# NEERC 11-13 题目选讲

北京大学 杜宇飞

Prologue

# ACM-ICPC 竞赛的一些特点

- 题目数量多。一般在 5 小时的竞赛中有 9 - 13 个题目不等。
- 没有部分分。题目只有 Accepted 与否两种得分。
  - 这决定了在 ACM 竞赛中，不会做的题很难拿到手。
  - 对选手的代码能力提出了很大的挑战。尤其是一些难题。
- 为了照顾弱队，ACM 竞赛中会有一些水题。
- 题目的思维性比较强。

(不过神犇看来都是水题)

- 很少用到 OI 比赛中的各种神奇数据结构。

11B - Binary Encoding

# 题目大意

- 为 0 至 m - 1 之内的数字进行二进制编码。
- 若 n 是最小的满足 $m \le 2^n$ 的 n，码长应为 n 或 n - 1。
- 要求：
  - 较小的数字的码长不应大于较大的数字的码长
  - 若将数字升序排列，编码也应是升序的
  - 所有数字编码唯一
  - 所有数字的编码都不是其他数字编码的前缀
  - 所有数字的码长总和最小化
- 请你给出编码方案。

# 讨论

- 这是一个水题。请神犇来秒！

# 解答

- 若有 t 个数字的码长是 n - 1，那么它们会令 2t 个长度是 n 的编码无法使用。
- 而长度是 n 的编码应有 m - t 个。所以 $m - t \le 2^n - 2t$。
- 解得 $t \le 2^n - m$。

# 解答

- 所以将前 $2^n - m$ 个 $n - 1$ 位二进制数用于编码前 $2^n - m$ 个数。
- 后 $m - 2^{n-1}$ 个 $n - 1$ 位二进制数，每个数分别带上后缀 0 和 1 作为后 $2m - 2^n$ 个数的编码。
- 即编码完成了 m 个数。

11D - Dictionary Size

# 题目大意

- 某国颁布了一本 n ($n \le 10\,000$) 个单词的词典。单词长度不超过 40。
- 该国的造词法是这样的:
  - 词典中的单词是一个词。
  - 能分为两部分的,其中前一部分是一个词典词或者其*非空前缀*,后一部分是一个词典词或者其*非空后缀*的词。
- 求能造出的*不相同*的词汇数目。

# 讨论

- 计数问题。
  - 如何不重不漏地计数？
  - 如何快速计数？

# 解答

- 对于一个词汇，可能可以在不同的地方分为两部分，使得它们都满足要求。如何消除这种重复计数？
  - 选取最靠后的分隔位置，也就是令前缀极长。
  - 不过这种方法有一种情况不能正确处理……

# 解答

- 例如，词典 {back, c}：
  - “bac” 是合法词汇：ba | c
  - 不过 ba 并不是极长前缀。
  - 但是 bac | 并不是正确的分隔（后缀应为非空）。
- 也就是说，词典词的非空前缀也可能是合法单词。
  - 这种情况仅仅会发生在该前缀本身不是一个词典词，并且前缀的*最后一个字母*是合法后缀的情况。
  - 特判即可。

# 解答

- 快速计数的方式就很简单了。
  - 建立一棵 Trie，以遍历所有合法前缀。
  - 在 Trie 的每个节点，若该节点没有孩子 c，则答案可以加上 c 开头的合法后缀的数目（极长前缀）。
    - 统计合法后缀可以将所有单词倒序插入另一个 Trie，以消除相同的后缀。
  - 若该节点有孩子 c，则可以执行前面的特判。

11G - GCD Guessing Game

# 题目大意

- Andrew 要猜测 Paul 的年龄，目前已知 Paul 的年龄在 1 至 n (n ≤ 10 000) 之间。
- Andrew 每次猜测一个 1 至 n 之间的数，Paul 会告知这个数与他的年龄的最大公约数。Andrew 可以根据 Paul 的答案决定下一步的问题。
- 求最坏情形下，Andrew 需要询问的次数。

11G - GCD Guessing Game

# 讨论

- 最好的策略是怎样的？
- 为什么这个策略是最好的？

11G - GCD Guessing Game

# 解答

- 对于询问 k，如果回答是 t，且 t > 1，那么相当于本次询问的是 $\frac{k}{t}$，回答是 1，问题规模由 n 下降到了 $\left\lfloor\frac{n}{t}\right\rfloor$。
- 在 n 比较大的时候，通过一次询问降低了问题的规模，这显然是比回答是 1 更有利的。而在 n 较小的时候，通过试验（状压 DP 等方式）我们发现这也不会让情况更坏。

# 解答

- 于是我们得出一个结论：

  Andrew 的年龄是 1 是最坏的情况。

- 为了确认 Andrew 的年龄是 1，需要询问所有的素数。
- 问题就演变成了用最少的询问次数，确定 Andrew 的年龄中不包含任何素因子。

# 讨论

- 如果两个素数 $p$ 和 $q$ 有 $pq \le n$，询问 $pq$ 就可以同时消除这两个素因子。
- 所以素数可以“打包”进行询问。
- 如何“打包”才能得到最少的询问次数？

11G - GCD Guessing Game

# 解答

- 我们先将 1 至 n 中所有的素数组成一个集合，然后从大到小考虑各个素数。
- 对于素数 p，如果集合中没有任一个数 q 满足 $pq \le n$，那么就只能单独地询问 p 了。
- 否则，将最大的满足条件的 q 与 p 打包，再继续在集合中查找。这显然比单独询问 p，或者将更小的 q 与之打包要优。
- 这样做的结果与下面做法在题目所给的范围中是等价的：每一个小于 $\sqrt{n}$ 的素数，都与能与它配对的最大的素数配对。也就是最多两个素数打包。

# 解答

- 于是本题便做完了。利用线性筛找出素数，时间复杂度是 O(n)。

11J - Journey

# 题目大意

- 在平面上有个机器人。作为一个有理想的机器人，它给自己写了个自动程序。
- 程序有 n ($n \le 100$) 个函数，每个函数都有不超过 100 条指令。指令分为以下几种:
  - GO：向机器人面向的方向走一步。
  - LEFT / RIGHT：左转/右转。
  - F$k$：调用第 k 个函数。
- 机器人初始站在原点，从 F1 函数开始执行。问机器人在程序执行过程中会到达的点中与原点的曼哈顿距离最大是多少。
- 注意，机器人可能走到无限远，程序也可能不会停止。

# 讨论

- 如何处理曼哈顿距离?
- 如何处理无限循环?

# 解答

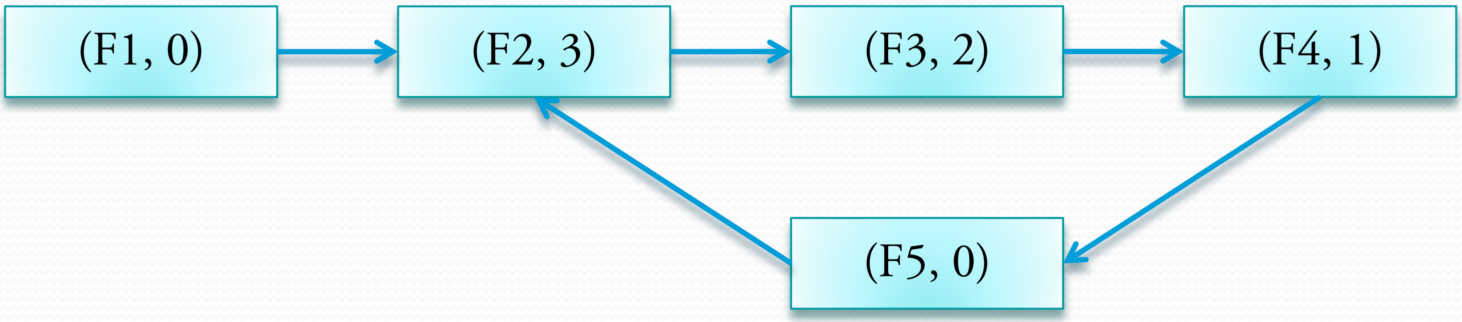
- $|x| = \max\{x, -x\}$
- $|x| + |y| = \max\{x + y, x - y, -x + y, -x - y\}$
- 所以可以分 $x + y$, $x - y$, $-x + y$, $-x - y$ 四个量来考虑。

