

数学题选讲

唐靖哲

北京航空航天大学计算机学院

2017 年 3 月 11 日

■ 为什么要讲数学题？

~~讲道理 51nod 到处都是数学题~~



写在前面

- 为什么要讲数学题？

讲道理 ~~51nod~~ 到处都是数学题

门槛低，初中生也可以听懂

- 这堂课像萌新 / 大佬这种人能听吗？

玄不救非，氪不改命



写在前面

- 为什么要讲数学题？

讲道理 ~~51nod~~ 到处都是数学题

门槛低，初中生也可以听懂

- 这堂课像萌新 / 大佬这种人能听吗？

~~玄不救非，氪不改命~~

难度已提前标明，我会尽量讲清楚，希望所有人能有启发

- 预习内容太难了怎么办？

我能怎么办呀？我也很绝望啊！—



■ 为什么要讲数学题？

讲道理 ~~51nod~~ 到处都是数学题

门槛低，初中生也可以听懂

■ 这堂课像萌新 / 大佬这种人能听吗？

~~玄不救非，氪不改命~~

难度已提前标明，我会尽量讲清楚，希望所有人能有启发

■ 预习内容太难了怎么办？

我能怎么办呀？~~我也很绝望啊！~~

预习内容会不加证明地 review，以便于直接使用

■ 我还想到一个有趣的问题，可惜这里空白太小，写不下

整体内容

- 质数筛法 versus 启发式分解
- 离散对数与原根
- 容斥原理与二项式系数
- ~~互动交流~~



质数筛法 VS. 启发式分解

■ 操作原理

- 根据题目特殊性，通过简单的试除法取代复杂的启发式分解

■ 适用场景

- 启发式分解真的很慢，Pollard's rho 分解 v 期望要 $O(v^{\frac{1}{3}} \log v)$
- 写出启发式分解比较困难或调试代价较高
- 怕随机化出现不存在的问题
- 喜欢分类讨论 渴求确定性算法

质数筛法 VS. 启发式分解

■ 操作原理

- 根据题目特殊性，通过简单的试除法取代复杂的启发式分解

■ 适用场景

- 启发式分解真的很慢，Pollard's rho 分解 v 期望要 $O(v^{\frac{1}{3}} \log v)$
- 写出启发式分解比较困难或调试代价较高
- 怕随机化出现不存在的问题
- 喜欢分类讨论 渴求确定性算法

质数筛法 VS. 启发式分解

■ 操作原理

- 根据题目特殊性，通过简单的试除法取代复杂的启发式分解

■ 适用场景

- 启发式分解真的很慢，Pollard's rho 分解 v 期望要 $\mathcal{O}(v^{\frac{1}{4}} \log v)$
- 写出启发式分解比较困难或调试代价较高
- 怕随机化出现不存在的问题
- 喜欢分类讨论 渴求确定性算法

质数筛法 VS. 启发式分解

■ 操作原理

- 根据题目特殊性，通过简单的试除法取代复杂的启发式分解

■ 适用场景

- 启发式分解真的很慢，Pollard's rho 分解 v 期望要 $\mathcal{O}(v^{\frac{1}{4}} \log v)$
- 写出启发式分解比较困难或调试代价较高
- 怕随机化出现不存在的问题
- 喜欢分类讨论 渴求确定性算法

质数筛法 VS. 启发式分解

■ 操作原理

- 根据题目特殊性，通过简单的试除法取代复杂的启发式分解

■ 适用场景

- 启发式分解真的很慢，Pollard's rho 分解 v 期望要 $\mathcal{O}(v^{\frac{1}{4}} \log v)$
- 写出启发式分解比较困难或调试代价较高
- 怕随机化出现不存在的问题
- 喜欢分类讨论 渴求确定性算法

质数筛法 VS. 启发式分解

■ 操作原理

- 根据题目特殊性，通过简单的试除法取代复杂的启发式分解

■ 适用场景

- 启发式分解真的很慢，Pollard's rho 分解 v 期望要 $\mathcal{O}(v^{\frac{1}{4}} \log v)$
- 写出启发式分解比较困难或调试代价较高
- 怕随机化出现不存在的问题
- 喜欢分类讨论 渴求确定性算法

质数筛法 VS. 启发式分解

■ 操作原理

- 根据题目特殊性，通过简单的试除法取代复杂的启发式分解

■ 适用场景

- 启发式分解真的很慢，Pollard's rho 分解 v 期望要 $\mathcal{O}(v^{\frac{1}{4}} \log v)$
- 写出启发式分解比较困难或调试代价较高
- 怕随机化出现不存在的问题
- 喜欢分类讨论 渴求确定性算法

- 有 n 个正整数 a_1, a_2, \dots, a_n 和 m 个正整数 b_1, b_2, \dots, b_m
- 定义分数 $Q = \frac{a_1 \cdot a_2 \cdots a_n}{b_1 \cdot b_2 \cdots b_m} = \frac{A}{B}$, 其中 A 与 B 互质
- 处理 k 个询问, 每个询问给出一个 M , 求一个整数 C 满足 $0 \leq C < M$ 且 $A \equiv BC \pmod{M}$, 即 $Q \equiv C \pmod{M}$, 不存在则输出 DIVISION BY ZERO
- $1 \leq n, m \leq 5000, 1 \leq k \leq 50, 2 \leq M \leq 10^{18},$
 $1 \leq a_i, b_j \leq 10^{18} \ (i = 1, 2, \dots, n, j = 1, 2, \dots, m)$

- 有 n 个正整数 a_1, a_2, \dots, a_n 和 m 个正整数 b_1, b_2, \dots, b_m
- 定义分数 $Q = \frac{a_1 \cdot a_2 \cdots a_n}{b_1 \cdot b_2 \cdots b_m} = \frac{A}{B}$, 其中 A 与 B 互质
- 处理 k 个询问, 每个询问给出一个 M , 求一个整数 C 满足 $0 \leq C < M$ 且 $A \equiv BC \pmod{M}$, 即 $Q \equiv C \pmod{M}$, 不存在则输出 DIVISION BY ZERO
- $1 \leq n, m \leq 5000, 1 \leq k \leq 50, 2 \leq M \leq 10^{18},$
 $1 \leq a_i, b_j \leq 10^{18} \ (i = 1, 2, \dots, n, j = 1, 2, \dots, m)$

- 有 n 个正整数 a_1, a_2, \dots, a_n 和 m 个正整数 b_1, b_2, \dots, b_m
- 定义分数 $Q = \frac{a_1 \cdot a_2 \cdots a_n}{b_1 \cdot b_2 \cdots b_m} = \frac{A}{B}$, 其中 A 与 B 互质
- 处理 k 个询问, 每个询问给出一个 M , 求一个整数 C 满足 $0 \leq C < M$ 且 $A \equiv BC \pmod{M}$, 即 $Q \equiv C \pmod{M}$, 不存在则输出 DIVISION BY ZERO
- $1 \leq n, m \leq 5000, 1 \leq k \leq 50, 2 \leq M \leq 10^{18},$
 $1 \leq a_i, b_j \leq 10^{18} \ (i = 1, 2, \dots, n, j = 1, 2, \dots, m)$

- 有 n 个正整数 a_1, a_2, \dots, a_n 和 m 个正整数 b_1, b_2, \dots, b_m
- 定义分数 $Q = \frac{a_1 \cdot a_2 \cdots a_n}{b_1 \cdot b_2 \cdots b_m} = \frac{A}{B}$, 其中 A 与 B 互质
- 处理 k 个询问, 每个询问给出一个 M , 求一个整数 C 满足 $0 \leq C < M$ 且 $A \equiv BC \pmod{M}$, 即 $Q \equiv C \pmod{M}$, 不存在则输出 DIVISION BY ZERO
- $1 \leq n, m \leq 5000, 1 \leq k \leq 50, 2 \leq M \leq 10^{18},$
 $1 \leq a_i, b_j \leq 10^{18} \ (i = 1, 2, \dots, n, j = 1, 2, \dots, m)$

■ Review

- 整除、因子、质数、最大公约数 (gcd)
- 互质：没有大于 1 的公共因子， $\gcd = 1$
- 唯一分解：任意正整数表示成质数幂次的乘积恰好有一种方法
- 质数筛法： $\mathcal{O}(n)$ 求出不超过 n 的所有质数
- 欧几里得算法： $\mathcal{O}(\log a + \log b)$ 求出 $\gcd(a, b)$
- 扩展欧几里得算法：在求出 $\gcd(a, b)$ 的同时求出 $sa + tb = \gcd(a, b)$ 的一组解 (s, t) ，并满足 $|s| + |t|$ 是所有解里最小的

■ Review

- 整除、因子、质数、最大公约数 (gcd)
- 互质：没有大于 1 的公共因子， $\gcd = 1$
- 唯一分解：任意正整数表示成质数幂次的乘积恰好有一种方法
- 质数筛法： $\mathcal{O}(n)$ 求出不超过 n 的所有质数
- 欧几里得算法： $\mathcal{O}(\log a + \log b)$ 求出 $\gcd(a, b)$
- 扩展欧几里得算法：在求出 $\gcd(a, b)$ 的同时求出 $sa + tb = \gcd(a, b)$ 的一组解 (s, t) ，并满足 $|s| + |t|$ 是所有解里最小的

■ Review

- 整除、因子、质数、最大公约数 (\gcd)
- 互质：没有大于 1 的公共因子， $\gcd = 1$
- 唯一分解：任意正整数表示成质数幂次的乘积恰好有一种方法
- 质数筛法： $\mathcal{O}(n)$ 求出不超过 n 的所有质数
- 欧几里得算法： $\mathcal{O}(\log a + \log b)$ 求出 $\gcd(a, b)$
- 扩展欧几里得算法：在求出 $\gcd(a, b)$ 的同时求出 $sa + tb = \gcd(a, b)$ 的一组解 (s, t) ，并满足 $|s| + |t|$ 是所有解里最小的

■ Review

- 整除、因子、质数、最大公约数 (\gcd)
- 互质：没有大于 1 的公共因子， $\gcd = 1$
- 唯一分解：任意正整数表示成质数幂次的乘积恰好有一种方法
- 质数筛法： $\mathcal{O}(n)$ 求出不超过 n 的所有质数
- 欧几里得算法： $\mathcal{O}(\log a + \log b)$ 求出 $\gcd(a, b)$
- 扩展欧几里得算法：在求出 $\gcd(a, b)$ 的同时求出 $sa + tb = \gcd(a, b)$ 的一组解 (s, t) ，并满足 $|s| + |t|$ 是所有解里最小的

■ Review

- 整除、因子、质数、最大公约数 (\gcd)
- 互质：没有大于 1 的公共因子， $\gcd = 1$
- 唯一分解：任意正整数表示成质数幂次的乘积恰好有一种方法
- 质数筛法： $\mathcal{O}(n)$ 求出不超过 n 的所有质数
- 欧几里得算法： $\mathcal{O}(\log a + \log b)$ 求出 $\gcd(a, b)$
- 扩展欧几里得算法：在求出 $\gcd(a, b)$ 的同时求出 $sa + tb = \gcd(a, b)$ 的一组解 (s, t) ，并满足 $|s| + |t|$ 是所有解里最小的

■ Review

- 整除、因子、质数、最大公约数 (\gcd)
- 互质：没有大于 1 的公共因子， $\gcd = 1$
- 唯一分解：任意正整数表示成质数幂次的乘积恰好有一种方法
- 质数筛法： $\mathcal{O}(n)$ 求出不超过 n 的所有质数
- 欧几里得算法： $\mathcal{O}(\log a + \log b)$ 求出 $\gcd(a, b)$
- 扩展欧几里得算法：在求出 $\gcd(a, b)$ 的同时求出 $sa + tb = \gcd(a, b)$ 的一组解 (s, t) ，并满足 $|s| + |t|$ 是所有解里最小的

■ 一个暴力的想法

- $A \equiv BC \pmod{M}$ 即 $sB + tM = A$ 有解的条件是 $\gcd(B, M) \mid A$
- 需要得到 A 和 B 的表示, 于是消除分子与分母的公因数
- 需要枚举 a'_i, b'_j 求出 $\gcd(a'_i, b'_j)$, 复杂度 $\mathcal{O}(nm \log v)$, 其中 v 表示数字最大值, 会超过时间限制

■ 一个暴力的想法

- $A \equiv BC \pmod{M}$ 即 $sB + tM = A$ 有解的条件是 $\gcd(B, M) \mid A$
- 需要得到 A 和 B 的表示，于是消除分子与分母的公因数
- 需要枚举 a'_i, b'_j 求出 $\gcd(a'_i, b'_j)$ ，复杂度 $\mathcal{O}(nm \log v)$ ，其中 v 表示数字最大值，会超过时间限制

■ 一个暴力的想法

- $A \equiv BC \pmod{M}$ 即 $sB + tM = A$ 有解的条件是 $\gcd(B, M) \mid A$
- 需要得到 A 和 B 的表示，于是消除分子与分母的公因数
- 需要枚举 a'_i, b'_j 求出 $\gcd(a'_i, b'_j)$ ，复杂度 $\mathcal{O}(nm \log v)$ ，其中 v 表示数字最大值，会超过时间限制

■ 一个不那么暴力的想法

- 将所有数利用启发式分解算法转换成质数幂次乘积的形式

- 消公因子转化为是幂指数的加减，复杂度

$\mathcal{O}((n + m + k)D(v) + (n + m)k \log v)$ ，其中 $D(v)$ 表示将 v 启发式分解的复杂度

- 大佬 10 分钟就写完并通过了

萌新还在抄模板、调试、思考人生

■ 一个不那么暴力的想法

- 将所有数利用启发式分解算法转换成质数幂次乘积的形式
- 消公因子转化为是幂指数的加减，复杂度

$\mathcal{O}((n + m + k)D(v) + (n + m)k \log v)$ ，其中 $D(v)$ 表示将 v 启发式分解的复杂度


- 大佬 10 分钟就写完并通过了


萌新还在抄模板、调试、思考人生

■ 一个不那么暴力的想法

- 将所有数利用启发式分解算法转换成质数幂次乘积的形式
- 消公因子转化为是幂指数的加减，复杂度

$\mathcal{O}((n + m + k)D(v) + (n + m)k \log v)$ ，其中 $D(v)$ 表示将 v 启发式分解的复杂度

- 大佬 10 分钟就写完并通过了 

萌新还在抄模板、调试、思考人生 

- 同余原理：对于 opt 是加、减、乘的情况，有

$$(((x \bmod M) \text{ opt } (y \bmod M)) \bmod M) = ((x \text{ opt } y) \bmod M)$$

- 乘法逆元：如果 $\gcd(x, M) = 1$ ，则 $x^{-1} \pmod{M}$ 有定义，
可以用扩展欧几里得算法 $\mathcal{O}(\log x)$ 求出
- a_i, b_j 只需要提取出与 M 不互质的部分，而互质的部分可以用逆元计算
- 看上去还是需要分解 M

- 同余原理：对于 opt 是加、减、乘的情况，有

$$(((x \bmod M) \text{ opt } (y \bmod M)) \bmod M) = ((x \text{ opt } y) \bmod M)$$

- 乘法逆元：如果 $\gcd(x, M) = 1$ ，则 $x^{-1} \pmod{M}$ 有定义，

可以用扩展欧几里得算法 $\mathcal{O}(\log x)$ 求出

- a_i, b_j 只需要提取出与 M 不互质的部分，而互质的部分可以

用逆元计算

- 看上去还是需要分解 M

- 同余原理：对于 opt 是加、减、乘的情况，有

$$(((x \bmod M) \text{ opt } (y \bmod M)) \bmod M) = ((x \text{ opt } y) \bmod M)$$

