FFT 的运用

Claris

Hangzhou Dianzi University

2017 年 8 月 6 日

Overview

- 众所周知，多项式有两种表示方法：系数表示法与点值表示法。

Overview

- 众所周知，多项式有两种表示方法：系数表示法与点值表示法。
- 两个度数为 n 的以系数表示的多项式的乘法时间复杂度为 $O(n^2)$。

Overview

- 众所周知，多项式有两种表示方法：系数表示法与点值表示法。
- 两个度数为 n 的以系数表示的多项式的乘法时间复杂度为 $O(n^2)$。
- 两个度数为 n 的以点值表示的多项式的乘法时间复杂度为 $O(n)$。

Overview

- 众所周知，多项式有两种表示方法：系数表示法与点值表示法。
- 两个度数为 n 的以系数表示的多项式的乘法时间复杂度为 $O(n^2)$。
- 两个度数为 n 的以点值表示的多项式的乘法时间复杂度为 $O(n)$。
- 将多项式在这两种表示法之间转换可以使用快速傅里叶变换 (Fast Fourier Transform) 在 $O(n \log n)$ 的时间内完成。

线性翻转位序

- 在 FFT 中，第一步是翻转位序，一般方法实现起来都是 $O(n \log n)$。

线性翻转位序

- 在 FFT 中，第一步是翻转位序，一般方法实现起来都是 $O(n \log n)$。
- 有一个小优化：

线性翻转位序

- 在 FFT 中，第一步是翻转位序，一般方法实现起来都是 $O(n \log n)$。
- 有一个小优化：
- `j=__builtin_ctz(n)-1;`
  `for(i=0;i<n;i++)pos[i]=pos[i>>1]>>1|((i&1)<<j);`

线性翻转位序

- 在 FFT 中，第一步是翻转位序，一般方法实现起来都是 $O(n \log n)$。
- 有一个小优化：
- `j=__builtin_ctz(n)-1;`
  `for(i=0;i<n;i++)pos[i]=pos[i>>1]>>1|((i&1)<<j);`
- 时间复杂度 $O(n)$，可以减小 FFT 的常数。

高精度乘法

给定两个正整数 A, B ，求 $A \times B$。

高精度乘法

给定两个正整数 A, B，求 $A \times B$。

- $1 \leq A, B \leq 10^{1000000}$。

Solution

- 将 A 和 B 分别从低位到高位写，得到 $a_0, a_1, ..., a_k$ 与 $b_0, b_1, ..., b_k$。

Solution

- 将 A 和 B 分别从低位到高位写，得到 $a_0, a_1, ..., a_k$ 与 $b_0, b_1, ..., b_k$。
- 则 $ans_i = \sum_{j=0}^{i} a_j b_{i-j}$。

Solution

- 将 A 和 B 分别从低位到高位写，得到 $a_0, a_1, ..., a_k$ 与 $b_0, b_1, ..., b_k$。
- 则 $ans_i = \sum_{j=0}^{i} a_j b_{i-j}$。
- 利用 FFT 加速卷积。

Solution

- 将 A 和 B 分别从低位到高位写，得到 $a_0, a_1, ..., a_k$ 与 $b_0, b_1, ..., b_k$。
- 则 $ans_i = \sum_{j=0}^{i} a_j b_{i-j}$。
- 利用 FFT 加速卷积。
- 时间复杂度 $O(n \log n)$。

Solution

- 将 A 和 B 分别从低位到高位写，得到 $a_0, a_1, ..., a_k$ 与 $b_0, b_1, ..., b_k$。
- 则 $ans_i = \sum_{j=0}^{i} a_j b_{i-j}$。
- 利用 FFT 加速卷积。
- 时间复杂度 $O(n \log n)$。
- 优化？

Solution

- 将 A 和 B 分别从低位到高位写，得到 $a_0, a_1, \ldots, a_k$ 与 $b_0, b_1, \ldots, b_k$。
- 则 $ans_i = \sum_{j=0}^{i} a_j b_{i-j}$。
- 利用 FFT 加速卷积。
- 时间复杂度 $O(n \log n)$。
- 优化？
- 压位处理。时间复杂度 $O(\frac{n}{w} \log \frac{n}{w})$。

