阵Psa转移到了下一个状态S1

然后。。。



强化学习：

状态价值函数：$v(s) = E[U_t|S_t = s]$

t 时刻的状态 s 能获得的未来回报的期望

价值函数用来衡量某一状态或状态-动作对的优劣价，累计奖励的期望

最优价值函数：所有策略下的最优累计奖励期望 $v_*(s) = \max_\pi v_\pi(s)$

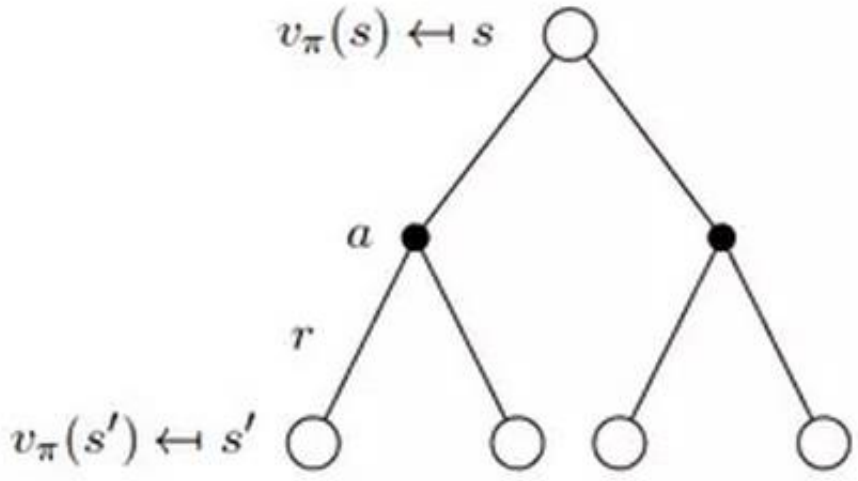
策略:已知状态下可能产生动作的概率分布

强化学习：

Bellman方程：当前状态的价值和下一步的价值及当前的奖励（Reward）有关

价值函数分解为当前的奖励和下一步的价值两部分

强化学习：



π 是给定状态s的情况下，动作a的概率分布

因为动作空间A，状态空间S均为有限集合，所以我们可以用求和来计算期望

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

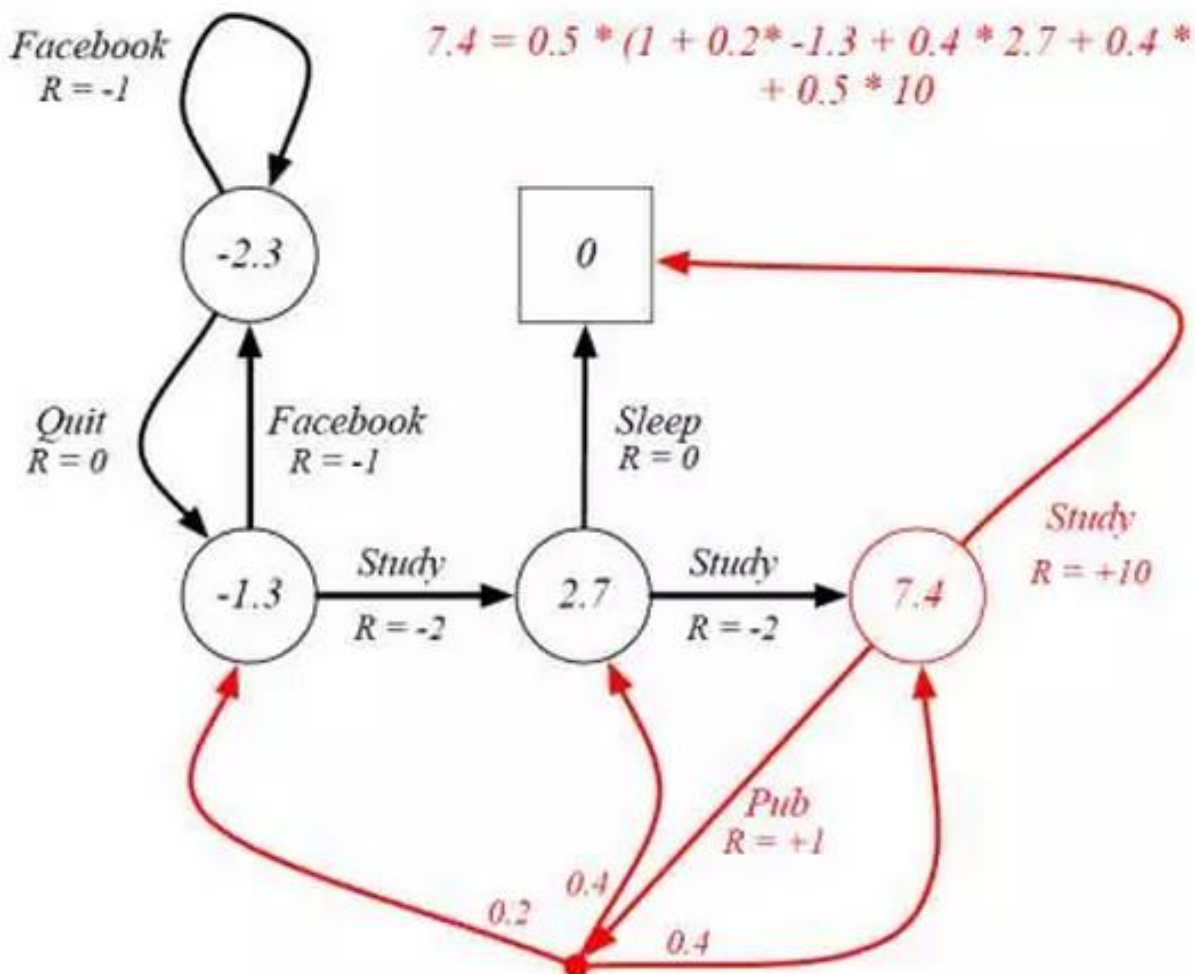
$$P_{ss'}^a = P(S_{t+1} = s' | S_t = s, A_t = a)$$