# 解答

- 考虑二元组 (F*k*, d)。其中 d 代表方向，一共四个值。
  - 从 (F1, 0) 开始，遇到函数调用就递归计算以下的值：
    - 函数执行完毕后的变化量 (dx, dy)
    - 函数执行完毕后面向的方向 d'
    - 函数执行过程中 x + y, x - y, -x + y, -x - y 的最大值
  - 调用如果返回了，就可以更新当前位置、当前方向以及过程中四个量的最大值等信息了。
  - 采用记忆化。

# 讨论

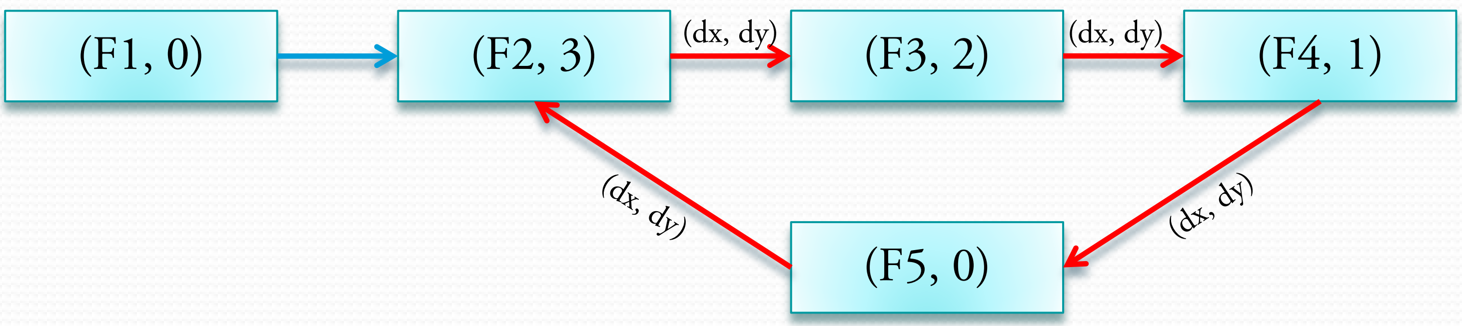
- 如何处理无限循环？
  - 也就是某次函数调用需要使用栈中正在计算的 (F*k*, d) 的信息。



11J - Journey

# 解答

- 计算循环中所有的 (dx, dy) 的和。
  - 如果为 (0, 0)，那么不会到达任何新点。运算结束。
  - 否则，机器人可以到达无限远。



# 解答

- 本题的最后一个难点是：高精度整数。
- 在现场只有 ITMO 队解出。

~~（不过 OI 现役集训队应该能轻松切掉吧 233）~~

11L - Lanes

# 题目大意

- 某座大桥上有 $n_1$ 条从左至右的车道，$n_2$ $(n_1, n_2 \le 10)$ 条从右至左的车道，还有一条潮汐车道。
- 在早上从左向右行驶的车辆较多，潮汐车道的方向是从左向右；晚上正好相反。
  - 也就是说早上有 $n_1 + 1$ 条左右车道，$n_2$ 条右左车道；晚上有 $n_1$ 条左右车道，$n_2 + 1$ 条右左车道。

11L - Lanes

# 题目大意

- 交通是很复杂的问题，我们有如下的抽象模型：
  - 从早到晚被分为 m ($m \le 100\,000$) 个离散的时刻，编号从 1 开始。
  - 在每个时刻，两侧均有一些车抵达。
  - 设左右车道和右左车道各有 L, R 条车道，那么左侧通行 L 辆车（即减少 L 辆车），右侧通行 R 辆车。
  - 剩下的车等待下一时刻。
- 如果在时刻 m 后仍有车辆在等待，那么后面的时刻仍按同样模型进行，只是不会有新到达的车辆了。

# 题目大意

- 你的任务是决定在哪一时刻改变潮汐车道的方向，能使所有车辆等待时间的总和最小化。
- 不过潮汐车道也不是瞬间就能改变方向的，它需要 r 个时刻来改变方向。
  - 也就是说，如果在 t 时刻改变方向，那么在 [t, t + r - 1] 时刻内潮汐车道不能使用，即左右和右左车道各有 $n_1$ 和 $n_2$ 条。

## 11L - Lanes

# 讨论

- 题面好长……
- 问题好乱……

# 解答

- 解决问题的钥匙就是发现，左右与右左实际上是两个不相关的问题。
  - 模型上两侧的到达和等待是不相关的。
  - 从某一时刻开始增加/减少了一条车道，两侧可以分别处理。
- 所以我们把问题分成左右和右左两个问题，最后再进行合并。

# 解答

- 某时刻的车道增加一条是不容易处理的。
- 不过减少一条车道就简单多了。若从 k 条车道减少到 k - 1 条车道:
    - 原本至多 k - 1 辆车在等待，什么都不做。
    - 否则，有一辆车不能在这个时刻通过。需要找到后面最早的有空余车道的时刻让其通过，然后更新总等待时间。

11L - Lanes

# 解答

- 处理左右的问题：
  - 由于减少一条车道相对简单，考虑改变潮汐车道方向的时刻*从晚至早*。
    - 这样计算的时候只需从后向前分别减少 1 条车道。
- 处理右左的问题：
  - 类似地，*从早至晚*考虑。

# 解答

- 如何找到最早的有空余车道的时刻？
  - 对于 1 至 m 之内的时刻，维护一张顺序表记录有空余车道的时刻。
  - 对于 m 以后的时刻，不会有新车辆驶来了，记录最晚的有车辆开走的时刻。如果前面的顺序表没有空余，只能将车辆放在最后开走了。
  - 时间复杂度每次 O(1)。

# 解答

- 至此，对于任一时刻，我们计算出了在此时刻改变车道方向，两侧的总等待时间。
- 合并无比简单：枚举改变时间，使得两侧总等待时间之和最小。
- 至此本问题得到了解决。

# 解答

- 本题现场只有 Moscow State University 1 队在比赛结束前 5 分钟解决。
- 题目理解困难、题意复杂是解决本题的障碍。
- 但是一旦混在一起的问题得到分离，解决就不难了。

12D - Disjoint Regular Expressions

# 题目大意

- 给出两个长度在 100 以内的正则表达式。
  - 包含选择、串接和 Kleene 星运算。
- 求是否存在一个非空字符串同时匹配两个正则表达式。如果有的话求最短的。

# 讨论

- 在昨天下午乔明达大神的课程中，大家已经学习了如何将以上的正则表达式转换为 NFA。
- 于是接下来的问题在 NFA 上做即可。

# 解答

- 首先用 Thompson's construction algorithm 将正则表达式转化为 NFA-ε。
- 问题转化成了，寻求一种输入字符的序列，使得两台自动机可以同时从初始状态转移到接受状态。
- 用 $q$ 表示初始状态，$f$ 表示接受状态。

# 解答

- BFS，用二元组 (α, β) 表示状态。
  - α 表示在第一台 NFA 上目前的状态，β 表示第二台。
- 两个状态同时转移。
  - 二者可以分别进行某个 ε 转移（或者认为自己转移到自己也是一种 ε 转移）。
  - 消耗输入字符的转移，二者必须同时进行。
- 如果同时转移到了接受状态，即认为找到解。
  - 即 $(q_1, q_2) \to (f_1, f_2)$。

