FZOI

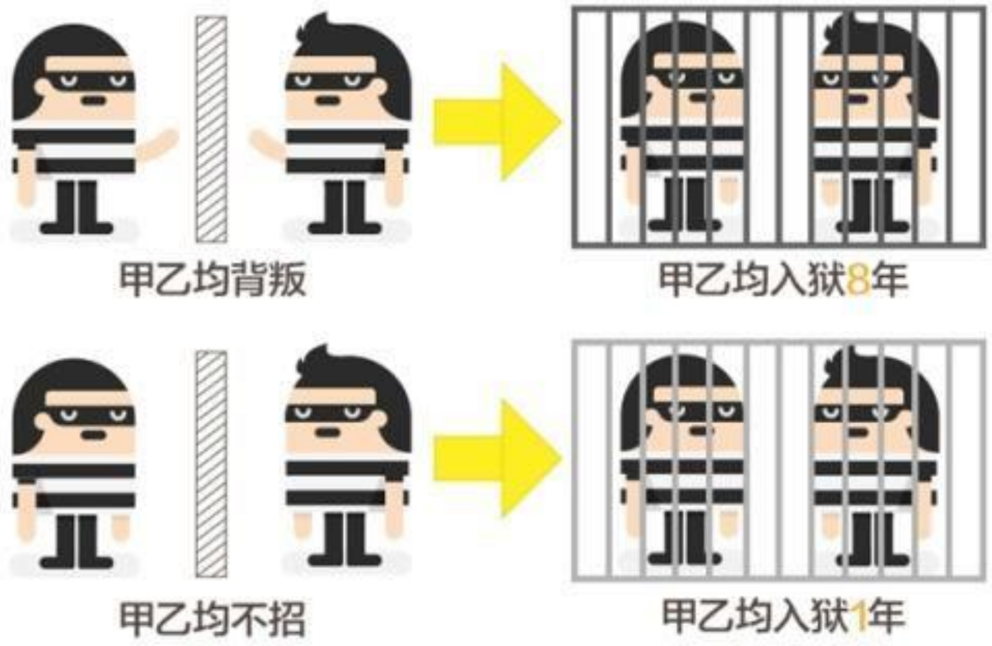
信息学

# 博弈论入门

(入门了是不是就可以自己修行了=_=)

# 囚徒困境

两个囚徒被关在监狱中，并彼此分开，无法接触。检察官告诉他们，如果两人都坦白事实，就都判刑 8 年;若只有一人坦白，则坦白者无罪释 放，不坦白者重判 15 年;若两人都不坦白，都判刑 1 年。若两个囚犯都足够聪明，则会采取什么样的策略，才能使自己(不考虑另一个囚徒)尽可能判刑减少?