$$v_\pi(s) = E_\pi[R_s^a + \gamma v_\pi(S_{t+1}) | S_t = s]$$

强化学习：

$$v_\pi(s_5) = \pi(a_3|s_5) * R_{a_3} + \pi(a_4) * (R_{a_4} + 1 * (P(s_3|s_5, a_4) * \pi(s_3) + P(s_4|s_5, a_4) * \pi(s_4) + P(s_5|s_5, a_4) * \pi(s_5))$$

$$7.4 = 0.5 * 10 + 0.5 * (1 + 0.2 * -1.3 + 0.4 * 2.7 + 0.4 * 7.4)$$

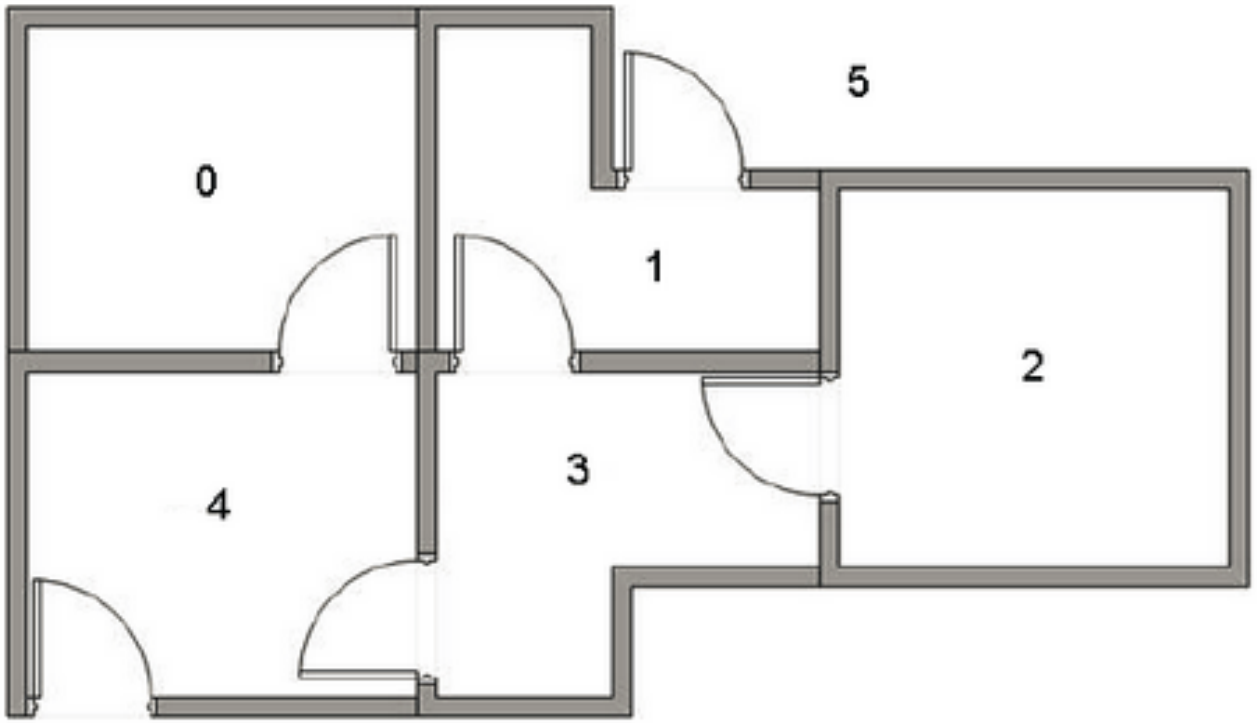


强化学习：

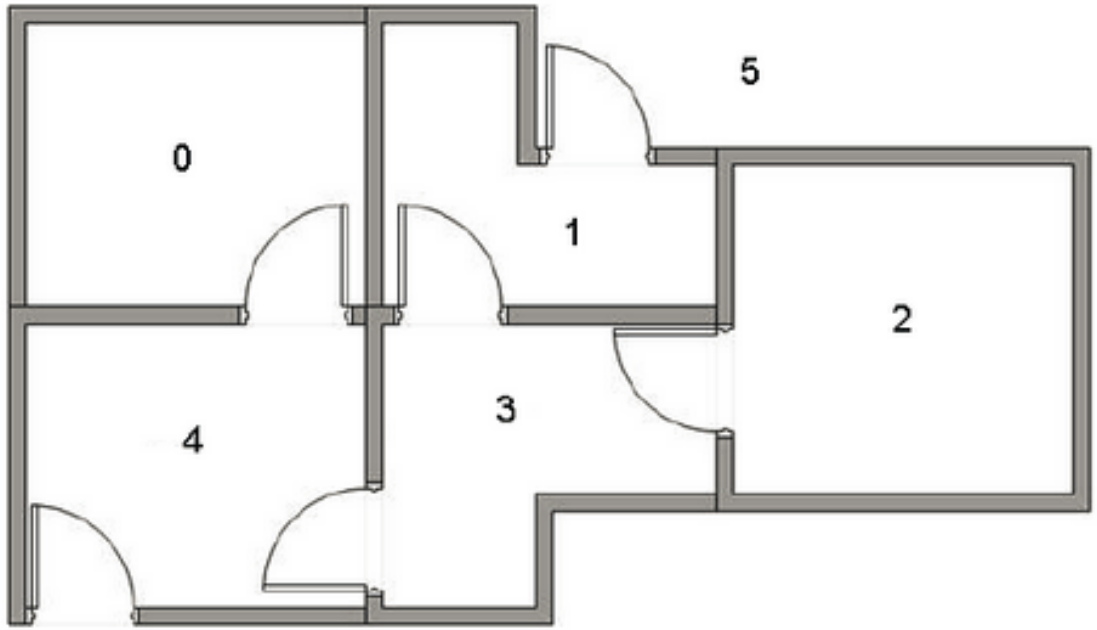
Bellman最优化方程： $v^*(s) = \max_a \mathcal{R}_s^a + \gamma \sum_{s' \in S} \mathcal{P}_{ss'}^a v_*(s')$

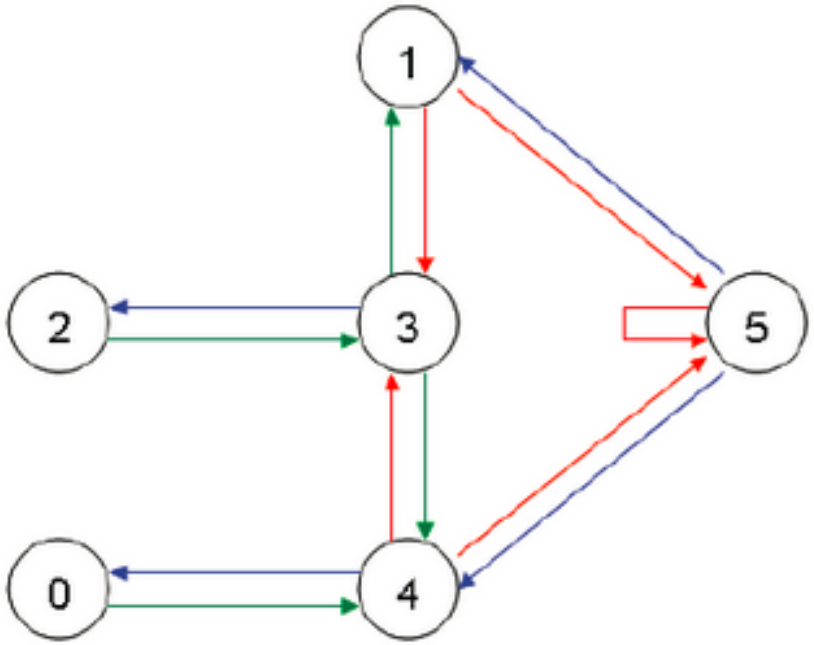
在某个状态（state）下最优价值函数的值，就是智能体（agent）在该状态下，所能获得的累积期望奖励值（cumulative expective rewards）的最大值.