- 乘法逆元：如果 $\gcd(x, M) = 1$ ，则 $x^{-1} \pmod{M}$ 有定义，
可以用扩展欧几里得算法 $\mathcal{O}(\log x)$ 求出
- a_i, b_j 只需要提取出与 M 不互质的部分，而互质的部分可以用逆元计算
- 看上去还是需要分解 M

- 同余原理：对于 opt 是加、减、乘的情况，有

$$(((x \bmod M) \text{ opt } (y \bmod M)) \bmod M) = ((x \text{ opt } y) \bmod M)$$

- 乘法逆元：如果 $\gcd(x, M) = 1$ ，则 $x^{-1} \pmod{M}$ 有定义，
可以用扩展欧几里得算法 $\mathcal{O}(\log x)$ 求出
- a_i, b_j 只需要提取出与 M 不互质的部分，而互质的部分可以用逆元计算
- 看上去还是需要分解 M

- 预处理不超过 $M^{\frac{1}{3}}$ 的质数，对 M 进行试除，最多剩下两个质因子，即 $M' = 1$ 或 $M' = p$ 或 $M' = pq$ 或 $M' = p^2$
- 此时 $\gcd(a_i, M'), \gcd(b_j, M')$ 至多有四种可能
- 若存在 $\gcd(\cdot, M')$ 不是 1 也不是 M' ，分解成功
- 否则 a_i, b_j 只可能含有 M' 的幂次，可以视为整体进行幂指数的加减，复杂度 $\mathcal{O}(kD(M) + (n + m)k \log v)$
- 原来代码还可以这么短

- 预处理不超过 $M^{\frac{1}{3}}$ 的质数, 对 M 进行试除, 最多剩下两个质因子, 即 $M' = 1$ 或 $M' = p$ 或 $M' = pq$ 或 $M' = p^2$
- 此时 $\gcd(a_i, M'), \gcd(b_j, M')$ 至多有四种可能
- 若存在 $\gcd(\cdot, M')$ 不是 1 也不是 M' , 分解成功
- 否则 a_i, b_j 只可能含有 M' 的幂次, 可以视为整体进行幂指数的加减, 复杂度 $\mathcal{O}(kD(M) + (n + m)k \log v)$
- 原来代码还可以这么短

- 预处理不超过 $M^{\frac{1}{3}}$ 的质数, 对 M 进行试除, 最多剩下两个质因子, 即 $M' = 1$ 或 $M' = p$ 或 $M' = pq$ 或 $M' = p^2$
- 此时 $\gcd(a_i, M'), \gcd(b_j, M')$ 至多有四种可能
- 若存在 $\gcd(\cdot, M')$ 不是 1 也不是 M' , 分解成功
- 否则 a_i, b_j 只可能含有 M' 的幂次, 可以视为整体进行幂指数的加减, 复杂度 $\mathcal{O}(kD(M) + (n + m)k \log v)$
- 原来代码还可以这么短

- 预处理不超过 $M^{\frac{1}{3}}$ 的质数, 对 M 进行试除, 最多剩下两个质因子, 即 $M' = 1$ 或 $M' = p$ 或 $M' = pq$ 或 $M' = p^2$
- 此时 $\gcd(a_i, M'), \gcd(b_j, M')$ 至多有四种可能
- 若存在 $\gcd(\cdot, M')$ 不是 1 也不是 M' , 分解成功
- 否则 a_i, b_j 只可能含有 M' 的幂次, 可以视为整体进行幂指数的加减, 复杂度 $\mathcal{O}(kD(M) + (n + m)k \log v)$
- 原来代码还可以这么短

- 预处理不超过 $M^{\frac{1}{3}}$ 的质数, 对 M 进行试除, 最多剩下两个质因子, 即 $M' = 1$ 或 $M' = p$ 或 $M' = pq$ 或 $M' = p^2$
- 此时 $\gcd(a_i, M'), \gcd(b_j, M')$ 至多有四种可能
- 若存在 $\gcd(\cdot, M')$ 不是 1 也不是 M' , 分解成功
- 否则 a_i, b_j 只可能含有 M' 的幂次, 可以视为整体进行幂指数的加减, 复杂度 $\mathcal{O}(kD(M) + (n + m)k \log v)$
- 原来代码还可以这么短 为什么你会这么熟练啊

- 对任意正整数 u ，定义 $f(u)$ 是 u 的所有质因子组成的集合
- 如果正整数 u 和 v 满足 u 整除 v 且 $f(u) = f(v)$ ，那么认为 u 对 v 来说是友好的
- 给出两个正整数 k_1 和 k_2 ，分别求有多少个数对它们来说是友好的，其中 k_1 和 k_2 拥有相同的最大质因子，不同的次大质因子（如果存在）
- $1 \leq k_1, k_2 \leq 10^{24}$

- 对任意正整数 u ，定义 $f(u)$ 是 u 的所有质因子组成的集合
- 如果正整数 u 和 v 满足 u 整除 v 且 $f(u) = f(v)$ ，那么认为 u 对 v 来说是友好的
- 给出两个正整数 k_1 和 k_2 ，分别求有多少个数对它们来说是友好的，其中 k_1 和 k_2 拥有相同的最大质因子，不同的次大质因子（如果存在）
- $1 \leq k_1, k_2 \leq 10^{24}$

- 对任意正整数 u ，定义 $f(u)$ 是 u 的所有质因子组成的集合
- 如果正整数 u 和 v 满足 u 整除 v 且 $f(u) = f(v)$ ，那么认为 u 对 v 来说是友好的
- 给出两个正整数 k_1 和 k_2 ，分别求有多少个数对它们来说是友好的，其中 k_1 和 k_2 拥有相同的最大质因子，不同的次大质因子（如果存在）
- $1 \leq k_1, k_2 \leq 10^{24}$

- 对任意正整数 u ，定义 $f(u)$ 是 u 的所有质因子组成的集合
- 如果正整数 u 和 v 满足 u 整除 v 且 $f(u) = f(v)$ ，那么认为 u 对 v 来说是友好的
- 给出两个正整数 k_1 和 k_2 ，分别求有多少个数对它们来说是友好的，其中 k_1 和 k_2 拥有相同的最大质因子，不同的次大质因子（如果存在）
- $1 \leq k_1, k_2 \leq 10^{24}$

- 积性函数：若 a, b 互质时有 $f(ab) = f(a)f(b)$ ，则 f 是积性的

- 若 $k = \prod_{i=1}^{\omega(k)} p_i^{e_i}$ ，则答案是 $\prod_{i=1}^{\omega(k)} e_i$ （不是 k 的约数个数）

- 注意 $k \leq 10^{24}$ ，可以用 `_int128_t` 进行加减乘除运算

- 积性函数：若 a, b 互质时有 $f(ab) = f(a)f(b)$ ，则 f 是积性的

- 若 $k = \prod_{i=1}^{\omega(k)} p_i^{e_i}$ ，则答案是 $\prod_{i=1}^{\omega(k)} e_i$ （不是 k 的约数个数）

- 注意 $k \leq 10^{24}$ ，可以用 `__int128_t` 进行加减乘除运算

- 积性函数：若 a, b 互质时有 $f(ab) = f(a)f(b)$ ，则 f 是积性的

- 若 $k = \prod_{i=1}^{\omega(k)} p_i^{e_i}$ ，则答案是 $\prod_{i=1}^{\omega(k)} e_i$ （不是 k 的约数个数）

- 注意 $k \leq 10^{24}$ ，可以用 `__int128_t` 进行加减乘除运算

珍爱生命，远离启发式分解



- 预处理不超过 $k^{\frac{1}{4}}$ 的质数，对 k 进行试除，最多会剩下三个质因子
- 令 $p_1 > p_2 > p_3$ ， k' 可能的情况有 $1, p_1, p_1^2, p_1p_2, p_1^3, p_1^2p_2, p_1p_2^2, p_1p_2p_3$
- 还可以利用的性质是 k_1' 与 k_2' 的 p_1 相等（如果存在）， p_2 不等（如果存在）

- 预处理不超过 $k^{\frac{1}{4}}$ 的质数，对 k 进行试除，最多会剩下三个质因子

萌新眉头一皱，发现事情并不简单

- 令 $p_1 > p_2 > p_3$ ， k' 可能的情况有 $1, p_1, p_1^2, p_1p_2, p_1^3, p_1^2p_2, p_1p_2^2, p_1p_2p_3$
- 还可以利用的性质是 k_1' 与 k_2' 的 p_1 相等（如果存在）， p_2 不等（如果存在）

- 预处理不超过 $k^{\frac{1}{4}}$ 的质数，对 k 进行试除，最多会剩下三个质因子

萌新眉头一皱，发现事情并不简单

- 令 $p_1 > p_2 > p_3$ ， k' 可能的情况有 $1, p_1, p_1^2, p_1p_2, p_1^3, p_1^2p_2, p_1p_2^2, p_1p_2p_3$
- 还可以利用的性质是 k_1' 与 k_2' 的 p_1 相等（如果存在）， p_2 不等（如果存在）

- 预处理不超过 $k^{\frac{1}{4}}$ 的质数，对 k 进行试除，最多会剩下三个质因子

萌新眉头一皱，发现事情并不简单

- 令 $p_1 > p_2 > p_3$ ， k' 可能的情况有 $1, p_1, p_1^2, p_1p_2, p_1^3, p_1^2p_2, p_1p_2^2, p_1p_2p_3$
- 还可以利用的性质是 k_1' 与 k_2' 的 p_1 相等（如果存在）， p_2 不等（如果存在）

- 通过 $\mathcal{O}(\log k)$ 的代价可以求出 $v = \gcd(k_1, k_2)$, 对于每个 k 可以知道 v 包含 p_1 (如果存在), $\frac{k}{v}$ 包含 p_2 (如果存在)

- 具体来说, $(v, \frac{k}{v})$ 有这么几种情况

$$\begin{array}{lll} (1, 1) & (p_1, p_2) & (p_1, p_2^2) \\ (p_1, 1) & (p_1, p_1^2), (p_1^2, p_1), (p_1^3, 1) & (p_1, p_2 p_3), (p_1 p_3, p_2) \\ (p_1, p_1), (p_1^2, 1) & (p_1, p_1 p_2), (p_1^2, p_2) & \end{array}$$

- 考虑分情况讨论

- 通过 $\mathcal{O}(\log k)$ 的代价可以求出 $v = \gcd(k_1, k_2)$, 对于每个 k 可以知道 v 包含 p_1 (如果存在), $\frac{k}{v}$ 包含 p_2 (如果存在)

- 具体来说, $(v, \frac{k}{v})$ 有这么几种情况

$$\begin{array}{lll} (1, 1) & (p_1, p_2) & (p_1, p_2^2) \\ (p_1, 1) & (p_1, p_1^2), (p_1^2, p_1), (p_1^3, 1) & (p_1, p_2 p_3), (p_1 p_3, p_2) \\ (p_1, p_1), (p_1^2, 1) & (p_1, p_1 p_2), (p_1^2, p_2) & \end{array}$$

- 考虑分情况讨论

- 通过 $\mathcal{O}(\log k)$ 的代价可以求出 $v = \gcd(k_1, k_2)$, 对于每个 k 可以知道 v 包含 p_1 (如果存在), $\frac{k}{v}$ 包含 p_2 (如果存在)

- 具体来说, $(v, \frac{k}{v})$ 有这么几种情况

$$\begin{array}{lll} (1, 1) & (p_1, p_2) & (p_1, p_2^2) \\ (p_1, 1) & (p_1, p_1^2), (p_1^2, p_1), (p_1^3, 1) & (p_1, p_2 p_3), (p_1 p_3, p_2) \\ (p_1, p_1), (p_1^2, 1) & (p_1, p_1 p_2), (p_1^2, p_2) & \end{array}$$

- 考虑分情况讨论: $\frac{k}{v} = 1$

- 通过 $\mathcal{O}(\log k)$ 的代价可以求出 $v = \gcd(k_1, k_2)$, 对于每个 k 可以知道 v 包含 p_1 (如果存在), $\frac{k}{v}$ 包含 p_2 (如果存在)

- 具体来说, $(v, \frac{k}{v})$ 有这么几种情况

$$\begin{array}{lll} (1, 1) & (p_1, p_2) & (p_1, p_2^2) \\ (p_1, 1) & (p_1, p_1^2), (p_1^2, p_1), (p_1^3, 1) & (p_1, p_2 p_3), (p_1 p_3, p_2) \\ (p_1, p_1), (p_1^2, 1) & (p_1, p_1 p_2), (p_1^2, p_2) & \end{array}$$

- 考虑分情况讨论: $\frac{k}{v} > 1$ 且为完全平方数

- 通过 $\mathcal{O}(\log k)$ 的代价可以求出 $v = \gcd(k_1, k_2)$, 对于每个 k 可以知道 v 包含 p_1 (如果存在), $\frac{k}{v}$ 包含 p_2 (如果存在)

- 具体来说, $(v, \frac{k}{v})$ 有这么几种情况

$$\begin{array}{lll} (1, 1) & (p_1, p_2) & (p_1, p_2^2) \\ (p_1, 1) & (p_1, p_1^2), (p_1^2, p_1), (p_1^3, 1) & (p_1, p_2 p_3), (p_1 p_3, p_2) \\ (p_1, p_1), (p_1^2, 1) & (p_1, p_1 p_2), (p_1^2, p_2) & \end{array}$$

- 考虑分情况讨论: $v > 1$ 且为完全平方数

- 通过 $\mathcal{O}(\log k)$ 的代价可以求出 $v = \gcd(k_1, k_2)$ ，对于每个 k 可以知道 v 包含 p_1 （如果存在）， $\frac{k}{v}$ 包含 p_2 （如果存在）

- 具体来说， $(v, \frac{k}{v})$ 有这么几种情况

$$\begin{array}{lll} (1, 1) & (p_1, p_2) & (p_1, p_2^2) \\ (p_1, 1) & (p_1, p_1^2), (p_1^2, p_1), (p_1^3, 1) & (p_1, p_2 p_3), (p_1 p_3, p_2) \\ (p_1, p_1), (p_1^2, 1) & (p_1, p_1 p_2), (p_1^2, p_2) & \end{array}$$

- 考虑分情况讨论： $v \mid \frac{k}{v}$

- 通过 $\mathcal{O}(\log k)$ 的代价可以求出 $v = \gcd(k_1, k_2)$, 对于每个 k 可以知道 v 包含 p_1 (如果存在), $\frac{k}{v}$ 包含 p_2 (如果存在)

- 具体来说, $(v, \frac{k}{v})$ 有这么几种情况

$$\begin{array}{lll} (1, 1) & (p_1, p_2) & (p_1, p_2^2) \\ (p_1, 1) & (p_1, p_1^2), (p_1^2, p_1), (p_1^3, 1) & (p_1, p_2 p_3), (p_1 p_3, p_2) \\ (p_1, p_1), (p_1^2, 1) & (p_1, p_1 p_2), (p_1^2, p_2) & \end{array}$$

- 考虑分情况讨论: $v < \frac{k}{v}$ (因为 $p_2 p_3 > M^{\frac{1}{2}}$), 此时已不需要继续分类讨论, 因为对答案的贡献都一样

- 通过 $\mathcal{O}(\log k)$ 的代价可以求出 $v = \gcd(k_1, k_2)$, 对于每个 k 可以知道 v 包含 p_1 (如果存在), $\frac{k}{v}$ 包含 p_2 (如果存在)

- 具体来说, $(v, \frac{k}{v})$ 有这么几种情况

$$\begin{array}{lll} (1, 1) & (p_1, p_2) & (p_1, p_2^2) \\ (p_1, 1) & (p_1, p_1^2), (p_1^2, p_1), (p_1^3, 1) & (p_1, p_2 p_3), (p_1 p_3, p_2) \\ (p_1, p_1), (p_1^2, 1) & (p_1, p_1 p_2), (p_1^2, p_2) & \end{array}$$

