# Game Theory

Owaski

雅礼中学

June 19, 2017

# Preface

今天讲一下 OI 中跟游戏论相关的内容，游戏论一直不算是一个热门话题，可能出于出题难度，也可能出于其他位置原因，whatever，但是基本的东西我们还是要知道的，高级的也应略有涉猎。

# Contents


1. Impartial Combinatorial Games
   - Nim
     - Examples
   - Coin Turning Games
     - Examples
   - Green Hackenbush
2. Two-Person Zero-Sum Games
3. Games on Bipartite Graph

# Definition

公平组合游戏具有以下特征：

- 1. 两个玩家都采取对自己最有利的决策，且轮流作出决策。
- 2. 玩家知道之前的所有决策。
- 3. 游戏的任意时刻，不包含随机成分。
- 4. 结局只有赢或输，没有平局。

在下文中将介绍多种经典模型。

# Nim

我们先从 Nim 游戏引入。

Nim 游戏指的是两个玩家轮流从若干个石头堆中拿走石头。每一轮一名玩家只能从某一堆中拿走至少一个石头。当某一方无法拿走石头的时候判定为输。

我们需要引入 SG 函数来解决这个问题。

## Sprague–Grundy Theorem

先将游戏中的每个局面抽象成一个有向图中的一个点，游戏中的一次决策看成一条边，那么每次游戏都可以看成一条从入度为 0 的点走到一个出度为 0 的点的路径。

SG 函数是对图中每个结点的评估函数，定义如下：

$$SG(x) = mex\{SG(y)|\exists(x, y) \text{ in graph}\}$$

其中 $mex$ 是定义在整数集合上的操作：

$$mex(S) = \min\{k|k \notin S, k \in N\}$$

在 Nim 游戏中，一个 $n$ 个石头的堆的 SG 值就是为 $n$。

# Sprague–Grundy Theorem

SG 函数有三个性质：

- 1. 若 $SG(x)=0$，那么 $x$ 所有后继点的 SG 值都不为 0。
- 2. 若 $SG(x)\neq 0$，那么 $x$ 的后继点中至少有一个点的 SG 值为 0。
- 3. 结束状态的 SG 值都为 0。

这样的性质可以帮助我们判定一个局面是否有必胜策略。如果 $SG(x)=0$，那么无论接下来走哪一步 $y$，$SG(y)>0$，另一个人都可以将其走到一个 $z$ 满足 $SG(z)=0$，结合最终的失败状态的 SG 值都为 0，$SG(x)=0$ 的时候是必败的，相反 $SG(x)>0$ 的时候是必胜的。

# Sprague–Grundy Theorem

现在考虑怎么样将多个石堆并起来考虑。

我们定义游戏的和：有若干个同时进行的游戏，玩家每轮可以选择任意一个游戏进行决策，最终无法操作的玩家输。那么 Nim 游戏相当于若干个单堆石头的和。

先给出结论，对于若干个游戏图 $G_1, G_2, \cdots, G_n$，有：

$$SG(G_1 + G_2 + \cdots + G_n) = SG(G_1) \oplus SG(G_2) \oplus \cdots \oplus SG(G_n)$$

思考一下怎么证明这个结论。

# Sprague–Grundy Theorem Proof

设 $G = G_1 + G_2 + \cdots + G_n$，考虑 $G$ 的转移。因为每次只能在一个游戏中操作，因此 $G$ 具有的转移为：

$$G_1 + G_2 + \cdots + G_i' + \cdots + G_n \qquad \text{for all } 1 \leq i \leq n$$

其中 $G_i'$ 为 $G_i$ 的一个转移。设 $F_G$ 为 $G$ 的转移集合，设 $b = SG(G_1) \oplus SG(G_2) \oplus \cdots \oplus SG(G_n)$，那么我们要证明的其实就是：

- 1. $\forall_{a<b, a \in N}, \exists_{g \in F_G} SG(g) = a$
- 2. $\forall_{g \in F_G} SG(g) \neq b$

## 1. $\forall_{a<b,a\in N}, \exists_{g\in F_G} SG(g) = a$

记 $d = b \oplus a$，$d$ 最高位为 $k$，一定存在 $SG(G_i)$ 满足第 $k$ 位上为 1。

因为 $SG(G_i) \oplus d < SG(G_i)$，因此必然存在 $G_i' \in F_{G_i}$，满足 $SG(G_i') = SG(G_i) \oplus d$。

因为 $d = b \oplus a \Rightarrow a = d \oplus b$，即：

$$a = SG(G_1) \oplus SG(G_2) \oplus \cdots \oplus SG(G_i) \oplus d \oplus \cdots \oplus SG(G_n)$$

$$a = SG(G_1) \oplus SG(G_2) \oplus \cdots \oplus SG(G_i') \oplus \cdots \oplus SG(G_n)$$

很明显 $G_1 + G_2 + \cdots + G_i' + \cdots + G_n \in F_G$，因此得证。

## 2. $\forall_{g \in F_G} SG(g) \neq b$

如果存在：

$$SG(G_1) \oplus SG(G_2) \oplus \cdots \oplus SG(G_i') \oplus \cdots \oplus SG(G_n) = b$$

那么有：

$$SG(G_1) \oplus SG(G_2) \oplus \cdots \oplus SG(G_i') \oplus \cdots \oplus SG(G_n) =$$

$$SG(G_1) \oplus SG(G_2) \oplus \cdots \oplus SG(G_i) \oplus \cdots \oplus SG(G_n)$$