# 解答

- ……题目要求非空串，因此以上的做法是有 Trick 不易处理的。
- 考虑

(ab)* 和 b*a*

两个正则表达式。

- 按照以上的做法会得出一个空串满足两个正则表达式。

# 解答

- 消除 Trick 很简单。
  - 改用三元组 $(\alpha, \beta, x)$ 描述状态。
    - 其中 $x$ 是一个布尔值，表示转移过程中是否有非 ε 转移。
  - 如果从 $(q_1, q_2, \mathit{false})$ 转移到了 $(f_1, f_2, \mathit{true})$，则找到了非空解。

# 题目大意

- Peter 在一个秘密化学实验室工作。他的一个新实验需要称重 $x$ ($x \le 10^{18}$) 纳克的试剂。
- 称重需要使用砝码，而砝码是装在 $n$ ($n \le 10^5$) 个被封起来的箱子的。第 $i$ 个箱子中装有 $q_i$ 个砝码，而它们的质量都是 $10^{k_i}$ ($q_i \cdot 10^{k_i} \le 10^{18}$) 纳克。
- 求至少需要开多少个箱子，才能称出 $x$ 纳克的质量。并给出一种开箱方案。

# 讨论

- 数据范围相当大……
- 考虑砝码质量都是 10 的幂这种特殊性。
- ……总之本题是个水题。神犇快来秒！

# 解答

- 先考虑 $x$ 的个位。
  - 我们发现只有质量是 $10^0$ 的砝码才能称出个位。
  - 因此，考虑所有质量是 $10^0$ 的砝码，若它们的总和不到 $x$ 的个位，那么称重是不可行的。
  - 否则，从装有最多砝码的箱子开始开，直到凑足 $x$ 的个位为止。

# 解答

- 在 $x$ 的个位处理好之后，就要处理 $x$ 的十位了。
- 这时可以发现：$10^0$ 的砝码和 $10^1$ 的砝码没有区别了。
  - 个位已经搞定了。
  - 处理十位时每 10 个 $10^0$ 的砝码等于一个 $10^1$ 的砝码。
- 凑足个位后，之前剩余的 $10^0$ 的砝码亦是如此。
- 考虑所有未开的 $10^1$ 的箱子和 $10^0$ 的箱子，从质量最大的箱子开始开，直到凑足十位为止。

# 解答

- 于是就能看出这题如何做了。
  - 从低位向上考虑。考虑到 $10^k$ 这一位的时候，就将 $10^k$ 的箱子加入待开箱子的集合，从质量大的开始开，直到凑足 $10^k$ 这一位。
- 本质上是贪心解法。正确性从考虑的过程可以即证。

12I - Identification of Protein

# 题目大意

- 蛋白质可以看做一个氨基酸序列。问题中对蛋白质做了如下规定：
  - 蛋白质最多由 400 个氨基酸组成。
  - 氨基酸只有 P 和 Q 两种。
  - P 的分子质量是 97.05276 Da，Q 的分子质量是 128.05858 Da。
- 现在通过测量，得到了n ($n \le 100\ 000$) 个*峰*的质量。所谓峰，就是指蛋白质的前缀或者后缀。

# 题目大意

- 由于测量手段的局限，测得的数值可能有如下问题：
  - 一些前缀/后缀没有被测量到。
  - 一些峰的质量不属于任一前缀/后缀（称为噪声）。
- 现在有如下约定：
  - 所有的峰的质量都是正数。
  - 最大的峰的质量一定是整个蛋白质的质量。
- 现在请你构建出一个蛋白质序列，使得最多数目的前/后缀可以与某个峰的质量匹配。注意一个峰的质量只能与一个前/后缀匹配，反之亦然。

# 简单转化

- $(9705276, 12805858) = 2$.
- 所以 $97.05276x + 128.05858y = m$ 在 $0 \le x + y \le 400$ 中最多只有一个解。
- 所以可以试图将峰表示成 $(x, y)$ 的形式，不能做这种表示的峰一定是噪声。
- 质量最大的峰代表整个蛋白质。所以蛋白质中 P 和 Q 的数量已经知道了。不妨设有 $p$ 和 $q$ 个。

# 讨论

- 采用什么算法解决这个问题?

# 讨论

- 凭着多年的 OI 经验，同学们果断看出本题应该使用 DP 处理。
- 将峰 $(x, y)$ 在矩阵 $A$ 中统计个数，方便 DP 处理。
- 最简单的想法是将前缀作为状态：$f(i, j)$ 表示长度为 $i$ 的前缀中有 $j$ 个 P 的情况，最多可以和多少个峰匹配。

# 讨论

- 一个有 $x$ 个 P 和 $y$ 个 Q 的前缀*暗示*一个有 $p-x$ 个 P 和 $q-y$ 个 Q 的后缀。
- $f(i, j)$ 对应有 $j$ 个 P 和 $i-j$ 个 Q 的情况。可以列出如下的 DP 方程:

$$f(i, j) = \max\{f(i-1, j), f(i, j-1)\}$$

$$+ (A[j][i-j]\ ?\ 1 : 0) + (A[p-j][q-i+j]\ ?\ 1 : 0)$$

# 讨论

- 但是这种做法是有问题的！

（否则标题会是“解答”。）

# 解答

- 前一页做法的问题在于，统计前后缀匹配的峰的时候会有重复。
- 例：

A[2][1] = 1（即只有一个峰的质量对应 $P^2Q^1$）

DP 过程中 PPQPP 会出现两次和这个峰的匹配。

PPQPP

PPQPP

# 解答

- 分析会发现，冲突只会发生在状态*表示*的前缀，和状态*暗示*的后缀之间。
- 因此我们对这个做法做如下改造：前缀和后缀同时转移，每次在前缀和后缀中各添上一个字母。

PQQPPQP

PQQPPQP

PQQPPQP

- 这样状态*表示*的峰的长度都在 $\left\lfloor\frac{p+q}{2}\right\rfloor$ 之内，就不会冲突了。

# 解答

- 状态用 $f(i, j, k)$ 表示，其中 $i$ 表示前缀和后缀的长度，$j$ 表示前缀中 P 的数目，$k$ 表示后缀中 P 的数目。
- $f(i, j, k)$ 可由以下四个状态转移而来：
  - $f(i-1, j-1, k-1)$: P…P
  - $f(i-1, j-1, k)$: P…Q
  - $f(i-1, j, k-1)$: Q…P
  - $f(i-1, j, k)$: Q…Q
- 统计和前面是类似的。

# 解答

- 注意以下两个特殊情况：
  - $j=k$：前后缀中的 P 和 Q 数目相等。
    - 前后缀和暗示的前后缀最多统计两次。
  - $j+k=p$ 且 $i=\frac{p+q}{2}$：前后缀加起来正好是整个蛋白质。
    - 不要统计暗示的前后缀。
- 注意蛋白质总长是奇数的情况，中间氨基酸枚举处理。

# 解答

- 最大匹配个数是 $f\left(\left\lfloor\frac{p+q}{2}\right\rfloor, j, p-j\right)$（$p+q$ 是偶数）或 $\max\left\{f\left(\left\lfloor\frac{p+q}{2}\right\rfloor, j, p-1-j\right), f\left(\left\lfloor\frac{p+q}{2}\right\rfloor, j, p-j\right)\right\}$（$p+q$ 是奇数）。
  - 奇数的情形下，-1 对应中间氨基酸是 P，否则对应中间氨基酸是 Q。
- 之后由 DP 的状态转移构造答案。