- 考虑分情况讨论: 只能对 v 判定素性才知道是哪种情况

- 通过 $\mathcal{O}(\log k)$ 的代价可以求出 $v = \gcd(k_1, k_2)$, 对于每个 k 可以知道 v 包含 p_1 (如果存在), $\frac{k}{v}$ 包含 p_2 (如果存在)

- 具体来说, $(v, \frac{k}{v})$ 有这么几种情况

$$\begin{array}{lll} (1, 1) & (p_1, p_2) & (p_1, p_2^2) \\ (p_1, 1) & (p_1, p_1^2), (p_1^2, p_1), (p_1^3, 1) & (p_1, p_2 p_3), (p_1 p_3, p_2) \\ (p_1, p_1), (p_1^2, 1) & (p_1, p_1 p_2), (p_1^2, p_2) & \end{array}$$

- 考虑分情况讨论, 搞定, 与启发式分解复杂度相仿, 但好写

■ Summarize

- 越是博学，越是想得复杂
- 磨刀不误砍柴功
- 对于不需要规约到大数分解的问题，试除法可能更加简便
- 此外，在处理较多数字的分解时，试除法或许比启发式分解更优秀，
例如区间筛问题（求 $L \leq x \leq R$ 的 $f(x)$ ，其中 $f(x)$ 是积性函数，
 $R \leq 10^{12}, R - L \leq 10^5$ ）

■ Summarize

- 越是博学，越是想得复杂
- 磨刀不误砍柴功
- 对于不需要规约到大数分解的问题，试除法可能更加简便
- 此外，在处理较多数字的分解时，试除法或许比启发式分解更优秀，
例如区间筛问题（求 $L \leq x \leq R$ 的 $f(x)$ ，其中 $f(x)$ 是积性函数，
 $R \leq 10^{12}, R - L \leq 10^5$ ）

■ Summarize

- 越是博学，越是想得复杂
- 磨刀不误砍柴功
- 对于不需要规约到大数分解的问题，试除法可能更加简便
- 此外，在处理较多数字的分解时，试除法或许比启发式分解更优秀，
例如区间筛问题（求 $L \leq x \leq R$ 的 $f(x)$ ，其中 $f(x)$ 是积性函数，
 $R \leq 10^{12}, R - L \leq 10^5$ ）

■ Summarize

- 越是博学，越是想得复杂
- 磨刀不误砍柴功
- 对于不需要规约到大数分解的问题，试除法可能更加简便
- 此外，在处理较多数字的分解时，试除法或许比启发式分解更优秀，
例如区间筛问题（求 $L \leq x \leq R$ 的 $f(x)$ ，其中 $f(x)$ 是积性函数，
 $R \leq 10^{12}, R - L \leq 10^5$ ）

离散对数与原根

- 缩系：模 m 意义下与 m 互质的元素组成缩系，缩系中任意两个元素的乘积还在缩系中，缩系的大小是 $\varphi(m)$
- 阶：满足 $x^r \equiv 1 \pmod{m}$ 最小正整数 r 称为 x 的阶 $\text{ord}_m(x)$
- 原根：缩系中存在元素 g 使得 g^i ($i = 1, 2, \dots, \varphi(m)$) 两两不同，则称 g 是模 m 意义下的原根，也意味着缩系中的元素可以表示成 g 的幂次，不难得到 $\text{ord}_m(g) = \varphi(m)$
- 指标：若缩系有原根 g ，则元素 $x \equiv g^i \pmod{m}$ 关于 g 指标为 $\text{ind}_{m,g}(x) = i \bmod \varphi(m)$ ，显然 $\text{ord}_m(x) = \frac{\varphi(m)}{\gcd(\varphi(m), \text{ind}_{m,g}(x))}$

离散对数与原根

- 缩系：模 m 意义下与 m 互质的元素组成缩系，缩系中任意两个元素的乘积还在缩系中，缩系的大小是 $\varphi(m)$
- 阶：满足 $x^r \equiv 1 \pmod{m}$ 最小正整数 r 称为 x 的阶 $\text{ord}_m(x)$
- 原根：缩系中存在元素 g 使得 g^i ($i = 1, 2, \dots, \varphi(m)$) 两两不同，则称 g 是模 m 意义下的原根，也意味着缩系中的元素可以表示成 g 的幂次，不难得到 $\text{ord}_m(g) = \varphi(m)$
- 指标：若缩系有原根 g ，则元素 $x \equiv g^i \pmod{m}$ 关于 g 指标为 $\text{ind}_{m,g}(x) = i \bmod \varphi(m)$ ，显然 $\text{ord}_m(x) = \frac{\varphi(m)}{\gcd(\varphi(m), \text{ind}_{m,g}(x))}$

离散对数与原根

- 缩系：模 m 意义下与 m 互质的元素组成缩系，缩系中任意两个元素的乘积还在缩系中，缩系的大小是 $\varphi(m)$
- 阶：满足 $x^r \equiv 1 \pmod{m}$ 最小正整数 r 称为 x 的阶 $\text{ord}_m(x)$
- 原根：缩系中存在元素 g 使得 g^i ($i = 1, 2, \dots, \varphi(m)$) 两两不同，则称 g 是模 m 意义下的原根，也意味着缩系中的元素可以表示成 g 的幂次，不难得到 $\text{ord}_m(g) = \varphi(m)$
- 指标：若缩系有原根 g ，则元素 $x \equiv g^i \pmod{m}$ 关于 g 指标为 $\text{ind}_{m,g}(x) = i \bmod \varphi(m)$ ，显然 $\text{ord}_m(x) = \frac{\varphi(m)}{\gcd(\varphi(m), \text{ind}_{m,g}(x))}$

离散对数与原根

- 缩系：模 m 意义下与 m 互质的元素组成缩系，缩系中任意两个元素的乘积还在缩系中，缩系的大小是 $\varphi(m)$
- 阶：满足 $x^r \equiv 1 \pmod{m}$ 最小正整数 r 称为 x 的阶 $\text{ord}_m(x)$
- 原根：缩系中存在元素 g 使得 g^i ($i = 1, 2, \dots, \varphi(m)$) 两两不同，则称 g 是模 m 意义下的原根，也意味着缩系中的元素可以表示成 g 的幂次，不难得到 $\text{ord}_m(g) = \varphi(m)$
- 指标：若缩系有原根 g ，则元素 $x \equiv g^i \pmod{m}$ 关于 g 指标为 $\text{ind}_{m,g}(x) = i \bmod \varphi(m)$ ，显然 $\text{ord}_m(x) = \frac{\varphi(m)}{\gcd(\varphi(m), \text{ind}_{m,g}(x))}$

离散对数与原根

- 对于阶为 u 的元素 x , x^k 的阶为 $\frac{u}{\gcd(u,k)}$, 所以任意元素的阶整除 $\varphi(m)$, 且原根 (如果存在) 个数为 $\varphi(\varphi(m))$

- 这里存在一个 $\mathcal{O}(\log^2 m)$ 求阶的算法, 也可用于找原根

- 缩系有原根的充要条件是 $m = 2, 4, p^n, 2p^n$, 这里 p 是奇质数, n 是任意整数

(证明见Elementary Number Theory第 8.3 节, 简单易懂)

- 若缩系没有原根, 则模 m 缩系可以表示成一系列有原根的缩系的笛卡儿积 (图示), 在 $8 \nmid m$ 时还可直接表示成生成元的幂次之积, 在 $m = 2^e$ ($e > 2$) 时, 5 的阶一定是 2^{e-2}

离散对数与原根

- 对于阶为 u 的元素 x , x^k 的阶为 $\frac{u}{\gcd(u,k)}$, 所以任意元素的阶整除 $\varphi(m)$, 且原根 (如果存在) 个数为 $\varphi(\varphi(m))$
- 这里存在一个 $\mathcal{O}(\log^2 m)$ 求阶的算法, 也可用于找原根
- 缩系有原根的充要条件是 $m = 2, 4, p^n, 2p^n$, 这里 p 是奇质数, n 是任意整数

(证明见Elementary Number Theory第 8.3 节, 简单易懂)

- 若缩系没有原根, 则模 m 缩系可以表示成一系列有原根的缩系的笛卡儿积 (图示), 在 $8 \nmid m$ 时还可直接表示成生成元的幂次之积, 在 $m = 2^e$ ($e > 2$) 时, 5 的阶一定是 2^{e-2}

离散对数与原根

- 对于阶为 u 的元素 x , x^k 的阶为 $\frac{u}{\gcd(u,k)}$, 所以任意元素的阶整除 $\varphi(m)$, 且原根 (如果存在) 个数为 $\varphi(\varphi(m))$
- 这里存在一个 $\mathcal{O}(\log^2 m)$ 求阶的算法, 也可用于找原根
- 缩系有原根的充要条件是 $m = 2, 4, p^n, 2p^n$, 这里 p 是奇质数, n 是任意整数

(证明见Elementary Number Theory第 8.3 节, 简单易懂)

- 若缩系没有原根, 则模 m 缩系可以表示成一系列有原根的缩系的笛卡儿积 (图示), 在 $8 \nmid m$ 时还可直接表示成生成元的幂次之积, 在 $m = 2^e$ ($e > 2$) 时, 5 的阶一定是 2^{e-2}

离散对数与原根

- 对于阶为 u 的元素 x , x^k 的阶为 $\frac{u}{\gcd(u,k)}$, 所以任意元素的阶整除 $\varphi(m)$, 且原根 (如果存在) 个数为 $\varphi(\varphi(m))$
- 这里存在一个 $\mathcal{O}(\log^2 m)$ 求阶的算法, 也可用于找原根
- 缩系有原根的充要条件是 $m = 2, 4, p^n, 2p^n$, 这里 p 是奇质数, n 是任意整数

(证明见Elementary Number Theory第 8.3 节, 简单易懂)

- 若缩系没有原根, 则模 m 缩系可以表示成一系列有原根的缩系的笛卡儿积 (图示), 在 $8 \nmid m$ 时还可直接表示成生成元的幂次之积, 在 $m = 2^e$ ($e > 2$) 时, 5 的阶一定是 2^{e-2}

- 考虑解方程 $A^B \equiv C \pmod{M}$, 已知其中三个元素
- 已知 A, B, M 求 C 是模幂问题
- 已知 A, C, M 求 B 是离散对数问题
- 已知 B, C, M 求 A 是高次剩余问题
- 已知 A, B, C 求 M 是大数分解问题

离散对数与原根

- 考虑解方程 $A^B \equiv C \pmod{M}$, 已知其中三个元素
- 已知 A, B, M 求 C 是模幂问题
- 已知 A, C, M 求 B 是离散对数问题
- 已知 B, C, M 求 A 是高次剩余问题
- 已知 A, B, C 求 M 是大数分解问题

离散对数与原根

- 解方程 $A^B \equiv C \pmod{M}$, 已知 A, C, M 求 B , M 是质数

- 设一个原根是 g , 问题等价于 $\text{Bind}_{M,g}(A) \equiv \text{ind}_{M,g}(C)$
 $\pmod{\varphi(M)}$

- 设 $r = \gcd(\text{ind}_{M,g}(A), \varphi(M)) = \frac{\varphi(M)}{\text{ord}_M(A)}$

问题转化为 $B \frac{\text{ind}_{M,g}(A)}{r} \equiv \frac{\text{ind}_{M,g}(C)}{r} \pmod{\frac{\varphi(M)}{r}}$

可得 $B \equiv \frac{\text{ind}_{M,g}(C)}{r} \left(\frac{\text{ind}_{M,g}(A)}{r} \right)^{-1} \pmod{\frac{\varphi(M)}{r}}$

- 满足 $1 \leq B \leq \varphi(M)$ 的解有 r 个, 它们在模 $\text{ord}_M(A)$ 意义下同余,
然而想算出具体值还是需要求解 $\text{ind}_{M,g}(A)$ 和 $\text{ind}_{M,g}(C)$, 或者说
 $\text{ind}_{M,A}(C)$

离散对数与原根

- 解方程 $A^B \equiv C \pmod{M}$, 已知 A, C, M 求 B , M 是质数

- 设一个原根是 g , 问题等价于 $\text{Bind}_{M,g}(A) \equiv \text{ind}_{M,g}(C)$
 $\pmod{\varphi(M)}$

- 设 $r = \gcd(\text{ind}_{M,g}(A), \varphi(M)) = \frac{\varphi(M)}{\text{ord}_M(A)}$

问题转化为 $B \frac{\text{ind}_{M,g}(A)}{r} \equiv \frac{\text{ind}_{M,g}(C)}{r} \pmod{\frac{\varphi(M)}{r}}$

可得 $B \equiv \frac{\text{ind}_{M,g}(C)}{r} \left(\frac{\text{ind}_{M,g}(A)}{r} \right)^{-1} \pmod{\frac{\varphi(M)}{r}}$

- 满足 $1 \leq B \leq \varphi(M)$ 的解有 r 个, 它们在模 $\text{ord}_M(A)$ 意义下同余,
然而想算出具体值还是需要求解 $\text{ind}_{M,g}(A)$ 和 $\text{ind}_{M,g}(C)$, 或者说
 $\text{ind}_{M,A}(C)$

离散对数与原根

- 解方程 $A^B \equiv C \pmod{M}$, 已知 A, C, M 求 B , M 是质数

- 设一个原根是 g , 问题等价于 $\text{Bind}_{M,g}(A) \equiv \text{ind}_{M,g}(C)$
 $\pmod{\varphi(M)}$

- 设 $r = \gcd(\text{ind}_{M,g}(A), \varphi(M)) = \frac{\varphi(M)}{\text{ord}_M(A)}$

问题转化为 $B \frac{\text{ind}_{M,g}(A)}{r} \equiv \frac{\text{ind}_{M,g}(C)}{r} \pmod{\frac{\varphi(M)}{r}}$

可得 $B \equiv \frac{\text{ind}_{M,g}(C)}{r} \left(\frac{\text{ind}_{M,g}(A)}{r} \right)^{-1} \pmod{\frac{\varphi(M)}{r}}$

- 满足 $1 \leq B \leq \varphi(M)$ 的解有 r 个, 它们在模 $\text{ord}_M(A)$ 意义下同余,
然而想算出具体值还是需要求解 $\text{ind}_{M,g}(A)$ 和 $\text{ind}_{M,g}(C)$, 或者说
 $\text{ind}_{M,A}(C)$

■ 大步小步算法 (Baby-Step Giant-Step Algorithm)

- 考虑求出 $1 \leq B \leq \text{ord}_M(A)$ 的唯一解, 设 $B = uT - v$, 其中 T 是设定的阈值, $1 \leq u \leq \frac{\text{ord}_M(A)}{T}, 0 \leq v < T$
- 由于 A^i ($i \in \mathbb{N}$) 的轨道是一个环, $A^B \equiv C$ 可化为 $A^{uT} \equiv CA^v$
- 预处理 A^v ($v = 0, 1, \dots, T$), 枚举 u 检查是否存在解, 若存在解则解唯一, 故只需哈希所需的 A^v
- 复杂度 $\mathcal{O}(T + \frac{\text{ord}_M(A)}{T})$, 取 $T = \mathcal{O}(\sqrt{\text{ord}_M(A)})$

■ 大步小步算法 (Baby-Step Giant-Step Algorithm)

