博弈

4182_543_731

2024/07

Contents

博弈问题

什么是一个博弈问题？通常情况下，

1. 有有限个**状态**（例如，图中棋子的位置，一堆石子的个数，...）
2. **两个**人**轮流**操作。第一个人先手，第二个人后手。
3. 每一个**状态**和当前操作的**人**会共同**决定**当前可用的**操作**，每个操作会使状态转移到另一个状态。

博弈问题

博弈问题还有多种可能的设定。我们给出最常见的设定可能：

1. **不能操作者输** (Normal)/不能操作者赢 (Misère)
2. **状态转移不存在环** (Normal)/状态转移存在环 (Loopy)
3. **可用操作与人无关** (平等)/**可用操作与人有关** (不平等)

平等博弈与不平等博弈是最常见的情况。

当然，在单个游戏，即不考虑多个独立游戏的情况下，平等与不平等没有区别：可以向状态中加入操作方；Normal 和 Misère 也没有区别：在最后加一步。但当我们考虑多个独立游戏的时候，情况则大有不同。

一般无环博弈

我们考虑最基础的情况：不能操作者输，转移不存在环，平等博弈。

经典问题

有一张**有向无环图**，图上有一枚棋子。双方轮流操作，每次操作可以将棋子沿着当前点的一条出边移动。不能操作者输，求获胜方。

$n \leq 10^6$

由于转移不存在环，游戏一定会结束。又由于游戏平等，双方最优操作下，每个状态的结果一定是如下二者之一：从该状态出发时，**先手必胜**或**后手必胜**。

记一个状态（棋子所在点）为 s，我们用 $f(s) = 1$ 表示当前局面下先手必胜，$f(s) = 0$ 表示当前局面下后手必胜。

一般无环博弈

经典问题

有一张**有向无环图**，图上有一枚棋子。双方轮流操作，每次操作可以将棋子沿着当前点的一条出边移动。不能操作者输，求获胜方。

$n, m \le 10^6$

按拓扑序**从后往前**考虑。在考虑一个点 s 时，它的所有后继点 $t_1, \cdots$ 的状态都被确定了。

- 如果存在后继点 t_i 使得 $f(t_i) = 0$，那么当前操作的人走过去就赢了。
- 否则，无论当前操作的人怎么走，状态都变成先手必胜。所以当前操作的人输。

如果 s 存在后继点 t_i 使得 $f(t_i) = 0$，则 $f(s) = 1$；否则 $f(s) = 0$。

有点像一个 DP，复杂度 $O(m)$

一般无环博弈

经典问题

有一张**有向无环图**，图上有一枚棋子。双方轮流操作，每次操作可以将棋子沿着当前点的一条出边移动。不能操作者输，求获胜方。

$n \leq 10^6$

对于更一般的情况，我们可以把状态看成点，转移看成边。那么任何一个无环博弈都可以表示为上述形式。(如果是不平等博弈，那这里我们可以直接把当前操作的人也放进状态中)，从而这是一个直接解决无环博弈的方式。

一般无环博弈

经典问题

有一张**有向无环图**，图上有一枚棋子。双方轮流操作，每次操作可以将棋子沿着当前点的一条出边移动。不能操作者输，求获胜方。

$n \leq 10^6$

对于更一般的情况，我们可以把状态看成点，转移看成边。那么任何一个无环博弈都可以表示为上述形式。(如果是不平等博弈，那这里我们可以直接把当前操作的人也放进状态中)，从而这是一个直接解决无环博弈的方式。

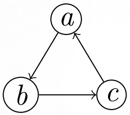
但一般情况下，问题的状态数都必然是非常大的，我们需要更多的观察以解决问题。

经典问题 2

有一张**有向图**，图上有一枚棋子。双方轮流操作，每次操作可以将棋子沿着当前点的一条出边移动。不能操作者输，求游戏结果。

$n \le 10^6$

现在每个状态的结果是什么？因为有环，游戏甚至可能永不停止。



那怎么定义胜负？我们认为无限循环等于平局，每个人都是胜 $>$ 平 $>$ 负。我们仍然用 $f(s)=1/0$ 表示先手胜/先手负，然后用 $f(s)=?$ 表示平局。

经典问题 2

有一张**有向图**，图上有一枚棋子。双方轮流操作，每次操作可以将棋子沿着当前点的一条出边移动。不能操作者输，求游戏结果。

$n \le 10^6$

之前的分析仍然成立：如果有让自己获胜的操作就可以直接走过去，如果所有操作都让自己必败则真的必败，即：

- 如果 s 存在后继点 t_i 满足 $f(t_i) = 0$，则 $f(s) = 1$。
- 如果 s 的每个后继点 t_i 都满足 $f(t_i) = 1$，则 $f(s) = 0$。

这可以用一个类似搜索与拓扑排序的方式实现：$f(t) = 0$ 则类似 bfs 的方式直接更新所有入边，$f(t) = 1$ 则类似拓扑排序的方式减少某个 s 的入度，减到 0 就更新。

一般有环博弈

- 如果 s 存在后继点 t_i 满足 $f(t_i)=0$，则 $f(s)=1$。
- 如果 s 的每个后继点 t_i 都满足 $f(t_i)=1$，则 $f(s)=0$。

这样显然不能确定所有的状态。剩下的状态怎么办？我们声称所有这样的状态都是平局。

证明.

注意到每个这样的状态都满足：不能转移到 $f=0$ 的状态，且可以转移到 $f=?$ 的状态。考虑一方采用如下策略：每步走到下一个 $f=?$ 的状态。那么另一方想跳出去只能走到 $f=1$，也就是输掉。所以另一方没法赢。这说明没有人能赢。 $\square$

这里的思想非常有意义：如果一方用一个简单的操作就能达到某种结果，那他的最终结果不会更差。

「联合省选 2023」过河卒 (Easy)

有一个 $n \times m$ 的棋盘。棋盘上有一些障碍。

第一个人有两枚棋子，第二个人有一枚棋子。

第一个人先手。每次操作可以选择一枚棋子，移动到一个四相邻的格子，不能移动到障碍。

- 第二个人不能向下移动
- 第一个人不能让他的两枚棋子在同一位置

不能操作者输。除此之外：

- 如果第二个人将棋子移动到第一行，则第二个人获胜。
- 如果一个人将棋子移动到另一方某枚棋子所在位置，则其获胜。

求游戏结果。$T, n, m \leq 10$

「联合省选 2023」过河卒 (Easy)

状态数是三枚棋子的 $O(n^3m^3)$，操作数 $O(1)$。

那么直接将状态变为点，然后做之前的方法。复杂度 $O(n^3m^3)$。

请自行实现状态标号，操作处理等问题，并注意题目中的**所有**细节。

「联合省选 2023」过河卒 (Easy)

状态数是三枚棋子的 $O(n^3m^3)$，操作数 $O(1)$。
那么直接将状态变为点，然后做之前的方法。复杂度 $O(n^3m^3)$。
请自行实现状态标号，操作处理等问题，并注意题目中的**所有**细节。