# 题目大意

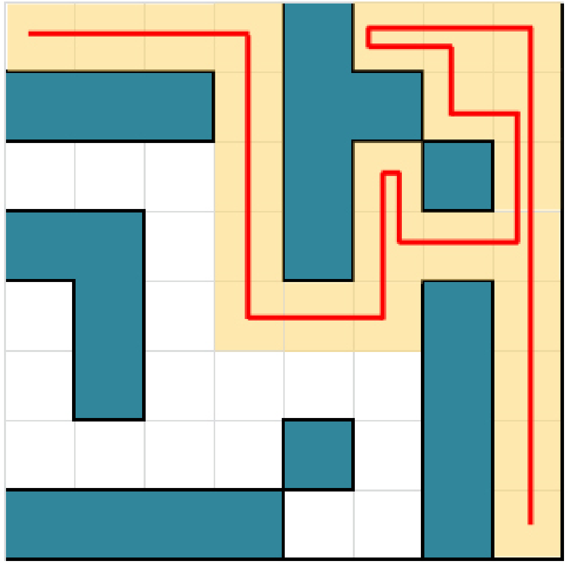
- 有一个四连通的 $h$ 行 $w$ 列 ($h, w \le 1\,500$) 的迷宫，有些格子里有障碍物。左上角的格子记作 (1, 1)，右下角的格子记作 $(h, w)$，这两个格子内都没有障碍物。
- 求一个最小的不包含 (1, 1) 和 $(h, w)$ 的正方形区域，使得这个区域中没有障碍物，并且在区域中填充障碍物之后，(1, 1) 和 $(h, w)$ 就不连通了。

# 讨论

- 如何才能阻断 $(1, 1)$ 与 $(h, w)$ 之间的连通呢？

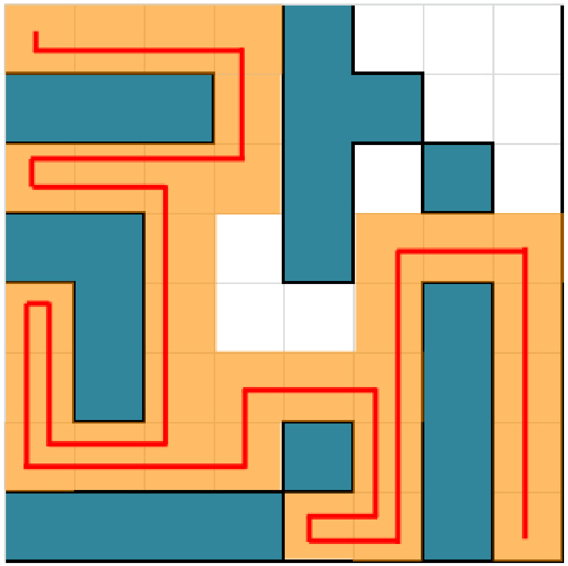
# 解答

- 所谓*左手路径*，就是从 (1, 1) 点开始向右，左手一直扶在墙上走出的路径。



# 解答

- *右手路径*呢，就是就是从 (1, 1) 点开始向下，右手一直扶在墙上走出的路径。



# 解答

- 如果一个连通的障碍，阻断了 (1, 1) 至 $(h, w)$ 之间的连通，这等价于它同时阻断了原来的左手和右手路径。
    - 显然，左手路径包围障碍物在右上部分的连通块，右手路径包围障碍物在左下部分的连通块。如果同时阻断这两者，相当于将两部分连通块连在一起，当然也就没有 (1, 1) 至 $(h, w)$ 之间的通路了。

# 解答

- 因此做法很简单，枚举正方形的左上角，二分正方形的长度，使得在其中没有障碍物，并且包含左手路径和右手路径经过的格子。

13C - Cactus Automorphisms

# 题目大意

- 给定一个 n (n ≤ 50 000) 个节点的仙人掌。
  - 仙人掌是指每条边最多在一个环中的无向图。
- 求仙人掌有多少种自同构。
  - 自同构指的是图的顶点集合 $V$ 到 $V$ 的变换 $m$，使得 $\forall v_1, v_2 \in V(\{v_1, v_2\} \in E \Rightarrow \{m(v_1), m(v_2)\} \in E)$。
- 以 $p_1^{a_1} p_2^{a_2} \dots p_k^{a_k}$ 的形式输出，其中 $p_k$ 是素数。

# 讨论

- 考虑仙人掌和树的相似性。

# 解答

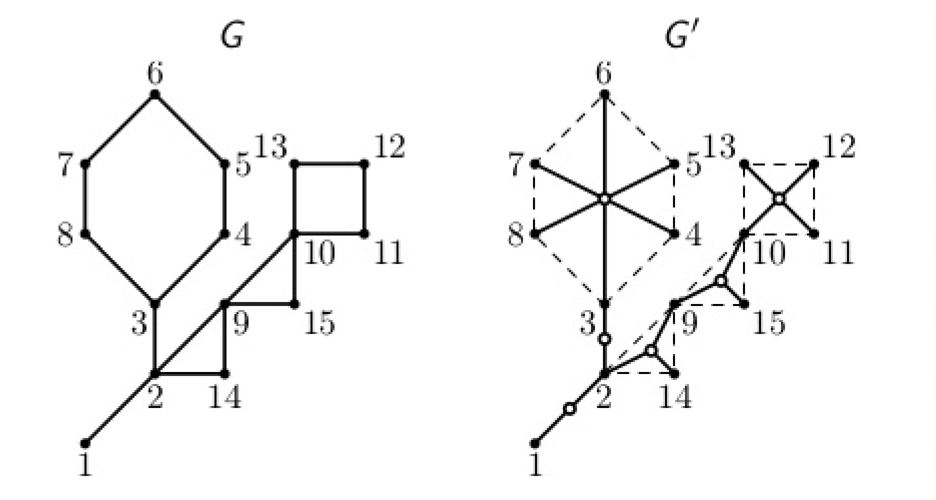
- 我们先研究树的自同构。考虑树的 *中心*。
    - 记 distance(v) 为树中距离 v 最大的点与 v 的距离，那么树的中心就是使得 distance(v) 最小的点 v。
- 树的中心可能不唯一。
    - 至多两个，这时代表树的中心是一条边。
- 为了消除这种情况，可以在树的每条边中间加入一个点。这时树的直径一定是偶数，中心是唯一的。

# 解答

- 显然，对于树的任何自同构，树的中心都是不动点。
  - 如果不做上一页的处理，树的两个中心可以映射到对方。
- 接下来考虑中心的子树。
  - 每有 $k$ 棵同构的子树，它们之间可以互相映射，答案要乘以 $k!$。
- 判断子树同构可以自底向上计算 Hash 值。
- 递归处理即可。

# 解答

- 接下来考虑如何将仙人掌转化成树。
- 仿照在边中加入节点的方法，对于仙人掌的每个圈，加入一个虚拟节点，并将它与圈中所有节点连边。



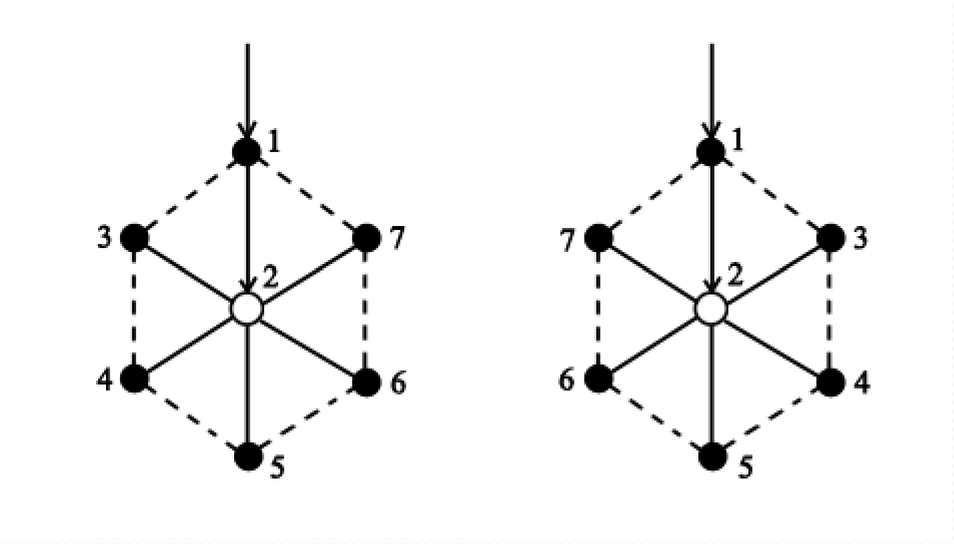
13C - Cactus Automorphisms

# 解答

- 现在仙人掌被转化成了树。它的直径仍是偶数，所以中心唯一。
  - 它的中心可能代表原来仙人掌的节点、边或者环。
- 对于代表环的节点——它的子树之间*不能*任意映射！