- 考虑求出 $1 \leq B \leq \text{ord}_M(A)$ 的唯一解, 设 $B = uT - v$, 其中 T 是设定的阈值, $1 \leq u \leq \frac{\text{ord}_M(A)}{T}, 0 \leq v < T$
- 由于 A^i ($i \in \mathbb{N}$) 的轨道是一个环, $A^B \equiv C$ 可化为 $A^{uT} \equiv CA^v$
- 预处理 A^v ($v = 0, 1, \dots, T$), 枚举 u 检查是否存在解, 若存在解则解唯一, 故只需哈希所需的 A^v
- 复杂度 $\mathcal{O}(T + \frac{\text{ord}_M(A)}{T})$, 取 $T = \mathcal{O}(\sqrt{\text{ord}_M(A)})$

■ 大步小步算法 (Baby-Step Giant-Step Algorithm)

- 考虑求出 $1 \leq B \leq \text{ord}_M(A)$ 的唯一解, 设 $B = uT - v$, 其中 T 是设定的阈值, $1 \leq u \leq \frac{\text{ord}_M(A)}{T}, 0 \leq v < T$
- 由于 A^i ($i \in \mathbb{N}$) 的轨道是一个环, $A^B \equiv C$ 可化为 $A^{uT} \equiv CA^v$
- 预处理 A^v ($v = 0, 1, \dots, T$), 枚举 u 检查是否存在解, 若存在解则解唯一, 故只需哈希所需的 A^v
- 复杂度 $\mathcal{O}(T + \frac{\text{ord}_M(A)}{T})$, 取 $T = \mathcal{O}(\sqrt{\text{ord}_M(A)})$

■ 大步小步算法 (Baby-Step Giant-Step Algorithm)

- 考虑求出 $1 \leq B \leq \text{ord}_M(A)$ 的唯一解, 设 $B = uT - v$, 其中 T 是设定的阈值, $1 \leq u \leq \frac{\text{ord}_M(A)}{T}, 0 \leq v < T$
- 由于 A^i ($i \in \mathbb{N}$) 的轨道是一个环, $A^B \equiv C$ 可化为 $A^{uT} \equiv CA^v$
- 预处理 A^v ($v = 0, 1, \dots, T$), 枚举 u 检查是否存在解, 若存在解则解唯一, 故只需哈希所需的 A^v
- 复杂度 $\mathcal{O}(T + \frac{\text{ord}_M(A)}{T})$, 取 $T = \mathcal{O}(\sqrt{\text{ord}_M(A)})$
- 需要保证 A^i ($i \in \mathbb{N}$) 的轨道是一个环, 或者说 A^{-1} 存在

■ 大步小步算法 (Baby-Step Giant-Step Algorithm)

- 考虑求出 $1 \leq B \leq \text{ord}_M(A)$ 的唯一解, 设 $B = uT - v$, 其中 T 是设定的阈值, $1 \leq u \leq \frac{\text{ord}_M(A)}{T}, 0 \leq v < T$
- 由于 A^i ($i \in \mathbb{N}$) 的轨道是一个环, $A^B \equiv C$ 可化为 $A^{uT} \equiv CA^v$
- 预处理 A^v ($v = 0, 1, \dots, T$), 枚举 u 检查是否存在解, 若存在解则解唯一, 故只需哈希所需的 A^v
- 复杂度 $\mathcal{O}(T + \frac{\text{ord}_M(A)}{T})$, 取 $T = \mathcal{O}(\sqrt{\text{ord}_M(A)})$
- 需要保证 A^i ($i \in \mathbb{N}$) 的轨道是一个环, 或者说 A^{-1} 存在

- 中国剩余定理：给定一系列同余方程 $x \equiv r_i \pmod{m_i}$ ，满足 m_i ($i = 1, 2, \dots, k$) 两两互质，则方程组存在唯一的通解 $x \equiv R \pmod{M}$ ，其中 $M = \text{lcm}(m_1, m_2, \dots, m_k)$ ， $R = \sum_{i=1}^k r_i M'_i \frac{M}{m_i}$ ， M'_i 表示 $\frac{M}{m_i}$ 在模 m_i 意义下乘法逆元
- 解方程 $A^B \equiv C \pmod{M}$ ，已知 A, C, M 求 B ，满足 $\gcd(A, M) = 1$, M 是奇数（题目：数论之神）
 - 由定理可知，解在模 $\text{ord}_M(A)$ 意义下唯一，大步小步算法适用

- 中国剩余定理：给定一系列同余方程 $x \equiv r_i \pmod{m_i}$ ，满足 m_i ($i = 1, 2, \dots, k$) 两两互质，则方程组存在唯一的通解 $x \equiv R \pmod{M}$ ，其中 $M = \text{lcm}(m_1, m_2, \dots, m_k)$ ， $R = \sum_{i=1}^k r_i M'_i \frac{M}{m_i}$ ， M'_i 表示 $\frac{M}{m_i}$ 在模 m_i 意义下乘法逆元
- 解方程 $A^B \equiv C \pmod{M}$ ，已知 A, C, M 求 B ，满足 $\gcd(A, M) = 1$, M 是奇数（题目：数论之神）
 - 由定理可知，解在模 $\text{ord}_M(A)$ 意义下唯一，大步小步算法适用

- 中国剩余定理：给定一系列同余方程 $x \equiv r_i \pmod{m_i}$ ，满足 m_i ($i = 1, 2, \dots, k$) 两两互质，则方程组存在唯一的通解 $x \equiv R \pmod{M}$ ，其中 $M = \text{lcm}(m_1, m_2, \dots, m_k)$ ， $R = \sum_{i=1}^k r_i M'_i \frac{M}{m_i}$ ， M'_i 表示 $\frac{M}{m_i}$ 在模 m_i 意义下乘法逆元
- 解方程 $A^B \equiv C \pmod{M}$ ，已知 A, C, M 求 B ，满足 $\gcd(A, M) = 1$, M 是奇数（题目：数论之神）
 - 由定理可知，解在模 $\text{ord}_M(A)$ 意义下唯一，大步小步算法适用

离散对数与原根

- $\gcd(A, M) > 1$ 时, 不妨考虑将 M 表示成 $\prod_{i=1}^{\omega(M)} p_i^{e_i}$ 的形式
- 令 $A \pmod{p_i^{e_i}} = p_i^u v$, 当 $u > 0$ 时, $ut \geq e_i$ 时 $A^t \equiv 0 \pmod{p_i^{e_i}}$, 会有一段不循环的结果, 并且之后的循环节是 1, 否则 $\text{ord}_{p_i^{e_i}}(A)$ 存在, 且 $\text{ord}_{p_i^{e_i}}(A) \mid \varphi(p_i^{e_i})$
- 经过不循环的段后, 必然产生循环, 循环的长度整除 $\text{lcm}(\varphi(p_1^{e_1}), \varphi(p_2^{e_2}), \dots, \varphi(p_{\omega(M)}^{e_{\omega(M)}})) \mid \varphi(M)$
- 不循环的段长度小于循环的长度
- 令 $A^i \bmod M$ 向 $A^{i+1} \bmod M$ ($i \in \mathbb{N}$) 连边, 轨道呈现 ρ 型

离散对数与原根

- $\gcd(A, M) > 1$ 时, 不妨考虑将 M 表示成 $\prod_{i=1}^{\omega(M)} p_i^{e_i}$ 的形式
- 令 $A \pmod{p_i^{e_i}} = p_i^u v$, 当 $u > 0$ 时, $ut \geq e_i$ 时 $A^t \equiv 0 \pmod{p_i^{e_i}}$, 会有一段不循环的结果, 并且之后的循环节是 1, 否则 $\text{ord}_{p_i^{e_i}}(A)$ 存在, 且 $\text{ord}_{p_i^{e_i}}(A) \mid \varphi(p_i^{e_i})$
- 经过不循环的段后, 必然产生循环, 循环的长度整除 $\text{lcm}(\varphi(p_1^{e_1}), \varphi(p_2^{e_2}), \dots, \varphi(p_{\omega(M)}^{e_{\omega(M)}})) \mid \varphi(M)$
- 不循环的段长度小于循环的长度
- 令 $A^i \bmod M$ 向 $A^{i+1} \bmod M$ ($i \in \mathbb{N}$) 连边, 轨道呈现 ρ 型

离散对数与原根

- $\gcd(A, M) > 1$ 时, 不妨考虑将 M 表示成 $\prod_{i=1}^{\omega(M)} p_i^{e_i}$ 的形式
- 令 $A \pmod{p_i^{e_i}} = p_i^u v$, 当 $u > 0$ 时, $ut \geq e_i$ 时 $A^t \equiv 0 \pmod{p_i^{e_i}}$, 会有一段不循环的结果, 并且之后的循环节是 1, 否则 $\text{ord}_{p_i^{e_i}}(A)$ 存在, 且 $\text{ord}_{p_i^{e_i}}(A) \mid \varphi(p_i^{e_i})$
- 经过不循环的段后, 必然产生循环, 循环的长度整除 $\text{lcm}(\varphi(p_1^{e_1}), \varphi(p_2^{e_2}), \dots, \varphi(p_{\omega(M)}^{e_{\omega(M)}})) \mid \varphi(M)$
- 不循环的段长度小于循环的长度
- 令 $A^i \bmod M$ 向 $A^{i+1} \bmod M$ ($i \in \mathbb{N}$) 连边, 轨道呈现 ρ 型

离散对数与原根

- $\gcd(A, M) > 1$ 时, 不妨考虑将 M 表示成 $\prod_{i=1}^{\omega(M)} p_i^{e_i}$ 的形式
- 令 $A \pmod{p_i^{e_i}} = p_i^u v$, 当 $u > 0$ 时, $ut \geq e_i$ 时 $A^t \equiv 0 \pmod{p_i^{e_i}}$, 会有一段不循环的结果, 并且之后的循环节是 1, 否则 $\text{ord}_{p_i^{e_i}}(A)$ 存在, 且 $\text{ord}_{p_i^{e_i}}(A) \mid \varphi(p_i^{e_i})$
- 经过不循环的段后, 必然产生循环, 循环的长度整除 $\text{lcm}(\varphi(p_1^{e_1}), \varphi(p_2^{e_2}), \dots, \varphi(p_{\omega(M)}^{e_{\omega(M)}})) \mid \varphi(M)$
- 不循环的段长度小于循环的长度
- 令 $A^i \bmod M$ 向 $A^{i+1} \bmod M$ ($i \in \mathbb{N}$) 连边, 轨道呈现 ρ 型

离散对数与原根

- $\gcd(A, M) > 1$ 时, 不妨考虑将 M 表示成 $\prod_{i=1}^{\omega(M)} p_i^{e_i}$ 的形式
- 令 $A \pmod{p_i^{e_i}} = p_i^u v$, 当 $u > 0$ 时, $ut \geq e_i$ 时 $A^t \equiv 0 \pmod{p_i^{e_i}}$, 会有一段不循环的结果, 并且之后的循环节是 1, 否则 $\text{ord}_{p_i^{e_i}}(A)$ 存在, 且 $\text{ord}_{p_i^{e_i}}(A) \mid \varphi(p_i^{e_i})$
- 经过不循环的段后, 必然产生循环, 循环的长度整除 $\text{lcm}(\varphi(p_1^{e_1}), \varphi(p_2^{e_2}), \dots, \varphi(p_{\omega(M)}^{e_{\omega(M)}})) \mid \varphi(M)$
- 不循环的段长度小于循环的长度
- 令 $A^i \bmod M$ 向 $A^{i+1} \bmod M$ ($i \in \mathbb{N}$) 连边, 轨道呈现 ρ 型

■ Pollard's rho Algorithm for Logarithms

- 把集合 $G = \{A^i \bmod M | i \in \mathbb{N}\}$ 分成三个部分 S_0, S_1, S_2 (比如根据模 3 的余值来划分), 并保证 $1 \notin S_1$
- 生成一系列 $x = A^i C^j$ 直到某个 x 另一种表示方法 $x = A^x C^y$, 则 $(i - x) \equiv B(j - y) \pmod{|G|}$, 方程可能有多解 (若不在环上?)
- 沿用 Floyd's Cycle-Finding Algorithm, 生成一系列元素 x_0, x_1, \dots 满足 $x_{i+1} = f(x_i)$ ($i = 0, 1, \dots$), 这里 $f(x) = Cx$ if $x \in S_0$,
 $f(x) = x^2$ if $x \in S_1$, $f(x) = Ax$ if $x \in S_2$
- 维护 x_i, x_{2i} 找环, 期望复杂度 $\mathcal{O}(\sqrt{\frac{\pi n}{2}})$, 不需要 G 关于 $*$ 成循环群, 证明见 Monte Carlo Methods for Index Computation (mod p)

■ Pollard's rho Algorithm for Logarithms

- 把集合 $G = \{A^i \bmod M | i \in \mathbb{N}\}$ 分成三个部分 S_0, S_1, S_2 (比如根据模 3 的余值来划分), 并保证 $1 \notin S_1$
- 生成一系列 $x = A^i C^j$ 直到某个 x 另一种表示方法 $x = A^x C^y$, 则 $(i - x) \equiv B(j - y) \pmod{|G|}$, 方程可能有多解 (若不在环上?)
- 沿用 Floyd's Cycle-Finding Algorithm, 生成一系列元素 x_0, x_1, \dots 满足 $x_{i+1} = f(x_i)$ ($i = 0, 1, \dots$), 这里 $f(x) = Cx$ if $x \in S_0$, $f(x) = x^2$ if $x \in S_1$, $f(x) = Ax$ if $x \in S_2$
- 维护 x_i, x_{2i} 找环, 期望复杂度 $\mathcal{O}(\sqrt{\frac{\pi n}{2}})$, 不需要 G 关于 $*$ 成循环群, 证明见 Monte Carlo Methods for Index Computation (mod p)

■ Pollard's rho Algorithm for Logarithms

- 把集合 $G = \{A^i \bmod M | i \in \mathbb{N}\}$ 分成三个部分 S_0, S_1, S_2 (比如根据模 3 的余值来划分), 并保证 $1 \notin S_1$
- 生成一系列 $x = A^i C^j$ 直到某个 x 另一种表示方法 $x = A^x C^y$, 则 $(i - x) \equiv B(j - y) \pmod{|G|}$, 方程可能有多解 (若不在环上?)
- 沿用 Floyd's Cycle-Finding Algorithm, 生成一系列元素 x_0, x_1, \dots 满足 $x_{i+1} = f(x_i)$ ($i = 0, 1, \dots$), 这里 $f(x) = Cx$ if $x \in S_0$,
 $f(x) = x^2$ if $x \in S_1$, $f(x) = Ax$ if $x \in S_2$
- 维护 x_i, x_{2i} 找环, 期望复杂度 $\mathcal{O}(\sqrt{\frac{\pi n}{2}})$, 不需要 G 关于 $*$ 成循环群, 证明见 Monte Carlo Methods for Index Computation (mod p)

■ Pollard's rho Algorithm for Logarithms

- 把集合 $G = \{A^i \bmod M | i \in \mathbb{N}\}$ 分成三个部分 S_0, S_1, S_2 (比如根据模 3 的余值来划分), 并保证 $1 \notin S_1$
- 生成一系列 $x = A^i C^j$ 直到某个 x 另一种表示方法 $x = A^x C^y$, 则 $(i - x) \equiv B(j - y) \pmod{|G|}$, 方程可能有多解 (若不在环上?)
- 沿用 Floyd's Cycle-Finding Algorithm, 生成一系列元素 x_0, x_1, \dots 满足 $x_{i+1} = f(x_i)$ ($i = 0, 1, \dots$), 这里 $f(x) = Cx$ if $x \in S_0$,
 $f(x) = x^2$ if $x \in S_1$, $f(x) = Ax$ if $x \in S_2$
- 维护 x_i, x_{2i} 找环, 期望复杂度 $\mathcal{O}(\sqrt{\frac{\pi n}{2}})$, 不需要 G 关于 $*$ 成循环群, 证明见 Monte Carlo Methods for Index Computation (mod p)