即：

$$SG(G_i') = SG(G_i)$$

矛盾。

# Take-Away

现在有一个石堆，包含 $n$ 个石子，每次可以拿走 $1 \sim k$ 个石子，两个玩家轮流操作，最后无法操作的玩家输。

# Take-Away Solution

很显然 $SG(n) = n \bmod (k+1)$。

# BZOJ3759 Hungergame

给 $n$ 个箱子，每个箱子内有一些石子，两个人轮流操作，每个玩家可以进行以下操作之一：

- 1. 打开任意多的箱子。
- 2. 从一个打开的箱子中拿走任意多的石子。

不能操作者判负，求先手是否必胜。

$n \le 20$

# BZOJ3759 Hungergame

定义必败状态：当前打开的箱子中石子异或和为 0，没打开的箱子中不存在一个子集满足异或和为 0。这个很容易证明。

那么只要一开始存在子集满足异或和为 0，那么先手必胜。

# Misère Nim

取到最后一个石子的玩家输，其他条件和普通 Nim 一样。

# Misère Nim Solution

先手必胜条件:

- 1. SG 和为 0 且不存在一个石堆中石子的数量大于 1。
- 2. SG 和不为 0 且存在一个石堆中石子的数量大于 1。

# Misère Nim Proof 1

考虑不存在石堆的大小大于 1 的情况。

这种情况下，如果 SG 和为 0，说明有偶数个 1，对方一定会取到最后一个，先手必胜。SG 不为 0 情况反之。

# Misère Nim Proof 2

考虑存在石子数量大于 1 的情况。

如果大于 1 的石堆有一个，那么 SG 和大于 0，我们选择取到只剩 0 或 1 个，根据 1 的石堆数量决定，先手必胜。

如果大于 1 的石堆至少有两个，如果 SG 和大于 0，我们按照常规 Nim 的最佳策略操作，直到我们的局面只剩一个石堆大于 1（为什么？），然后和上面操作一样，先手必胜。如果 SG 和等于 0，那么一次操作后 SG 大于 0，留给对面的是必胜局面。

因此这种情况下 SG 大于 0 即可保证先手必胜。

# Moore's $Nim_k$

每次可以在至多 $k$ 堆石子中取任意多个石子，其他条件和普通 Nim 一样。

# Moore's Nim$_k$ Solution

考虑 $n$ 堆石子的二进制展开，令 $x_i$ 表示第 $i$ 堆石子的数量，设 $x_i = \sum_{j=0}^{m} g_{ij} \cdot 2^j$，如果满足 $\forall 0 \le j \le m, \sum_{i=0}^{n-1} g_{ij} \equiv 0 \pmod{k+1}$，那么先手必败，否则先手必胜。

# Moore's Nim$_k$ Proof

首先我们知道当没有石子的时候是先手必败局面。

当满足 $\forall 0 \le j \le m, \sum_{i=0}^{n-1} g_{ij} \equiv 0 \pmod{k+1}$ 的时候，接下来拿走任意的石子都不会满足所有列上的和 mod $k+1$ 为 0。因为我们一次最多改变 $k$ 个石堆的个数，因此每一列的 1 的个数的变化是 $[-k, k]$ 的。采用反证法，如果每一列的变化都为 0，表示每一个 $0 \to 1$ 都对应了 $1 \to 0$，而 $0 \to 1$ 在高位必定对应了一个 $1 \to 0$，$1 \to 0$ 在它所在的列又对应了一个 $0 \to 1$，而 $0 \to 1$ 在更高位必定又对应了一个 $0 \to 1$，以此类推，在最高位会存在 $0 \to 1$ 的情况，不合法，因此不会每一列的变化都为 0。

# Moore's Nim$_k$ Proof

当满足 $\exists 0 \le j \le m, \sum_{i=0}^{n-1} g_{ij} \not\equiv 0 \pmod{k+1}$ 的时候，存在操作使得接下来多有列的和 mod $k+1$ 为 0。考虑和 mod $k+1$ 不为 0 的最高列，这一列将 $m(m \le k)$ 个 $1 \to 0$，考虑下一个不为 0 的列，设之前 $m$ 个更改的石堆在这一列有 $a$ 个 1，$b$ 个 0，设这一列 mod $k+1$ 为 $r$，分情况讨论：

- 1. $a \ge r$，直接在 $a$ 个 1 中选出 $r$ 个 1 变成 0。
- 2. $b \ge k+1-r$，直接在 $b$ 个 0 中选出 $k+1-r$ 个 0 变成 1。
- 3. $a < r, b < k+1-r$，考虑将另外的 $r-a$ 个和原来的 $a$ 个 1 变成 0，考虑这个时候我们操作的石堆数 $= m+r-a = a+b+r-a = b+r < k+1-r+r = k+1$，合法。将 $m$ 更新成 $m+r-a$。

# Misère Moore's Nim$_k$

取到最后一个石子的玩家输，其他条件和 Moore's Nim$_k$ 一致。

## Misère Moore's $\text{Nim}_k$ Solution

先手必胜的条件：

- 1. 当不存在 $>1$ 的石堆的时候，堆数 $\bmod k+1 \neq 1$。
- 2. 当存在 $>1$ 的石堆的时候，存在某一列和 $\bmod k+1 \neq 0$

# Misère Moore's Nim$_k$ Proof 1

当不存在 $>1$ 的石堆的时候，如果堆数 $\bmod k+1 \neq 1$，先手可以第一步使得堆数 $\bmod k+1=1$，然后对于后手每次移除 $r$ 堆，先手对应的移走 $k+1-r$ 堆，这样最后一堆就会留给后手了。