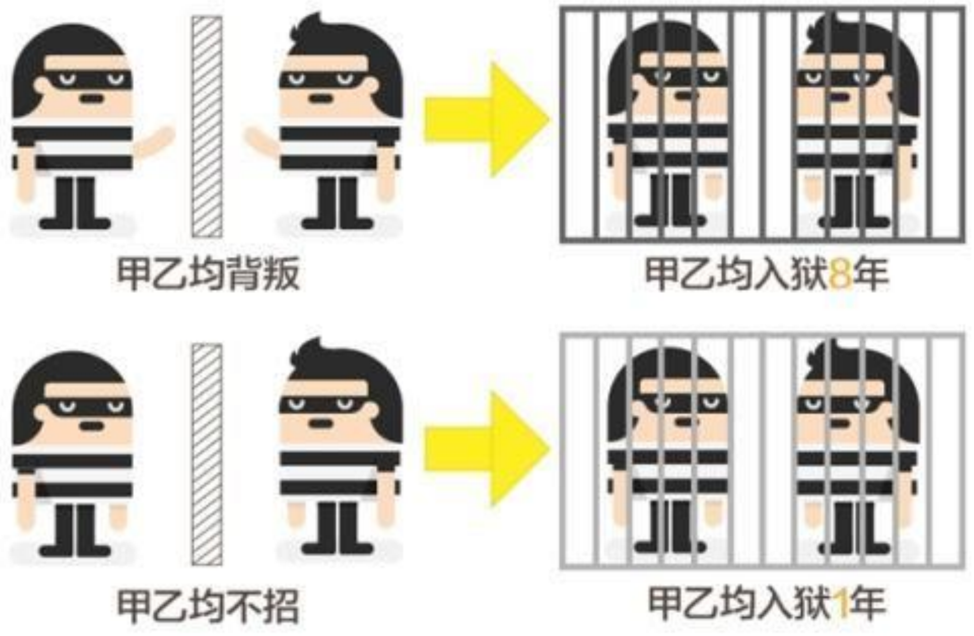



分析一下
若对方坦白：自己坦白：8年，比自己抗争15年好。
若对方抗争：自己坦白：0年，比自己抗争 1年好。

# 囚徒困境



两个囚徒被关在监狱中，并彼此分开，无法接触。检察官告诉他们，如果两人都坦白事实，就都判刑 8 年;若只有一人坦白，则坦白者无罪释 放，不坦白者重判 15 年;若两人都不坦白，都判刑 1 年。若两个囚犯都足够聪明，则会采取什么样的策略，才能使自己(不考虑另一个囚徒)尽可能判刑减少?




两人按照最优策略思考，则会得到2人坦白的局面
尽管这不是总体最优的（都不坦白总体最优）
这种矛盾的状态称为囚徒困境（其实这个模型在社会上普遍存在）

囚徒困境的2人都坦白这一局面，就是纳什均衡点。

什么是纳什均衡?

在一个博弈的过程中，存在这样一种策略，使得在外界条件、他人决策变化的情况下，任何人都没有理由打破这种均衡的状态。

前提：最优策略博弈

两只猪被关在一个猪圈中，猪圈的一侧有一个踏板，另一侧是食槽。
任何一只猪去踩踏板，都会立即在食槽中喂食。
若大猪踩了踏板，小猪会先开始吃，等到大猪赶回来时，还有一半食料留给大猪；
若小猪踩踏板，大猪在小猪回来时刚好将食吃完。试问，两只猪那个占优势？

问题简化一下：
大猪踩踏板，大猪小猪1/2
小猪踩踏板，小猪0 大猪1

分析

小猪不会动
因为不踩比踩不会更坏

结论

小猪占优势

# 智猪问题-改



大猪踩踏板，大猪3/4小猪1/4
小猪踩踏板，小猪大猪1/2

结论

猪们都争着踩

修改纳什均衡点，提高猪们的积极性

# 智猪问题-模型应用



企业中的员工。

一个企业的奖惩策略不合适，员工的纳什均衡点在“躺平”位置企业是无法长期发展的。

修改奖惩策略（例如多劳多得）让每一个员工（大猪/小猪）的纳什均衡点变化到努力工作（踩板）上。

- 囚徒困境与智猪问题，
- 其实都是一种分析多玩家，采取何种策略结果最优的博弈过程。

**博弈论，是经济学的一个分支，主要研究具有竞争或对抗性质的对象，在一定规则下产生的各种行为。**
**博弈论考虑游戏中的个体的预测行为和实际行为，并研究它们的优化策略。**
**通俗地讲，博弈论主要研究:在一个游戏中，进行游戏的多位玩家的策略。**
OI-Wiki

# 合作？竞争？有限次博弈

- 我们应当相信，无论中间有多少不公与挫折，社会的合作与进步终究是历史的必然。

假定进行囚徒困境100次。
最终的结果大家分别关了多少年？

考虑第100次情况，因为当前决定不会影响后面的决策。
所以肯定考虑坦白。
而正因为第100次的坦白，第99次也会选择坦白（不会取得更差）
结论：一直坦白（而不是合作抵抗）
但是这与上方红字矛盾了。

# 合作？竞争？无限次博弈



- 我们应当相信，无论中间有多少不公与挫折，社会的合作与进步终究是历史的必然。

假定进行囚徒困境无限次。
最终的结果大家分别关了多少年？

有科学家用计算机模拟了十几种策略。找出优胜者（关的少的）
Tit-for-tat策略非常棒：首次合作，然后把对方上次的策略当作自己的策略。
（ Tit-for-tat：以牙还牙）

# 一点心得



- 博弈研究是博弈过程中的最优策略。
- 假定按照最优策略博弈，会存在一个纳什均衡点。
- 无限次博弈之下：Tit-for-tat策略非常棒：首次合作，然后把对方上次的策略当作自己的策略。

# 博弈经典问题
# Nim游戏

- 有 n 堆石子，第 i 堆的个数为 ai 。两人轮流从任意一堆中拿走任意个 石头，如果轮到某个人时所有的石子堆都已经被拿空了，则输。

- - 两人均无失误，都采用最优策略拿石头。
- - 问什么情况下必胜?