- 给定整数 $seed, p, n$ 和 k , 求解满足方程

$((seed^{2^x} \bmod p) \bmod n) = k$ 的最小正整数解 x , 无解输出 -1

- $1 \leq seed < p \leq 10^9, 0 \leq k < n \leq 10^9, p$ 是质数

- 给定整数 $seed, p, n$ 和 k , 求解满足方程

$((seed^{2^x} \bmod p) \bmod n) = k$ 的最小正整数解 x , 无解输出 -1

- $1 \leq seed < p \leq 10^9, 0 \leq k < n \leq 10^9, p$ 是质数

■ 问题可以划分成几个阶段

- 找到 u ($0 \leq u < p$) 满足 $u \equiv k \pmod{n}$
- 找到 v ($1 \leq v \leq \text{ord}_p(\text{seed})$) 满足 $\text{seed}^v \equiv u \pmod{p}$
- 找到 x ($1 \leq x \leq \text{ord}_{\text{ord}_p(\text{seed})}(2)$) 满足 $2^x \equiv v \pmod{\text{ord}_p(\text{seed})}$

■ 问题可以划分成几个阶段

- 找到 u ($0 \leq u < p$) 满足 $u \equiv k \pmod{n}$
- 找到 v ($1 \leq v \leq \text{ord}_p(\text{seed})$) 满足 $\text{seed}^v \equiv u \pmod{p}$
- 找到 x ($1 \leq x \leq \text{ord}_{\text{ord}_p(\text{seed})}(2)$) 满足 $2^x \equiv v \pmod{\text{ord}_p(\text{seed})}$

■ 问题可以划分成几个阶段

- 找到 u ($0 \leq u < p$) 满足 $u \equiv k \pmod{n}$
- 找到 v ($1 \leq v \leq \text{ord}_p(\text{seed})$) 满足 $\text{seed}^v \equiv u \pmod{p}$
- 找到 x ($1 \leq x \leq \text{ord}_{\text{ord}_p(\text{seed})}(2)$) 满足 $2^x \equiv v \pmod{\text{ord}_p(\text{seed})}$


■ 问题可以划分成几个阶段

- 找到 u ($0 \leq u < p$) 满足 $u \equiv k \pmod{n}$
- 找到 v ($1 \leq v \leq \text{ord}_p(\text{seed})$) 满足 $\text{seed}^v \equiv u \pmod{p}$
- 找到 x ($1 \leq x \leq \text{ord}_{\text{ord}_p(\text{seed})}(2)$) 满足 $2^x \equiv v \pmod{\text{ord}_p(\text{seed})}$

■ 问题可以划分成几个阶段

- 找到 u ($0 \leq u < p$) 满足 $u \equiv k \pmod{n}$
- 找到 v ($1 \leq v \leq \text{ord}_p(\text{seed})$) 满足 $\text{seed}^v \equiv u \pmod{p}$
- ~~找到 x ($1 \leq x \leq \text{ord}_{\text{ord}_p(\text{seed})}(2)$) 满足 $2^x \equiv v \pmod{\text{ord}_p(\text{seed})}$)~~

$1 \leq \text{seed} < p$ 且 p 是质数, 所以 $\text{ord}_p(\text{seed})$ 一定存在

但是 $\text{ord}_{\text{ord}_p(\text{seed})}(2)$ 不一定存在 

- 采用 Pollard's rho Algorithm for Logarithms 算法

■ 问题可以划分成几个阶段

- 找到 u ($0 \leq u < p$) 满足 $u \equiv k \pmod{n}$
- 找到 v ($1 \leq v \leq \text{ord}_p(\text{seed})$) 满足 $\text{seed}^v \equiv u \pmod{p}$
- ~~找到 x ($1 \leq x \leq \text{ord}_{\text{ord}_p(\text{seed})}(2)$) 满足 $2^x \equiv v \pmod{\text{ord}_p(\text{seed})}$)~~

$1 \leq \text{seed} < p$ 且 p 是质数, 所以 $\text{ord}_p(\text{seed})$ 一定存在

但是 $\text{ord}_{\text{ord}_p(\text{seed})}(2)$ 不一定存在

- 采用 Pollard's rho Algorithm for Logarithms 算法


■ 问题可以划分成几个阶段

- 找到 u ($0 \leq u < p$) 满足 $u \equiv k \pmod{n}$
- 找到 v ($1 \leq v \leq \text{ord}_p(\text{seed})$) 满足 $\text{seed}^v \equiv u \pmod{p}$
- ~~找到 x ($1 \leq x \leq \text{ord}_{\text{ord}_p(\text{seed})}(2)$) 满足 $2^x \equiv v \pmod{\text{ord}_p(\text{seed})}$)~~

$1 \leq \text{seed} < p$ 且 p 是质数, 所以 $\text{ord}_p(\text{seed})$ 一定存在

但是 $\text{ord}_{\text{ord}_p(\text{seed})}(2)$ 不一定存在

- ~~采用 Pollard's rho Algorithm for Logarithms 算法~~

最优复杂度 $\mathcal{O}(p^{\frac{3}{4}})$, 会超过时间限制 

不妨从 $G = \{2^i \bmod \text{ord}_p(\text{seed}) \mid i \in \mathbb{N}\}$ 的形状入手

- 解方程 $A^B \equiv C \pmod{M}$, 已知 A, C, M 求最小非负整数 B
 - 只需最小解, 若 $\gcd(A, M) = 1$, A^{-1} 有定义, 大步小步算法适用
- 扩展大步小步算法 (Extended Baby-Step Giant-Step Algorithm)
 - 把方程化为 $A^{B-\delta} A^\delta \equiv C \pmod{M}$, 消去公因子变为
$$A^{B-\delta} A' \equiv C' \pmod{M'}$$
 , 枚举 $\delta = 0, 1, \dots$ 进行下面的步骤
 - 若 $\gcd(A, M') = 1$, 套用大步小步算法
 - 否则检验是否有 $A' \equiv C' \pmod{M'}$, 如果有则找到解
 - 如果没有, 则增加 δ , 尝试将 A', C', M' 消去公因子 $\gcd(A, M')$
 - 只会进行至多 $\log_2 M$ 步消因子操作

- 解方程 $A^B \equiv C \pmod{M}$, 已知 A, C, M 求最小非负整数 B
 - 只需最小解, 若 $\gcd(A, M) = 1$, A^{-1} 有定义, 大步小步算法适用
- 扩展大步小步算法 (Extended Baby-Step Giant-Step Algorithm)
 - 把方程化为 $A^{B-\delta} A^\delta \equiv C \pmod{M}$, 消去公因子变为
$$A^{B-\delta} A' \equiv C' \pmod{M'}$$
 , 枚举 $\delta = 0, 1, \dots$ 进行下面的步骤
 - 若 $\gcd(A, M') = 1$, 套用大步小步算法
 - 否则检验是否有 $A' \equiv C' \pmod{M'}$, 如果有则找到解
 - 如果没有, 则增加 δ , 尝试将 A', C', M' 消去公因子 $\gcd(A, M')$
 - 只会进行至多 $\log_2 M$ 步消因子操作

- 解方程 $A^B \equiv C \pmod{M}$, 已知 A, C, M 求最小非负整数 B
 - 只需最小解, 若 $\gcd(A, M) = 1$, A^{-1} 有定义, 大步小步算法适用
- 扩展大步小步算法 (Extended Baby-Step Giant-Step Algorithm)
 - 把方程化为 $A^{B-\delta} A^\delta \equiv C \pmod{M}$, 消去公因子变为
$$A^{B-\delta} A' \equiv C' \pmod{M'}$$
 , 枚举 $\delta = 0, 1, \dots$ 进行下面的步骤
 - 若 $\gcd(A, M') = 1$, 套用大步小步算法
 - 否则检验是否有 $A' \equiv C' \pmod{M'}$, 如果有则找到解
 - 如果没有, 则增加 δ , 尝试将 A', C', M' 消去公因子 $\gcd(A, M')$
 - 只会进行至多 $\log_2 M$ 步消因子操作

- 解方程 $A^B \equiv C \pmod{M}$, 已知 A, C, M 求最小非负整数 B
 - 只需最小解, 若 $\gcd(A, M) = 1$, A^{-1} 有定义, 大步小步算法适用
- 扩展大步小步算法 (Extended Baby-Step Giant-Step Algorithm)
 - 把方程化为 $A^{B-\delta} A^\delta \equiv C \pmod{M}$, 消去公因子变为
$$A^{B-\delta} A' \equiv C' \pmod{M'}$$
 , 枚举 $\delta = 0, 1, \dots$ 进行下面的步骤
 - 若 $\gcd(A, M') = 1$, 套用大步小步算法
 - 否则检验是否有 $A' \equiv C' \pmod{M'}$, 如果有则找到解
 - 如果没有, 则增加 δ , 尝试将 A', C', M' 消去公因子 $\gcd(A, M')$
 - 只会进行至多 $\log_2 M$ 步消因子操作

- 解方程 $A^B \equiv C \pmod{M}$, 已知 A, C, M 求最小非负整数 B
 - 只需最小解, 若 $\gcd(A, M) = 1$, A^{-1} 有定义, 大步小步算法适用
- 扩展大步小步算法 (Extended Baby-Step Giant-Step Algorithm)
 - 把方程化为 $A^{B-\delta} A^\delta \equiv C \pmod{M}$, 消去公因子变为
$$A^{B-\delta} A' \equiv C' \pmod{M'}$$
 , 枚举 $\delta = 0, 1, \dots$ 进行下面的步骤
 - 若 $\gcd(A, M') = 1$, 套用大步小步算法
 - 否则检验是否有 $A' \equiv C' \pmod{M'}$, 如果有则找到解
 - 如果没有, 则增加 δ , 尝试将 A', C', M' 消去公因子 $\gcd(A, M')$
 - 只会进行至多 $\log_2 M$ 步消因子操作

- 解方程 $A^B \equiv C \pmod{M}$, 已知 A, C, M 求最小非负整数 B
 - 只需最小解, 若 $\gcd(A, M) = 1$, A^{-1} 有定义, 大步小步算法适用
- 扩展大步小步算法 (Extended Baby-Step Giant-Step Algorithm)
 - 把方程化为 $A^{B-\delta} A^\delta \equiv C \pmod{M}$, 消去公因子变为
$$A^{B-\delta} A' \equiv C' \pmod{M'}$$
 , 枚举 $\delta = 0, 1, \dots$ 进行下面的步骤
 - 若 $\gcd(A, M') = 1$, 套用大步小步算法
 - 否则检验是否有 $A' \equiv C' \pmod{M'}$, 如果有则找到解
 - 如果没有, 则增加 δ , 尝试将 A', C', M' 消去公因子 $\gcd(A, M')$
 - 只会进行至多 $\log_2 M$ 步消因子操作

- 解方程 $A^B \equiv C \pmod{M}$, 已知 A, C, M 求最小非负整数 B
 - 只需最小解, 若 $\gcd(A, M) = 1$, A^{-1} 有定义, 大步小步算法适用
- 扩展大步小步算法 (Extended Baby-Step Giant-Step Algorithm)
 - 把方程化为 $A^{B-\delta} A^\delta \equiv C \pmod{M}$, 消去公因子变为
$$A^{B-\delta} A' \equiv C' \pmod{M'}$$
 , 枚举 $\delta = 0, 1, \dots$ 进行下面的步骤
 - 若 $\gcd(A, M') = 1$, 套用大步小步算法
 - 否则检验是否有 $A' \equiv C' \pmod{M'}$, 如果有则找到解
 - 如果没有, 则增加 δ , 尝试将 A', C', M' 消去公因子 $\gcd(A, M')$
 - 只会进行至多 $\log_2 M$ 步消因子操作

- 解方程 $A^B \equiv C \pmod{M}$, 已知 A, C, M 求最小非负整数 B
 - 只需最小解, 若 $\gcd(A, M) = 1$, A^{-1} 有定义, 大步小步算法适用
- 扩展大步小步算法 (Extended Baby-Step Giant-Step Algorithm)
 - 把方程化为 $A^{B-\delta} A^\delta \equiv C \pmod{M}$, 消去公因子变为
$$A^{B-\delta} A' \equiv C' \pmod{M'}$$
 , 枚举 $\delta = 0, 1, \dots$ 进行下面的步骤
 - 若 $\gcd(A, M') = 1$, 套用大步小步算法
 - 否则检验是否有 $A' \equiv C' \pmod{M'}$, 如果有则找到解
 - 如果没有, 则增加 δ , 尝试将 A', C', M' 消去公因子 $\gcd(A, M')$
 - 只会进行至多 $\log_2 M$ 步消因子操作

■ 回到本题

- 找到 u ($0 \leq u < p$) 满足 $u \equiv k \pmod{n}$

- 找到 v ($1 \leq v \leq \text{ord}_p(\text{seed})$) 满足 $\text{seed}^v \equiv u \pmod{p}$

- 令 $\text{ord}_p(\text{seed}) = 2^e \cdot r$ ，求出 $\text{ord}_r(2)$ ，并设定阈值 Q

- 当 $n \leq Q$ 时，枚举 $1 \leq x \leq e + \text{ord}_r(2)$ 检查，模值在模 n 意义下或
可视为随机分布，期望复杂度 $\mathcal{O}(Q)$

- 当 $n > Q$ 时， u 有不超过 $\frac{p}{Q}$ 种取值，枚举 u 求解 v ，再求解 x ，
期望复杂度 $\mathcal{O}(T + \frac{p}{Q}(\log p + \frac{p}{T}))$ ，取 $T = \mathcal{O}(\frac{p}{\sqrt{Q}})$

- 为均衡两种情况的复杂度，取 $Q = \mathcal{O}(p^{\frac{2}{3}})$

■ 回到本题

- 找到 u ($0 \leq u < p$) 满足 $u \equiv k \pmod{n}$
- 找到 v ($1 \leq v \leq \text{ord}_p(\text{seed})$) 满足 $\text{seed}^v \equiv u \pmod{p}$
- 令 $\text{ord}_p(\text{seed}) = 2^e \cdot r$ ，求出 $\text{ord}_r(2)$ ，并设定阈值 Q
 - 当 $n \leq Q$ 时，枚举 $1 \leq x \leq e + \text{ord}_r(2)$ 检查，模值在模 n 意义下或
可视为随机分布，期望复杂度 $\mathcal{O}(Q)$
 - 当 $n > Q$ 时， u 有不超过 $\frac{p}{Q}$ 种取值，枚举 u 求解 v ，再求解 x ，
期望复杂度 $\mathcal{O}(T + \frac{p}{Q}(\log p + \frac{p}{T}))$ ，取 $T = \mathcal{O}(\frac{p}{\sqrt{Q}})$
 - 为均衡两种情况的复杂度，取 $Q = \mathcal{O}(p^{\frac{2}{3}})$

■ 回到本题

- 找到 u ($0 \leq u < p$) 满足 $u \equiv k \pmod{n}$
- 找到 v ($1 \leq v \leq \text{ord}_p(\text{seed})$) 满足 $\text{seed}^v \equiv u \pmod{p}$
- 令 $\text{ord}_p(\text{seed}) = 2^e \cdot r$ ，求出 $\text{ord}_r(2)$ ，并设定阈值 Q
 - 当 $n \leq Q$ 时，枚举 $1 \leq x \leq e + \text{ord}_r(2)$ 检查，模值在模 n 意义下或可视为随机分布，期望复杂度 $\mathcal{O}(Q)$
 - 当 $n > Q$ 时， u 有不超过 $\frac{p}{Q}$ 种取值，枚举 u 求解 v ，再求解 x ，期望复杂度 $\mathcal{O}(T + \frac{p}{Q}(\log p + \frac{p}{T}))$ ，取 $T = \mathcal{O}(\frac{p}{\sqrt{Q}})$
 - 为均衡两种情况的复杂度，取 $Q = \mathcal{O}(p^{\frac{2}{3}})$