同上，但是胜方会在此基础上最小化操作次数，负方则会最大化操作次数，在游戏不平局时额外求出几步后结束。

「联合省选 2023」过河卒

考虑 $f = 1/0$ 的状态。记 $T(s)$ 表示这个状态的步数。对于无法继续操作的状态 $T = 0$，其它情况下：

- 对于获胜状态 $f(s) = 1$，这里要最小化时间，且需要走到 $f = 0$ 的态。那么 $T(s)$ 等于最小的满足 $f(t) = 0$ 的后继 $T(t)$ 加一。
- 对于 $f(s) = 0$，输的一方会最大化时间，因此 $T(s)$ 等于最大的后继 $T(t)$ 加一。

「联合省选 2023」过河卒

考虑 $f=1/0$ 的状态。记 $T(s)$ 表示这个状态的步数。对于无法继续操作的状态 $T=0$，其它情况下：

- 对于获胜状态 $f(s)=1$，这里要最小化时间，且需要走到 $f=0$ 的态。那么 $T(s)$ 等于最小的满足 $f(t)=0$ 的后继 $T(t)$ 加一。
- 对于 $f(s)=0$，输的一方会最大化时间，因此 $T(s)$ 等于最大的后继 $T(t)$ 加一。

怎么求 T？回想刚才的过程：

$f(t)=0$ 则类似 bfs 的方式直接更新所有入边，$f(t)=1$ 则类似拓扑排序的方式减少某个 s 的入度，减到 0 就更新。

如果我们按照 T 从小到大更新，那么 $f=1$ 的情况在最小的 $T(t)$ 处得到更新，而 $f=0$ 的情况在最大的 $T(t)$ 处得到更新。所以 bfs 实现，同时更新 T 即可。

「联合省选 2023」过河卒

请自行实现状态标号，操作处理等问题，并注意题目中的**所有**细节。
请自行实现状态标号，操作处理等问题，并注意题目中的**所有**细节。
请自行实现状态标号，操作处理等问题，并注意题目中的**所有**细节。

「联合省选 2023」过河卒

请自行实现状态标号，操作处理等问题，并注意题目中的**所有**细节。
请自行实现状态标号，操作处理等问题，并注意题目中的**所有**细节。
请自行实现状态标号，操作处理等问题，并注意题目中的**所有**细节。

...第一个人不能让他的两枚棋子在同一位置...

```
O
O
.
X
.
```

[NEERC2016] Game on Graph

有一张**有向无环图**，图上有一枚棋子。双方轮流操作，每次操作可以将棋子沿着当前点的一条出边移动。不能操作者输。

但是现在第一个人认为平局（循环）$>$ 赢 $>$ 输，第二个人认为赢 $>$ 输 $>$ 平。求每个状态（起始位置和先手方）的结果。

$n, m \leq 2 \times 10^5$

第一个人希望循环，第二个人希望不循环。那么先考虑游戏能不能停止。可以发现

- 如果第一个人操作但所有后继状态都停止，那么必定停止。
- 如果第二个人操作且存在后继状态停止，那么停止。

这样可以 bfs 出一些停止状态。对于剩下的状态，考虑类似之前的分析：第二个人一定不能停，第一个人可以不停，那么游戏一定不停止。

[NEERC2016] Game on Graph

考虑游戏能停止的那些状态，此时双方正常地比胜负。
那么考虑用之前的方式。但这里图还是可能有环，因此用有环的版本。

[NEERC2016] Game on Graph

考虑游戏能停止的那些状态，此时双方正常地比胜负。

那么考虑用之前的方式。但这里图还是可能有环，因此用有环的版本。

但这样还是可能有无法确定的位置（之前是填平局，但这里显然不对）。这些位置是怎么回事？

之前的分析：双方都不能赢，因此平局。但这里第二个人是输 $>$ 平局，因此第二个人此时会选择主动去输。而刚才的分析说明他可以这样做。

复杂度 $O(n+m)$。

观察法

在一些问题结构比较好（数，网格，...）的问题上，博弈的结果可能高度复杂，甚至极其反直觉。此时可能难以直接推导出结果。

因此一种有时非常有效的方式：打表，观察。

观察法

在一些问题结构比较好（数，网格，...）的问题上，博弈的结果可能高度复杂，甚至极其反直觉。此时可能难以直接推导出结果。

因此一种有时非常有效的方式：打表，观察。

请大家拿出自己喜欢的 IDE 开始尝试。

[AGC002E] Candy Piles

有 n 堆石子。双方轮流操作，每次操作可以：

- 在每堆非空石子中拿走一个，或者
- 拿走最大的一堆石子。

拿走最后一枚石子的人输，求谁获胜。

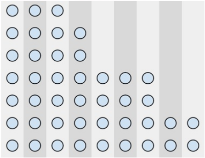
$n \leq 10^5$

[AGC002E] Candy Piles

将石子堆从大到小排序，那么操作 2 相当于拿走左侧第一堆棋子。

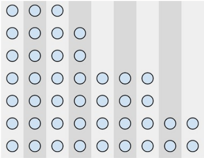
如果看成一个柱状图的形式，那么操作 2 就是拿掉第一行，而操作 1 每堆拿掉一个正好相当于拿掉最下面一列。

因此问题相当于，有一个



每次删掉第一行或第一列，删完的人输。

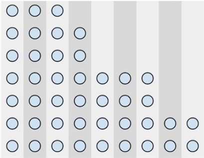
[AGC002E] Candy Piles



考虑当前删到了哪，“左下角”在哪。那么相当于：初始“左下角”在左下，每次可以向右走或者向上走，走出去的人输。

但直接做是 $O(nv)$ 的。

[AGC002E] Candy Piles



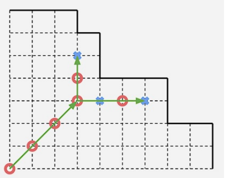
考虑当前删到了哪，“左下角”在哪。那么相当于：初始“左下角”在左下，每次可以向右走或者向上走，走出去的人输。

但直接做是 $O(nv)$ 的。

```
13
14 14 14 14 14 10 10 10 6 6 6 6 6
0 1 0 1 0
1 0 1 0 1
0 1 0 1 0
1 0 1 0 1
0 1 0 1 1 0 1 0
1 0 1 1 0 1 0 1
0 1 1 0 1 0 1 0
1 1 0 1 0 1 0 1
1 0 1 0 1 0 1 1 0 1 0 1 0
0 1 0 1 0 1 1 0 1 0 1 0 1
1 0 1 0 1 1 0 1 0 1 0 1 0
0 1 0 1 1 0 1 0 1 0 1 0 1
1 0 1 1 0 1 0 1 0 1 0 1 0
0 1 1 0 1 0 1 0 1 0 1 0 1
```

结论：除去边界外部分外，每条对角线上值相同。

证明：考虑每条对角线怎么求值。从最顶部看：



每个边界点的值和它上面及右侧有多少位置相关。除去向内拐角的位置外，每个位置只有一个方向的边界，那么是 01 交替。再考虑角上的情况，可以发现 0 对角线两侧都是 1，1 对角线最多连续两条，那么接下来还会是这个样子。

上面的证明给出了一种求值方式。复杂度 $O(n)$。

一道题

有一堆 n 个石子，双方轮流操作，每次操作可以拿掉若干个棋子：第一次操作只能拿一个；设上一次拿了 c 个，则下一次可以拿不超过 $c+k$ 个。拿掉最后一枚石子的人获胜。