力

在数轴上有 n 个电荷，第 i 个电荷坐标为 i，电量为 q_i。

力

在数轴上有 n 个电荷，第 i 个电荷坐标为 i，电量为 q_i。

求每个电荷受到的场强大小。

力

在数轴上有 n 个电荷，第 i 个电荷坐标为 i，电量为 q_i。

求每个电荷受到的场强大小。

即求 $E_i = \sum_{j<i} \frac{q_j}{(i-j)^2} - \sum_{j>i} \frac{q_j}{(i-j)^2}$。

力

在数轴上有 n 个电荷，第 i 个电荷坐标为 i，电量为 q_i。

求每个电荷受到的场强大小。

即求 $E_i = \sum_{j<i} \frac{q_j}{(i-j)^2} - \sum_{j>i} \frac{q_j}{(i-j)^2}$。

- $1 \leq n \leq 100000$。

力

在数轴上有 n 个电荷，第 i 个电荷坐标为 i，电量为 q_i。

求每个电荷受到的场强大小。

即求 $E_i = \sum_{j<i} \frac{q_j}{(i-j)^2} - \sum_{j>i} \frac{q_j}{(i-j)^2}$。

- $1 \leq n \leq 100000$。
- $1 \leq q_i \leq 10^9$。

力

在数轴上有 n 个电荷，第 i 个电荷坐标为 i，电量为 q_i。

求每个电荷受到的场强大小。

即求 $E_i = \sum_{j<i} \frac{q_j}{(i-j)^2} - \sum_{j>i} \frac{q_j}{(i-j)^2}$。

- $1 \leq n \leq 100000$。
- $1 \leq q_i \leq 10^9$。
- Source： ZJOI 2014

Solution

- 设 $f_i = \sum_{j<i} \frac{q_j}{(i-j)^2}$ ，求出 f 后将 q 翻转即可得到另一半答案。

Solution

- 设 $f_i = \sum_{j<i} \frac{q_j}{(i-j)^2}$，求出 f 后将 q 翻转即可得到另一半答案。
- 构造 $a_0 = b_0 = 0, a_i = q_i, b_i = \frac{1}{i^2}$。

Solution

- 设 $f_i = \sum_{j<i} \frac{q_j}{(i-j)^2}$，求出 f 后将 q 翻转即可得到另一半答案。
- 构造 $a_0 = b_0 = 0, a_i = q_i, b_i = \frac{1}{i^2}$。
- 则 a 与 b 卷积 $c_i = \sum_{j=0}^{i} a_j b_{i-j} = \sum_{j=1}^{i-1} \frac{q_j}{(i-j)^2} = f_i$。

Solution

- 设 $f_i = \sum_{j<i} \frac{q_j}{(i-j)^2}$，求出 f 后将 q 翻转即可得到另一半答案。
- 构造 $a_0 = b_0 = 0, a_i = q_i, b_i = \frac{1}{i^2}$。
- 则 a 与 b 卷积 $c_i = \sum_{j=0}^{i} a_j b_{i-j} = \sum_{j=1}^{i-1} \frac{q_j}{(i-j)^2} = f_i$。
- FFT 求出 c 即可。

Solution

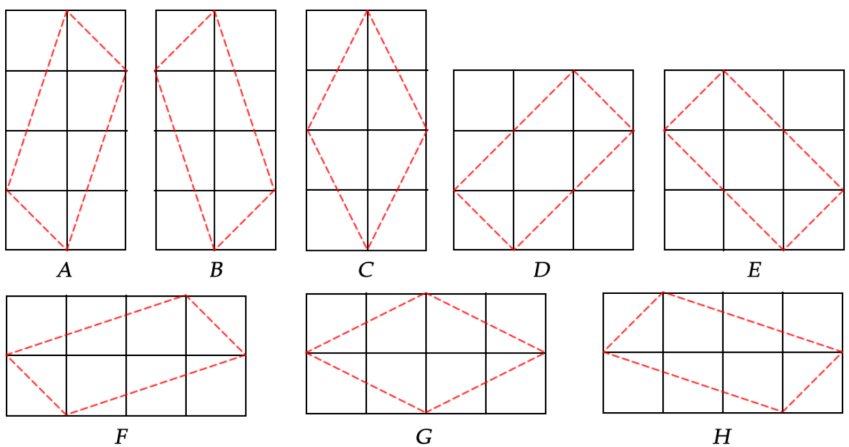
- 设 $f_i = \sum_{j<i} \frac{q_j}{(i-j)^2}$，求出 f 后将 q 翻转即可得到另一半答案。
- 构造 $a_0 = b_0 = 0, a_i = q_i, b_i = \frac{1}{i^2}$。
- 则 a 与 b 卷积 $c_i = \sum_{j=0}^{i} a_j b_{i-j} = \sum_{j=1}^{i-1} \frac{q_j}{(i-j)^2} = f_i$。
- FFT 求出 c 即可。
- 时间复杂度 $O(n \log n)$。