■ 回到本题

- 找到 u ($0 \leq u < p$) 满足 $u \equiv k \pmod{n}$
- 找到 v ($1 \leq v \leq \text{ord}_p(\text{seed})$) 满足 $\text{seed}^v \equiv u \pmod{p}$
- 令 $\text{ord}_p(\text{seed}) = 2^e \cdot r$ ，求出 $\text{ord}_r(2)$ ，并设定阈值 Q
 - 当 $n \leq Q$ 时，枚举 $1 \leq x \leq e + \text{ord}_r(2)$ 检查，模值在模 n 意义下或可视为随机分布，期望复杂度 $\mathcal{O}(Q)$
 - 当 $n > Q$ 时， u 有不超过 $\frac{p}{Q}$ 种取值，枚举 u 求解 v ，再求解 x ，期望复杂度 $\mathcal{O}(T + \frac{p}{Q}(\log p + \frac{p}{T}))$ ，取 $T = \mathcal{O}(\frac{p}{\sqrt{Q}})$
 - 为均衡两种情况的复杂度，取 $Q = \mathcal{O}(p^{\frac{2}{3}})$

■ 回到本题

- 找到 u ($0 \leq u < p$) 满足 $u \equiv k \pmod{n}$
- 找到 v ($1 \leq v \leq \text{ord}_p(\text{seed})$) 满足 $\text{seed}^v \equiv u \pmod{p}$
- 令 $\text{ord}_p(\text{seed}) = 2^e \cdot r$, 求出 $\text{ord}_r(2)$, 并设定阈值 Q
 - 当 $n \leq Q$ 时, 枚举 $1 \leq x \leq e + \text{ord}_r(2)$ 检查, 模值在模 n 意义下或可视为随机分布, 期望复杂度 $\mathcal{O}(Q)$
 - 当 $n > Q$ 时, u 有不超过 $\frac{p}{Q}$ 种取值, 枚举 u 求解 v , 再求解 x , 期望复杂度 $\mathcal{O}(T + \frac{p}{Q}(\log p + \frac{p}{T}))$, 取 $T = \mathcal{O}(\frac{p}{\sqrt{Q}})$
 - 为均衡两种情况的复杂度, 取 $Q = \mathcal{O}(p^{\frac{2}{3}})$

- 给定整数 B, C 和 M , 求解满足方程 $A^B \equiv C \pmod{M}$ 且 $A \leq M$ 的所有非负整数解 A , 无解输出 No Solution
- 保证解的数量不超过 \sqrt{M}
- $1 \leq B, C < M \leq 10^9$

- 给定整数 B, C 和 M , 求解满足方程 $A^B \equiv C \pmod{M}$ 且 $A \leq M$ 的所有非负整数解 A , 无解输出 No Solution
- 保证解的数量不超过 \sqrt{M}
- $1 \leq B, C < M \leq 10^9$

- 给定整数 B, C 和 M , 求解满足方程 $A^B \equiv C \pmod{M}$ 且 $A \leq M$ 的所有非负整数解 A , 无解输出 No Solution
- 保证解的数量不超过 \sqrt{M}
- $1 \leq B, C < M \leq 10^9$

■ 解方程 $A^B \equiv C \pmod{M}$, 已知 B, C, M 求 A

■ 高次剩余问题

- M 有原根时问题会好办许多, 考虑 M 是质数幂次的情况, 然后利用中国剩余定理合并
- $M = 2^e$ 时没有原根, 需要完善做法

- 解方程 $A^B \equiv C \pmod{M}$, 已知 B, C, M 求 A
 - 高次剩余问题
 - M 有原根时问题会好办许多, 考虑 M 是质数幂次的情况, 然后利用中国剩余定理合并
 - $M = 2^e$ 时没有原根, 需要完善做法

- 解方程 $A^B \equiv C \pmod{M}$, 已知 B, C, M 求 A
 - 高次剩余问题
 - M 有原根时问题会好办许多, 考虑 M 是质数幂次的情况, 然后利用中国剩余定理合并
 - $M = 2^e$ 时没有原根, 需要完善做法

■ 解高次剩余 $A^B \equiv C \pmod{M}$, $M = p^e$, p 是奇质数

- 若 $C \equiv 0 \pmod{M}$, 则 $x = p^u v$ ($\gcd(p, v) = 1$) 满足 $uB \geq e$ 即可,

即 $p^{\lceil \frac{e}{B} \rceil} \mid x$

- 若 $\gcd(C, M) = 1$, 可以取一原根 g 将问题转化为

$\text{Bind}_{M,g}(x) \equiv \text{ind}_{M,g}(C) \pmod{\varphi(M)}$, 消公因子后检查是否有解,
有解则利用扩展欧几里得算法求出通解即可

- 若 $1 < \gcd(C, M) < M$, 令 $C = p^a b$ ($\gcd(a, b) = 1$), 那么 $B \mid a$,
 $p^{\frac{a}{B}} \mid x$, 消因子后转化为 $\gcd(C, M) = 1$ 的情况, 转化回来时需要
扩张解的所在域 (例子)

- 解高次剩余 $A^B \equiv C \pmod{M}$, $M = p^e$, p 是奇质数
 - 若 $C \equiv 0 \pmod{M}$, 则 $x = p^u v$ ($\gcd(p, v) = 1$) 满足 $uB \geq e$ 即可, 即 $p^{\lceil \frac{e}{B} \rceil} \mid x$
 - 若 $\gcd(C, M) = 1$, 可以取一原根 g 将问题转化为 $\text{Bind}_{M,g}(x) \equiv \text{ind}_{M,g}(C) \pmod{\varphi(M)}$, 消公因子后检查是否有解, 有解则利用扩展欧几里得算法求出通解即可
 - 若 $1 < \gcd(C, M) < M$, 令 $C = p^a b$ ($\gcd(a, b) = 1$), 那么 $B \mid a$, $p^{\frac{a}{B}} \mid x$, 消因子后转化为 $\gcd(C, M) = 1$ 的情况, 转化回来时需要扩张解的所在域 (例子)

■ 解高次剩余 $A^B \equiv C \pmod{M}$, $M = p^e$, p 是奇质数

- 若 $C \equiv 0 \pmod{M}$, 则 $x = p^u v$ ($\gcd(p, v) = 1$) 满足 $uB \geq e$ 即可,

$$\text{即 } p^{\lceil \frac{e}{B} \rceil} \mid x$$

- 若 $\gcd(C, M) = 1$, 可以取一原根 g 将问题转化为

$$\text{Bind}_{M,g}(x) \equiv \text{ind}_{M,g}(C) \pmod{\varphi(M)}, \text{ 消公因子后检查是否有解,}$$

有解则利用扩展欧几里得算法求出通解即可

- 若 $1 < \gcd(C, M) < M$, 令 $C = p^a b$ ($\gcd(a, b) = 1$), 那么 $B \mid a$,

$$p^{\frac{a}{B}} \mid x, \text{ 消因子后转化为 } \gcd(C, M) = 1 \text{ 的情况, 转化回来时需要}$$

扩张解的所在域 (例子)

- 解高次剩余 $A^B \equiv C \pmod{M}$, $M = 2^e$
 - 当 $e > 2$ 时缩系可以表示成两个循环群的直和 $C_2 \times C_{2^{e-2}}$, 然而没有办法使用中国剩余定理
 - 看到解的数量不超过 \sqrt{M} , 一个暴力的想法是生成出所有的解
 - 如果有 $A^B \equiv C \pmod{2^e}$, 那么一定有 $A^B \equiv C \pmod{2^{e-1}}$
 - 假设已知 $x^B \equiv C \pmod{2^{e-1}}$, 那么可能有 $x^B \equiv C \pmod{2^e}$ 或 $(x + 2^{e-1})^B \equiv C \pmod{2^e}$, 利用这个必要条件进行 BFS 即可

- 解高次剩余 $A^B \equiv C \pmod{M}$, $M = 2^e$
 - 当 $e > 2$ 时缩系可以表示成两个循环群的直和 $C_2 \times C_{2^{e-2}}$, 然而没有办法使用中国剩余定理
 - 看到解的数量不超过 \sqrt{M} , 一个暴力的想法是生成出所有的解
 - 如果有 $A^B \equiv C \pmod{2^e}$, 那么一定有 $A^B \equiv C \pmod{2^{e-1}}$
 - 假设已知 $x^B \equiv C \pmod{2^{e-1}}$, 那么可能有 $x^B \equiv C \pmod{2^e}$ 或 $(x + 2^{e-1})^B \equiv C \pmod{2^e}$, 利用这个必要条件进行 BFS 即可

- 解高次剩余 $A^B \equiv C \pmod{M}$, $M = 2^e$
 - 当 $e > 2$ 时缩系可以表示成两个循环群的直和 $C_2 \times C_{2^{e-2}}$, 然而没有办法使用中国剩余定理
 - 看到解的数量不超过 \sqrt{M} , 一个暴力的想法是生成出所有的解
 - 如果有 $A^B \equiv C \pmod{2^e}$, 那么一定有 $A^B \equiv C \pmod{2^{e-1}}$
 - 假设已知 $x^B \equiv C \pmod{2^{e-1}}$, 那么可能有 $x^B \equiv C \pmod{2^e}$ 或 $(x + 2^{e-1})^B \equiv C \pmod{2^e}$, 利用这个必要条件进行 BFS 即可

- 解高次剩余 $A^B \equiv C \pmod{M}$, $M = 2^e$
 - 当 $e > 2$ 时缩系可以表示成两个循环群的直和 $C_2 \times C_{2^{e-2}}$, 然而没有办法使用中国剩余定理
 - 看到解的数量不超过 \sqrt{M} , 一个暴力的想法是生成出所有的解
 - 如果有 $A^B \equiv C \pmod{2^e}$, 那么一定有 $A^B \equiv C \pmod{2^{e-1}}$
 - 假设已知 $x^B \equiv C \pmod{2^{e-1}}$, 那么可能有 $x^B \equiv C \pmod{2^e}$ 或 $(x + 2^{e-1})^B \equiv C \pmod{2^e}$, 利用这个必要条件进行 BFS 即可
 - 由于模 2^e 意义的特殊性, 这个方法是可以通过的, 直到有一天昨天我又翻了一遍《数论讲义》……

- 解高次剩余 $A^B \equiv C \pmod{M}$, $M = 2^e$
 - 当 $e > 2$ 时缩系可以表示成两个循环群的直和 $C_2 \times C_{2^{e-2}}$, 然而没有办法使用中国剩余定理
 - 看到解的数量不超过 \sqrt{M} , 一个暴力的想法是生成出所有的解
 - 如果有 $A^B \equiv C \pmod{2^e}$, 那么一定有 $A^B \equiv C \pmod{2^{e-1}}$
 - 假设已知 $x^B \equiv C \pmod{2^{e-1}}$, 那么可能有 $x^B \equiv C \pmod{2^e}$ 或 $(x + 2^{e-1})^B \equiv C \pmod{2^e}$, 利用这个必要条件进行 BFS 即可
 - 由于模 2^e 意义的特殊性, 这个方法是可以通过的, 直到有一天昨天我又翻了一遍《数论讲义》……

$$A^B \bmod M$$

震惊!

模 2^e 意义也有“原根”!



加油，编的已经快像真的了

- 当 $e > 2$ 时, 可以归纳证明 $5^{2^{e-3}} \equiv 1 + 2^{e-1} \pmod{2^e}$, 从而得到 $\text{ord}_{2^e}(5) = 2^{e-2}$
- 这意味着 5 的幂次可以生成 2^{e-2} 个形如 $4k+1$ 的数字, 而缩系中恰好有 2^{e-2} 个形如 $4k+1$ 的数字
- 形如 $4k+1$ 的数字乘以 (-1) 即可生成剩下的 2^{e-2} 个与 2^e 互质的数字, 它们都是 $4k+3$ 的形式
- 对于 $\gcd(A, 2^e) = 1$ 的情况, 有 $A \equiv (-1)^{\frac{A-1}{2}} 5^u \pmod{2^e}$, 根据 B 的奇偶性讨论一下即可转化为离散对数问题, 不用受解数的限制

- 当 $e > 2$ 时, 可以归纳证明 $5^{2^{e-3}} \equiv 1 + 2^{e-1} \pmod{2^e}$, 从而得到 $\text{ord}_{2^e}(5) = 2^{e-2}$
- 这意味着 5 的幂次可以生成 2^{e-2} 个形如 $4k + 1$ 的数字, 而缩系中恰好有 2^{e-2} 个形如 $4k + 1$ 的数字
- 形如 $4k + 1$ 的数字乘以 (-1) 即可生成剩下的 2^{e-2} 个与 2^e 互质的数字, 它们都是 $4k + 3$ 的形式
- 对于 $\gcd(A, 2^e) = 1$ 的情况, 有 $A \equiv (-1)^{\frac{A-1}{2}} 5^u \pmod{2^e}$, 根据 B 的奇偶性讨论一下即可转化为离散对数问题, 不用受解数的限制

- 当 $e > 2$ 时, 可以归纳证明 $5^{2^{e-3}} \equiv 1 + 2^{e-1} \pmod{2^e}$, 从而得到 $\text{ord}_{2^e}(5) = 2^{e-2}$
- 这意味着 5 的幂次可以生成 2^{e-2} 个形如 $4k + 1$ 的数字, 而缩系中恰好有 2^{e-2} 个形如 $4k + 1$ 的数字
- 形如 $4k + 1$ 的数字乘以 (-1) 即可生成剩下的 2^{e-2} 个与 2^e 互质的数字, 它们都是 $4k + 3$ 的形式
- 对于 $\gcd(A, 2^e) = 1$ 的情况, 有 $A \equiv (-1)^{\frac{A-1}{2}} 5^u \pmod{2^e}$, 根据 B 的奇偶性讨论一下即可转化为离散对数问题, 不用受解数的限制

- 当 $e > 2$ 时, 可以归纳证明 $5^{2^{e-3}} \equiv 1 + 2^{e-1} \pmod{2^e}$, 从而得到 $\text{ord}_{2^e}(5) = 2^{e-2}$
- 这意味着 5 的幂次可以生成 2^{e-2} 个形如 $4k + 1$ 的数字, 而缩系中恰好有 2^{e-2} 个形如 $4k + 1$ 的数字
- 形如 $4k + 1$ 的数字乘以 (-1) 即可生成剩下的 2^{e-2} 个与 2^e 互质的数字, 它们都是 $4k + 3$ 的形式
- 对于 $\gcd(A, 2^e) = 1$ 的情况, 有 $A \equiv (-1)^{\frac{A-1}{2}} 5^u \pmod{2^e}$, 根据 B 的奇偶性讨论一下即可转化为离散对数问题, 不用受解数的限制

■ Summarize

- 掉线的同学可以准备重连了
- 大步小步算法是分块算法中的经典算法，使用时可以记住一点
一次使用，多次受用
- 通过原根可以将问题降低层次，或许会转化为更简单的问题，从这一点来看原根可以一定程度上代替单位复根
- 高次剩余问题是一个不比离散对数问题难的问题
- 扩张域的技巧有时很有用（循环探求、模意义贝尔数、模意义斐波那契数等）