# Misère Moore's Nim$_k$ Proof 2

当存在 $>1$ 的石堆且存在某一列和 $\bmod k+1 \neq 0$ 时，我们采用 Moore's Nim$_k$ 的最优策略，这样会保证留给对方每一列和为 0 的局面，这种局面包含的大于 1 的石碓个数 $\geq k+1$，因此 $>1$ 的石堆个数 $[1,k]$ 的局面会留给我们，用类似 Moore's Nim$_k$ Proof 2 的方法能够证明这种局面可以达到 Misère Moore's Nim$_k$ Proof 1 中对方的必败局面。

# Staircase

有 $n$ 层阶梯，编号 $1 \sim n$，每层阶梯上有一些硬币。现在有两个玩家轮流操作，每次操作可以将第 $j$ 层阶梯上任意多个（至少一个）硬币放到 $j-1$ 层阶梯上，第 0 层阶梯即为地板。将最后一枚从阶梯移到地板上的玩家胜利。

# Staircase Solution

只要将奇数层的硬币数看成 Nim 堆即可。假设我方现在是先手，我方按照 Nim 的最优策略移动奇数层的硬币，然后对方有两种上选择：

- 1. 如果移动奇数层的硬币，那么我们继续移动奇数层的硬币。
- 2. 如果移动偶数层的硬币，我们将对方移动的硬币移动到偶数层。

可以发现这样的话，如果对面移动奇数层的硬币，实际上就是在做传统的 Nim 游戏，如果对面移动偶数层的硬币，我们有将其无效化的方法，因此我们只需要取奇数层上的硬币即可。至于为什么是奇数层，是因为地板是 0 层。

# POI2004 Gra

有 $m$ 个格子排成一行，从左到右编号 1 到 $m$，其中 $n$ 个给定的格子里有石子，且编号为 $m$ 的格子里没有石子。两个人轮流操作，每次操作要求选择一个石子，石子会移动到它右边第一个不含石子的格子里。将某个石子移动到编号为 $m$ 的格子的人胜利，问先手有多少种操作方案能使先手必胜。

$2 \le m \le 10^9, 1 \le n \le 10^6, n < m$

# POI2004 Gra Solution

如果 $m-1$ 有石子，那么先手获胜，否则会尽量避免将石子放到 $m-1$ 位置上去。

当所有石子排列在 $m-n-1 \sim m-2$ 上，后手只能将某个石子移到 $m-1$，先手再移一步即获胜。因此我们只要考虑将 $n$ 个石子移到 $m-n-1 \sim m-2$ 上即可。

考虑将连续的一段石子看成一堆，空白格子看成空白阶梯，那么每次操作即是将某一堆石子中取一部分放到下一级阶梯上，因此直接转换成了 Staircase 的模型。

# Wythoff Game

国际象棋中的皇后现在位置在 $(n, m)$，两个玩家轮流操作，每次操作可以移动皇后到：

- 1. $(a, m), 0 \le a < n$
- 2. $(n, b), 0 \le b < m$
- 3. $(n - x, m - x), 1 \le x \le \min(n, m)$

将皇后移动到 $(0, 0)$ 的玩家获得胜利。

## Wythoff Game

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 7 | 8 | 6 | 9 | 0 | 1 | 4 | 5 |
| 6 | 7 | 8 | 1 | 9 | 10 | 3 | 4 |
| 5 | 3 | 4 | 0 | 6 | 8 | 10 | 1 |
| 4 | 5 | 3 | 2 | 7 | 6 | 9 | 0 |
| 3 | 4 | 5 | 6 | 2 | 0 | 1 | 9 |
| 2 | 0 | 1 | 5 | 3 | 4 | 8 | 6 |
| 1 | 2 | 0 | 4 | 5 | 3 | 7 | 8 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |

这是打出来的 SG 表，首先表是对称的，我们只考虑上三角矩阵，那么先手必败的点编号为 $(0,0),(1,2),(3,5),(4,7)\cdots$。

# Wythoff Game

Wythoff 给出了结论，第 $n$ 个必败点的坐标为 $(\lfloor n\phi \rfloor, \lfloor n\phi^2 \rfloor)$，其中 $\phi = \frac{1+\sqrt{5}}{2}$。怎么判断一个点是不是必败点呢？

# Wythoff Game

Wythoff 给出了结论，第 $n$ 个必败点的坐标为 $(\lfloor n\phi \rfloor, \lfloor n\phi^2 \rfloor)$，其中 $\phi = \frac{1+\sqrt{5}}{2}$。怎么判断一个点是不是必败点呢？

其实很简单，首先如果是下三角的我们先翻到上三角，我们注意到 $\lfloor n\phi^2 \rfloor = n + \lfloor n\phi \rfloor$，因此对于一个点 $(x, y)$，其中 $x \leq y$，我们令 $n = y - x$，只要检查一下 $x, y$ 是否是 $\lfloor n\phi \rfloor, \lfloor n\phi^2 \rfloor$ 即可。

# Take-and-Break

Take-and-Break 是允许拿走或者特定情况下将一堆分成分成若干堆的游戏，看一道很老很经典的题。

David 玩一个石子游戏。游戏中，有 $n$ 堆石子, 被编号为 $0 \cdots n-1$。两名玩家轮流取石子。每一轮游戏，每名玩家选取 3 堆石子 $i, j, k(i < j, j \leq k$, 且至少有一枚石子在第 $i$ 堆石子中)，从 $i$ 中取出一枚石子，并向 $j, k$ 中各放入一枚石子 (如果 $j = k$ 则向 $k$ 中放入 2 颗石子)。最先不能取石子的人输。