# 解答

- 如果非中心点代表环，那么除了该点的父节点，如果剩下的点可以翻转，答案乘 2，否则乘 1。



# 解答

- 如果中心点（即根）代表环，那么环中的节点可以旋转或镜像同构。答案应乘以这样的同构数。
- 由于环中可能有很多节点，需要使用 KMP 算法等进行匹配。

13D - Dictionary

# 题目大意

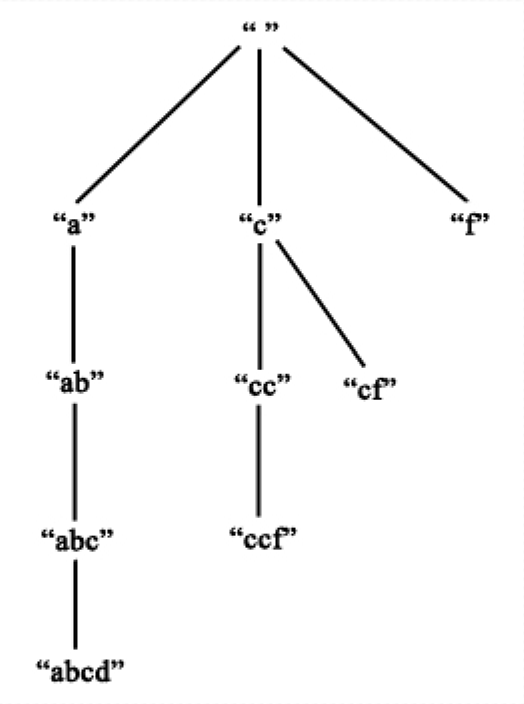
- 给定不超过 50 个单词，每个单词长度在 10 以内。
- 要求构造一棵节点数最少的树，使得：
  - 树的边带有字母。
  - 所有给定的单词都可以由树的一条自上而下的路径表出。

# 讨论

- 如何转化此题给定的模型？

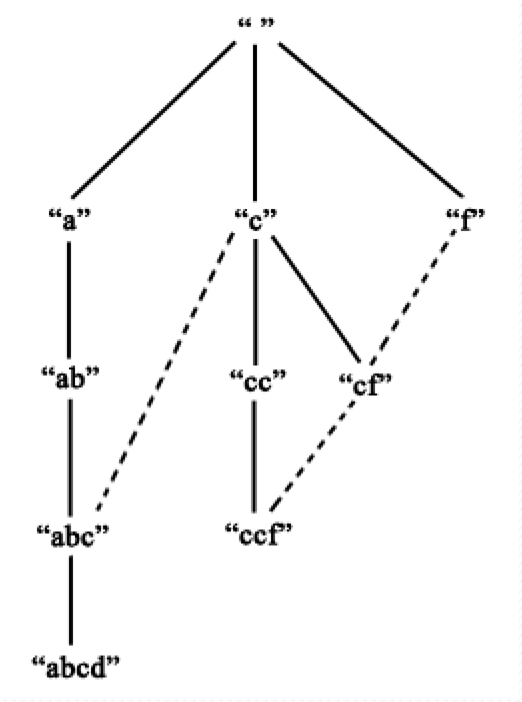
# 解答

- 如果不计节点数目，最简单的解决方案是 Trie。



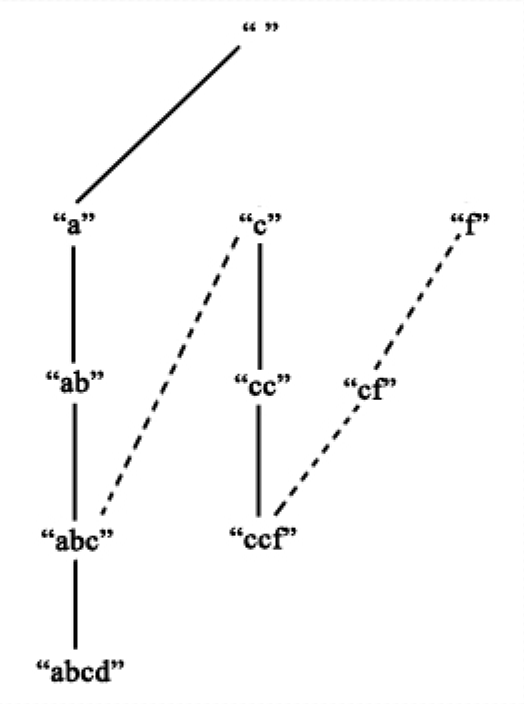
# 解答

- 但是 Trie 很浪费，因为有些节点是另外节点的后缀!



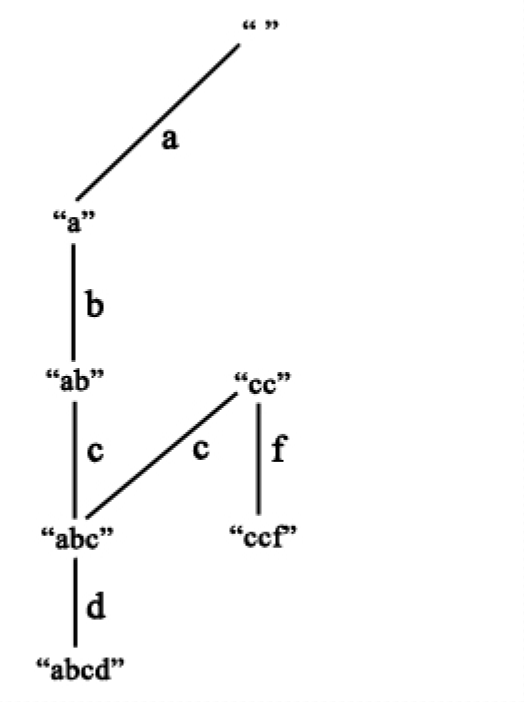
# 解答

- 一个节点可以直接包含它的后缀。



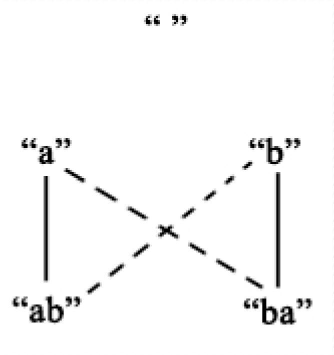
# 解答

- 从而这些节点是不需要的，这样就减少了节点的数目。



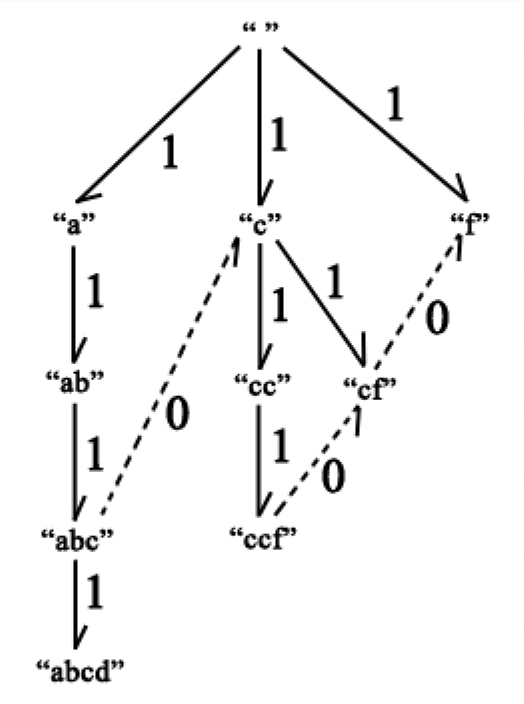
# 解答

- 不过也不能直接取所有的后缀转移，否则可能会有环。



# 解答

- 经过进一步的分析，我们发现就是求下面这张图的最小树形图：



# 解答

- 于是本题解法如下：
  - 建立一个表示空字符串的节点。
  - 对于字符串集合中的每个字符串，将其所有前缀建立一个节点表示，并建立权值为 1 的有向边连接它们（Trie转移）。
  - 若某个前缀是另一前缀的后缀，建立权值为 0 的有向边连接它们（后缀转移）。
  - 求本图从空出发的最小树形图。
  - 根据最小树形图建树即可。