■ Summarize

- 掉线的同学可以准备重连子 学习使我快乐
- 大步小步算法是分块算法中的经典算法，使用时可以记住一点
一次使用，多次受用
- 通过原根可以将问题降低层次，或许会转化为更简单的问题，从这一点来看原根可以一定程度上代替单位复根
- 高次剩余问题是一个不比离散对数问题难的问题
- 扩张域的技巧有时很有用（循环探求、模意义贝尔数、模意义斐波那契数等）

■ Summarize

- 掉线的同学可以准备重连子 学习使我快乐
- 大步小步算法是分块算法中的经典算法，使用时可以记住一点
一次使用，多次受用
- 通过原根可以将问题降低层次，或许会转化为更简单的问题，从这一点来看原根可以一定程度上代替单位复根
- 高次剩余问题是一个不比离散对数问题难的问题
- 扩张域的技巧有时很有用（循环探求、模意义贝尔数、模意义斐波那契数等）

■ Summarize

- 掉线的同学可以准备重连子 学习使我快乐
- 大步小步算法是分块算法中的经典算法，使用时可以记住一点
一次使用，多次受用
- 通过原根可以将问题降低层次，或许会转化为更简单的问题，从这一点来看原根可以一定程度上代替单位复根
- 高次剩余问题是一个不比离散对数问题难的问题
- 扩张域的技巧有时很有用（循环探求、模意义贝尔数、模意义斐波那契数等）

■ Summarize

- 掉线的同学可以准备重连子 学习使我快乐
- 大步小步算法是分块算法中的经典算法，使用时可以记住一点
一次使用，多次受用
- 通过原根可以将问题降低层次，或许会转化为更简单的问题，从这一点来看原根可以一定程度上代替单位复根
- 高次剩余问题是一个不比离散对数问题难的问题
- 扩张域的技巧有时很有用（循环探求、模意义贝尔数、模意义斐波那契数等）

■ 容斥原理基础版

- 在某个全集 U 上有 n 个集合 A_1, A_2, \dots, A_n , 那么

$$\left| \bigcup_{i=1}^n A_i \right| = \sum_{k=1}^n (-1)^{k-1} \sum_{1 \leq i_1 < i_2 < \dots < i_k \leq n} |A_{i_1} \cap A_{i_2} \cap \dots \cap A_{i_k}|$$

- 例如 $|A \cup B| = |A| + |B| - |A \cap B|$,

$$|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|$$

- 推论: 将交和并推广到偏序关系上也成立, 例如在某个全集 U 中有

n 个元素 x_1, x_2, \dots, x_n , 那么 $\max\{x_1, x_2, \dots, x_n\} =$

$$\sum_{k=1}^n (-1)^{k-1} \sum_{1 \leq i_1 < i_2 < \dots < i_k \leq n} \min\{x_{i_1}, x_{i_2}, \dots, x_{i_k}\}, \text{ 将 } \min, \max \text{ 互}$$

换也成立, 改成 \gcd, lcm 也成立

■ 容斥原理基础版

- 在某个全集 U 上有 n 个集合 A_1, A_2, \dots, A_n , 那么

$$\left| \bigcup_{i=1}^n A_i \right| = \sum_{k=1}^n (-1)^{k-1} \sum_{1 \leq i_1 < i_2 < \dots < i_k \leq n} |A_{i_1} \cap A_{i_2} \cap \dots \cap A_{i_k}|$$

- 例如 $|A \cup B| = |A| + |B| - |A \cap B|$,

$$|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|$$

- 推论: 将交和并推广到偏序关系上也成立, 例如在某个全集 U 中有

n 个元素 x_1, x_2, \dots, x_n , 那么 $\max\{x_1, x_2, \dots, x_n\} =$

$$\sum_{k=1}^n (-1)^{k-1} \sum_{1 \leq i_1 < i_2 < \dots < i_k \leq n} \min\{x_{i_1}, x_{i_2}, \dots, x_{i_k}\}, \text{ 将 } \min, \max \text{ 互}$$

换也成立, 改成 \gcd, lcm 也成立

■ 容斥原理基础版

- 在某个全集 U 上有 n 个集合 A_1, A_2, \dots, A_n , 那么

$$\left| \bigcup_{i=1}^n A_i \right| = \sum_{k=1}^n (-1)^{k-1} \sum_{1 \leq i_1 < i_2 < \dots < i_k \leq n} |A_{i_1} \cap A_{i_2} \cap \dots \cap A_{i_k}|$$

- 例如 $|A \cup B| = |A| + |B| - |A \cap B|$,

$$|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|$$

- 推论: 将交和并推广到偏序关系上也成立, 例如在某个全集 U 中有

n 个元素 x_1, x_2, \dots, x_n , 那么 $\max\{x_1, x_2, \dots, x_n\} =$

$$\sum_{k=1}^n (-1)^{k-1} \sum_{1 \leq i_1 < i_2 < \dots < i_k \leq n} \min\{x_{i_1}, x_{i_2}, \dots, x_{i_k}\}, \text{ 将 } \min, \max \text{ 互}$$

换也成立, 改成 \gcd, lcm 也成立

■ 容斥原理基础版

- 在某个全集 U 上有 n 个集合 A_1, A_2, \dots, A_n , 那么

$$\left| \bigcup_{i=1}^n A_i \right| = \sum_{k=1}^n (-1)^{k-1} \sum_{1 \leq i_1 < i_2 < \dots < i_k \leq n} |A_{i_1} \cap A_{i_2} \cap \dots \cap A_{i_k}|$$

- 例如 $|A \cup B| = |A| + |B| - |A \cap B|$,

$$|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|$$

- 推论: 将交和并推广到偏序关系上也成立, 例如在某个全集 U 中有

n 个元素 x_1, x_2, \dots, x_n , 那么 $\max\{x_1, x_2, \dots, x_n\} =$

$$\sum_{k=1}^n (-1)^{k-1} \sum_{1 \leq i_1 < i_2 < \dots < i_k \leq n} \min\{x_{i_1}, x_{i_2}, \dots, x_{i_k}\} , \text{ 将 } \min, \max \text{ 互}$$

换也成立, 改成 \gcd, lcm 也成立

■ 容斥原理进阶版

- 有 n 个属性集合 U_1, U_2, \dots, U_n ，和 n 个特征子集 A_1, A_2, \dots, A_n ，满足 $A_i \subseteq U_i$ ($i = 1, 2, \dots, n$)，需要对每个属性都满足特征时进行一些计数，而满足特征的较难计数，满足属性全集和满足特征的补集较易计数，则有

$$\sum_{x_1, x_2, \dots, x_n} [x_1 \in A_1][x_2 \in A_2] \cdots [x_n \in A_n] \text{way}(x_1, x_2, \dots, x_n) = \sum_{x_1} ([x_1 \in U_1] - [x_1 \in (U_1 - A_1)]) \sum_{x_2} ([x_2 \in U_2] - [x_2 \in (U_2 - A_2)]) \cdots \sum_{x_n} ([x_n \in U_n] - [x_n \in (U_n - A_n)]) \text{way}(x_1, x_2, \dots, x_n)$$

- $\frac{\varphi(m)}{m} = \prod_{p|m, p \text{ is prime}} (1 - \frac{1}{p})$, $\text{Stirling2}(n, k) = \frac{1}{k!} \sum_{r=0}^k (-1)^{k-r} \binom{k}{r} r^n$

■ 容斥原理进阶版

- 有 n 个属性集合 U_1, U_2, \dots, U_n ，和 n 个特征子集 A_1, A_2, \dots, A_n ，满足 $A_i \subseteq U_i$ ($i = 1, 2, \dots, n$)，需要对每个属性都满足特征时进行一些计数，而满足特征的较难计数，满足属性全集和满足特征的补集较易计数，则有

$$\sum_{x_1, x_2, \dots, x_n} [x_1 \in A_1][x_2 \in A_2] \cdots [x_n \in A_n] \text{way}(x_1, x_2, \dots, x_n) = \sum_{x_1} ([x_1 \in U_1] - [x_1 \in (U_1 - A_1)]) \sum_{x_2} ([x_2 \in U_2] - [x_2 \in (U_2 - A_2)]) \cdots \sum_{x_n} ([x_n \in U_n] - [x_n \in (U_n - A_n)]) \text{way}(x_1, x_2, \dots, x_n)$$

- $\frac{\varphi(m)}{m} = \prod_{p|m, p \text{ is prime}} (1 - \frac{1}{p})$, $\text{Stirling2}(n, k) = \frac{1}{k!} \sum_{r=0}^k (-1)^{k-r} \binom{k}{r} r^n$

■ 容斥原理进阶版

- 有 n 个属性集合 U_1, U_2, \dots, U_n ，和 n 个特征子集 A_1, A_2, \dots, A_n ，满足 $A_i \subseteq U_i$ ($i = 1, 2, \dots, n$)，需要对每个属性都满足特征时进行一些计数，而满足特征的较难计数，满足属性全集和满足特征的补集较易计数，则有

$$\sum_{x_1, x_2, \dots, x_n} [x_1 \in A_1][x_2 \in A_2] \cdots [x_n \in A_n] \text{way}(x_1, x_2, \dots, x_n) = \sum_{x_1} ([x_1 \in U_1] - [x_1 \in (U_1 - A_1)]) \sum_{x_2} ([x_2 \in U_2] - [x_2 \in (U_2 - A_2)]) \cdots \sum_{x_n} ([x_n \in U_n] - [x_n \in (U_n - A_n)]) \text{way}(x_1, x_2, \dots, x_n)$$

- $\frac{\varphi(m)}{m} = \prod_{p|m, p \text{ is prime}} (1 - \frac{1}{p})$, $\text{Stirling2}(n, k) = \frac{1}{k!} \sum_{r=0}^k (-1)^{k-r} \binom{k}{r} r^n$

■ 二项式系数

- 二项式定理: $(x + y)^n = \sum_{r=0}^n \binom{n}{r} x^r y^{n-r}$, 其中

$$\binom{n}{r} = \frac{n!}{r!(n-r)!} = \prod_{i=1}^r \frac{n+1-i}{i}$$

- 从 n 个元素的无序集合里选出包含 r 个元素的子集的方案数为 $\binom{n}{r}$
- 隔板法: 长度为 n 的序列拆成 r 段非空序列的方案数为 $\binom{n-1}{r-1}$
- $x_1 + x_2 + \cdots + x_r = n, x_i \in \mathbb{N}^+ (i = 1, 2, \cdots, r)$ 的解数为 $\binom{n-1}{r-1}$
- $x_1 + x_2 + \cdots + x_r = n, x_i \in \mathbb{N} (i = 1, 2, \cdots, r)$ 的解数为 $\binom{n+r-1}{r-1}$

■ 二项式系数

- 二项式定理: $(x + y)^n = \sum_{r=0}^n \binom{n}{r} x^r y^{n-r}$, 其中

$$\binom{n}{r} = \frac{n!}{r!(n-r)!} = \prod_{i=1}^r \frac{n+1-i}{i}$$

- 从 n 个元素的无序集合里选出包含 r 个元素的子集的方案数为 $\binom{n}{r}$
- 隔板法: 长度为 n 的序列拆成 r 段非空序列的方案数为 $\binom{n-1}{r-1}$
- $x_1 + x_2 + \cdots + x_r = n, x_i \in \mathbb{N}^+ (i = 1, 2, \cdots, r)$ 的解数为 $\binom{n-1}{r-1}$
- $x_1 + x_2 + \cdots + x_r = n, x_i \in \mathbb{N} (i = 1, 2, \cdots, r)$ 的解数为 $\binom{n+r-1}{r-1}$

■ 二项式系数

- 二项式定理: $(x + y)^n = \sum_{r=0}^n \binom{n}{r} x^r y^{n-r}$, 其中

$$\binom{n}{r} = \frac{n!}{r!(n-r)!} = \prod_{i=1}^r \frac{n+1-i}{i}$$

- 从 n 个元素的无序集合里选出包含 r 个元素的子集的方案数为 $\binom{n}{r}$
- 隔板法: 长度为 n 的序列拆成 r 段非空序列的方案数为 $\binom{n-1}{r-1}$
- $x_1 + x_2 + \cdots + x_r = n, x_i \in \mathbb{N}^+ (i = 1, 2, \cdots, r)$ 的解数为 $\binom{n-1}{r-1}$
- $x_1 + x_2 + \cdots + x_r = n, x_i \in \mathbb{N} (i = 1, 2, \cdots, r)$ 的解数为 $\binom{n+r-1}{r-1}$

■ 二项式系数

- 二项式定理: $(x + y)^n = \sum_{r=0}^n \binom{n}{r} x^r y^{n-r}$, 其中

$$\binom{n}{r} = \frac{n!}{r!(n-r)!} = \prod_{i=1}^r \frac{n+1-i}{i}$$

- 从 n 个元素的无序集合里选出包含 r 个元素的子集的方案数为 $\binom{n}{r}$
- 隔板法: 长度为 n 的序列拆成 r 段非空序列的方案数为 $\binom{n-1}{r-1}$
- $x_1 + x_2 + \cdots + x_r = n, x_i \in \mathbb{N}^+ (i = 1, 2, \cdots, r)$ 的解数为 $\binom{n-1}{r-1}$
- $x_1 + x_2 + \cdots + x_r = n, x_i \in \mathbb{N} (i = 1, 2, \cdots, r)$ 的解数为 $\binom{n+r-1}{r-1}$

■ 二项式系数

- 二项式定理: $(x + y)^n = \sum_{r=0}^n \binom{n}{r} x^r y^{n-r}$, 其中

$$\binom{n}{r} = \frac{n!}{r!(n-r)!} = \prod_{i=1}^r \frac{n+1-i}{i}$$

- 从 n 个元素的无序集合里选出包含 r 个元素的子集的方案数为 $\binom{n}{r}$
- 隔板法: 长度为 n 的序列拆成 r 段非空序列的方案数为 $\binom{n-1}{r-1}$
- $x_1 + x_2 + \cdots + x_r = n, x_i \in \mathbb{N}^+ (i = 1, 2, \cdots, r)$ 的解数为 $\binom{n-1}{r-1}$
- $x_1 + x_2 + \cdots + x_r = n, x_i \in \mathbb{N} (i = 1, 2, \cdots, r)$ 的解数为 $\binom{n+r-1}{r-1}$

■ 二项式系数

- 二项式定理: $(x + y)^n = \sum_{r=0}^n \binom{n}{r} x^r y^{n-r}$, 其中

$$\binom{n}{r} = \frac{n!}{r!(n-r)!} = \prod_{i=1}^r \frac{n+1-i}{i}$$

- 从 n 个元素的无序集合里选出包含 r 个元素的子集的方案数为 $\binom{n}{r}$
- 隔板法: 长度为 n 的序列拆成 r 段非空序列的方案数为 $\binom{n-1}{r-1}$
- $x_1 + x_2 + \cdots + x_r = n, x_i \in \mathbb{N}^+ (i = 1, 2, \cdots, r)$ 的解数为 $\binom{n-1}{r-1}$
- $x_1 + x_2 + \cdots + x_r = n, x_i \in \mathbb{N} (i = 1, 2, \cdots, r)$ 的解数为 $\binom{n+r-1}{r-1}$

- 有 $k_1 + k_2$ 个区间 $[L_i, R_i]$ ，在第 i 个区间里选出一个整数

x_i ($i = 1, 2, \dots, k_1 + k_2$)，令 $S = \sum_{i=1}^{k_1+k_2} x_i$ ，求 $S < 0$,

$S = 0, S > 0$ 的概率

- 假设概率可以表示成最简分数 $\frac{A}{B}$ ，则需要找到整数 C 满足

$0 \leq C < M$ 且 $A \equiv BC \pmod{M}$ ，然后给出 C 而不是 A

和 B ，其中 $M = 10^9 + 7$

- $1 \leq k_1, k_2 \leq 8, -10^7 \leq L_i \leq R_i \leq 10^7$ ($i = 1, 2, \dots, k_1 + k_2$)