# Take-and-Break Solution

相信大部分同学都做过这道题，将编号倒过来，我们将每个石子看成独立的石堆，其大小为其所在的位置，那么其实就是一个大小为 $x$ 的石堆，变成了两个大小分别为 $y, z < x$ 的石堆，这样我们可以 $O(n^3)$ 计算出一个在所有位置的石子的 SG 值，最后 Nim 和即可。

其实有很多 Take-and-Break 这一类型的游戏，但是大多都要打表找规律，甚至根本就没规律。

# Take-and-Break Variants

- 1. Lasker's Nim，每次可以从一堆中拿走若干个石子，或者将一堆至少有两个石子的石堆分裂成两个非空石堆。
- 2. Grundy's Game，每次可以将一堆石子分成两堆大小不同的非空石子。

# Definition

$n$ 个硬币排成一行，有些正面朝上，有些反面朝上。将硬币从左向右按 $1 \sim n$ 编号。游戏者根据某些约束翻转硬币，但翻转的硬币中，最右边的硬币必须从正面翻到反面（为什么？），谁不能操作谁输。

来看一些经典模型。

# Turning Turtles

每次操作要将一枚硬币从正面翻转到反面，假设其编号为 $x$，额外的，如果有必要的话，可以选择一枚硬币 $y(y < x)$ 翻转（正面到反面 or 反面到正面），无法操作的玩家输。

# Turning Turtles Solution

设有 $k$ 个正面的硬币，编号分别为 $x_1, x_2, \cdots, x_k$，那么其实相当于有 $k$ 堆石子，第 $i$ 堆大小为 $x_i$ 的 Nim 游戏，证明：

- 1. 如果将硬币 $x$ 从正面翻到反面，不选择 $y$，那么相当于取完一堆大小为 $x$ 的石子。
- 2. 如果将硬币 $x$ 从正面翻到反面，选择 $y(y < x)$，且硬币 $y$ 是从反面到正面，那么相当于取走 $x - y$ 个石子。
- 3. 如果将硬币 $x$ 从正面翻到反面，选择 $y(y < x)$，且硬币 $y$ 是从正面到反面，等价于取走了两堆大小分别为 $x, y$ 的石子，也就是相当于取走了 $x - y$ 个石子，因为 $x - (x - y) \oplus y = 0$。

# Extension

其实 Turning Turtles 的做法可以扩展到一般的 Coin Turning Game，设有 $n$ 个硬币，其中有 $k$ 个硬币是正面的，那么游戏可以变成 $k$ 个子游戏的和。设正面硬币的编号分别为 $x_1 \sim x_k$，定义子游戏 $X_i$：有 $x_i$ 个硬币且只有最右边的硬币为正面，设初始局面为 $X$，那么：

$$SG(X) = SG(X_1) \oplus SG(X_2) \oplus \cdots SG(X_k)$$

比如 $X = 01101$，那么：

$$k = 3, X_1 = 01, X_2 = 001, X_3 = 00001$$

$$SG(01101) = SG(01) \oplus SG(001) \oplus SG(00001)$$

# Mock Turtles

规则：每次可以翻转一个、两个或三个硬币，最右边的硬币必须从正面翻到反面。

# Mock Turtles SG

我们可以将 SG 表打出来，出于某些原因，下标从 0 开始：

| position $x$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| $SG(x)$ | 1 | 2 | 4 | 7 | 8 | 11 | 13 | 14 | 16 | 19 |

可以发现 $SG(x) = 2x$ 或者 $2x + 1$，那么什么情况下要加一呢？能不能证明呢？

# Mock Turtles One of Solutions

可以发现下面的 SG 值二进制上 1 的个数全都是奇数，而且是从小排到大的不遗漏的，我们定义二进制位上有奇数个 1 的数是 good 数，偶数个 1 的是 bad 数，那么有以下计算法则：

$$\text{good} \oplus \text{good} = \text{bad} \oplus \text{bad} = \text{bad}$$

$$\text{good} \oplus \text{bad} = \text{bad} \oplus \text{good} = \text{good}$$

考虑证明这个猜想。

# Mock Turtles One of Solutions

我们采用归纳法，设前 $0 \sim x-1$ 项都是符合要求的，考虑第 $x$ 项。

- 1. 翻一个硬币，只能翻 $x$，状态变成 $SG = 0$
- 2. 翻两个硬币，翻了 $x$ 还能翻一个 $y(y < x)$，状态变成 $SG(y)$，即之前的所有 good 数
- 3. 翻三个硬币，翻了 $x$ 还能翻两个 $y, z(y < z < x)$，状态变成 $SG(y) \oplus SG(z)$，根据之前的计算法则，我们可以取遍之前的所有 bad 数

因此当前的 $SG(x)$ 即为下一个 good 数。

# Two-Dimensional Coin Turning Games

其实也就是把一行硬币改成了一个矩阵，之前的限制条件（每次最右边的被翻的硬币必须从正面翻到反面）改成：其中一枚硬币 $(x, y)$ 必须从正面翻到反面，其他所有被翻的硬币必须在矩形 $\{(a, b)|0 \leq a \leq x, 0 \leq b \leq y\}$ 内。