Q-Learning:

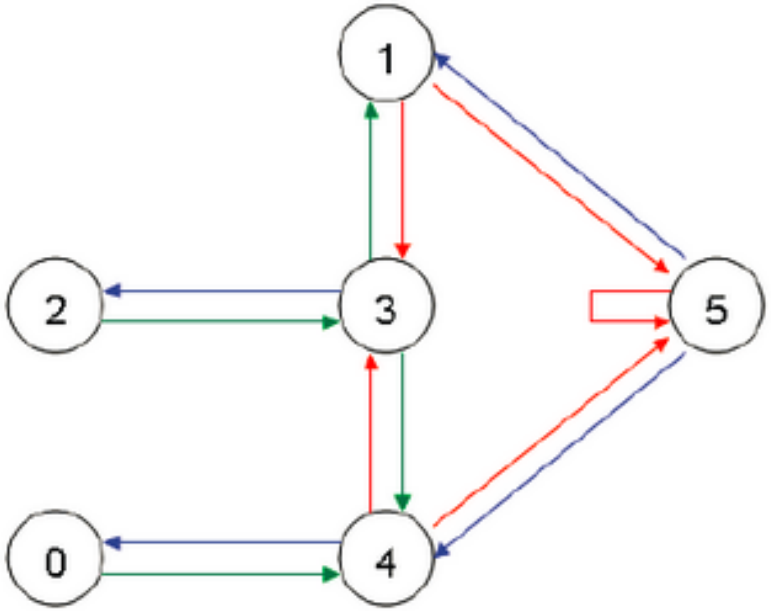


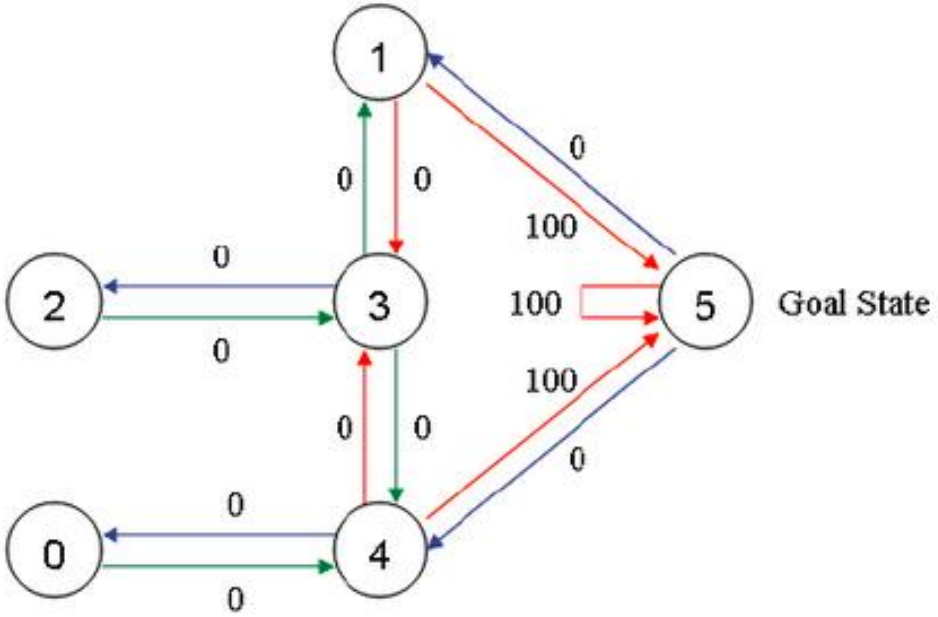
Q-Learning:



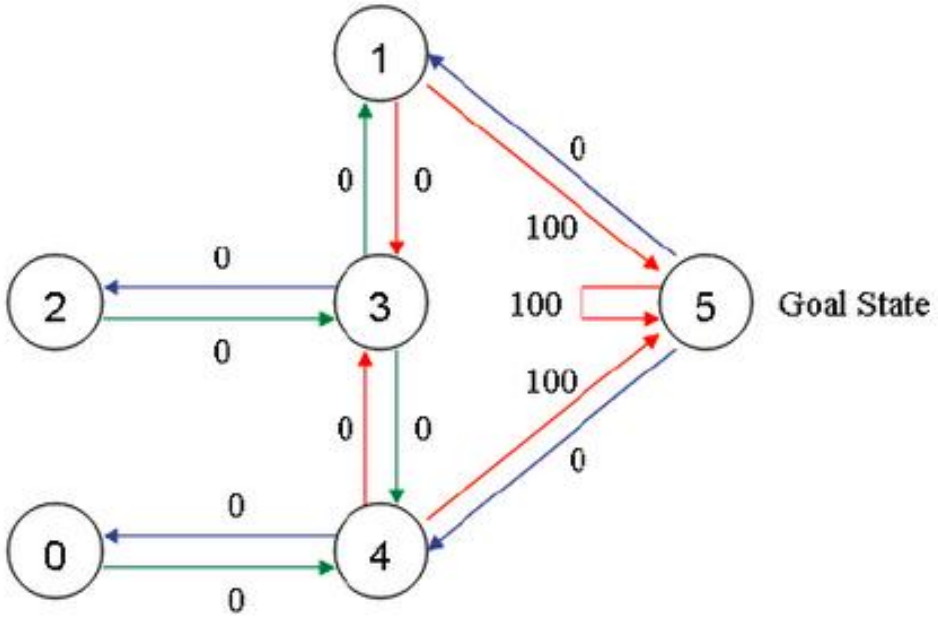


Q-Learning:





Q-Learning:



$$R = \begin{array}{c} \\ \text{State} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \text{Action} \\ \begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} -1 & -1 & -1 & -1 & 0 & -1 \\ -1 & -1 & -1 & 0 & -1 & 100 \\ -1 & -1 & -1 & 0 & -1 & -1 \\ -1 & 0 & 0 & -1 & 0 & -1 \\ 0 & -1 & -1 & 0 & -1 & 100 \\ -1 & 0 & -1 & -1 & 0 & 100 \end{bmatrix} \end{array}$$

Q-Learning:

$$Q(s,a) = R(s,a) + \gamma \cdot \max_{\widetilde{a}}\{Q(\widetilde{s},\widetilde{a})\}$$

其中 s, a 表示当前的状态和行为, $\widetilde{s}, \widetilde{a}$ 表示 s 的下一个状态及行为, 学习参数 γ 为满足 $0 \leq \gamma < 1$ 的常数.

Step 1 给定参数 γ 和 *reward* 矩阵 R.

Step 2 令 $Q := \mathbf{0}$.

Step 3 *For each episode:*

3.1 随机选择一个初始的状态 s.

3.2 若未达到目标状态, 则执行以下几步

(1) 在当前状态 s 的所有可能行为中选取一个行为 a.

(2) 利用选定的行为 a, 得到下一个状态 $\widetilde{s}$.

(3) 按照 *(1.1)* 计算 $Q(s,a)$.

(4) 令 $s := \widetilde{s}$.

Q-Learning：

首先取学习参数 $\gamma = 0.8$, 初始状态为房间 1, 并将 Q 初始化为一个零矩阵

$$Q = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

Q-Learning:

观察矩阵 R 的第二行 (对应房间 1 或状态 1), 它包含两个非负值, 即当前状态 1 的下一步行为有两种可能: 转至状态 3 或转至状态 5. 随机地, 我们选取转至状态 5.

$$R = \begin{array}{c} \\ \text{State} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \text{Action} \\ \begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} -1 & -1 & -1 & -1 & 0 & -1 \\ -1 & -1 & -1 & 0 & -1 & 100 \\ -1 & -1 & -1 & 0 & -1 & -1 \\ -1 & 0 & 0 & -1 & 0 & -1 \\ 0 & -1 & -1 & 0 & -1 & 100 \\ -1 & 0 & -1 & -1 & 0 & 100 \end{bmatrix} \end{array}$$

Q-Learning:

想象一下, 当我们的 agent 位于状态 5 以后, 会发生什么事情呢? 观察矩阵 R 的第 6 行它对应三个可能的行为: 转至状态 1, 4 或 5.

$$
\begin{aligned}
Q(1,5) &= R(1,5) + 0.8 * \max\{Q(5,1), Q(5,4), Q(5,5)\} \\
&= 100 + 0.8 * \max\{0,0,0\} \\
&= 100.
\end{aligned}
$$

$$
R = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}
\begin{array}{c}
\begin{array}{cccccc} \text{Action} & & & & & \\ 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{bmatrix}
-1 & -1 & -1 & -1 & 0 & -1 \\
-1 & -1 & -1 & 0 & -1 & 100 \\
-1 & -1 & -1 & 0 & -1 & -1 \\
-1 & 0 & 0 & -1 & 0 & -1 \\
0 & -1 & -1 & 0 & -1 & 100 \\
-1 & 0 & -1 & -1 & 0 & 100
\end{bmatrix}
\end{array}
\quad \text{State}
$$

$$
Q = \begin{array}{c}
\begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 100 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\end{array}
$$

Q-Learning:

接下来, 进行下一次 episode 的迭代, 首先随机地选取一个初始状态, 这次我们选取状态 3 作为初始状态.

观察矩阵 R 的第四行 (对应状态 3), 它对应三个可能的行为: 转至状态 1, 2 或 4. 随机地, 我们选取转至状态 1. 因此观察矩阵 R 的第二行 (对应状态 1), 它对应两个可能的行为: 转至状态 3 或 5.