Tile Cutting

n 次询问，每次给定 S，问在面积为 $2S$ 的长宽均为整数的长方形方格纸四条边上 (不包括角落) 选 4 个点形成面积为 S 的平行四边形个数。

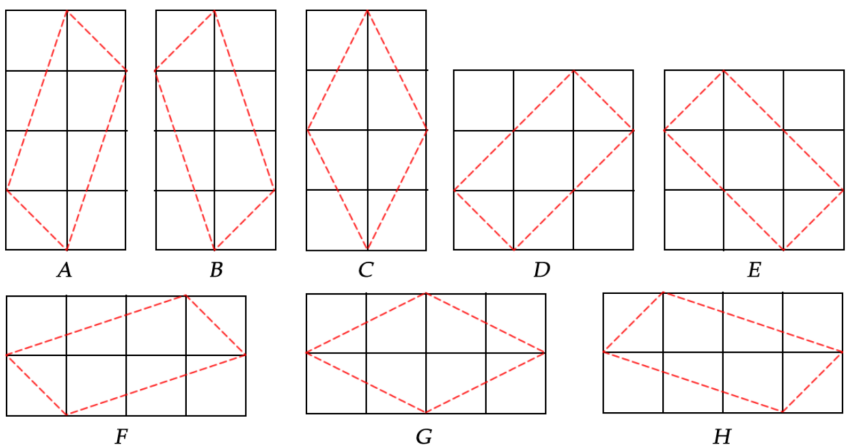
Tile Cutting

n 次询问，每次给定 S，问在面积为 $2S$ 的长宽均为整数的长方形方格纸四条边上 (不包括角落) 选 4 个点形成面积为 S 的平行四边形个数。



Tile Cutting

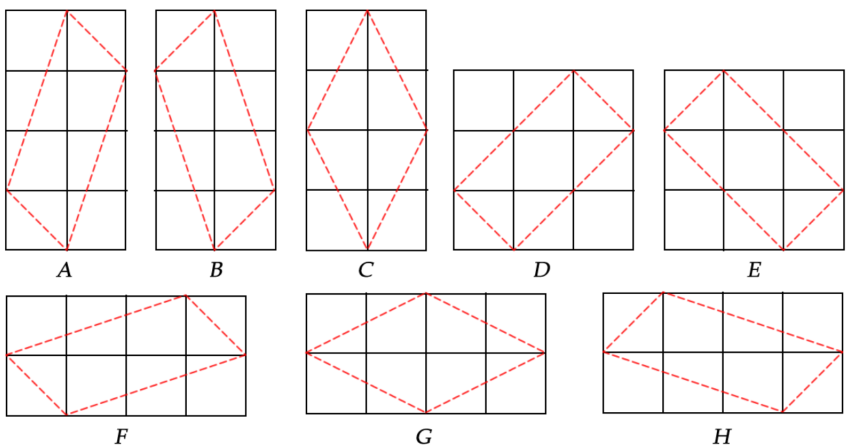
n 次询问，每次给定 S，问在面积为 $2S$ 的长宽均为整数的长方形方格纸四条边上 (不包括角落) 选 4 个点形成面积为 S 的平行四边形个数。



- $1 \leq n, S \leq 500000$。

Tile Cutting

n 次询问，每次给定 S，问在面积为 $2S$ 的长宽均为整数的长方形方格纸四条边上 (不包括角落) 选 4 个点形成面积为 S 的平行四边形个数。



- $1 \leq n, S \leq 500000$。
- Source： WF 2015

Solution

- 如果左下右 3 个顶点都确定，那么整个平行四边形就定了。

Solution

- 如果左下右 3 个顶点都确定，那么整个平行四边形就定了。
- 不妨设左顶点为 $(-a, b)$，下顶点为 $(0, 0)$，右顶点为 (c, d)，则 $S = ac + bd$。

Solution

- 如果左下右 3 个顶点都确定，那么整个平行四边形就定了。
- 不妨设左顶点为 $(-a, b)$，下顶点为 $(0, 0)$，右顶点为 (c, d)，则 $S = ac + bd$。
- $ans_i = \sum_{j=1}^{i-1} \sum_{a|j} \sum_{b|i-j} 1$。

