中国科学技术大学
University of Science and Technology of China

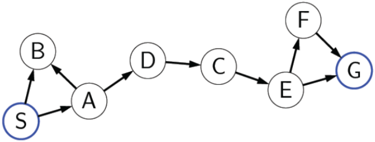
# 人工智能讲义

## 马尔科夫决策过程

March 24, 2022

# Outline

# 搜索问题的改变




## 搜索问题的一点小改变

- 任何一个状态用 $n-D$ 的向量描述，每一维称为状态的一个影响因素或表现特性。
- 现实应用问题中，后继函数 $successors(s,a)$ 在当前状态 $s$ 和行动 $a$ 的联合作用下，产生的后继通常不是 **确定的**？Why?
- 模型的的不确定性，太多的影响因素没有被模型所考虑，用一种不确定的概率 $p$ 来综合所有其它影响因素。

| 节点/状态 | 行动 | 结果状态 | 代价/收益 |
|---|---|---|---|
| ... | ... | ... | ... |

$\Downarrow$

| 节点/状态 (部分) | 节点/状态 (部分)$\Rightarrow p$ | 行动 | 结果状态 | 代价 |
|---|---|---|---|---|
| ... | ... | | | ... |

# 搜索问题的改变

状态s，
行动/输入a

随机的

后继状态1
Successors(s,a)

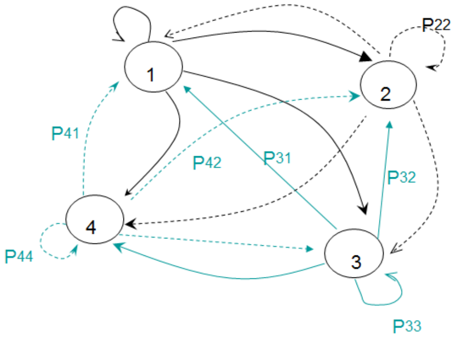
后继状态2
Successors(s,a)

| 节点/状态 | 行动 | 结果状态 | 代价/收益 |
|---|---|---|---|
| ... | ... | ... | ... |

$\Downarrow$

| 节点/状态（部分） | 节点/状态（部分）$\Rightarrow p$ | 行动 | 结果状态 | 代价 |
|---|---|---|---|---|
| ... | ... | | | ... |

# 新搜索问题描述




## 解释说明

- 如图，1,2,3,4 表示 4 个状态
- $p_{ij}$: 表示状态 $i \to j$ 的跳转概率
- 上图只表示了一个行动 $a$ 导致各个状态之间的发生的状态跳转可能性，不同行动得到类似上图的，不同的状态转移图
- 而经典搜索问题，图上的任何一条边就是一个具体的"行动"

# 新搜索问题的解

## 经典搜索问题

- 行动 $a$，确定地让搜索沿一条边前进；
- 解：从初态到终态的一条路径或状态序列，表示为（初态，边 1，边 2，. . . ，边 $m$，终态）或者（初态，状态 1，状态 2，. . . ，状态 $m$，终态）
- 解也可以用行动来表示：（初态，行动 1，行动 2，. . . ，行动 $m$，终态）

## 新搜索问题：MDP

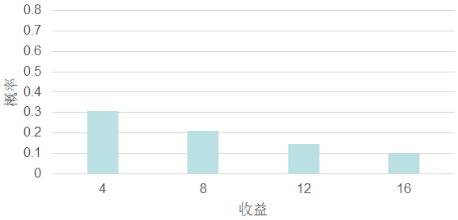
- 行动 $a$，随机地让搜索沿多条边前进；
- 我们的目的是从找到初态到终态的路径/最优路径，但是行动 $a$ 带来后果是随机的，如何描述解？
- 问题的解：每个状态，给出一个“最优行动” $a_i^*$，所谓最优，即尽管行动的后果不确定，但是“平均”看来，该行动得到的好处对于找“初态到终态”的路径是最大的。
- 此时问题的解，不是一条路径，而是“策略”：一组从状态到行动的映射关系。

# 随机数游戏

## 说明

- 游戏为回合制，每个回合开始时，你有两个选择：继续游戏或者退出游戏；
- 若你选择退出游戏，则你得到￥15，且游戏结束；
- 若你选择继续游戏，则你得到￥4，游戏继续；此时产生一个 0∼9 的随机数，若该随机数为 0，1，2，则游戏直接结束；否则，该回合结束；
- 你的决策是什么？为了获得最多的钱！