给定 k，求 $n \in [l, r]$ 中有多少位置后手必胜。

$k, l, r \le 10^{14}$

一道题

好像可以正经 DP 做，但是我们选择先打表。

一道题

好像可以正经 DP 做，但是我们选择先打表。

```
500 11
0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1
0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1
0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1
0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1
0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1
0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1
0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1
0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1
0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1
0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1
0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0
```

容易发现有循环节，容易发现每一段是：

一道题

好像可以正经 DP 做，但是我们选择先打表。

```
500 11
0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1
0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1
0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1
0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1
0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1
0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1
0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1
0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1
0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1
0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1
0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 0
```

容易发现有循环节，容易发现每一段是：

- 一个 0 和 k 个 1，然后
- 一个 0 和 2^0-1 个 1，一个 0 和 2^1-1 个 1，一个 0 和 2^2-1 个 $1...$，然后
- 一个 0 和 k 个 1，然后
- 一个 0 和 2^0-1 个 1，一个 0 和 2^1-1 个 1，一个 0 和 2^2-1 个 $1...$，然后

只需要再看出中间两段的长度。

一道题

记第一段走到 2^x-1，第二段走到 2^y-1，然后再观察。

```
1: 2 1 2
2: 3 1 2 3 1
3: 4 1 2 4
4: 5 1 2 4 5 1 2
5: 6 1 2 4 6 1 2
6: 7 1 2 4 7 1
7: 8 1 2 4 8
8: 9 1 2 4 8 9 1 2 4
9: 10 1 2 4 8 10 1 2 4
10: 11 1 2 4 8 11 1 2 4
11: 12 1 2 4 8 12 1 2 4
12: 13 1 2 4 8 13 1 2
13: 14 1 2 4 8 14 1 2
14: 15 1 2 4 8 15 1
15: 16 1 2 4 8 16
16: 17 1 2 4 8 16 17 1 2 4 8
17: 18 1 2 4 8 16 18 1 2 4 8
18: 19 1 2 4 8 16 19 1 2 4 8
19: 20 1 2 4 8 16 20 1 2 4 8
20: 21 1 2 4 8 16 21 1 2 4 8
21: 22 1 2 4 8 16 22 1 2 4 8
22: 23 1 2 4 8 16 23 1 2 4 8
23: 24 1 2 4 8 16 24 1 2 4 8
24: 25 1 2 4 8 16 25 1 2 4
25: 26 1 2 4 8 16 26 1 2 4
26: 27 1 2 4 8 16 27 1 2 4
27: 28 1 2 4 8 16 28 1 2 4
28: 29 1 2 4 8 16 29 1 2
29: 30 1 2 4 8 16 30 1 2
30: 31 1 2 4 8 16 31 1
31: 32 1 2 4 8 16 32
32: 33 1 2 4 8 16 32 33 1 2 4 8 16
```

可以看出规律。循环节里面只有 $O(\log k)$ 个 0，所以复杂度 $O(\log k)$

「SNOI2020」取石子

有一堆 n 个石子，双方轮流操作，每次操作可以拿掉若干个棋子：第一次操作只能拿不超过 k 个；设上一次拿了 c 个，则下一次可以拿不超过 $2c$ 个。拿掉最后一枚石子的人**失败**。

给定 k，求 $n \in [1, m]$ 中有多少位置后手必胜。

$k, m \leq 10^{18}, T \leq 10^5$

「SNOI2020」取石子

尝试打表。像上一个题那样固定 k 构造，可以发现不循环，但貌似只和斐波那契数有关。

考虑同时枚举 n, k，此时容易看出固定 n，k 越大答案越容易是 1。这是直接的：先手有更多操作就更可能赢。

「SNOI2020」取石子

尝试打表。像上一个题那样固定 k 构造，可以发现不循环，但貌似只和斐波那契数有关。

考虑同时枚举 n, k，此时容易看出固定 n，k 越大答案越容易是 1。这是直接的：先手有更多操作就更可能赢。

打个表：

```
500
1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 21 1 2 3 1 5 1 2 8 1 2 3 1 34 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 55
1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 21 1 2 3 1 5 1 2 8 1 2 3 1 89 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 21
1 2 3 1 5 1 2 8 1 2 3 1 34 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 144 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 21
 1 2 3 1 5 1 2 8 1 2 3 1 34 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 55 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 21
 1 2 3 1 5 1 2 8 1 2 3 1 233 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 21 1 2 3 1 5 1 2 8 1 2 3 1 34 1 2 3 1 5 1 2 8
 1 2 3 1 13 1 2 3 1 5 1 2 55 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 21 1 2 3 1 5 1 2 8 1 2 3 1 89 1 2 3 1 5 1 2 8
 1 2 3 1 13 1 2 3 1 5 1 2 21 1 2 3 1 5 1 2 8 1 2 3 1 34 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 377 1 2 3 1 5 1 2
8 1 2 3 1 13 1 2 3 1 5 1 2 21 1 2 3 1 5 1 2 8 1 2 3 1 34 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 55 1 2 3 1 5 1 2
8 1 2 3 1 13 1 2 3 1 5 1 2 21 1 2 3 1 5 1 2 8 1 2 3 1 89 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 21 1 2 3 1 5 1 2
8 1 2 3 1
```

「SNOI2020」取石子

```
500
1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 21 1 2 3 1 5 1 2 8 1 2 3 1 34 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 55
1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 21 1 2 3 1 5 1 2 8 1 2 3 1 89 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 21
1 2 3 1 5 1 2 8 1 2 3 1 34 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 144 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 21
 1 2 3 1 5 1 2 8 1 2 3 1 34 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 55 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 21
 1 2 3 1 5 1 2 8 1 2 3 1 233 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 21 1 2 3 1 5 1 2 8 1 2 3 1 34 1 2 3 1 5 1 2 8
 1 2 3 1 13 1 2 3 1 5 1 2 55 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 21 1 2 3 1 5 1 2 8 1 2 3 1 89 1 2 3 1 5 1 2 8
 1 2 3 1 13 1 2 3 1 5 1 2 21 1 2 3 1 5 1 2 8 1 2 3 1 34 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 377 1 2 3 1 5 1 2
8 1 2 3 1 13 1 2 3 1 5 1 2 21 1 2 3 1 5 1 2 8 1 2 3 1 34 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 55 1 2 3 1 5 1 2
8 1 2 3 1 13 1 2 3 1 5 1 2 21 1 2 3 1 5 1 2 8 1 2 3 1 89 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 21 1 2 3 1 5 1 2
8 1 2 3 1
```