Solution

- 如果左下右 3 个顶点都确定，那么整个平行四边形就定了。
- 不妨设左顶点为 $(-a, b)$，下顶点为 $(0, 0)$，右顶点为 (c, d)，则 $S = ac + bd$。
- $ans_i = \sum_{j=1}^{i-1} \sum_{a|j} \sum_{b|i-j} 1$。
- 令 $d_0 = 0, d_i$ 为 i 的约数个数，则 $ans_i = \sum_{j=0}^{i} d_j d_{i-j}$。

Solution

- 如果左下右 3 个顶点都确定，那么整个平行四边形就定了。
- 不妨设左顶点为 $(-a, b)$，下顶点为 $(0, 0)$，右顶点为 (c, d)，则 $S = ac + bd$。
- $ans_i = \sum_{j=1}^{i-1} \sum_{a|j} \sum_{b|i-j} 1$。
- 令 $d_0 = 0, d_i$ 为 i 的约数个数，则 $ans_i = \sum_{j=0}^{i} d_j d_{i-j}$。
- FFT 求出 ans 即可。

Solution

- 如果左下右 3 个顶点都确定，那么整个平行四边形就定了。
- 不妨设左顶点为 $(-a, b)$，下顶点为 $(0, 0)$，右顶点为 (c, d)，则 $S = ac + bd$。
- $ans_i = \sum_{j=1}^{i-1} \sum_{a|j} \sum_{b|i-j} 1$。
- 令 $d_0 = 0, d_i$ 为 i 的约数个数，则 $ans_i = \sum_{j=0}^{i} d_j d_{i-j}$。
- FFT 求出 ans 即可。
- 时间复杂度 $O(n \log n)$。

3-idiots

给定 n 根长度分别为 $a_1, a_2, \ldots, a_n$ 的木棒，问随机选择 3 根木棒能够拼成三角形的概率。

3-idiots

给定 n 根长度分别为 $a_1, a_2, \ldots, a_n$ 的木棒，问随机选择 3 根木棒能够拼成三角形的概率。

- $3 \leq n \leq 100000$。

3-idiots

给定 n 根长度分别为 $a_1, a_2, \ldots, a_n$ 的木棒，问随机选择 3 根木棒能够拼成三角形的概率。

- $3 \leq n \leq 100000$。
- $1 \leq a_i \leq 100000$。

3-idiots

给定 n 根长度分别为 $a_1, a_2, \ldots, a_n$ 的木棒，问随机选择 3 根木棒能够拼成三角形的概率。

- $3 \leq n \leq 100000$。
- $1 \leq a_i \leq 100000$。
- Source：HDU 4609

Solution

- 假设 $a_i \leq a_j \leq a_k$，则它们能拼成三角形当且仅当 $a_i + a_j > a_k$。

Solution

- 假设 $a_i \leq a_j \leq a_k$，则它们能拼成三角形当且仅当 $a_i + a_j > a_k$。
- 考虑反面：不能拼成三角形当且仅当 $a_i + a_j \leq a_k$。

Solution

- 假设 $a_i \leq a_j \leq a_k$，则它们能拼成三角形当且仅当 $a_i + a_j > a_k$。
- 考虑反面：不能拼成三角形当且仅当 $a_i + a_j \leq a_k$。
- 枚举 a_k，计算 $a_i + a_j \leq a_k$ 的方案数。

Solution

- 假设 $a_i \leq a_j \leq a_k$，则它们能拼成三角形当且仅当 $a_i + a_j > a_k$。
- 考虑反面：不能拼成三角形当且仅当 $a_i + a_j \leq a_k$。
- 枚举 a_k，计算 $a_i + a_j \leq a_k$ 的方案数。
- 显然的卷积形式，FFT 求解。

Solution

- 假设 $a_i \le a_j \le a_k$，则它们能拼成三角形当且仅当 $a_i + a_j > a_k$。
- 考虑反面：不能拼成三角形当且仅当 $a_i + a_j \le a_k$。
- 枚举 a_k，计算 $a_i + a_j \le a_k$ 的方案数。
- 显然的卷积形式，FFT 求解。
- 时间复杂度 $O(a \log a)$。