# 题目大意

- 在二维世界中，有片地方的地形是折线形的，并且顶点的横坐标严格单调上升，顶点不超过 10 000 个。
- 在这么一片地方要修建不超过 10 000 座太阳能塔。每座塔的塔高均已确定。
- 阳光的角度是 $\alpha$。
- 太阳能塔和地形之间会相互遮盖。求一个建设方案，使得每座塔被阳光照射到的长度总和最大。

# 讨论

- 本题难度不大，请神犇来秒！

# 解答

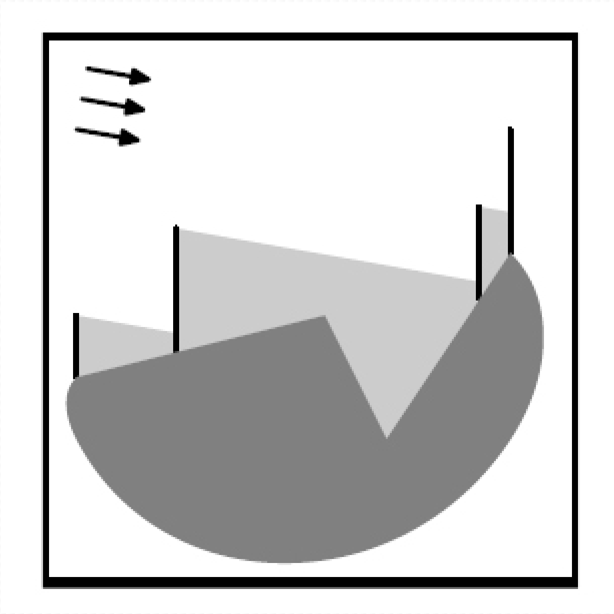
- 不太常见的几何技巧。做下面的变换：

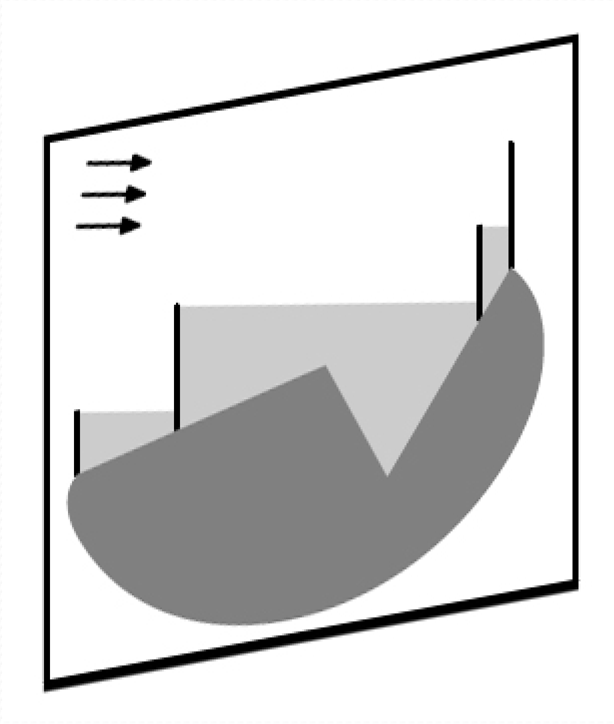
$$\begin{cases} x' = x \\ y' = y + x\tan\alpha \end{cases}$$

(另外其实不做这个变换本题也可做，只是略显复杂)

# 解答

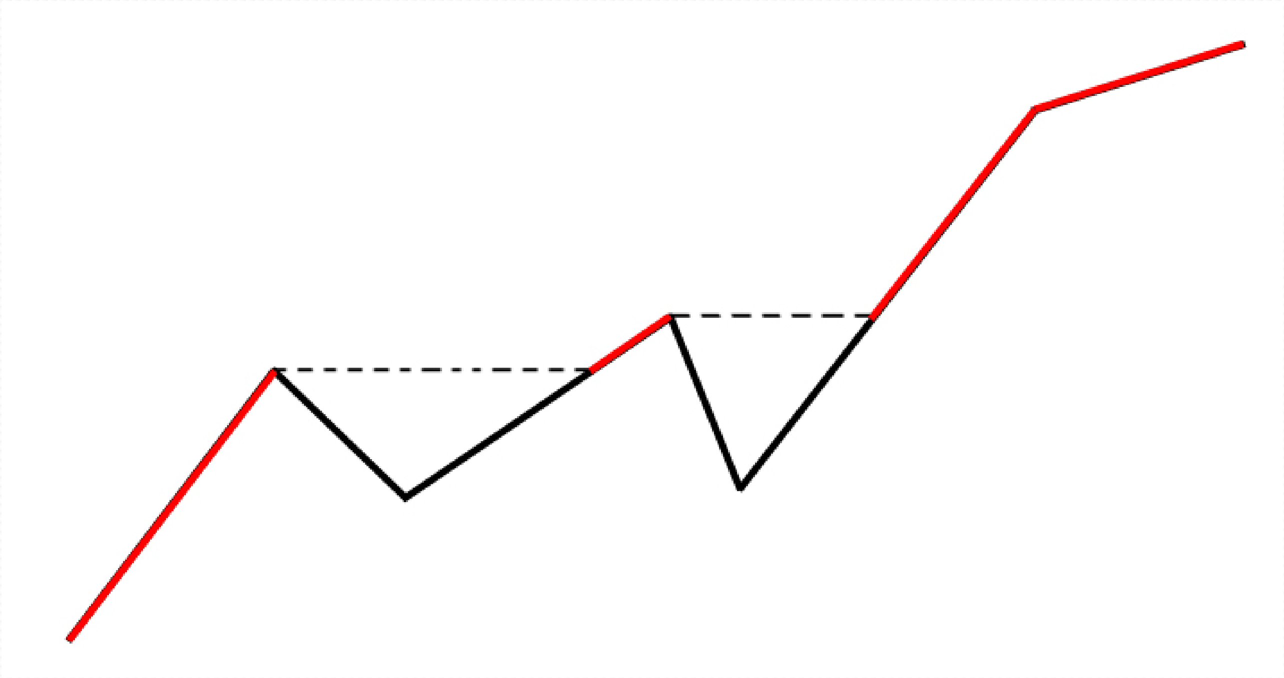
- ……就可以将阳光的角度修正为水平的!