可以看出全是斐波那契数。

容易看出这个序列是怎么生成的：$f_0 = f_1 = \{1\}$，f_i 等于 f_{i-1} 拼接 f_{i-2}，然后把最后一个位置换成 F_i。

「SNOI2020」取石子

```
500
1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 21 1 2 3 1 5 1 2 8 1 2 3 1 34 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 55
1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 21 1 2 3 1 5 1 2 8 1 2 3 1 89 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 21
1 2 3 1 5 1 2 8 1 2 3 1 34 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 144 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 21
 1 2 3 1 5 1 2 8 1 2 3 1 34 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 55 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 21
 1 2 3 1 5 1 2 8 1 2 3 1 233 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 21 1 2 3 1 5 1 2 8 1 2 3 1 34 1 2 3 1 5 1 2 8
 1 2 3 1 13 1 2 3 1 5 1 2 55 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 21 1 2 3 1 5 1 2 8 1 2 3 1 89 1 2 3 1 5 1 2 8
 1 2 3 1 13 1 2 3 1 5 1 2 21 1 2 3 1 5 1 2 8 1 2 3 1 34 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 377 1 2 3 1 5 1 2
8 1 2 3 1 13 1 2 3 1 5 1 2 21 1 2 3 1 5 1 2 8 1 2 3 1 34 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 55 1 2 3 1 5 1 2
8 1 2 3 1 13 1 2 3 1 5 1 2 21 1 2 3 1 5 1 2 8 1 2 3 1 89 1 2 3 1 5 1 2 8 1 2 3 1 13 1 2 3 1 5 1 2 21 1 2 3 1 5 1 2
8 1 2 3 1
```

可以看出全是斐波那契数。

容易看出这个序列是怎么生成的：$f_0 = f_1 = \{1\}$，f_i 等于 f_{i-1} 拼接 f_{i-2}，然后把最后一个位置换成 F_i。

再大眼观察一下，这就是斐波那契表示下的最后一位，或者说每次减最大的斐波那契数，减到 0 那次减掉的数。

斐波那契表示：将 n 写成一堆不同斐波那契数的和，要求相邻的斐波那契数不同时出现。
构造：每次减最大的能减的。因为 $F_i = F_{i-1} + F_{i-2}$ 这样一定对。

「SNOI2020」取石子

能赢的 k 是斐波那契表示下的最低位。

证明.

归纳。如果 k 不小于这个数，那么直接减这一位，接下来低位是 $F_{c+2} > 2F_c$。
否则，按位考虑可以证明如果减一个最低位是 $F_d(d < c)$ 的数，则减完结果的最低位至多是 F_{d+1}，这不超过 $2F_d$。 □

然后是斐波那契表示下的“数位 DP”。

性质分析

另一方面，在问题结构比较复杂（例如图上博弈）的问题上，问题结构可能提供结果，复杂的结构使得打表观察变得很难。

此时，通常直接分析性质可以得到不错的结果。一种有效的方式是，从简单的情形开始。

一道题 2

有一棵 n 个点的树，双方进行如下游戏：有一枚棋子初始在 1，每次操作将棋子移动到另一个点，要求每次移动的距离递增，不能操作者输。

~~然后求有多少个包含1 的连通块使得先手必胜。~~

$n \leq 10^6$

部分分：图是一条链。

一道题 2

先考虑一条链怎么做。设 1 两侧链的长度为 a, b。

一道题 2

先考虑一条链怎么做。设 1 两侧链的长度为 a, b。

如果 $a = b$，那么后手可以**镜像**操作：每次先手往外移动后，后手移动到上一个位置关于中心的镜像，这样先手只能再向外移动。这直接说明后手必胜。

一道题 2

先考虑一条链怎么做。设 1 两侧链的长度为 a, b。

如果 $a = b$，那么后手可以**镜像**操作：每次先手往外移动后，后手移动到上一个位置关于中心的镜像，这样先手只能再向外移动。这直接说明后手必胜。

否则，先手第一步可以移动到刚才的情况：移到一个位置使得两侧没移动过的位置相同，然后就变成 $a = b$ 的情况。

因此先手败当且仅当 $a = b$，即 1 是链中点。

如何扩展到树上？

一道题 2

直观的想法:

$$\text{链} \Leftrightarrow \text{直径}$$

$$\text{链中点} \Leftrightarrow \text{直径中点}$$

在直径上考虑上述博弈，可以发现是对的：先手必败当且仅当 1 是直径中点。

一道题 2

直观的想法：

$$\text{链} \Leftrightarrow \text{直径}$$

$$\text{链中点} \Leftrightarrow \text{直径中点}$$

在直径上考虑上述博弈，可以发现是对的：先手必败当且仅当 1 是直径中点。
~~然后是长链剖分优化 DP。~~

「雅礼集训 2017 Day2」棋盘游戏 / 「LibreOJ Round #6」花札 / [CERC2022] Combination Locks

有一个 $n \times m$ 的网格，上面有一些位置是障碍。

先放一枚棋子，然后双方轮流移动棋子（四相邻），棋子不能经过重复位置。不能移动者输。

求哪些位置使得先手必胜。

$n, m \leq 100$

有一个无向图，双方轮流移动，不能重复经过点。不能操作者输。

顶级结论题：先手必胜当且仅当任意一个最大匹配包含起点 s。

有一个无向图，双方轮流移动，不能重复经过点。不能操作者输。

顶级结论题：先手必胜当且仅当任意一个最大匹配包含起点 s。

Part 1：如果任意最大匹配包含 s，则先手胜。

证明.

先手采用如下策略：任取一最大匹配，每次沿最大匹配走到另一侧。

若后手走到一个不在匹配中的点，则按照整个游戏走的路径进行增广：先手走的边在匹配中，后手走的边不在，终点不在匹配中；这可以得到一个不包含 s 的最大匹配，矛盾。 □

Part 2: 如果存在最大匹配不包含 s，则后手胜。

证明.

后手采用如下策略：取一不包含 s 的最大匹配，每次沿最大匹配走到另一侧。
先手第一步必定走到匹配中的点（否则匹配可以加边）。类似刚才的分析，如果先手又走到一个不在匹配中的点，则这又是一条增广路：开头结尾不在匹配中，中间后手走的边是匹配。这样增广可以得到更大的匹配，矛盾。 □

虽然结论对于一般图也成立，但一般图最大匹配比较困难，所以题一般是二分图。

「雅礼集训 2017 Day2」棋盘游戏

然后就是判定是否一个点必然在最大匹配中，但这里不能做 $O(nm)$ 次最大匹配。

匈牙利算法后好像可以判定，复杂度 $O(n^2m^2)$。

可以直接网络流：dinic 做完最大匹配后，问一个点是否一定在最大匹配内相当于问是否存在一条增广路把它移出去，那么相当于能否从自己这一侧的源/汇点到达它。在残量网络上 dfs 即可。

复杂度应该是 $O((nm)^{1.5})$

[NOI2011] 兔兔与蛋蛋游戏

有一个 $n \times m$ 的网格图，上面除了一个空位，每个格子上都有一枚黑棋或者白棋。

双方轮流操作，先手黑色后手白色，每次可以将一枚自己颜色，且与空格相邻的棋子移到空格。求谁获胜。

$n, m \leq 40$

[NOI2011] 兔兔与蛋蛋游戏

关键结论：空格不会回到原先的位置。

证明.

如果空格绕了一圈回去，那么因为网格是二分图，将空格移回去的人和第一步操作的人不同。但放进第一个格子的棋子是第一步操作的人的颜色。 □

所以还是之前的问题。

有一棵 n 个点的树和 m 条额外边。Alice 和 Bob 在图上进行如下博弈：

两人轮流行动，每次走一条边或者不走，Alice 先手。Alice 可以走树边和额外边，Bob 只能走树边。若两人相遇则 Bob 胜。如果始终不能相遇则 Alice 胜。

求多少种初始状态使得 Alice 必胜。

$n, m \leq 10^5$

部分分：$m = 1$

$m=1$ 怎么做?