组队活动

ACM 校队一共有 n 名队员，从 1 到 n 标号，现在 n 名队员要组成若干支队伍，每支队伍至多有 m 名队员。

组队活动

ACM 校队一共有 n 名队员，从 1 到 n 标号，现在 n 名队员要组成若干支队伍，每支队伍至多有 m 名队员。

你需要计算出一共有多少种不同的组队方案，方案数模 998244353。

组队活动

ACM 校队一共有 n 名队员，从 1 到 n 标号，现在 n 名队员要组成若干支队伍，每支队伍至多有 m 名队员。

你需要计算出一共有多少种不同的组队方案，方案数模 998244353。

- $1 \leq n \leq 100000$。

组队活动

ACM 校队一共有 n 名队员，从 1 到 n 标号，现在 n 名队员要组成若干支队伍，每支队伍至多有 m 名队员。

你需要计算出一共有多少种不同的组队方案，方案数模 998244353。

- $1 \leq n \leq 100000$。
- $1 \leq m \leq 100000$。

组队活动

ACM 校队一共有 n 名队员，从 1 到 n 标号，现在 n 名队员要组成若干支队伍，每支队伍至多有 m 名队员。

你需要计算出一共有多少种不同的组队方案，方案数模 998244353。

- $1 \leq n \leq 100000$。
- $1 \leq m \leq 100000$。
- Source： quailty's Contest #1

Solution

- 设 f_i 表示 i 个人自由分组的方案数，初始值 $f_0 = 1$。

Solution

- 设 f_i 表示 i 个人自由分组的方案数，初始值 $f_0 = 1$。
- 考虑枚举第 i 个人所在队伍剩下的人数，假设有 j 人，那么有 $f_{i-j-1}C(i-1,j)$ 种方案，所以

$$\begin{aligned} f_i &= \sum_{j=0}^{\min(i,m)-1} f_{i-j-1}C(i-1,j) \\ &= (i-1)! \sum_{j=0}^{\min(i,m)-1} \frac{f_{i-j-1}}{(i-j-1)!} \times \frac{1}{j!} \end{aligned}$$

Solution

- 设 f_i 表示 i 个人自由分组的方案数，初始值 $f_0 = 1$。
- 考虑枚举第 i 个人所在队伍剩下的人数，假设有 j 人，那么有 $f_{i-j-1}C(i-1,j)$ 种方案，所以

$$\begin{aligned} f_i &= \sum_{j=0}^{\min(i,m)-1} f_{i-j-1}C(i-1,j) \\ &= (i-1)! \sum_{j=0}^{\min(i,m)-1} \frac{f_{i-j-1}}{(i-j-1)!} \times \frac{1}{j!} \end{aligned}$$

- 这个转移显然是个卷积的形式，用分治 NTT 即可做到 $O(n\log^2 n)$。

FFT 模任意数

- NTT 的模数只能是费马素数，具有很大的局限性。

FFT 模任意数

- NTT 的模数只能是费马素数，具有很大的局限性。
- 对于任意模数 P，一般 P 在 int 范围内。

FFT 模任意数

- NTT 的模数只能是费马素数，具有很大的局限性。
- 对于任意模数 P，一般 P 在 int 范围内。
- 设 $k = \lceil \sqrt{P} \rceil$，则 $P-1$ 以内每个数 x 都可以表示为 $ak+b$ 的形式，其中 $a, b \leq \sqrt{P}$。

FFT 模任意数

- NTT 的模数只能是费马素数，具有很大的局限性。
- 对于任意模数 P，一般 P 在 int 范围内。
- 设 $k = \lceil \sqrt{P} \rceil$，则 $P-1$ 以内每个数 x 都可以表示为 $ak+b$ 的形式，其中 $a, b \leq \sqrt{P}$。
- $(ak+b)(ck+d) = ack^2 + (ad+bc)k + bd$，其中 $ac, ad+bc, bd$ 的每一项都不超过 nP，可以用 double 或者 long double 存储。

FFT 模任意数

- NTT 的模数只能是费马素数，具有很大的局限性。
- 对于任意模数 P，一般 P 在 int 范围内。
- 设 $k = \lceil \sqrt{P} \rceil$，则 $P-1$ 以内每个数 x 都可以表示为 $ak+b$ 的形式，其中 $a, b \leq \sqrt{P}$。
- $(ak+b)(ck+d) = ack^2 + (ad+bc)k + bd$，其中 $ac, ad+bc, bd$ 的每一项都不超过 nP，可以用 double 或者 long double 存储。
- 因此普通的 FFT 就可以完成这 3 部分的计算，最后将 3 部分拼起来即可得到答案。