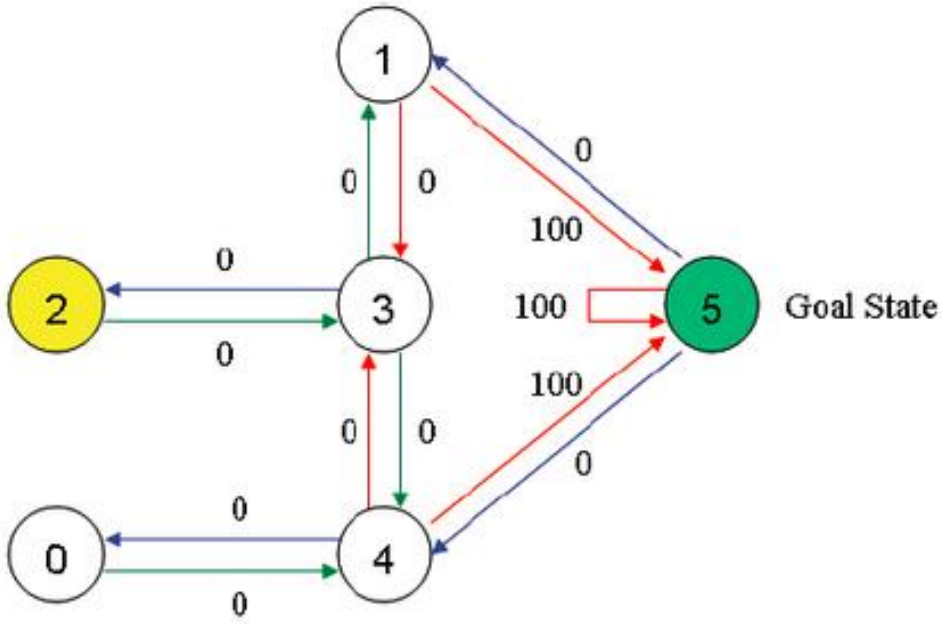
$$
\begin{aligned}
Q(3,1) &= R(3,1) + 0.8 * \max\{Q(1,3), Q(1,5)\} \\
&= 0 + 0.8 * \max\{0, 100\} \\
&= 80.
\end{aligned}
$$

$$
R = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}
\begin{array}{c}
\begin{array}{cccccc} \text{Action} & & & & & \\ 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{bmatrix}
-1 & -1 & -1 & -1 & 0 & -1 \\
-1 & -1 & -1 & 0 & -1 & 100 \\
-1 & -1 & -1 & 0 & -1 & -1 \\
-1 & 0 & 0 & -1 & 0 & -1 \\
0 & -1 & -1 & 0 & -1 & 100 \\
-1 & 0 & -1 & -1 & 0 & 100
\end{bmatrix}
\end{array}
\quad \text{State}
$$

$$
Q = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}
\begin{array}{c}
\begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 100 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 80 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\end{array}
$$

Q-Learning：

现在状态 1 变成了当前状态. 因为状态 1 还不是目标状态, 因此我们需要继续往前探索. 状态 1 对应三个可能的行为: 转至状态 3 或 5. 不妨假定我们幸运地选择了状态 5.



Q-Learning:

$$
\begin{aligned}
Q(1,5) &= R(1,5) + 0.8 * \max\{Q(5,1), Q(5,4), Q(5,5)\} \\
&= 100 + 0.8 * \max\{0,0,0\} \\
&= 100.
\end{aligned}
$$

$$
Q = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}
\begin{array}{c}
\begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 100 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 80 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\end{array}
$$

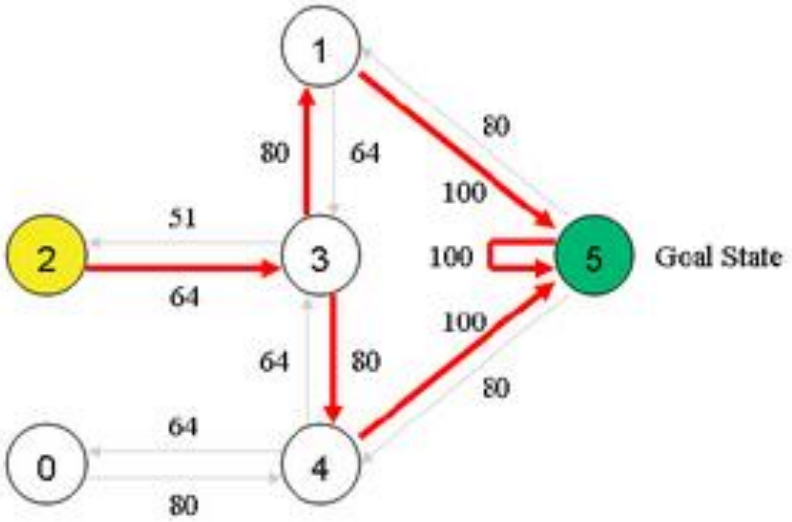
$$
R = \begin{array}{c} \text{State} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}
\begin{array}{c}
\begin{array}{cccccc} \multicolumn{6}{c}{\text{Action}} \\ 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{bmatrix}
-1 & -1 & -1 & -1 & 0 & -1 \\
-1 & -1 & -1 & 0 & -1 & 100 \\
-1 & -1 & -1 & 0 & -1 & -1 \\
-1 & 0 & 0 & -1 & 0 & -1 \\
0 & -1 & -1 & 0 & -1 & 100 \\
-1 & 0 & -1 & -1 & 0 & 100
\end{bmatrix}
\end{array}
$$