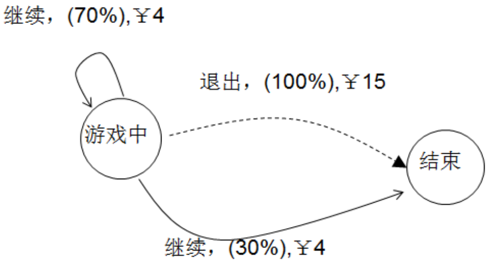
# 随机数游戏




## 问题分析

- 第一回合主动退出收益：$15, 100\%$;
- 第二回合主动退出收益：$19, 70\%$;
- 第三回合主动退出收益：$23, 49\%$;
- ...(第 $k+1$ 回合主动退出)：　　收益：$15 + k * 4$, 获得该收益的概率 $0.7^k$
- 每回合都不主动退出的期望收益，参考上图
  $0.3 * 4 + 0.7 * 0.3 * 8 + 0.7 * 0.7 * 0.3 * 12 + \ldots = 40/3$
  理解为：$n$ 个人参加游戏，都选择不主动退出，大家的平均收益

# 随机数游戏




## 随机数游戏/一种掷骰子游戏: 描述为状态转移图

- 如上图所示，线型表示不同行动
- 概率为 0 的边删除

# MDP 描述

## MDP/马尔科夫决策过程: 一种搜索问题的变型

- $\mathbf{S}$: 状态空间
- 初态: $s_0$
- 行动: $Action(s)$, 给定状态 $s \in \mathbf{S}$, 合法行动集合
- 状态转移概率: $T(s, a, s')$, 从状态 $s$ 出发, 采用行动 $a$, 导致结果状态 $s'$ 的概率;
- 奖励: $Reward(s, a, s')$, 状态转移 $(s, a, s')$ 得到的收益
- 目标测试: $isEnd(s)$

## 解释说明

- 行动和状态转移概率一起定义了经典搜索问题的后继函数。
- 奖励就是经典搜索问题中的路径耗散, 这里我们关注最大化奖励, 区别于最小化路径耗散。
- 马尔科夫性: 行动 $a$ 的确定只和当前状态 $s$ 相关。

# 随机数游戏形式化为 MDP

## 例子：形式化为 MDP

- $\mathbf{S} = \{$ 结束，游戏中$\}$
- $s_0 = \{$ 游戏中$\}$
- 行动：$Action(s) = \{$ 继续，退出$\}$
- 状态转移概率：$T(s, a, s')$

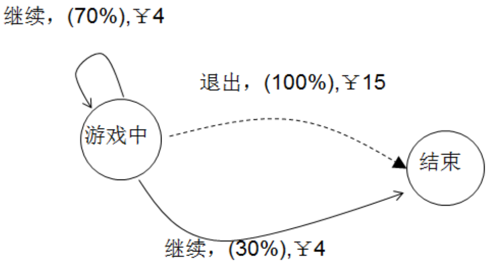
$$T(s, \text{继续}, s') = \begin{array}{cc} \begin{array}{cc} \text{结束} & \text{游戏中} \end{array} & \\ \begin{pmatrix} 1 & 0 \\ 0.3 & 0.7 \end{pmatrix} & \begin{array}{l} \text{结束} \\ \text{游戏中} \end{array} \end{array} \quad T(s, \text{退出}, s') = \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix}$$

- 奖励：$Reward(s, a, s')$，状态转移 $(s, a, s')$ 得到的收益

$$Reward(s, \text{继续}, s') = \begin{pmatrix} 0 & 0 \\ 4 & 4 \end{pmatrix} \quad Reward(s, \text{退出}, s') = \begin{pmatrix} 0 & 0 \\ 15 & 0 \end{pmatrix}$$

- 目标测试：$isEnd(s)$

# MDP 进一步理解




## 理解 MDP：定义在有向图上的搜索

- $n$ 个节点，每个节点的每个行动/策略会有一定的概率转移到其它的状态，故每个节点的每个行动/策略有 $n$ 条“出边”，每条边用“行动/策略，概率，收益”来标记
- 经典搜索问题是 MDP 在概率只能取值为 0 或 1 时的特例