# Acrostic Twins

每次操作可以翻转两枚硬币，两枚硬币要么在同一行，要么在同一列。

# Acrostic Twins SG

这个游戏的 SG 函数非常容易看出来，因为 $(x, y)$ 可以看成两堆大小分别为 $x, y$ 的石子，$(x, y) \to (a, y)$ 可以看成第一堆石子拿走了 $x - a$ 个，$(x, y) \to (x, b)$ 可以看成第二堆石子拿走了 $y - b$ 个，因此 $SG(x, y) = x \oplus y$，接下来我们来看一个更加典型的例子。

# Turning Corners

每次操作可以翻转四个硬币（分别在矩形的四个角上），即 $(a, b), (a, y), (x, b), (x, y)$，其中 $0 \leq a < x, 0 \leq b < y$。

# Turning Corners SG

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 2 | 0 | 2 | 3 | 1 | 8 | 10 | 11 | 9 | 12 | 14 | 15 |
| 3 | 0 | 3 | 1 | 2 | 12 | 15 | 13 | 14 | 4 | 7 | 5 |
| 4 | 0 | 4 | 8 | 12 | 6 | 2 | 14 | 10 | 11 | 15 | 3 |
| 5 | 0 | 5 | 10 | 15 | 2 | 7 | 8 | 13 | 3 | 6 | 9 |
| 6 | 0 | 6 | 11 | 13 | 14 | 8 | 5 | 3 | 7 | 1 | 12 |
| 7 | 0 | 7 | 9 | 14 | 10 | 13 | 3 | 4 | 15 | 8 | 6 |
| 8 | 0 | 8 | 12 | 4 | 11 | 3 | 7 | 15 | 13 | 5 | 1 |
| 9 | 0 | 9 | 14 | 7 | 15 | 6 | 1 | 8 | 5 | 12 | 11 |
| 10 | 0 | 10 | 15 | 5 | 3 | 9 | 12 | 6 | 1 | 11 | 14 |

这个表看上去非常的乱，但其实背后是存在数学规律的。

# Nim Multiplication

我们定义 Nim 积：

$$x \otimes y = mex\{(a \otimes b) \oplus (a \otimes y) \oplus (x \otimes b), 0 \le a < x, 0 \le b < y\}$$

根据之前的表，我们可以很显然的得出：

$$x \otimes 0 = 0 \otimes x = 0, x \otimes 1 = 1 \otimes x = x, x \otimes y = y \times x$$

不加证明的（其实是不会）给出两个计算法则：

$$x \otimes (y \otimes z) = (x \otimes y) \otimes z$$

$$x \otimes (y \oplus z) = (x \otimes y) \oplus (x \otimes z)$$

# Nim Multiplication

有了刚才的法则，结论和之前的那张表，我们可以计算稍微复杂一些的 Nim 积了，比如 $12 \otimes 9$。

# Nim Multiplication

有了刚才的法则，结论和之前的那张表，我们可以计算稍微复杂一些的 Nim 积了，比如 $12 \otimes 9$。

$$12 \otimes 9 = (8 \oplus 4) \otimes (8 \oplus 1) = (8 \otimes 8) \oplus (8 \otimes 1) \oplus (4 \otimes 8) \oplus (4 \otimes 1)$$

$$= 13 \oplus 8 \oplus 11 \oplus 4 = 10$$

但是用这种方法计算更大的 Nim 会变的非常麻烦，我们需要打出一张很大的表。

# Nim Multiplication

经过数学家们的努力，我们现在有两个更加强大的运算法则，我们首先定义 Fermat 2-power 为 $2^{2^n}, n = 0, 1, 2, \cdots$。

- 1. 一个 Fermet 2-power 与任意小于它的数的 Nim 积为一般意义下的乘法的积，e.g. $2 \otimes 16 = 32$。
- 2. 一个 Fermet 2-power 与自己的 Nim 积为自己的 $3/2$ 倍，e.g. $16 \otimes 16 = 24$，注意我们可以写成 $a \otimes a = a \oplus \frac{a}{2}$。

# Nim Multiplication

现在我们能计算更复杂的积了，比如 $24 \otimes 17$。

## Nim Multiplication

现在我们能计算更复杂的积了，比如 $24 \otimes 17$。

$$24 \otimes 17 = (16 \oplus 8) \otimes (16 \oplus 1) = (16 \otimes 16) \oplus (16 \otimes 1) \oplus (8 \otimes 16) \oplus (8 \otimes 1)$$

$$= 24 \oplus 16 \oplus 128 \oplus 8 = 128$$

那么我们有没有什么更快捷的方法计算 Nim 积呢？

# Nim Multiplication

设两个计算函数，$F(x, y) = x \otimes y, G(x, y) = x \otimes y$(x 是 2 的幂)。

对于 $F$，找到 $a$ 满足 $2^{2^a} \le x < 2^{2^{a+1}}$，令 $M = 2^{2^a}$，将 $x, y$ 表示成 $x = p \otimes M \oplus q, y = s \otimes M \oplus t$，那么乘法可以表示成：

$$(p \otimes M \oplus q) \otimes (s \otimes M \oplus t) = p \otimes s \otimes M^2 \oplus (p \otimes t \oplus q \otimes s) \otimes M \oplus q \otimes t$$

$$= F(p, s) \otimes (M \oplus \frac{M}{2}) \oplus (F(p, t) \oplus F(q, s)) \otimes M \oplus F(q, t)$$

$$= F(p, s) \otimes M \oplus G(\frac{M}{2}, F(p, s)) \oplus (F(p, t) \oplus F(q, s)) \otimes M \oplus F(q, t)$$