Q-Learning:

若我们继续执行更多的 episode, 矩阵 Q 将最终收敛成

$$Q=\begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}\begin{array}{c} \begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 400 & 0 \\ 0 & 0 & 0 & 320 & 0 & 500 \\ 0 & 0 & 0 & 320 & 0 & 0 \\ 0 & 400 & 256 & 0 & 400 & 0 \\ 320 & 0 & 0 & 320 & 0 & 500 \\ 0 & 400 & 0 & 0 & 400 & 500 \end{bmatrix} \end{array}$$

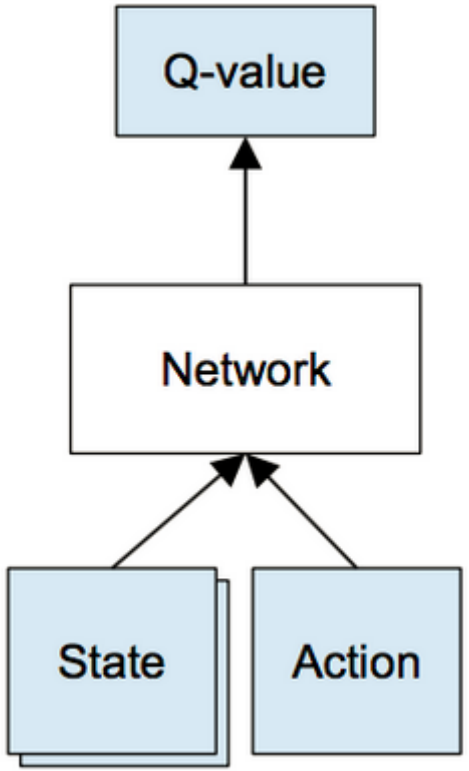
$$Q=\begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}\begin{array}{c} \begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 80 & 0 \\ 0 & 0 & 0 & 64 & 0 & 100 \\ 0 & 0 & 0 & 64 & 0 & 0 \\ 0 & 80 & 51 & 0 & 80 & 0 \\ 64 & 0 & 0 & 64 & 0 & 100 \\ 0 & 80 & 0 & 0 & 80 & 100 \end{bmatrix} \end{array}$$