显然不用特殊边必败，那么我们必须走到特殊边和其对应的路径上去。此时 Alice 可以走一个环，Bob 只能走除去特殊边之外的链。记环长为 l。

$l=2$ 相当于没有，所以 Bob 胜。

$l=3$ 时 Bob 可以在中间控制所有点，然后 Alice 被迫走下环，Bob 就赢了。

$m=1$ 怎么做?

显然不用特殊边必败，那么我们必须走到特殊边和其对应的路径上去。此时 Alice 可以走一个环，Bob 只能走除去特殊边之外的链。记环长为 l。

$l=2$ 相当于没有，所以 Bob 胜。

$l=3$ 时 Bob 可以在中间控制所有点，然后 Alice 被迫走下环，Bob 就赢了。

$l \geq 4$ 时，Bob 最多控制三个点，所以 Alice 在特殊边两侧反复横跳就赢了。那么 Alice 只要走到环上且下一步不被抓住，就一定能走过去。

那么，Alice 只需要尽快走到环上，Bob 则会去拦截。因为此时双方都走树边，那么 Alice 胜当且仅当 Alice 向上走到环上点的距离小于 Bob 走到这个点的距离。

有很多种做法，比如枚举 Alice 位置，然后点分治算 Bob 可行位置数。也可以长链剖分。

一般情况怎么做?

类似之前的分析，只要 Alice 到了某个环长 ≥ 4 的额外边的路径上、且不被立刻抓住就赢了。记这些点为特殊点。

一般情况怎么做?

类似之前的分析，只要 Alice 到了某个环长 ≥ 4 的额外边的路径上、且不被立刻抓住就赢了。记这些点为特殊点。

另一方面，如果所有环长都是 3，那 Alice 在树上最多跳一步，这样跳出去也会被 Bob 抓住，因此 Alice 此时必败。所以 Alice 还是得走到特殊点。

如果树上往两个方向走都有特殊点，那么 Alice 必胜：往 Bob 不在的方向走即可。因此这些点也相当于特殊点。

那么还是变成往上走到特殊点的形式，但这里还有环长为 3 的边。

然后是超级细节讨论。

- 如果 Alice, Bob 在同一个子树内，那么由于 Alice 可以通过非树边跳，Bob 最好抓的时机必定是两人初始位置的 LCA：Bob 能先到达这里胜。但还需要注意 Alice 可以跳过 LCA 的情况，此时 Bob 需要提前到达这里。
- 如果 Bob 在子树外，那时机还是根处。但类似地有特殊情况：Alice 可以有很多种方式跳过根，但不能跳到 Bob 的方向。

实现可以点分治，可以长链剖分，可以多种技巧一起用。
细节极其恐怖。
复杂度可以是 $O(n+m)$，但是 $O((n+m)\log^2)$ 也不是问题只要你写对

Contents

权值博弈

从这里开始，我们默认考虑无环情况。

一种常见的博弈变体是，双方不只是比一个胜负，而是博弈地最大/最小化一个权值：每个终止状态会给出一个分数 v，通常先手希望最大化 v，后手希望最小化 v。问题问双方最优操作下的结果。

这完全可以看成胜负情况的一个扩展：我们用 $+1$ 表示第一个人获胜，-1 表示第二个人获胜。唯一的区别在于之前是记录先手胜/后手胜，但在单个游戏的情况下，我们可以在状态中标记先后手，然后给后手态乘 -1。

权值博弈

从这里开始，我们默认考虑无环情况。

一种常见的博弈变体是，双方不只是比一个胜负，而是博弈地最大/最小化一个权值：每个终止状态会给出一个分数 v，通常先手希望最大化 v，后手希望最小化 v。问题问双方最优操作下的结果。

这完全可以看成胜负情况的一个扩展：我们用 $+1$ 表示第一个人获胜，-1 表示第二个人获胜。唯一的区别在于之前是记录先手胜/后手胜，但在单个游戏的情况下，我们可以在状态中标记先后手，然后给后手态乘 -1。

容易得到直接的转移方式：记 $f(s)$ 表示状态 s 的值，记 t_i 为 s 的所有后继状态，则

- 先手操作的状态：$f(s) = \max f(t_i)$
- 后手操作的状态：$f(s) = \min f(t_i)$

比之前的情况更像一个 DP。

权值博弈

简单 DP 的情况就不讲了。

我们之前提过一个思想：

如果一方用一个简单的操作就能达到某种结果，那他的最终结果不会更差。

在权值的情况下它更有用：如果先手能保证不小于某个值，后手能保证不大于某个值，那么答案就是这个。

Fox and Card Game(Easy)

有 n 个非负序列。每个序列长度 l_i 都是**偶数。**

双方轮流操作，第一个人可以从序列开头拿走一个数，第二个人可以从序列结尾拿走一个数。双方最大化自己得分（显然等价于第二个人最小化第一个人得分），求最优操作下双方得分。

$n, l_i \leq 100$

Fox and Card Game(Easy)

如果第二个人复读第一个人的操作，那么双方平分序列。

如果第一个人先随便操作一下，然后复读第二个人操作直到第二个人操作回来，然后重复，那么双方平分序列。

所以答案就是第一个人拿前一半，第二个人拿后一半。

[CF388C] Fox and Card Game

有 n 个非负序列。每个序列长度 l_i 任意。

双方轮流操作，第一个人可以从序列开头拿走一个数，第二个人可以从序列结尾拿走一个数。双方最大化自己得分（显然等价于第二个人最小化第一个人得分），求最优操作下双方得分。

$n, l_i \leq 100$

[CF388C] Fox and Card Game

如果序列长度变为偶数，那么双方都可以用上述操作平分序列。
那么对于长度为奇数的序列，一定是一个人先拿一个，然后双方保证平分。
于是直接变成，两侧归双方，然后先操作的人拿到中心元素。

有 n 个非负元素，双方轮流拿，求最优操作下的结果。

显然是排序后从大到小依次拿。$O(n\log n + nl)$

[CF794E] Choosing Carrot

有一个长度为 n 的序列，双方轮流操作，每次从开头或结尾拿走一个数。

剩一个数时停止，这个数即为得分。第一个人先手，希望最大化得分；第二个人希望最小化得分。

进一步，对于每个 k，求出如果第一个人可以先操作 k 次时，游戏的结果。

$n \le 10^5$

[CF794E] Choosing Carrot

先考虑 $k=0$。~~二分答案然后打表 01 情况~~

和刚才类似，后手可以镜像先手的操作。这样当 n 是奇数时，后手可以确保不大于中心元素。而 n 是偶数时，后手可以确保不大于中心两元素的较大值（先手最后操作）。

考虑另一方向。如果 n 是偶数，那么先手可以做到这个界：删掉中心较小元素那一侧，然后就变成了较大的在中心，接着镜像操作。因此 n 为偶数时，答案就是中心两元素的较大者。

先考虑 $k=0$。~~二分答案然后打表 01 情况~~

和刚才类似，后手可以镜像先手的操作。这样当 n 是奇数时，后手可以确保不大于中心元素。而 n 是偶数时，后手可以确保不大于中心两元素的较大值（先手最后操作）。

考虑另一方向。如果 n 是偶数，那么先手可以做到这个界：删掉中心较小元素那一侧，然后就变成了较大的在中心，接着镜像操作。因此 n 为偶数时，答案就是中心两元素的较大者。