提示：NIM博弈不存在平局，只会有先手必败，先手必胜2种情况

# Nim 游戏 – 分析



假设有n堆石子

- 当 $A_1 \oplus A_2 \oplus A_3 \ldots \oplus A_n = x \neq 0$时
- 假设x的最高位为k
- 说明至少有1堆石子$A_i$的k位为1
- 只需要从$A_i$中取到$x \oplus A_i$个石子，使得
- $A_1 \oplus A_2 \oplus A_3 \ldots \oplus A_n = 0$
- 这样，下一轮对手操作的时候就不得不打破xor=0的局面，此时你只需要跟着对手完成同样的操作即可。

所以

NIM先手必胜的条件，当且仅当：

$$A_1 \oplus A_2 \oplus A_3 \ldots \oplus A_n = x \neq 0$$

同理

NIM先手必输的条件，当且仅当：

$$A_1 \oplus A_2 \oplus A_3 \ldots \oplus A_n = 0$$

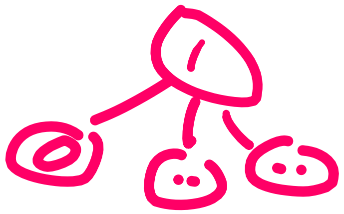
Bouton’s 理论

# Nim 游戏 – 一些概念介绍与推广

- 凡是n堆物品，每堆任意个，2个玩家先后轮流取任意个称为NIM游戏。
- NIM游戏无平均，只有先手必胜/必败两种情况。
  - 第一个取的称为先手，第二个取的称为后手
  - 游戏过程的每个状态称为局面
- 这里讨论的博弈问题一般考虑理想情况
  - 2个人无失误
  - 均采用最优策略博弈

# NIM游戏-深入分析

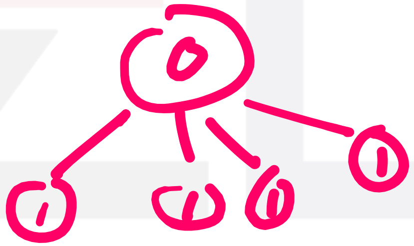




$SG_x != 0$

- 若当前状态为，先手必胜，则其子局面至少有一个先手必败
- 若当前状态为，先手必败，则其子局面全部为先手必胜。

$SG_x == 0$



- 多堆石头情况下取成（0，0,…,0，0）这个状态是必胜状态。

# 一些概念介绍与推广——公平组合游戏ICG



若一个游戏满足:

1.由两名玩家交替行动。

2.在游戏进程的任意时刻，可以执行的合法行动与轮到哪名玩家无关。

3.不能行动的玩家判负。

则称该游戏为一个公平组合游戏。

# 一些概念介绍与推广——公平组合游戏ICG –特点



OI中一般研究公平组合游戏 ~~（以及非公平组合游戏）~~

公平组合游戏特点

1.某一局面的决策集合只与局面有关，与决策者无关。

2.同一局面不能多次抵达。

NIM游戏属于公平组合游戏。

围棋属于么?

# 一些概念介绍与推广——有向图游戏（博弈图）

- 任意一个公平组合游戏，都可将每个局面看作顶点，每个局面通过有向边指向其子局面，进而抽象出一个有向图游戏。
  - ICG的图一定是一个有向无环图

**有向图游戏：一个有向无环图，图中有一个唯一的起点。起点上放一枚棋子。两名玩家交替地把这个棋子沿着有向边进行移动。每次可以移动一步，无法移动者判负。**

感性理解（不准确）

# 一个操作：mex



$$\operatorname{mex}(S) = \min\{x\} \quad (x \notin S, x \in N)$$

mex{0,1,2,4}=3、mex{2,3,5}=0、mex{}=0。

Mex的值= 不属于集合S中的 最小非负整数

# 一些概念介绍与推广——SG函数

**任何一个公平组合问题都可以转化为有向无环图进行处理。**

对于图上每个顶点，定义其SG函数为

sg(x)=mex({sg(y)}) y是x的子局面

一个局面的SG函数值，为其后继局面SG函数值集合的mex

例如1堆石头任取x个石头（1维Nim游戏）

SG(0)=0 (0堆石头没有后继)