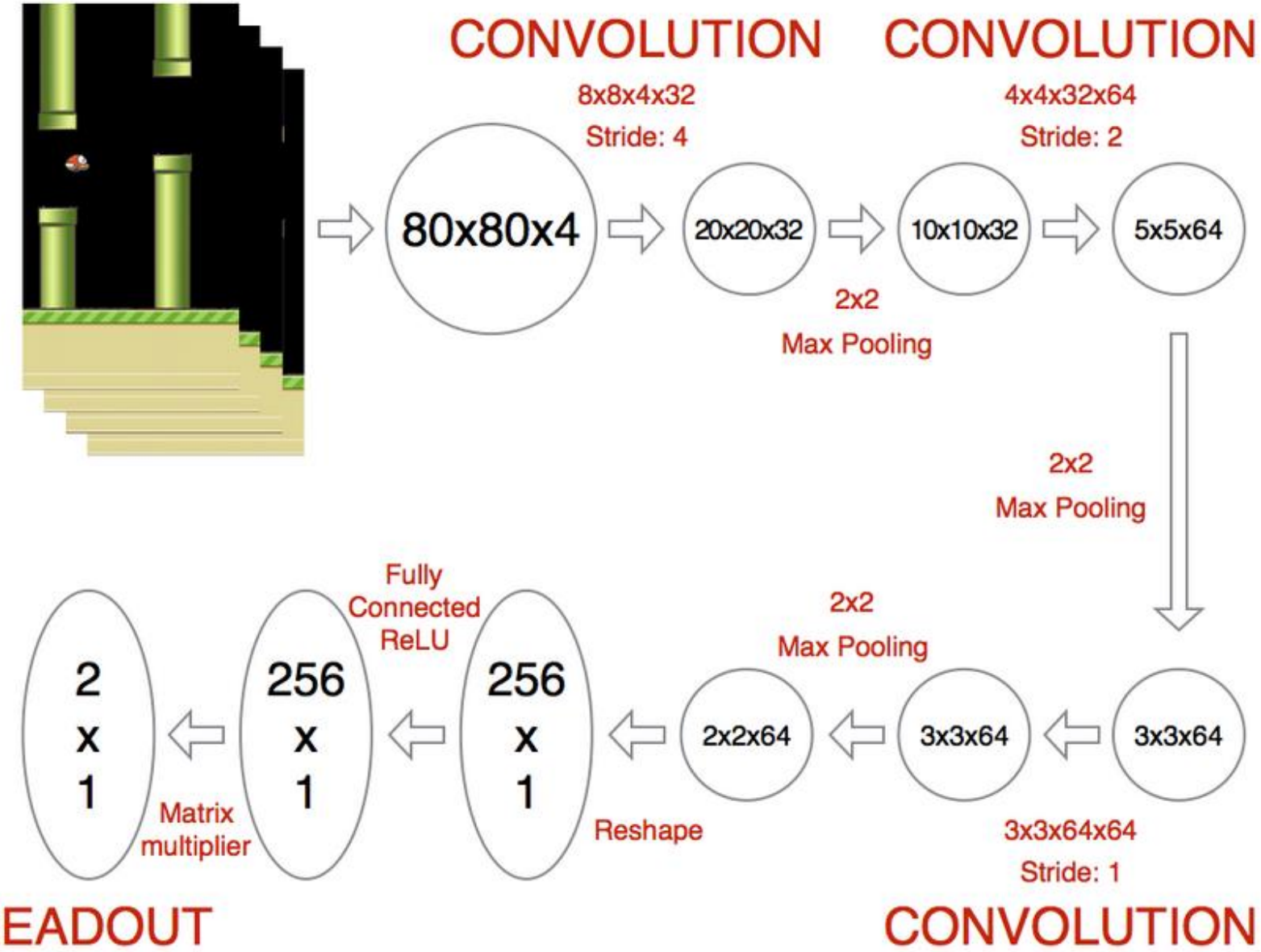
Deep Q Network

假设一个batch是四张图，那么我们的Q-table该如何设定呢？

$$256^{84x84x4} \approx 10^{67970}$$



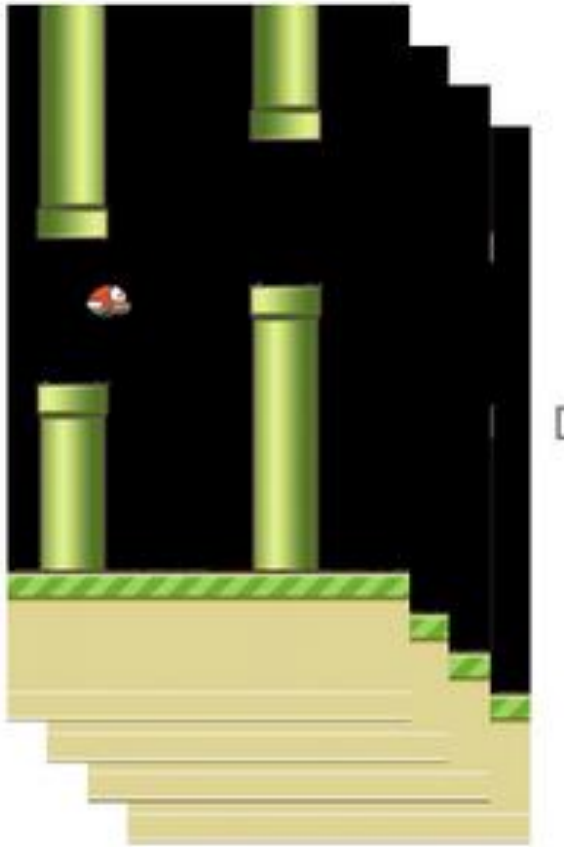
Deep Q Network



Deep Q Network

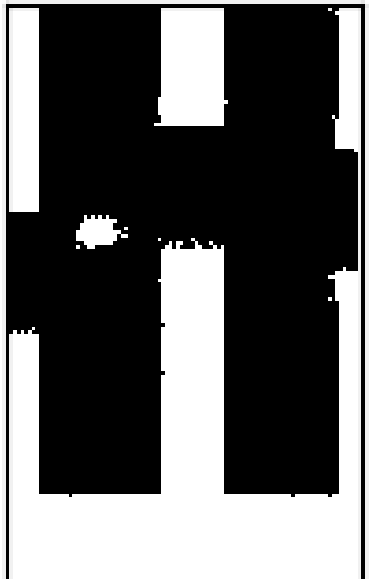
Layer	Input	Filter size	Stride	Num filters	Activation	Output
conv1	84x84x4	8x8	4	32	ReLU	20x20x32
conv2	20x20x32	4x4	2	64	ReLU	9x9x64
conv3	9x9x64	3x3	1	64	ReLU	7x7x64
fc4	7x7x64			512	ReLU	512
fc5	512			18	Linear	18

Deep Q Network



1. Convert image to grayscale
2. Resize image to 80x80
3. Stack last 4 frames to produce an 80x80x4 input array for network

BINARY



Deep Q Network

Exploration VS Exploitation：探索还是开发？都要！

ε-greedy exploration：有机会去探索

强化学习：

强化学习：

强化学习：

强化学习：

强化学习：

智能体（agent）

策略（policy）

行为(action)