He is Flying

给定一个长度为 n 的正整数序列 $a_1, a_2, \ldots, a_n$ ，总和为 s。

He is Flying

给定一个长度为 n 的正整数序列 $a_1, a_2, \ldots, a_n$ ，总和为 s。

请对于 1 到 s 的每个数 k 统计：有多少区间 $[l, r]$ 的和恰好为 k。

He is Flying

给定一个长度为 n 的正整数序列 $a_1, a_2, \ldots, a_n$ ，总和为 s。

请对于 1 到 s 的每个数 k 统计：有多少区间 $[l, r]$ 的和恰好为 k。

- $1 \leq n \leq 100000$。

He is Flying

给定一个长度为 n 的正整数序列 $a_1, a_2, \ldots, a_n$ ，总和为 s。

请对于 1 到 s 的每个数 k 统计：有多少区间 $[l, r]$ 的和恰好为 k。

- $1 \leq n \leq 100000$。
- $1 \leq s \leq 50000$。

He is Flying

给定一个长度为 n 的正整数序列 $a_1, a_2, \ldots, a_n$ ，总和为 s。

请对于 1 到 s 的每个数 k 统计：有多少区间 $[l, r]$ 的和恰好为 k。

- $1 \leq n \leq 100000$。
- $1 \leq s \leq 50000$。
- Source： HDU 5307

Solution

- 设 p 为 a 的前缀和，c_i 为 p 中等于 i 的个数。

Solution

- 设 p 为 a 的前缀和，c_i 为 p 中等于 i 的个数。
- 则 $ans_i = \sum_{j=i}^{s} c_j c_{j-i}$。

Solution

- 设 p 为 a 的前缀和，c_i 为 p 中等于 i 的个数。
- 则 $ans_i = \sum_{j=i}^{s} c_j c_{j-i}$。
- 设 $d_i = c_{s-i}$，则 c 与 d 的卷积
  $f_{i+s} = \sum_{j=0}^{i+s} c_j d_{i+s-j} = \sum_{j=0}^{i+s} c_j c_{j-i} = ans_i$。

Solution

- 设 p 为 a 的前缀和，c_i 为 p 中等于 i 的个数。
- 则 $ans_i = \sum_{j=i}^{s} c_j c_{j-i}$。
- 设 $d_i = c_{s-i}$，则 c 与 d 的卷积
  $f_{i+s} = \sum_{j=0}^{i+s} c_j d_{i+s-j} = \sum_{j=0}^{i+s} c_j c_{j-i} = ans_i$。
- 时间复杂度 $O(s \log s)$。

Rusty String

字符串 T 是字符串 S 的循环节，当且仅当 T 重复拼接次后恰好为 S。

Rusty String

字符串 T 是字符串 S 的循环节，当且仅当 T 重复拼接次后恰好为 S。

给定一个长度为 n 的字符串 S，每个位置要么为 V，要么为 K，要么还没确定，但只能填 V 或 K。

Rusty String

字符串 T 是字符串 S 的循环节，当且仅当 T 重复拼接次后恰好为 S。

给定一个长度为 n 的字符串 S，每个位置要么为 V，要么为 K，要么还没确定，但只能填 V 或 K。

请找出所有可能是 S 循环节的字符串 T 的长度。

Rusty String

字符串 T 是字符串 S 的循环节，当且仅当 T 重复拼接次后恰好为 S。

给定一个长度为 n 的字符串 S，每个位置要么为 V，要么为 K，要么还没确定，但只能填 V 或 K。

请找出所有可能是 S 循环节的字符串 T 的长度。

- $1 \leq n \leq 500000$。

Rusty String

字符串 T 是字符串 S 的循环节，当且仅当 T 重复拼接次后恰好为 S。

给定一个长度为 n 的字符串 S，每个位置要么为 V，要么为 K，要么还没确定，但只能填 V 或 K。

请找出所有可能是 S 循环节的字符串 T 的长度。

- $1 \leq n \leq 500000$。
- Source： Codeforces Round #423 Div. 1 E

Solution

- 若循环节长度为 k，那么所有关于 k 同余的位置中不能既出现 V，又出现 K。

Solution

- 若循环节长度为 k，那么所有关于 k 同余的位置中不能既出现 V，又出现 K。
- 假如有一对 V, K 的距离为 t，那么所有 $d|t$ 作为循环节都是非法的。