# 状态转移

状态转移：$s \to s'$

- 任意给定一个状态 $s$ 和任意一个行动 $a$，其状态转移到一个可能的"后继状态"集合，而转移到这些可能后继状态的概率形成一个分布，即概率和为 1
- 用公式描述，即$\sum_{s' \in \mathbf{S}} T(s, a, s') = 1$
- 若重新定义概念"后继状态"：若 $s'$ 是 $s$ 的后继状态，当且仅当 $T(s, a, s') > 0$
- 那么，经典搜索问题相当于，对任意给定的一组状态 $s$ 和行动 $a$，有唯一后继状态 $s'$ 或没有后继状态。

# MDP 的解

**解：对任何状态 $s$，定义一个最优行动 $a^*$**

- MDP 的解被称为“策略”，映射表
- 通常会造成许多从初态到终态的随机路径；
- 路径的数目是状态数目的指数函数；

**如何评价 MDP 的解**

- 策略评估！
- 经典搜索问题，任何行动的结果是唯一确定的，所以只有唯一一条从初态到终态的最优路径。

# 奖励、收益和解

## 行动收益

- 对每个行动 $s$，定义其收益为它导致状态转移带来的奖励：$U(s,a,s') = Reward(s,a,s')$
- 该收益获得的概率是 $T(s,a,s')$
- 该行动的期望收益：$\sum_{s'\in\mathbf{S}} T(s,a,s')\,U(s,a,s')$

## 路径收益

- 路径 $s_0, a_0, s_1, a_1, \ldots, s_e$ 上所有行动带来的收益之和；

## 策略收益

- 一个策略（记为 $\pi$）会造成多条从初态到终态的路径，每条路径的出现概率可能不一样，收益可能也不一样；
- 所有从初态到终态的路径的收益期望，我们将之定义为 策略的价值，即策略收益，记为 $V_\pi$。

# MDP 的策略评估

## 计算一个策略的价值/值：枚举法

- 计算每条随机路径的收益 $u_i$
- 计算每条随机路径的出现概率 $p_i$
- 求加权和 $\sum_i u_i p_i$

## 存在问题

- 有多少条随机路径？无穷条路径！如随机数游戏，可能永远不会退出；可造成任意长度的路径；
- 因此，上述计算一个策略的价值的方法，实际上是无法实现的；时间需求太大/无限大。
- 策略评估存在困难，基于策略评估的问题：如何在多个策略中选择“最优策略”？会更困难。

# 随机数游戏的策略价值

## 例子：如何计算一个策略的价值

- 记策略/policy 为 $\pi = \{$游戏中：继续, 退出：*$\}$
- 策略 $\pi$ 的收益为：
  $V_\pi(\text{游戏中}) = 30\% \times 4 + 70\% \times (4 + V_\pi(\text{游戏中}))$，解之，得到 $V_\pi(\text{游戏中}) = 40/3$

## 推广：计算策略价值的方法

- 思路：找到递推公式，后解之；
- 通用地推公式：$V_\pi(s) = \sum_{s' \in \mathbf{S}} T(s, a, s')[U(s, a, s') + V_\pi(s')]$
- 上述公式中 $s$ 表示初态/当前状态；每个状态都可以列出一个上述递推式，联立这些递推式，得到方程组（$n$ 个状态，$n$ 个未知数 $V_\pi(s)$,$n$ 个线性方程)，解之。

# 策略评估

## 策略评估算法思想：解递推方程组的方法

- 初始化所有状态的策略价值 $V_\pi(s) = 0$
- Repeat $T$ 次:
  - 对每个状态 $s \in \mathbf{S}$，执行:
    - 利用递推方程循环更新 $V_\pi(s) = \sum_{s' \in \mathbf{S}} T(s, a, s')[U(s, a, s') + V_\pi(s')]$

## 算法评述

- 算法停止条件：相邻两次对 $V_\pi(s)$ 的更新足够小，不妨设为 $T$ 次
- 仅仅需要存储最近一次的 $V_\pi(s)$ 的值
- 时间代价：$O($ 状态数目 $\times$ 循环次数 $\times$ 后继状态数目$)$