SG(1)=mex(SG(0)) =1

SG(2)=mex(SG(0),SG(1))=2

…

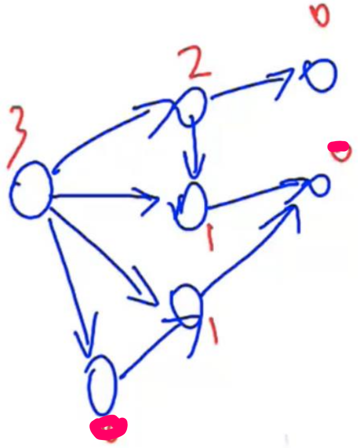
SG(n)=n

# 一些概念介绍与推广——SG函数

**任何一个公平组合问题都可以转化为有向无环图进行处理。**

对于图上每个顶点，定义其SG函数为
sg(x)=mex({sg(y)}) y是x的子局面

一个局面的SG函数值=其后继局面SG函数值集合的mex




红色为sg(x)值

叶子结点sg(x)=0

**任何多个公平组合问题都可以转化为多个有向无环图进行处理。**

定义SG(G1) SG(G2) SG(G3) … SG(Gm)
分别是多个有向无环图的结果.

有结论

SG 定理

$$SG(G1,G2,G3..Gm) = SG(G1) \oplus SG(G2) \oplus SG(G3) \oplus \dots \oplus SG(Gm)$$

**SG函数：把复杂博弈问题拆分成多个Nim游戏进行处理**

SG定理

若sg(G)>0,则先手必胜
若sg(G)==0,则先手必败

结论证明：https://blog.csdn.net/qq_39239844/article/details/81100916

# 复杂博弈问题的处理方法



1把原游戏分解成多个独立的子游戏

2分别考虑每一个子游戏，计算其SG值 How?

3利用下列公式，合成原游戏的SG值

$$SG(G1,G2,G3..Gm) = SG(G1) \oplus SG(G2) \oplus SG(G3) \oplus \ldots \oplus SG(Gm)$$

4利用SG定理得出结论

若sg(G)>0,则先手必胜，若sg(G)==0,则先手必败

# 如何计算SG函数?

1.如果你的可选操作为任意步数

  则SG(x)=x

2.如果你的可选操作为1～m的连续整数

  则直接取模 SG(x) – x%(m+1)

3.如果你的操作数为一系列不连续的数

  则。。。打表（优先预处理） OR DFS（如果不能打表再DFS）

# 如何计算SG函数?



## 打表

```
//f[]: 可以取走的石子个数
//sg[]:0~n的SG函数值
//hash[]:mex{}
int f[N],sg[N],hash[N];
void getSG(int n)
{
    int i,j;
    memset(sg,0,sizeof(sg));
    for(i=1;i<=n;i++)
    {
        memset(hash,0,sizeof(hash));
        for(j=1;f[j]<=i;j++)
            hash[sg[i-f[j]]]=1;
        for(j=0;j<=n;j++)    //求mes{}中未出现的最小的非负整数
        {
            if(hash[j]==0)
            {
                sg[i]=j;
                break;
            }
        }
    }
}
```

## DFS

```
//注意 S数组要按从小到大排序 SG函数要初始化为-1 对于每个集合只需初始化1遍
//n是集合s的大小 S[i]是定义的特殊取法规则的数组
int s[110],sg[10010],n;
int SG_dfs(int x)
{
    int i;
    if(sg[x]!=-1)
        return sg[x];
    bool vis[110];
    memset(vis,0,sizeof(vis));
    for(i=0;i<n;i++)
    {
        if(x>=s[i])
        {
            SG_dfs(x-s[i]);
            vis[sg[x-s[i]]]=1;
        }
    }
    int e;
    for(i=0;;i++)
        if(!vis[i])
        {
            e=i;
            break;
        }
    return sg[x]=e;
}
```