Solution

- 若循环节长度为 k，那么所有关于 k 同余的位置中不能既出现 V，又出现 K。
- 假如有一对 V, K 的距离为 t，那么所有 $d|t$ 作为循环节都是非法的。
- 问题转化为找出所有可能的 t，FFT 求解即可。

Solution

- 若循环节长度为 k，那么所有关于 k 同余的位置中不能既出现 V，又出现 K。
- 假如有一对 V, K 的距离为 t，那么所有 $d|t$ 作为循环节都是非法的。
- 问题转化为找出所有可能的 t，FFT 求解即可。
- 时间复杂度 $O(n \log n)$。

Eades

给定一个序列 $a_1, a_2, \ldots, a_n$，对于 1 到 n 的每个 k，统计区间最大值个数恰好为 k 的区间个数。

Eades

给定一个序列 $a_1, a_2, \ldots, a_n$，对于 1 到 n 的每个 k，统计区间最大值个数恰好为 k 的区间个数。

- $1 \leq n \leq 60000$。

Eades

给定一个序列 $a_1, a_2, \ldots, a_n$，对于 1 到 n 的每个 k，统计区间最大值个数恰好为 k 的区间个数。

- $1 \leq n \leq 60000$。
- $1 \leq a_i \leq n$。

Eades

给定一个序列 $a_1, a_2, ..., a_n$，对于 1 到 n 的每个 k，统计区间最大值个数恰好为 k 的区间个数。

- $1 \leq n \leq 60000$。
- $1 \leq a_i \leq n$。
- Source：HDU 5751

Solution

- 枚举每个 i，通过单调栈求出 a_i 作为最大值 (不要求唯一) 的区间 $[l_i, r_i]$。

Solution

- 枚举每个 i，通过单调栈求出 a_i 作为最大值 (不要求唯一) 的区间 $[l_i, r_i]$。
- 枚举每个值作为最大值，将对应的位置按照 $[l_i, r_i]$ 分组，可以得到若干个连续段，每个连续段相互独立。

Solution

- 枚举每个 i，通过单调栈求出 a_i 作为最大值 (不要求唯一) 的区间 $[l_i, r_i]$。
- 枚举每个值作为最大值，将对应的位置按照 $[l_i, r_i]$ 分组，可以得到若干个连续段，每个连续段相互独立。
- 对于每个连续段，枚举最终选定的区间落在哪两个位置之间，那么最大值个数等于两个位置的差值，FFT 求解即可。

Solution

- 枚举每个 i，通过单调栈求出 a_i 作为最大值 (不要求唯一) 的区间 $[l_i, r_i]$。
- 枚举每个值作为最大值，将对应的位置按照 $[l_i, r_i]$ 分组，可以得到若干个连续段，每个连续段相互独立。
- 对于每个连续段，枚举最终选定的区间落在哪两个位置之间，那么最大值个数等于两个位置的差值，FFT 求解即可。
- 考虑一个数，若出现了 $size$ 次，则时间复杂度为 $O(size \log size)$。

Solution

- 枚举每个 i，通过单调栈求出 a_i 作为最大值 (不要求唯一) 的区间 $[l_i, r_i]$。
- 枚举每个值作为最大值，将对应的位置按照 $[l_i, r_i]$ 分组，可以得到若干个连续段，每个连续段相互独立。
- 对于每个连续段，枚举最终选定的区间落在哪两个位置之间，那么最大值个数等于两个位置的差值，FFT 求解即可。
- 考虑一个数，若出现了 $size$ 次，则时间复杂度为 $O(size \log size)$。
- 故总时间复杂度为 $O(n \log n)$。