# 解答

- 之后修整地形。在下图所示的坑中放塔是不划算的，也就是塔只放在红线的区域内。

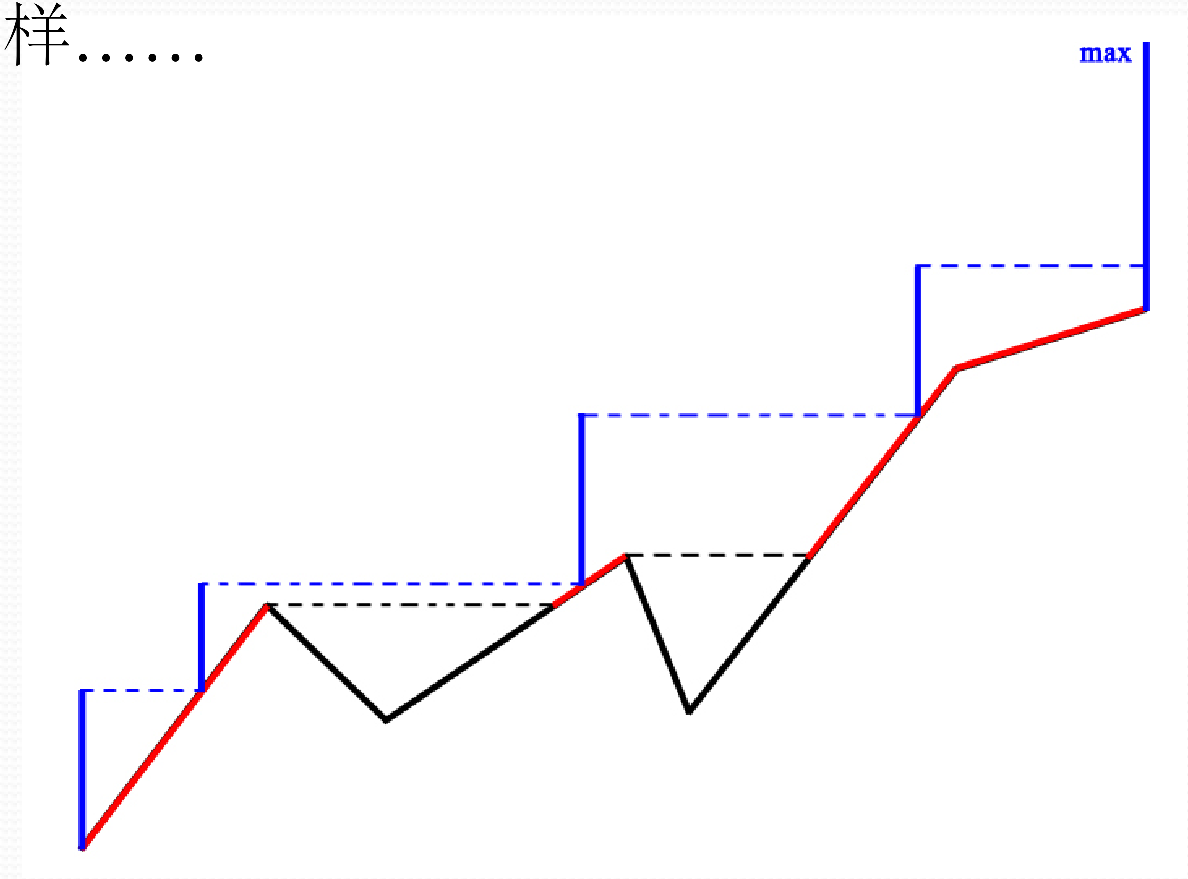


# 解答

- 之后先将最高的塔放在最右边。
- 然后从最左开始，依次将塔放在当前最左的不会被别的塔覆盖的地方。
- 如果覆盖满了，就将剩余的塔全扔在最后就行了。

# 解答

- 就像这样……



# 题目大意

- 给定一个长度在 100 000 以内的非负整数序列，序列中元素的大小在 $2^{31}$ - 1 以内。
- 求有多少个连续子列，满足子列中所有元素的 XOR 与 AND 相等。

# 讨论

- 比较简单的题都放在后面了……

# 解答

- 对于起点在某个固定位置的所有连续子序列而言，它们 AND 的值不会超过 31 个。
- 并且会随着终点向右移动单调下降。

111011→101011→100011→000011

# 解答

- 询问一段的 AND 值很简单。如果没有修改操作，用 ST 算法即可。
- 因此可以二分出 AND 值变化的边界。
- 之后，就是询问存在多少个起点固定，终点在某区间内的连续子序列，使得其 XOR 等于某定值了。
- 计算出 XOR 的前缀和，对于每个值建立一个连续序列，二分计数就行了。

# 扩展

- AND 满足如下几个性质：
  - 区间可加性：

    $\mathrm{AND}\{[l,r]\} = \mathrm{AND}\{[l,m]\}\ \&\ \mathrm{AND}\{[m+1,r]\}$
  - 单调性：$[l_1,r_1] \subseteq [l_2,r_2] \Rightarrow \mathrm{AND}\{[l_1,r_1]\} \geq \mathrm{AND}\{[l_2,r_2]\}$
  - 可取的值不多。
- 满足这些性质的运算还有 OR，GCD 等。他们都可以用类似的方法处理。

# 扩展

- CERC ‘13 C - Magical GCD
  - 长度在 100 000 以内的正整数序列，大小不超过 $10^{12}$。求一个连续子序列，使得在所有的连续子序列中，它们的 GCD 值乘以它们的长度最大。
  - $10^{12} < 2^{40}$。因此若起点确定，GCD 值不超过 40 个。
  - 仿照上题，二分 GCD 值变化的边界即可。

CERC 13A - Rubik's Rectangle

# 题目大意

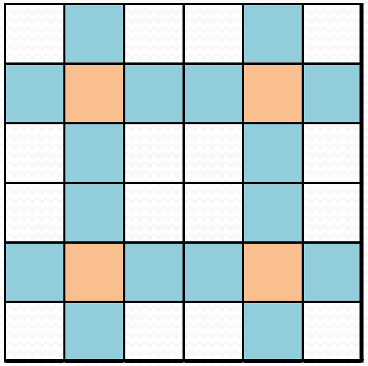
(如果进行到这个题了，说明我题目准备少了……QAQ)

- 有一个 $H \times W$ $(H, W \leq 100)$ 的矩阵，矩阵中是 1 至 HW 的一个排列。
- 每次可以进行的操作是：
  - 将一行中的数翻转。
  - 将一列中的数翻转。
- 构造一个 10HW 次操作以内的方案，将矩阵操作成顺序的，或者断言不可能。

CERC 13A - Rubik's Rectangle

# 解答

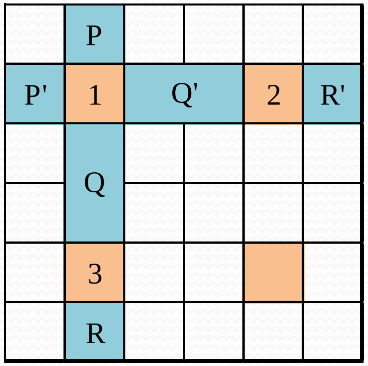
- 处在对称位置的两行及两列会确定一个四元组。
- 无论什么操作都不能将数字移出四元组。