但 n 是奇数的时候，答案显然不是这个界：考虑 0 1 0。

[CF794E] Choosing Carrot

n 为偶数时，答案就是中心两元素的较大者。

但解决了偶数的情况后，奇数的情况就容易了：操作一步变成偶数情况，但先后手反过来，所以先手会选择较大的答案。那么答案就是 $\max(\min(a_{m-1}, a_m), \min(a_m, a_{m+1}))$。

[CF794E] Choosing Carrot

n 为偶数时，答案就是中心两元素的较大者。

但解决了偶数的情况后，奇数的情况就容易了：操作一步变成偶数情况，但先后手反过来，所以先手会选择较大的答案。那么答案就是 $\max(\min(a_{m-1}, a_m), \min(a_m, a_{m+1}))$。

然后考虑原问题，任意删相当于在一个区间内选择中心，随便做即可。复杂度 $O(n)$。

Contents

多个独立博弈

在很多时候，博弈问题可以被分为若干个独立的子问题，满足如下性质：

- 每个人每次选择**一个**子博弈，在上面进行一步操作。如果都不能操作则输。

这时，之前不同设定的区别就显现出来了：将之前对不同设定的处理方式套用到每个子游戏上显然不对。事实上不同设定的表现通常**完全不一样**。

对于最一般的情况（Normal（不能操作输）/Normal（无环）/Impartial（平等）），我们有 SG 定理：每个状态可以被表示为一个非负整数，或者 Nimber。

~~对于不平等博弈，有超现实数等理论，但这极其高深。~~

~~对于（平等）Loopy Game，有 Loopy value 和 Smith's Rule，甚至有人这样出题 (DoubleXorGame)~~

~~对于 Misère Game，……~~

如果有兴趣，可以读读 Winning Ways for Your Mathematical Plays，保证内容超纲。

平等博弈与 Sprague-Garundy 定理

在有限的平等博弈下，SG 定理可以用如下方式描述：

- 每个状态可以用一个非负整数的 SG 值表示。状态 S 为先手必胜当且仅当 $SG(s) \neq 0$。
- 一个状态的 SG 值可以使用如下方式计算：记其所有后继状态为 t_i，则 $SG(s) = \text{mex}(SG(t_1), SG(t_2), \cdots)$，即所有后继 SG 值中第一个没有出现的非负整数。
- SG 定理：如果状态 s 由两个独立的子博弈 s_1, s_2 组成，则 $SG(s) = SG(s_1) \oplus SG(s_2)$。

众所周知的例子：

Nim

有多堆石子。每次操作选择一堆石子，从其中拿走任意个（至少一个)。

每堆石子是独立的（注意这里没有其它限制，例如之前的那几个题多堆情况都不是独立的)。用 i 表示一堆 i 个石子的状态。

i 可以转移到 $0, 1, 2, \cdots, i-1$。直接归纳可得 $SG(i) = i$。

那么多堆情况的 SG 值就是每一堆大小的异或和。

SG 的直接求法

经典问题

有一张**有向无环图**，图上有很多枚棋子。双方轮流操作，每次操作可以将棋子沿着当前点的一条出边移动。不能操作者输，求获胜方。

$n \le 10^6$

每枚棋子是一个独立问题。我们只需要求出每个点出发的 SG 函数，然后异或起来。

那么直接拓扑序后用 $SG(s) = \text{mex}(SG(t_1), SG(t_2), \cdots)$ 求即可。

小结论：$SG(s) = O(\sqrt{m})$：在有了 $0, 1, \cdots, v-1$ 的情况下，我们还需要 v 条边以构造 $SG = v$。

观察法 2

和之前一样，除了直接求外，SG 函数也有两种常见处理方法：观察和推性质。

[ARC107E] Mex Mat

有一个 $n \times n$ 的矩阵 a。给定第一行第一列，然后剩下的满足 $a_{i,j} = \mathrm{mex}(a_{i-1,j}, a_{i,j-1})$。求矩阵中 $0, 1, 2$ 的个数。

$n \leq 2 \times 10^5$

[ARC107E] Mex Mat

```
30
2 2 0 2 1 1 2 2 2 0 2 0 1 2 1 0 0 0 2 1 0 0 2 1 0 0 1 2 2 1
2 0 1 0 2 0 1 0 1 2 0 1 0 1 0 1 2 1 0 2 1 2 0 2 1 2 0 1 0 2
0 1 0 1 0 1 0 1 0 1 2 0 1 0 1 0 1 0 1 0 2 0 1 0 2 0 1 0 1 0
0 2 1 0 1 0 1 0 1 0 1 2 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
1 0 2 1 0 1 0 1 0 1 0 1 2 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
0 1 0 2 1 0 1 0 1 0 1 0 1 2 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
2 0 1 0 2 1 0 1 0 1 0 1 0 1 2 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
0 1 0 1 0 2 1 0 1 0 1 0 1 0 1 2 0 1 0 1 0 1 0 1 0 1 0 1 0 1
2 0 1 0 1 0 2 1 0 1 0 1 0 1 0 1 2 0 1 0 1 0 1 0 1 0 1 0 1 0
2 1 0 1 0 1 0 2 1 0 1 0 1 0 1 0 1 2 0 1 0 1 0 1 0 1 0 1 0 1
0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 1 0 1 2 0 1 0 1 0 1 0 1 0 1 0
2 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 1 0 1 2 0 1 0 1 0 1 0 1 0 1
2 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 1 0 1 2 0 1 0 1 0 1 0 1 0
2 0 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 1 0 1 2 0 1 0 1 0 1 0 1
0 1 0 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 1 0 1 2 0 1 0 1 0 1 0
0 2 1 0 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 1 0 1 2 0 1 0 1 0 1
0 1 0 1 0 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 1 0 1 2 0 1 0 1 0
0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 1 0 1 2 0 1 0 1
2 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 1 0 1 2 0 1 0
1 2 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 1 0 1 2 0 1
0 1 2 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 1 0 1 2 0
0 2 0 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 1 0 1 2
1 0 1 0 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 1 0 1
0 1 0 1 0 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 1 0
0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 1
1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0
2 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1
2 0 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 2 1 0 1 0
0 1 0 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 2 1 0 1 0 1 0 2 1 0 1
```