# Nim Multiplication

对于 $G$，找到 $a$ 满足 $2^{2^a} \le x < 2^{2^{a+1}}$，令 $M = 2^{2^a}$，将 $x, y$ 表示成 $x = p \otimes M, y = s \otimes M \oplus t$，那么乘法可以表示成：

$$(p \otimes M) \otimes (s \otimes M \oplus t) = p \otimes s \otimes M^2 \oplus (p \otimes t) \otimes M$$

$$= G(p, s) \otimes (M \oplus \frac{M}{2}) \oplus G(p, t) \otimes M$$

$$= G(p, s) \otimes M \oplus G(\frac{M}{2}, G(p, s)) \oplus G(p, t) \otimes M$$

# Nim Multiplication

设 $f(n)$ 表示 $F(x,y), x,y < 2^{2^n}$ 情况下的最坏复杂度，$g(n)$ 表示 $G$ 的最坏复杂度，那么有：

$$g(n) = 3g(n-1) = O(3^n)$$

$$f(n) = 4f(n-1) + g(n-1) = 4f(n-1) + 3^{n-1}$$

$$f(n) + 3^n = 4(f(n-1) + 3^{n-1}) = O(4^n)$$

$$f(n) = O(4^n) - O(3^n) = O(4^n)$$

因此复杂度就是 $O(4^n) = O(4^{\log\log x}) = O(\log^2 x)$，至此我们对于 Nim 可以完成高效的加法和乘法。

# Tartan Games

既然 Nim 都有 Nim 和，Nim 积了，游戏怎么能只有游戏和呢，肯定还是有游戏积的！

Tartan Games 指的是给定两个一维翻硬币游戏 $G_1$ 和 $G_2$，定义 $G_1 \times G_2$ 是二维翻硬币游戏，操作如下：假设 $G_1$ 中翻转 $x_1 \sim x_n$ 是一个合法操作，$G_2$ 中翻转 $y_1 \sim y_m$ 是一个合法操作，那么在 $G_1 \times G_2$ 中翻转 $(x_i, y_j), 1 \leq i \leq n, 1 \leq j \leq m$ 是一个合法操作。

可以发现之前的 Turning Corners 其实是一个叫做 Twins 的游戏的平方，Twins 指的是一行硬币，每次必须要翻转两个硬币。

# Tartan Theorem

如果 $g_1(x)$ 是 $G_1$ 的 SG 函数，$g_2(y)$ 是 $G_2$ 的 SG 函数，那么 $G_1 \times G_2$ 的 SG 函数 $g(x, y)$ 即为 Nim 积：

$$g(x, y) = g_1(x) \otimes g_2(y)$$

比如在 Twins 中的 $SG(x) = x$，那么 Turning Corners 的 $SG(x, y) = x \otimes y$。

# Turning Turtles Squared

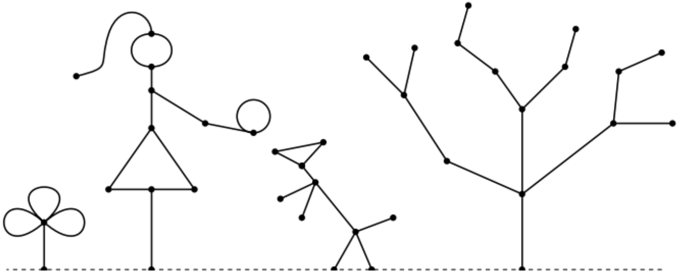
在游戏 Turning Turtles $\times$ Turning Turtles 中，我们可以翻转矩形的四个角，也可以翻转同一行的两个，或者同一列的两个，或者只翻转一个。

我们之前求出了 Turning Turtles 的 $SG(x) = x$，有个细节：之前是下标从 1 开始的情况，转换成 0 的话 $SG(x, y) = (x + 1) \otimes (y + 1)$。

有兴趣的同学可以自己发明一些游戏，然后乘起来观察规律（其实是拿去出题）。

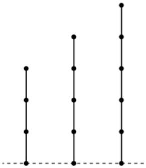
# Definition

简称有根图删边游戏。有根图指的是一个无向图，满足每条边都有路径连接到一个根结点或者是地面，如下图。每次操作可以任意删除一条边，然后移除那些不与地面相连的部分。



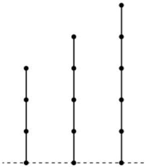
# Bamboo Stalks

有若干个直立在地面上的竹竿，每个竹竿有若干节，每次可以从某个竹竿的某节劈开，将上面的部分拿走，无法操作的玩家输。



# Bamboo Stalks

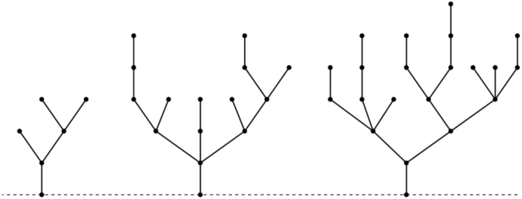
有若干个直立在地面上的竹竿，每个竹竿有若干节，每次可以从某个竹竿的某节劈开，将上面的部分拿走，无法操作的玩家输。