CERC 13A - Rubik's Rectangle

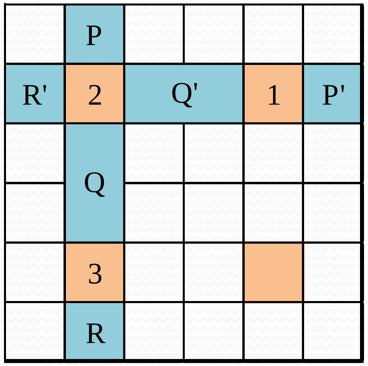
# 解答

- 可以构造出下面的一种操作序列：



# 解答

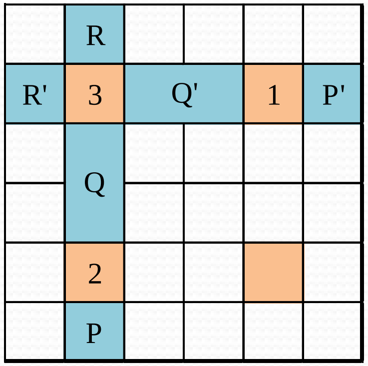
- 可以构造出下面的一种操作序列：
  - 翻转行。



## CERC 13A - Rubik’s Rectangle

# 解答

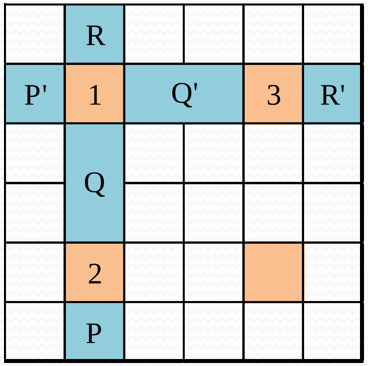
- 可以构造出下面的一种操作序列：
  - 翻转行。
  - 翻转列。



CERC 13A - Rubik's Rectangle

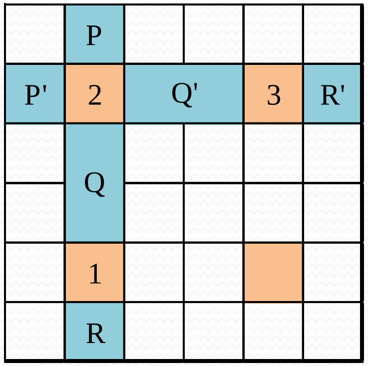
# 解答

- 可以构造出下面的一种操作序列：
  - 翻转行。
  - 翻转列。
  - 翻转行。



# 解答

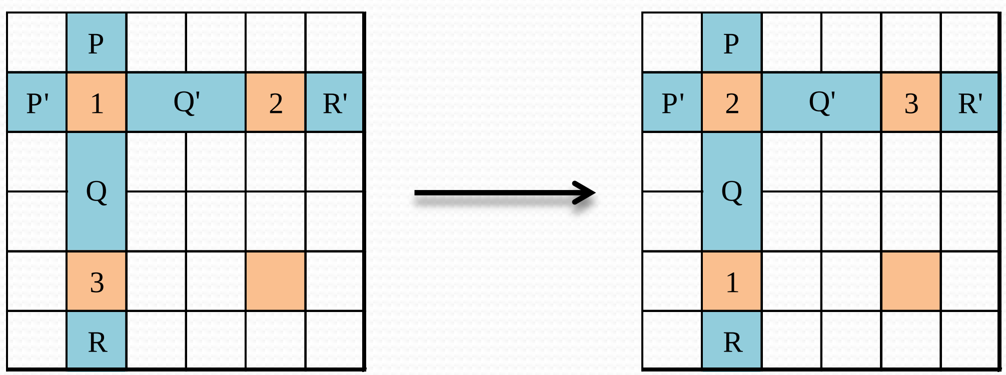
- 可以构造出下面的一种操作序列：
  - 翻转行。
  - 翻转列。
  - 翻转行。
  - 翻转列。



CERC 13A - Rubik’s Rectangle

# 解答

- 通过这个操作序列，可以将四元组中的三个数旋转，而不影响其他位置上的数。
- 也即是说，任何一个四元组的*偶排列*都能够调整得到。



# 解答

- 奇排列如何处理？
  - 将四元组的奇偶性记为 Q(i, j)，即(i, j)，(H - 1 - i, j), (i, W - 1 - j), (H - 1 - i, W - 1 - j) 这四个位置上的数的奇偶性。
    - 其中 $i < \left\lceil \frac{H}{2} \right\rceil, j < \left\lceil \frac{W}{2} \right\rceil$。
  - Q(i, j) = 0 即为偶排列。
  - 翻转第 i 行，会改变所有 Q(i, k) $(0 \le k < \left\lceil \frac{W}{2} \right\rceil)$ 的奇偶性。
  - 翻转第 j 列类似。

CERC 13A - Rubik's Rectangle

# 解答

- 用 R(i) = 1 表示第 i 行翻转，否则 R(i) = 0。
- 列是否翻转用 C(j) 表示。
- 也就是令

$$Q(i,j) \oplus R(i) \oplus C(j) = 0$$

可以用 Gauss 消元解。

# 解答

- 另外，由于应有

$$Q(i,j) \oplus Q(0,j) \oplus Q(i,0) \oplus Q(0,0) = 0$$

也可以贪心地将第 0 行和第 0 列的四元组先做完，然后如果仍有某个 Q(i, j) = 1，就断言无解。

- 时间复杂度 O(HW)。
- 所需的次数远远小于 10HW。

SWERC 11B - Coin Collecting

# 题目大意

(如果进行到这个题了，说明我题目准备得太少了……QAQ)

- 你是一个硬币收藏家，你收集了很多国家的硬币。
- 你的朋友也是一个硬币收藏家，然后她看上了你的收藏，打算和你玩一个游戏。
- 赢家可以拿走输家的全部或部分收藏。虽然你有多年辛苦毁于一旦的风险，但是你也看上了她的收藏，所以打算玩一玩。

SWERC 11B - Coin Collecting

# 题目大意

- 她准备了 r ($r \le 300$) 对信封，每个信封里面有两个不同国家的硬币。
- 你可以在每对信封中选择一个或者不选。
- 当你选择结束后，如果她可以选出你选的信封的一个非空子集，使得所有国家的硬币都出现偶数次，那么她赢得游戏，拿走你的全部收藏。
- 否则你获胜，拿走你选的所有信封中的硬币。
- 请问你最多可以拿走多少枚硬币。

SWERC 11B - Coin Collecting

# 解答

- 一个很容易想到的想法：
  - 将每个信封 (a, b) 表为无向边 (a, b)。
  - 一个非空信封集合，使得所有国家的硬币出现偶数次，即为一个*环*。
- 于是做如此转化。
  - 本题转化为：给定 r 对无向边，每对中最多选出一条，使得构成的图中没有环。你需要使边集大小最大。

SWERC 11B - Coin Collecting

# 讨论

- 然后这个题目怎么处理呢？
- 提示：
  - 考虑每次加入新的一对边。
  - 然后修正前面的选择。

# 解答

- 考虑新的一对边 $(e_1, e_2)$。原来选择的边构成边集 E。
  - 由于 E 无环，所以 E 是树或者森林。
  - 如果 $E + e_1$ 或者 $E + e_2$ 无环，将其加入 E 即可。
  - 否则，记 $C(E + e)$ 为加入 e 后形成的唯一的环。如果存在 $f \in C(E + e)$，使得与 f 属于同一边对的另一边 f’ 满足 $E + e - f + f'$ 无环，那么 E 的大小就增加了。
  - 否则，继续考虑 $C(E + e - f + f')$……

# 解答

- 最后会形成类似于 $E + e - f_1 + f_1' - f_2 + f_2' - \ldots + f_n'$ 的图。这很类似于二分图最大匹配中的***增广路***。
- 我们的任务即为，在一个新的边对 $(e_1, e_2)$ 进入之后，寻找图中的增广路。

# 解答

- 加边 / 删边的操作会改变图。然而直接迭代搜索的话太慢了。
- 考虑不改变这张图，也就是只检查 E + $f_i$’ 中的环。
  - 情况变得相当有限，搜索的代价降低了。
  - 不过会出现很多问题！

# 解答

- 如果 $C(E + f_i')$ 中包含已经删除过的边 d……
  - E - d 会有两个连通块。在当前实际的图 E + e - f + f' - … 中，考虑连接这两个连通块的路径上的某边 g，也可以删除 g 来消除环。
    - 不过，这可以在删除 d 的时候进行。容易发现，如果删除 g 会产生增广路，在删除 d 的时候改删除 g 也是可以的。
  - d 可能会被删除两次。
    - 这是不合法的。后面的搜索过程会避免这一情况。

SWERC 11B - Coin Collecting

# 解答

- 由此，寻求增广路的过程可以归纳为下：
  - 对于每条边 e（新加入的和之前边对中的），将其拆为 +e 和 -e 两点。
  - -e 与 +e’ 相连。其中 e’ 是 e 处于同一边对中的边。
  - +e 与 -f 相连，其中 $f \in C(E + e)$。
  - 源点：$+e_1$ 和 $+e_2$
  - 汇点：所有出度为 0 的点。
  - 寻找从源点到汇点的路径即可。

# 解答

- 如何避免删除同一条边两次?
  - 亦即两次经过同一点 -d。
  - 很简单，只需要寻找从源点到汇点的最短路即可。

# 解答

- 时间复杂度是 O($r^3$)。
- 现场没有队伍解出该题。在解题报告上被标为

***Difficulty*: very hard (I mean, impossible)**

# Thank you!