每条对角线上几乎相同——除了开头几个。

可以大力证明，第四行第四列之后都和之前一样——$a_{4,4} = a_{5,5}$。
然后就是 $O(n)$

简单 SG 练习

Nim，但是每次只能拿不超过 k 个。

简单 SG 练习

Nim，但是每次只能拿不超过 k 个。

$0, 1, 2, \cdots, k, 0, 1, 2, \cdots k$。

显然是 $i \bmod (k+1)$

简单 SG 练习

Nim，但是每次只能拿 $[l, r]$ 个。

Nim，但是每次只能拿 $[l, r]$ 个。

```
357 7 53
0 0 0 0 0 0 0 1 1 1 1 1 1 1 2 2 2 2 2 2 2 3 3 3 3 3 3 3 4 4 4 4 4 4 4 5 5 5 5 5 5 5 6 6 6 6 6 6 6 7 7 7 7 7 7 7 8 8 8 8
0 0 0 0 0 0 0 1 1 1 1 1 1 1 2 2 2 2 2 2 2 3 3 3 3 3 3 3 4 4 4 4 4 4 4 5 5 5 5 5 5 5 6 6 6 6 6 6 6 7 7 7 7 7 7 7 8 8 8 8
0 0 0 0 0 0 0 1 1 1 1 1 1 1 2 2 2 2 2 2 2 3 3 3 3 3 3 3 4 4 4 4 4 4 4 5 5 5 5 5 5 5 6 6 6 6 6 6 6 7 7 7 7 7 7 7 8 8 8 8
0 0 0 0 0 0 0 1 1 1 1 1 1 1 2 2 2 2 2 2 2 3 3 3 3 3 3 3 4 4 4 4 4 4 4 5 5 5 5 5 5 5 6 6 6 6 6 6 6 7 7 7 7 7 7 7 8 8 8 8
0 0 0 0 0 0 0 1 1 1 1 1 1 1 2 2 2 2 2 2 2 3 3 3 3 3 3 3 4 4 4 4 4 4 4 5 5 5 5 5 5 5 6 6 6 6 6 6 6 7 7 7 7 7 7 7 8 8 8 8
0 0 0 0 0 0 0 1 1 1 1 1 1 1 2 2 2 2 2 2 2 3 3 3 3 3 3 3 4 4 4 4 4 4 4 5 5 5 5 5 5 5 6 6 6 6 6 6 6 7 7 7 7 7 7 7 8 8
```

$\left\lfloor \frac{i \bmod (l+r)}{l} \right\rfloor$。直接归纳就能证。

简单 SG 练习

Nim，但是每次不能拿正好 l 个。

简单 SG 练习

Nim，但是每次不能拿正好 l 个。

```
300 11
000 001 002 003 004 005 006 007 008 009 010 000 001 002 003 004 005 006 007 008 009 010
011 012 013 014 015 016 017 018 019 020 021 011 012 013 014 015 016 017 018 019 020 021
022 023 024 025 026 027 028 029 030 031 032 022 023 024 025 026 027 028 029 030 031 032
033 034 035 036 037 038 039 040 041 042 043 033 034 035 036 037 038 039 040 041 042 043
044 045 046 047 048 049 050 051 052 053 054 044 045 046 047 048 049 050 051 052 053 054
055 056 057 058 059 060 061 062 063 064 065 055 056 057 058 059 060 061 062 063 064 065
066 067 068 069 070 071 072 073 074 075 076 066 067 068 069 070 071 072 073 074 075 076
077 078 079 080 081 082 083 084 085 086 087 077 078 079 080 081 082 083 084 085 086 087
088 089 090 091 092 093 094 095 096 097 098 088 089 090 091 092 093 094 095 096 097 098
099 100 101 102 103 104 105 106 107 108 109 099 100 101 102 103 104 105 106 107 108 109
110 111 112 113 114 115 116 117 118 119 120 110 111 112 113 114 115 116 117 118 119 120
121 122 123 124 125 126 127 128 129 130 131 121 122 123 124 125 126 127 128 129 130 131
132 133 134 135 136 137 138 139 140 141 142 132 133 134 135 136 137 138 139 140 141 142
143 144 145 146 147 148 149 150 151 152 153 143 144 145 146
```

简单 SG 练习

有一个矩形，每次选择一个的矩形的一边，选择边长 l 的一个因子 d，将矩形这一边切成 d 份。

简单 SG 练习

有一个矩形，每次选择一个的矩形的一边，选择边长 l 的一个因子 d，将矩形这一边切成 d 份。

打表...吗?

$a \oplus a = 0$，切偶数份显然得到 $SG = 0$，切奇数份相当于留一份。

如果只有奇数，相当于每次在一侧拿掉任意多个质因子。那么记 d_u 表示 u 的质因子数(重复出现记多次)，问题就是两堆质因子的 Nim。答案是 $d_n \oplus d_m$。

如果有偶数会发生啥?

简单 SG 练习

有一个矩形，每次选择一个的矩形的一边，选择边长 l 的一个因子 d，将矩形这一边切成 d 份。

打表...吗?

$a \oplus a = 0$，切偶数份显然得到 $SG = 0$，切奇数份相当于留一份。

如果只有奇数，相当于每次在一侧拿掉任意多个质因子。那么记 d_u 表示 u 的质因子数 (重复出现记多次)，问题就是两堆质因子的 Nim。答案是 $d_n \oplus d_m$。

如果有偶数会发生啥？操作偶数都变成 0，相当于只操作奇数的情况下，每个转移又多出去一个 $SG = 0$。因为 SG 都是 mex，这样相当于在 mex 上去掉 0，那么一定全部加一。答案是不考虑 2 的情况下，$(d_n \oplus d_m) + 1$。

“mex 上去掉一些元素”的思想有一个超级好题 (loj3633)，但这里没时间讲了。

翻石子模型

有一些石子，每次操作可以拿掉一个位置 x 上的石子，然后根据 x，翻转 x 之后某些格子上石子的状态：有就删掉，没有就加入。

注意到 $x \oplus x = 0$，那么有的情况下再放一枚和这个等价。这样就是独立问题了。

「HAOI2015」数组游戏

有一个长度为 n 的 01 序列。每次操作可以：选择一个 1，记这个位置为 x，则再选择一个正整数 k，翻转 $x, 2x, 3x, \cdots, kx$。不能操作者输。

求谁获胜。

$n \leq 10^9$，只有 100 个 1。

根据刚才的结论，每个位置可以有一个 sg_i，然后 $sg_i = \text{mex}_{k\geq 1}(sg_{2i} \oplus sg_{3i} \oplus \cdots sg_{ki})$。
打表：

「HAOI2015」数组游戏

根据刚才的结论，每个位置可以有一个 sg_i，然后 $sg_i = \text{mex}_{k\geq 1}(sg_{2i} \oplus sg_{3i} \oplus \cdots sg_{ki})$。
打表：

```
233
6 4 6 2 2 5 2 4 1 1 1 4 1 1 1 1 4 4 4 4 4 4 4 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1
```

结论：sg_i 只和 $\lfloor \frac{n}{i} \rfloor$ 有关。冷静一下这是显然的：只用到了 i 的倍数。

「HAOI2015」数组游戏

每段选一个位置求，考虑暴力往后面 $+k$，但一整个段可以快速跳过去。

复杂度分析：对于 $\frac{n}{i} = k$ 的一段，其长度为 $O(\frac{n}{k^2})$。因此以 k 为距离向后跳时，大概在 $\frac{n}{i} = \sqrt{\frac{n}{k}}$ 时后面每一段可以跳。后面有 $\sqrt{\frac{n}{k}}$ 段，此部分积分得 $O(n^{3/4})$。前面距离为 $\sqrt{nk}$，因此跳 $\sqrt{\frac{n}{k}}$ 次，结果一样。

所以复杂度是 $O(n^{3/4})$

Contents

[AGC043C] Giant Graph

给定三张 n 个点的图 G_1, G_2, G_3。使用如下方式定义一张 n^3 个点的大图：

- 点标号为 (u_1, u_2, u_3)，点权为 $(10^{18})^{u_1+u_2+u_3}$。
- 两点 $(u_1, u_2, u_3), (v_1, v_2, v_3)$ 间有边当且仅当存在 i 使得 u_i, v_i 在 G_i 间有边，且对于其它下标 $u_j = v_j$。