其实就是 Nim 对吧。

# Green Hackenbush on Trees

现在我们的竹竿换成了一棵棵的有根树，如下图。



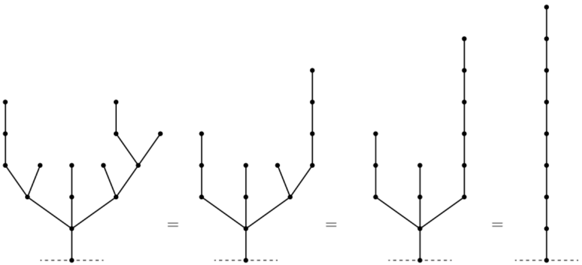
怎样计算一棵树的 SG 值呢？

# Colon Principle

考虑一个点有 $x(x>1)$ 个分叉，设这 $x$ 个分叉的 Nim 和为 $v$，那么可以将这些分叉替换成一个长度为 $v$ 的链。

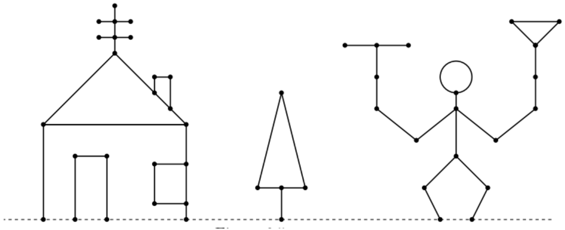
考虑在之前的树上应用这个法则。

# Colon Principle



# Green Hackenbush on General Rooted Graph

我们现在考虑将树换成一般的图，如下图。



怎么求 SG 值呢？

# The Fusion Principle

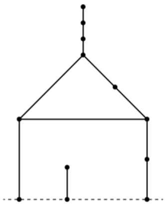
如果两个相邻的点在一个环上，那么我们可以将其合并成一个点。

换句话说，对于一个 $x$ 条边的环，我们可以缩成一个点再加上 $x$ 个自环。

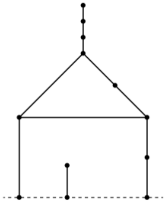
# Examples

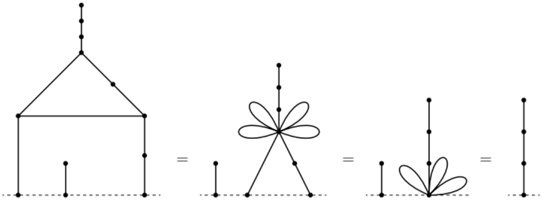
因为地面其实相当于一个点。

# Examples



# Examples





# Conclusion

那么做法就很明显了，我们直接用 tarjan 缩出边双连通分量，然后可以变成一棵树，一棵树就用之前的 Colon Principle 即可。

# Definition

在这一部分中我们只讨论两个玩家之间的，一个玩家赢另一个玩家输的游戏。我们称两个玩家分别 I 和 II。

# Strategic Form

两人零和游戏的策略形式指的是一个三元组 $(X, Y, A)$，其中：

- 1. $X$ 是一个非空集合，表示 I 的策略。
- 2. $Y$ 是一个非空集合，表示 II 的策略。
- 3. $A$ 是一个定义在 $X \times Y$ 上的实数函数，$A(x, y)$ 表示 I 做 $x$ 决策，II 做 $y$ 决策时 I 的收益（II 的损失）。

# Odd or Even

I 和 II 同时做出决策，每次决策表示选择 1 还是选择 2，玩家 I 获胜当且仅当两个玩家选择的数的和为奇数，II 获胜当且仅当是偶数，输的玩家要给赢的玩家和那么多的钱。

# Strategic Form of Odd or Even

$X = Y = \{1, 2\}$，函数 $A$ 如下：

| $X \backslash Y$ | 1 | 2 |
|---|---|---|
| 1 | $-2$ | $+3$ |
| 2 | $+3$ | $-4$ |

# Analysis of Odd or Even

让我们从 I 的角度来分析，假设一共玩了无限局游戏，I 出了 3/5 的 1 和 2/5 个 2，这种情况下：

- 1. 如果 II 全出 1，那么 I 的平均收益为 $((-2)\times 3+3\times 2)/5=0$。
- 2. 如果 II 全出 2，那么 I 的平均收益为 $(3\times 3+(-4)\times 2)/5=1/5$。

这种策略下，最坏收益是 0，那有没有策略可以让我们的最坏收益更好呢？

# Pure and Mixed Strategy

我们将 $X, Y$ 中的元素称为 pure。
如果对 $X, Y$ 中每个元素赋予一个选择的概率，那么称为 mixed。
我们接下来讨论的是 mixed strategy 下的最优解。

## Analysis of Odd or Even

设 $p$ 表示 I 出 1 的概率，$1-p$ 即为 I 出 2 的概率，那么：

- 1. II 出 1 的收益为 $-2p+3(1-p)=-5p+3$。
- 2. II 出 2 的收益为 $3p-4(1-p)=7p-4$。

对于一个固定的 $p$，我们的最坏收益即为：

$$\min\{-5p+3, 7p-4\}$$

玩家 I 要最大化最坏收益，即求：

$$\max_p \min\{-5p+3, 7p-4\}$$

# Analysis of Odd or Even

细心的同学们应该早已经发现，我们要求的即为 $y = -5p + 3$ 与 $y = 7p - 4$ 的交点：

$$-5p + 3 = 7p - 4 \Rightarrow p = 7/12, y = 1/12$$

因此当 $p = 7/12$ 的时候，最坏收益是 $1/12$，并且我们可以发现当 II 也采用最优策略的时候，II 的损失不会超过 $1/12$，因此 $1/12$ 是一个类似于临界点的值。