- 有 $k_1 + k_2$ 个区间 $[L_i, R_i]$ ，在第 i 个区间里选出一个整数 x_i ($i = 1, 2, \dots, k_1 + k_2$)，令 $S = \sum_{i=1}^{k_1+k_2} x_i$ ，求 $S < 0$, $S = 0$, $S > 0$ 的概率
- 假设概率可以表示成最简分数 $\frac{A}{B}$ ，则需要找到整数 C 满足 $0 \leq C < M$ 且 $A \equiv BC \pmod{M}$ ，然后给出 C 而不是 A 和 B ，其中 $M = 10^9 + 7$
- $1 \leq k_1, k_2 \leq 8, -10^7 \leq L_i \leq R_i \leq 10^7$ ($i = 1, 2, \dots, k_1 + k_2$)

- 有 $k_1 + k_2$ 个区间 $[L_i, R_i]$ ，在第 i 个区间里选出一个整数 x_i ($i = 1, 2, \dots, k_1 + k_2$)，令 $S = \sum_{i=1}^{k_1+k_2} x_i$ ，求 $S < 0$, $S = 0$, $S > 0$ 的概率
- 假设概率可以表示成最简分数 $\frac{A}{B}$ ，则需要找到整数 C 满足 $0 \leq C < M$ 且 $A \equiv BC \pmod{M}$ ，然后给出 C 而不是 A 和 B ，其中 $M = 10^9 + 7$
- $1 \leq k_1, k_2 \leq 8, -10^7 \leq L_i \leq R_i \leq 10^7$ ($i = 1, 2, \dots, k_1 + k_2$)

- 总方案数为 $\prod_{i=1}^{k_1+k_2} R_i - L_i + 1$ ，在模 M 意义下非 0，只需要求出 $S < 0, S = 0, S > 0$ 的方案数即可求得概率
- $[L_i \leq x_i \leq R_i] = [x_i \geq L_i] - [x_i \geq R_i + 1]$
 $= [(x_i - L_i + 1) \in \mathbb{N}^+] - [(x_i - R_i) \in \mathbb{N}^+]$
- $S = 0$ 可以直接容斥转化为隔板问题， $S < 0$ 可以增加一个变元 $x_{k_1+k_2+1} \in \mathbb{N}^+$ 转化成 $S + x_{k_1+k_2+1} = 0$ 再容斥
- 最坏复杂度 $\mathcal{O}(2^{k_1+k_2+1}(k_1 + k_2 + 1))$ ，枚举二进制子集会得到最坏复杂度，利用 DFS 枚举可以剪枝掉不必要的情况

- 总方案数为 $\prod_{i=1}^{k_1+k_2} R_i - L_i + 1$ ，在模 M 意义下非 0，只需要求出 $S < 0, S = 0, S > 0$ 的方案数即可求得概率
- $[L_i \leq x_i \leq R_i] = [x_i \geq L_i] - [x_i \geq R_i + 1]$
 $= [(x_i - L_i + 1) \in \mathbb{N}^+] - [(x_i - R_i) \in \mathbb{N}^+]$
- $S = 0$ 可以直接容斥转化为隔板问题， $S < 0$ 可以增加一个变元 $x_{k_1+k_2+1} \in \mathbb{N}^+$ 转化成 $S + x_{k_1+k_2+1} = 0$ 再容斥
- 最坏复杂度 $\mathcal{O}(2^{k_1+k_2+1}(k_1 + k_2 + 1))$ ，枚举二进制子集会得到最坏复杂度，利用 DFS 枚举可以剪枝掉不必要的情况

- 总方案数为 $\prod_{i=1}^{k_1+k_2} R_i - L_i + 1$ ，在模 M 意义下非 0，只需要求出 $S < 0, S = 0, S > 0$ 的方案数即可求得概率
- $[L_i \leq x_i \leq R_i] = [x_i \geq L_i] - [x_i \geq R_i + 1]$
 $= [(x_i - L_i + 1) \in \mathbb{N}^+] - [(x_i - R_i) \in \mathbb{N}^+]$
- $S = 0$ 可以直接容斥转化为隔板问题， $S < 0$ 可以增加一个变元 $x_{k_1+k_2+1} \in \mathbb{N}^+$ 转化成 $S + x_{k_1+k_2+1} = 0$ 再容斥
- 最坏复杂度 $\mathcal{O}(2^{k_1+k_2+1}(k_1 + k_2 + 1))$ ，枚举二进制子集会得到最坏复杂度，利用 DFS 枚举可以剪枝掉不必要的情况

- 总方案数为 $\prod_{i=1}^{k_1+k_2} R_i - L_i + 1$ ，在模 M 意义下非 0，只需要求出 $S < 0, S = 0, S > 0$ 的方案数即可求得概率
- $[L_i \leq x_i \leq R_i] = [x_i \geq L_i] - [x_i \geq R_i + 1]$
 $= [(x_i - L_i + 1) \in \mathbb{N}^+] - [(x_i - R_i) \in \mathbb{N}^+]$
- $S = 0$ 可以直接容斥转化为隔板问题， $S < 0$ 可以增加一个变元 $x_{k_1+k_2+1} \in \mathbb{N}^+$ 转化成 $S + x_{k_1+k_2+1} = 0$ 再容斥
- 最坏复杂度 $\mathcal{O}(2^{k_1+k_2+1}(k_1 + k_2 + 1))$ ，枚举二进制子集会得到最坏复杂度，利用 DFS 枚举可以剪枝掉不必要的情况

- 有 n 个格子排成一行，每个格子可以有黑或白两种颜色，初始每个格子都是白色的
- 现在要进行 m 次涂绘，第 i 次涂绘有一个限制 a_i ($i = 1, 2, \dots, m$)，意味着这次要选出连续的恰好 a_i 个格子，不论它们之前的颜色如何，现在都要将它们涂成黑色
- 不同的操作可能涂绘出相同的局面，如果两个局面是不同的，那么必然存在一个格子在两种局面里被涂上的颜色不同，问经过 m 次涂绘后可能有多少种不同的局面
- $1 \leq n \leq 10^9, 1 \leq m \leq 4, 1 \leq a_i \leq n$ ($i = 1, 2, \dots, m$)

- 有 n 个格子排成一行，每个格子可以有黑或白两种颜色，初始每个格子都是白色的
- 现在要进行 m 次涂绘，第 i 次涂绘有一个限制 a_i ($i = 1, 2, \dots, m$)，意味着这次要选出连续的恰好 a_i 个格子，不论它们之前的颜色如何，现在都要将它们涂成黑色
- 不同的操作可能涂绘出相同的局面，如果两个局面是不同的，那么必然存在一个格子在两种局面里被涂上的颜色不同，问经过 m 次涂绘后可能有多少种不同的局面
- $1 \leq n \leq 10^9, 1 \leq m \leq 4, 1 \leq a_i \leq n$ ($i = 1, 2, \dots, m$)

- 有 n 个格子排成一行，每个格子可以有黑或白两种颜色，初始每个格子都是白色的
- 现在要进行 m 次涂绘，第 i 次涂绘有一个限制 a_i ($i = 1, 2, \dots, m$)，意味着这次要选出连续的恰好 a_i 个格子，不论它们之前的颜色如何，现在都要将它们涂成黑色
- 不同的操作可能涂绘出相同的局面，如果两个局面是不同的，那么必然存在一个格子在两种局面里被涂上的颜色不同，问经过 m 次涂绘后可能有多少种不同的局面

■ $1 \leq n \leq 10^9, 1 \leq m \leq 4, 1 \leq a_i \leq n$ ($i = 1, 2, \dots, m$)

- 有 n 个格子排成一行，每个格子可以有黑或白两种颜色，初始每个格子都是白色的
- 现在要进行 m 次涂绘，第 i 次涂绘有一个限制 a_i ($i = 1, 2, \dots, m$)，意味着这次要选出连续的恰好 a_i 个格子，不论它们之前的颜色如何，现在都要将它们涂成黑色
- 不同的操作可能涂绘出相同的局面，如果两个局面是不同的，那么必然存在一个格子在两种局面里被涂上的颜色不同，问经过 m 次涂绘后可能有多少种不同的局面
- $1 \leq n \leq 10^9, 1 \leq m \leq 4, 1 \leq a_i \leq n$ ($i = 1, 2, \dots, m$)

- 涂绘的结果必然是黑白相间的，黑色段的数量不超过 m
- 考虑 m 次操作之间的关系，操作的顺序不影响答案，操作的位置可能相交或不相交，而不相交时有先后关系
- 可能的情况有 $\sum_{k=1}^m \text{Stirling2}(m, k) k! \leq 75$ 种
- 受到第 i_1, i_2, \dots, i_k 次操作影响的黑段的长度有上下界限制
- 方案数与 $x_1 + x_2 + \dots + x_{2k+1} = n$ 的解数相同，其中
$$x_1, x_{2k+1} \in \mathbb{N}, x_{2i+1} \in \mathbb{N}^+ \quad (i = 1, 2, \dots, k-1),$$
$$x_{2i} \in [L_i, R_i] \quad (i = 1, 2, \dots, k)$$

- 涂绘的结果必然是黑白相间的，黑色段的数量不超过 m
- 考虑 m 次操作之间的关系，操作的顺序不影响答案，操作的位置可能相交或不相交，而不相交时有先后关系
- 可能的情况有 $\sum_{k=1}^m \text{Stirling2}(m, k)k! \leq 75$ 种
- 受到第 i_1, i_2, \dots, i_k 次操作影响的黑段的长度有上下界限制
- 方案数与 $x_1 + x_2 + \dots + x_{2k+1} = n$ 的解数相同，其中
$$x_1, x_{2k+1} \in \mathbb{N}, x_{2i+1} \in \mathbb{N}^+ \quad (i = 1, 2, \dots, k-1),$$
$$x_{2i} \in [L_i, R_i] \quad (i = 1, 2, \dots, k)$$

- 涂绘的结果必然是黑白相间的，黑色段的数量不超过 m
- 考虑 m 次操作之间的关系，操作的顺序不影响答案，操作的位置可能相交或不相交，而不相交时有先后关系
- 可能的情况有 $\sum_{k=1}^m \text{Stirling2}(m, k)k! \leq 75$ 种
- 受到第 i_1, i_2, \dots, i_k 次操作影响的黑段的长度有上下界限制
- 方案数与 $x_1 + x_2 + \dots + x_{2k+1} = n$ 的解数相同，其中
$$x_1, x_{2k+1} \in \mathbb{N}, x_{2i+1} \in \mathbb{N}^+ \quad (i = 1, 2, \dots, k-1),$$
$$x_{2i} \in [L_i, R_i] \quad (i = 1, 2, \dots, k)$$

- 涂绘的结果必然是黑白相间的，黑色段的数量不超过 m
- 考虑 m 次操作之间的关系，操作的顺序不影响答案，操作的位置可能相交或不相交，而不相交时有先后关系
- 可能的情况有 $\sum_{k=1}^m \text{Stirling2}(m, k)k! \leq 75$ 种
- 受到第 i_1, i_2, \dots, i_k 次操作影响的黑段的长度有上下界限制
- 方案数与 $x_1 + x_2 + \dots + x_{2k+1} = n$ 的解数相同，其中

$$x_1, x_{2k+1} \in \mathbb{N}, x_{2i+1} \in \mathbb{N}^+ \quad (i = 1, 2, \dots, k-1),$$

$$x_{2i} \in [L_i, R_i] \quad (i = 1, 2, \dots, k)$$

- 涂绘的结果必然是黑白相间的，黑色段的数量不超过 m
- 考虑 m 次操作之间的关系，操作的顺序不影响答案，操作的位置可能相交或不相交，而不相交时有先后关系
- 可能的情况有 $\sum_{k=1}^m \text{Stirling2}(m, k)k! \leq 75$ 种
- 受到第 i_1, i_2, \dots, i_k 次操作影响的黑段的长度有上下界限制
- 方案数与 $x_1 + x_2 + \dots + x_{2k+1} = n$ 的解数相同，其中

$$x_1, x_{2k+1} \in \mathbb{N}, x_{2i+1} \in \mathbb{N}^+ \quad (i = 1, 2, \dots, k-1),$$

$$x_{2i} \in [L_i, R_i] \quad (i = 1, 2, \dots, k)$$

- 75 种情况里有些情况可能有交集
- 需要统计的局面满足至少一种情况

- 75 种情况里有些情况可能有交集
- 需要统计的局面满足至少一种情况
- 外面套一层容斥

- 75 种情况里有些情况可能有交集
- 需要统计的局面满足至少一种情况
- 外面套一层容斥
- 最坏情况有 396 种有意义的集合, 例如 $a = \{2, 3, 4, 5\}$

- 75 种情况里有些情况可能有交集
- 需要统计的局面满足至少一种情况
- 外面套一层容斥
- 最坏情况有 396 种有意义的集合，例如 $a = \{2, 3, 4, 5\}$

■ Summarize

- 熟练使用 **容斥原理**，理解算反面的意义
- 涉及到排列组合时，注意组合意义与形式推导相结合
- 考虑情况需要面面俱到

■ Extension

- 错排公式
- 带标号连通图计数
- 禁位棋盘多项式

■ Summarize

- 熟练使用 **容斥原理**，理解算反面的意义
- 涉及到排列组合时，注意组合意义与形式推导相结合
- 考虑情况需要面面俱到

■ Extension

- 错排公式
- 带标号连通图计数
- 禁位棋盘多项式

■ Summarize

- 熟练使用 **容斥原理**，理解算反面的意义
- 涉及到排列组合时，注意组合意义与形式推导相结合
- 考虑情况需要面面俱到

■ Extension

- 错排公式
- 带标号连通图计数
- 禁位棋盘多项式

■ Summarize

- 熟练使用 **容斥原理**，理解算反面的意义
- 涉及到排列组合时，注意组合意义与形式推导相结合
- 考虑情况需要面面俱到

■ Extension

- 错排公式
- 带标号连通图计数
- 禁位棋盘多项式

■ Summarize

- 熟练使用 **容斥原理**，理解算反面的意义
- 涉及到排列组合时，注意组合意义与形式推导相结合
- 考虑情况需要面面俱到

■ Extension

- 错排公式
- 带标号连通图计数
- 禁位棋盘多项式

■ Summarize

- 熟练使用 **容斥原理**，理解算反面的意义
- 涉及到排列组合时，注意组合意义与形式推导相结合
- 考虑情况需要面面俱到

■ Extension

- 错排公式
- 带标号连通图计数
- 禁位棋盘多项式

- Feel free to ask any questions



感谢

感谢工作人员提供技术支持

感谢听课的各位的积极参与

祝大家在学习训练中有所收获，在比赛考试中旗开得胜

祝 51nod 越办越好

感谢

感谢工作人员提供技术支持

感谢听课的各位的积极参与

祝大家在学习训练中有所收获，在比赛考试中旗开得胜

祝 51nod 越办越好

感谢

感谢工作人员提供技术支持

感谢听课的各位的积极参与

祝大家在学习训练中有所收获，在比赛考试中旗开得胜

祝 51nod 越办越好

感谢

感谢工作人员提供技术支持

感谢听课的各位的积极参与

祝大家在学习训练中有所收获，在比赛考试中旗开得胜

祝 51nod 越办越好

感谢又善良，又仁慈，又有钱的夹克老爷

感谢

感谢工作人员提供技术支持

感谢听课的各位的积极参与

祝大家在学习训练中有所收获，在比赛考试中旗开得胜

祝 51nod 越办越好

感谢又善良，又仁慈，又有钱的夹克老爷全程防冷场