求该图最大权独立集点权和，模 998244353。

$n \le 10^5$

[AGC043C] Giant Graph

先考虑怎么选答案。

由于 10^{18} 大于点数，可以发现最优解一定是先尽量让 $u_1 + u_2 + u_3$ 最大（$= 3n$）的点选的最多，在此基础上让下标和等于 $3n - 1$ 的选的最多，接下来同理。

考虑一种下标和 $u_1 + u_2 + u_3$ 内的问题。由于两点相邻当且仅当两维相同、另一维原图上相邻，同一种下标和内部不存在连边。因此一个下标和内的一定是选当前所有能选的。

又因为这样个数最大的方案唯一，不需要考虑此处选择对剩余问题的影响。从而如下贪心可以得到最优解：

按照 $u_1 + u_2 + u_3$ 从大到小考虑所有点，如果不存在与其相邻的已选点则选择当前点。

但直接做是 $O(n^3)$ 的。

[AGC043C] Giant Graph

按照 $u_1 + u_2 + u_3$ 从大到小考虑所有点，如果不存在与其相邻的已选点则选择当前点。

考虑贪心的过程，设 $f_{i,j,k}$ 表示是否选 (i, j, k)。可以得到如下形式：

对于 $f_{i,j,k}$，考虑与其相邻且下标和更大的点集。

- 如果其中存在 (u_1, u_2, u_3) 使得 $f_{u_1,u_2,u_3} = 1$，则 $f_{i,j,k} = 0$。
- 否则，$f_{i,j,k} = 1$。

这是什么？

[AGC043C] Giant Graph

按照 $u_1 + u_2 + u_3$ 从大到小考虑所有点，如果不存在与其相邻的已选点则选择当前点。

考虑贪心的过程，设 $f_{i,j,k}$ 表示是否选 (i, j, k)。可以得到如下形式：

对于 $f_{i,j,k}$，考虑与其相邻且下标和更大的点集。

- 如果其中存在 (u_1, u_2, u_3) 使得 $f_{u_1,u_2,u_3} = 1$，则 $f_{i,j,k} = 0$。
- 否则，$f_{i,j,k} = 1$。

这是什么？博弈问题。

$f_{i,j,k}$ 表示是否后手必胜。其中每一步能走到一个相邻且下标和更大的点集。

[AGC043C] Giant Graph

两点 $(u_1, u_2, u_3), (v_1, v_2, v_3)$ 间有边当且仅当存在 i 使得 u_i, v_i 在 G_i 间有边，且对于其它下标 $u_j = v_j$。

从博弈的角度观察这个问题。每一次移动为在 (i, j, k) 中选择一个，将其移动到原图上与其相邻且下标更大的点。

那么这是三个完全独立的游戏。根据 SG 定理，可以在三张图上分别求出 i, j, k 对应的 SG 值，然后 $f_{i,j,k} = 1$ 当且仅当 $SG_1(i) \oplus SG_2(j) \oplus SG_3(k) = 0$。求出 SG 值后容易计算这个结果。注意到 $SG(i) \le O(\sqrt{m})$ 则更容易计算。

复杂度 $O(n + m)$。

[AGC016F] Games on DAG

给一个 n 个点的有向无环图。$1, 2$ 上分别有两枚棋子，做传统的移边博弈。现在可以任意删一些边，求有多少种方式使得先手必胜。

$n \leq 15$

先手必胜当且仅当 $SG(1) \neq SG(2)$，因此考虑对着 SG 做。
因为 SG 是 mex，因此如果确定了每个点的 SG 值，则图需要满足的条件显然为：

对于一个 $SG = i$ 的点，其连向了 $SG = 0, 1, \cdots, i-1$ 的每种点，且不连向 $SG = i$ 的点。

但我们显然不能枚举 SG，因此考虑 DP。

[AGC016F] Games on DAG

对于一个 $SG=i$ 的点，其连向了 $SG=0,1,\cdots,i-1$ 的每种点，且不连向 $SG=i$ 的点。

考虑按照 SG 值从小到大加进去，如果之前已经加入了 $SG=0,1,\cdots,i-1$ 的点，现在需要加入 $SG=i$ 的，则它们需要连向之前每一个 SG 值，这显然没法维护。

[AGC016F] Games on DAG

对于一个 $SG=i$ 的点，其连向了 $SG=0,1,\cdots,i-1$ 的每种点，且不连向 $SG=i$ 的点。

考虑按照 SG 值从小到大加进去，如果之前已经加入了 $SG=0,1,\cdots,i-1$ 的点，现在需要加入 $SG=i$ 的，则它们需要连向之前每一个 SG 值，这显然没法维护。

换一个方向，加入时限制 $SG=i$ 的需要连向剩余每个点，这样转移就对了。

dp_S 表示当前加入到 S 的方案数，枚举下一个 SG 值的集合 T 转移。转移系数是一个集合需要连向剩下每个点，且集合内没有边。先处理每个点到集合的系数，然后再求和，复杂度 $O(3^n)$。

[CF1033G] Chip Game

我们只考虑博弈，外层计数留作练习。

有 n 堆石子，Alice 每次必须从一堆拿正好 a 个，Bob 每次拿正好 b 个，求谁获胜。

这是不平等博弈，游戏有四种情况：Alice 胜，Bob 胜，先手胜，后手胜，求是哪种情况。

[CF1033G] Chip Game

首先，双方都可以进行镜像操作：如果对手操作一堆，我也操作一堆。可以发现这相当于每一堆对 $a+b$ 取模。

[CF1033G] Chip Game

首先，双方都可以进行镜像操作：如果对手操作一堆，我也操作一堆。可以发现这相当于每一堆对 $a+b$ 取模。

我们用 $\{S|T\}$ 表示如果 Alice 操作，可以得到 S 中的状态，如果 Bob 操作可以得到 T 中的状态。不妨设 $a \le b$。分类讨论：

- 如果 $s < a$，则都不能进行操作，此时 $\{|\} = 0$。
- 如果 $a \le s < b$，则只有 Alice 能操作，此时是 $\{0|\}$。这个状态对 Alice 严格更优，记作 1。
- 如果 $b \le s \le 2a$，则双方都能操作，但总共只能操作一次。此时是 $\{0|0\} = *$，或者说 Nimber 1。注意这两个 1 不是同一个东西。
- 如果 $b \le s, 2a \le s$，那么 Bob 操作一次就没了，Alice 操作完还有，这是 $\{c|0\}(c > 0)$。

[CF1033G] Chip Game

$\{0|\} = 1, \{0|0\} = *, \{c|0\}(c > 0)$

超现实数入门。

$\{c|0\}$ 不比 0 差。$\{0|\} = 1$ 严格比 0 好。如果有 $\{0|\} = 1$，则 Alice 必胜：先操作别的东西，最后再动 1 即可。

否则，$\{c|0\}(c > 0)$ 是 Hot Game，操作这个比操作 Nimber 更优：Alice 操作一次这个就赢了。所以如果有一个，则 Alice 能先手胜。如果有至少两个，则 Alice 必胜。

如果都没有，就只剩下 Nimber，然后是平等博弈。

Thanks!