# The Minimax Theorem

我们定义有限的游戏：$X, Y$ 均为有限集合。

对于每个有限的两个人的零和游戏：

- 1. 存在一个数 $V$ 为游戏的值。
- 2. 对于 I，存在混合策略使得无论 II 怎么决策，I 的收益不小于 $V$。
- 3. 对于 II，存在混合策略使得无论 I 怎么决策，II 的损失不大于 $V$。

# Optimal Strategy

我们定义对于 I 的最佳混合策略为向量 $p = (p_1, p_2, \cdots, p_n)$，对于 II 的最佳混合策略为向量 $q = (q_1, q_2, \cdots, q_m)$，满足：

$$\sum_{i=1}^{n} p_i \cdot a_{ij} \geq V \qquad \text{for all } j = 1 \sim m$$

$$\sum_{j=1}^{m} a_{ij} \cdot q_j \leq V \qquad \text{for all } i = 1 \sim n$$

# Find Optimal Strategy

我们可以考虑将 $V$ 也看成一个未知量，用单纯形计算：

$$maximize \qquad V$$

$$\sum_{i=1}^{n} p_i \cdot a_{ij} \geq V \qquad \text{for all } j = 1 \sim m$$

$$\sum_{i=1}^{n} p_i = 1$$

# Codeforces 98E

A 君有 $n$ 张牌，B 君有 $m$ 张牌，桌上还有一张反扣着的牌，每张牌都不一样。

每个回合可以做两件事中的一件：

- 1. 猜测桌上的牌是什么，猜对则胜，猜败则输。
- 2. 询问对方是否有某张牌，若有则需要将其示出，否则继续游戏。

A 和 B 都很聪明，问 A 的胜率。

$n, m \leq 5000$

# Codeforces 98E Solution

首先如果不确定反扣着的牌的话是不会猜桌上的牌的。

假设对方如果问了一张自己没有的牌，我们可能怀疑桌上的牌就是这张。而我们可以用自己的牌去询问对方，如果对方相信的话就会输掉，我们称这种行为叫做欺骗。

设 $f(n, m)$ 表示先手胜率，那么有这样一个 Strategic Form：

| | 相信 | 不相信 |
|---|---|---|
| 猜测 | $\frac{m}{m+1}(1 - f(m-1, n))$ | $\frac{1}{m+1} + \frac{m}{m+1}(1 - f(m-1, n))$ |
| 欺骗 | $1$ | $1 - f(m, n-1)$ |

# Codeforces 98E Solution

显然这个问题可以用我们之前的知识解决，不过这个 $2 \times 2$ 的矩阵可能只需要直线求个交即可，不需要用单纯形。

# Topcoder SRM682 div1 hard

Bob 被选中去参加一个游戏，规则如下：有一行 $n$ 个格子，从左到右编号为 $0 \sim n-1$，每个格子上有一个权值 $value_i$。Bob 从 0 处开始游戏，每次操作有两种选择：1. 在目前的格子停止游戏；2. 移到右边的格子。最终 Bob 的游戏结果为停止游戏时所在的格子的权值。然而，游戏的主办方为了刁难玩家，在游戏开始时设定了一个秘密整数 $k$，满足 $0 \le k \le n-1$，$k$ 表示了 Bob 能移动到的最右边的格子，如果 Bob 到了格子 $k$，不管 Bob 怎么选择，游戏都会停止。Bob 并不知道 $k$ 的具体值，只知道 $k$ 是随机选择的，也不知道 $k$ 在 $0 \sim n-1$ 的概率分布，但他知道对于一个正整数 $x$，$k < x$ 的概率不超过 $\frac{x}{n}$。求 Bob 采取最佳策略时的胜率。

# Topcoder SRM682 div1 hard

考虑 $k = i$ 的概率为 $a_i$，那么我们现在的限制条件有：

$$\sum_{i=0}^{n-1} a_i = 1$$

$$\sum_{i=0}^{x-1} a_i \le \frac{x}{n} \qquad \text{for all } 1 \le x \le n$$

同时我们可以得到 Strategic Form，那么我们单纯形一发就好啦。

# Definition

通常为在二分图上按照某种规则走的游戏。

# JSOI2009 Game

在 $n \times m$ 的迷宫中有一个棋子 AA 首先任意选择棋子放置的位置。然后，YY 和 AA 轮流将棋子移动到相邻的格子里。

游戏的规则规定，在一次游戏中，同一个格子不能进入两次，且不能将棋子移动到某些格子中去。当玩家无法继续移动棋子时，游戏结束，最后一个移动棋子的玩家赢得了游戏。

求 AA 初始将棋子放在哪些格子会有必胜策略。

$n, m \leq 100$

# JSOI2009 Game

黑白染色后变成二分图上，AA 选择一个起点，然后 YY 和 AA 轮流将棋子移动到相邻的点上，不能移动的输。

定义最大匹配关键点为一定在最大匹配上的点，而且容易证明这样的点是先手必胜点。也容易证明如果一个点不一定在最大匹配上，这样的点为先手必败点。

具体怎么求最大匹配关键点参考 2015 年国家集训队论文中陈胤伯的《浅谈图的匹配算法及应用》。

# THANKS FOR LISTENING

# Reference


- 王晓珂《解析一类组合游戏》
- GAME THEORY–Thomas S.Ferguson
- 曹钦翔《从“k 倍动态减法游戏”出发探究一类组合游戏问题》
- 张一飞《由感性认识到理性认识——透析一类搏弈游戏的解答过程》
- 贾志豪《组合游戏略述——浅谈 SG 游戏的若干拓展及变形》
- 翁文涛 Games