## 策略的数目

策略 $\pi : s \in \mathbf{S} \rightarrow a \in Action(s)$

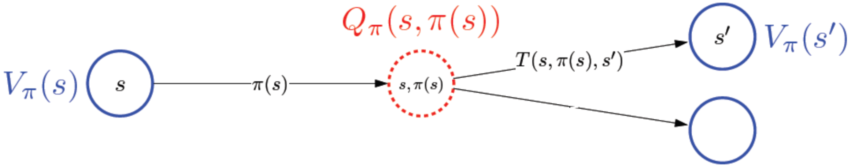
- 即 $a = \pi(s)$
- 策略评估算法能计算一个策略的价值 $V_\pi(s)$

策略空间的大小

- 假设有 $n$ 个状态 $s_1, s_2, \ldots, s_n$
- 策略空间的大小：$\Pi_{s_i \in \mathbf{S}} |Action(s_i)|$，状态数目 $|\mathbf{S}|$ 的指数函数

如何找到最优策略？

# Q-value




## Q-value

- Q-value: $Q_\pi(s,a)$ 定义为从状态 $s$ 出发，采用行动 $a$ 后继续采用策略 $\pi$ 的价值/收益；
- Q-value 与 $V_\pi$ 的区别：在状态 $s$ 时，采用了不同的行动，故 $V_\pi(s)$ 仅仅是 $\pi, s$ 的函数，而 $Q_\pi(s,a)$ 是 $\pi, s, a$ 的函数
- 为什么要引入 Q-value? 讨论在状态 $s$ 下，哪一个行动会得到更多的收益。

$V_\pi(s) = \sum_{s'\in \mathbf{S}} T(s,a,s')[U(s,a,s') + V_\pi(s')] = \sum_{s'\in \mathbf{S}} T(s,\pi(s),s')[U(s,\pi(s),s') + V_\pi(s')]$
$Q_\pi(s,a) = \sum_{s'\in \mathbf{S}} T(s,a,s')[U(s,a,s') + V_\pi(s')]$

# MDP：策略改进算法

## 策略改进算法

- 输入一个策略 $\pi$，输出一个更新的改进策略 $\pi_{new}$，充分利用马尔科夫性
- 对任意状态 $s$，执行下述操作：
  - 对不同的行动 $a \in Action(s)$，计算 $Q_\pi(s, a)$；
  - 更新 $\pi_{new} = \arg\max_{a \in Action(s)} Q_\pi(s, a)$

## 评述

- 优点类似爬山法，把状态 $s$ 所有可能的行动都尝试一遍，找到期望奖励最大的行动，用来更新策略 $\pi$。

# 策略迭代算法

## 计算最优的策略

- $\pi \leftarrow$ 任意初始化值
- Repeat $T_1$ 次（或者 $\pi$ 不再变化为止）:
  - 评估策略，计算 $V_\pi$;
  - 执行策略改进算法，得到 $\pi_{new}$
  - $\pi \leftarrow \pi_{new}$

## 算法评述

- 保证全局最优性
- 时间复杂度和初始解、状态数、行动数、后继状态数、迭代次数等相关

## 问题:

- 每次循环都要精确计算“经历过”的每个策略的价值，没有必要！我们只要最优值！

# 值迭代算法

## 计算最优的策略

- 对所有状态 $s$ 初始化 $V_{opt}^{0}(s) \leftarrow 0$;
- for $t = 1, 2, \ldots, T_0$ 次:
  - 对每一个状态 $s$，执行:
    - $V_{opt}^{t}(s) \leftarrow \max_{a \in Action(s)} \sum_{s'} T(s, a, s')[Reward(s, a, s') + V_{opt}^{t-1}(s')]$

## 值迭代算法

- 把策略评估和策略改进两个独立的过程结合在一起，放入一个过程中完成
- 同样能保证全局最优性
- 对中间经历过的策略没有完整评估过。

# MDP：折扣因子

计算路径收益时，因为未经过的路径是否会经历，不确定

- 所以，添加一个折扣因子 $\lambda$，来重新计算/估计路径收益

MDP：引入折扣因子的变化

- 递推方程：$V_\pi = \sum_{s'} T(s, a, s')[Reward(s, a, s') + \lambda V_\pi(s')]$
- $\lambda = 1$ 的特殊情形