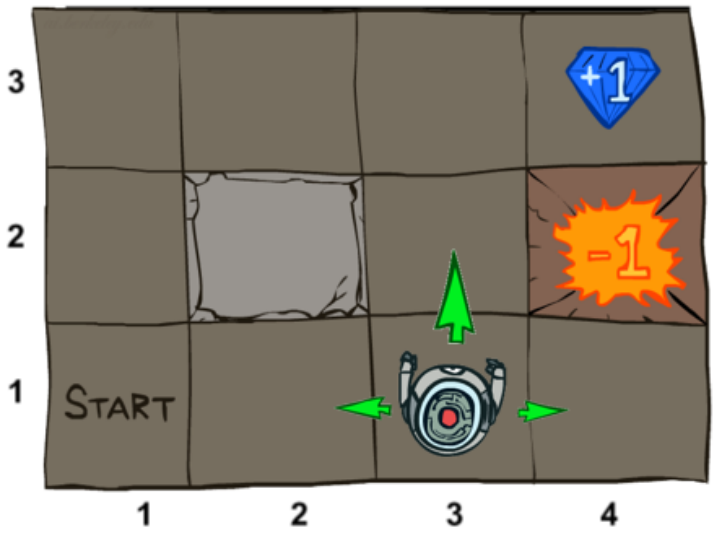
# 实验三-CS188

*潘林朝*

## 网格世界

包含：初始位置、墙、两个结束状态（获得1或则-1）奖励、活着的移动奖励（living reward）。
目标：尽可能获得多的奖励(reward)。




注意：智能体(agent)做出动作north后，形成q值，由于转移概率函数的影响，不一定就执行north动作，这可以认为是实际环境的影响。在这里有80%的几率做出north动作，有10%和10%的几率做出west和east，有0%做出相反动作。

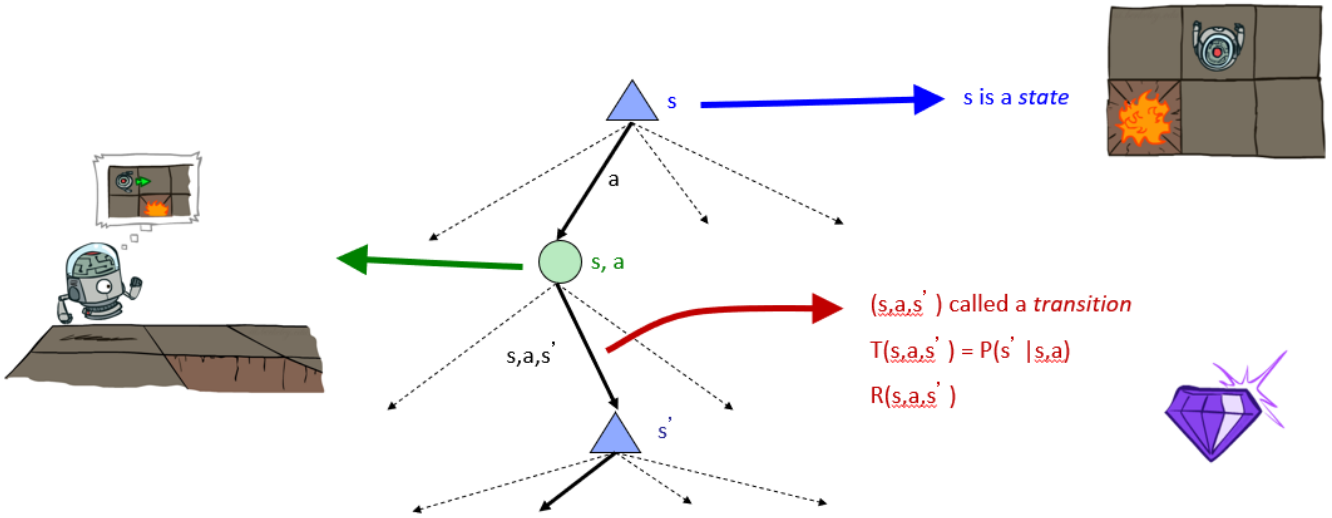
## MDPs

描述马尔科夫决策过程(Markov Decision Processes)的因素有：

- 状态集合$S$
- 动作集合$A$
- 状态转移概率函数$T(s,a,s')$
- 奖励函数$R(s,a,s')$

## 一个示例

可以发现，对比min-max搜索树，它多出一个q值，即下图绿色的。




## 价值迭代（动态规划算法）

基于动态规划的强化学习算法要求事先知道环境的状态转移函数和奖励函数，也就是需要知道整个马尔可夫决策过程。已知MDP所有因素（包括状态转移概率函数，即环境模型），去尽可能的获得最多的奖励。换而言之，找到最优策略，使得奖励的期望（价值/回报）最大。

参考链接，在加入策略π和折扣因子γ的因素后，可以得到贝尔曼最优方程如下：

$$V^*(s) = \max_{a \in A} \{ r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V^*(s') \}$$

$$Q^*(s,a) = r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) \max_{a' \in A} Q^*(s',a')$$

可以递归定义：

$$V^*(s) = \max_a Q^*(s,a)$$

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

$$V^*(s) = \max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V^*(s')]$$

使用动态规划价值迭代的目标是：

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

也即它的目标是获取到最优的价值估计。

伪代码如下[1]：

**Algorithm – Value Iteration**

**Input:** MDP $M = \langle S, s_0, A, P_a(s' \mid s), r(s,a,s') \rangle$
**Output:** Value function $V$

Set $V$ to arbitrary value function; e.g., $V(s) = 0$ for all $s$

Repeat
  $\Delta \leftarrow 0$
  For each $s \in S$
    $\underbrace{V'(s) \leftarrow \max_{a \in A(s)} \sum_{s' \in S} P_a(s' \mid s) \left[ r(s,a,s') + \gamma V(s') \right]}_{\text{Bellman equation}}$
    $\Delta \leftarrow \max(\Delta, |V'(s) - V(s)|)$
  $V \leftarrow V'$
Until $\Delta \leq \theta$

加入Q值后，可改写为：

$\Delta \leftarrow 0$
For each $s \in S$
  For each $a \in A(s)$
    $Q(s,a) \leftarrow \sum_{s' \in S} P_a(s' \mid s) \left[ r(s,a,s') + \gamma V(s') \right]$
  $\Delta \leftarrow \max(\Delta, |\max_{a \in A(s)} Q(s,a) - V(s)|)$
  $V(s) \leftarrow \max_{a \in A(s)} Q(s,a)$

注意：在本项目中，使用选项`-i`控制收敛条件，即迭代的深度。

完成`RL指引.doc`文件的问题一和问题二。

---

1. https://gibberblot.github.io/rl-notes/single-agent/value-iteration.html ↩