反函数

给定一棵有 n 个节点的无根树，每个节点是一个括号。

反函数

给定一棵有 n 个节点的无根树，每个节点是一个括号。

定义 $S(x, y)$ 为从 x 开始沿着最短路走到 y，将沿途经过的点上的字符依次连起来得到的字符串。

反函数

给定一棵有 n 个节点的无根树，每个节点是一个括号。

定义 $S(x, y)$ 为从 x 开始沿着最短路走到 y，将沿途经过的点上的字符依次连起来得到的字符串。

函数 $f(x, y)$ 等于对 $S(x, y)$ 进行划分，使得每一个部分都是平衡的括号串，能得到的最大的段数。

反函数

给定一棵有 n 个节点的无根树，每个节点是一个括号。

定义 $S(x, y)$ 为从 x 开始沿着最短路走到 y，将沿途经过的点上的字符依次连起来得到的字符串。

函数 $f(x, y)$ 等于对 $S(x, y)$ 进行划分，使得每一个部分都是平衡的括号串，能得到的最大的段数。

m 次询问，每次输入一个 k，查询有多少点对的 f 值为 k。

反函数

给定一棵有 n 个节点的无根树，每个节点是一个括号。

定义 $S(x, y)$ 为从 x 开始沿着最短路走到 y，将沿途经过的点上的字符依次连起来得到的字符串。

函数 $f(x, y)$ 等于对 $S(x, y)$ 进行划分，使得每一个部分都是平衡的括号串，能得到的最大的段数。

m 次询问，每次输入一个 k，查询有多少点对的 f 值为 k。

- $1 \leq n \leq 50000$。

反函数

给定一棵有 n 个节点的无根树，每个节点是一个括号。

定义 $S(x, y)$ 为从 x 开始沿着最短路走到 y，将沿途经过的点上的字符依次连起来得到的字符串。

函数 $f(x, y)$ 等于对 $S(x, y)$ 进行划分，使得每一个部分都是平衡的括号串，能得到的最大的段数。

m 次询问，每次输入一个 k，查询有多少点对的 f 值为 k。

- $1 \leq n \leq 50000$。
- $1 \leq m \leq 25000$。

反函数

给定一棵有 n 个节点的无根树，每个节点是一个括号。

定义 $S(x, y)$ 为从 x 开始沿着最短路走到 y，将沿途经过的点上的字符依次连起来得到的字符串。

函数 $f(x, y)$ 等于对 $S(x, y)$ 进行划分，使得每一个部分都是平衡的括号串，能得到的最大的段数。

m 次询问，每次输入一个 k，查询有多少点对的 f 值为 k。

- $1 \leq n \leq 50000$。
- $1 \leq m \leq 25000$。
- Source： BZOJ 2016 NOI 十连测第五场

Solution

- 考虑如何快速回答单个询问。

Solution

- 考虑如何快速回答单个询问。
- 对这棵树进行树分治，求出重心到每个点的前缀和 s。

Solution

- 考虑如何快速回答单个询问。
- 对这棵树进行树分治，求出重心到每个点的前缀和 s。
- 对于两个点 i, j，假设是从 i 开始走到 j，那么它们的 s 互为相反数，且 i 的 s 是它到重心路径上最大的，j 的 s 则是最小的。

Solution

- 同时维护一下最值出现的个数，即可得到将两条链拼起来的链的 f 值。

Solution

- 同时维护一下最值出现的个数，即可得到将两条链拼起来的链的 f 值。
- 将所有点按 s 桶排序，对于每种 s 值单独计算，因为 k 固定，所以知道了某一项，另一项也知道，直接计数即可。

Solution

- 同时维护一下最值出现的个数，即可得到将两条链拼起来的链的 f 值。
- 将所有点按 s 桶排序，对于每种 s 值单独计算，因为 k 固定，所以知道了某一项，另一项也知道，直接计数即可。
- 需要根据 s 是不是 0 分两种情况讨论，细节比较多。

Solution

- 同时维护一下最值出现的个数，即可得到将两条链拼起来的链的 f 值。
- 将所有点按 s 桶排序，对于每种 s 值单独计算，因为 k 固定，所以知道了某一项，另一项也知道，直接计数即可。
- 需要根据 s 是不是 0 分两种情况讨论，细节比较多。
- 时间复杂度 $O(mn \log n)$。

Solution

- 考虑用一次树分治直接预处理出每个询问的答案。

Solution

- 考虑用一次树分治直接预处理出每个询问的答案。
- 注意到上一个算法里最后的计数实际上可以转化成两个多项式的卷积，因此用 FFT 优化这个过程即可。

Solution

- 考虑用一次树分治直接预处理出每个询问的答案。
- 注意到上一个算法里最后的计数实际上可以转化成两个多项式的卷积，因此用 FFT 优化这个过程即可。
- 对于每次计算，假设一共有 $size$ 个点，对于某个 s，有 t 个点，那么处理这个 s 的时候，多项式的次数上界不超过 t，因此每层分治的复杂度为 $O(size \log size)$。

Solution

- 考虑用一次树分治直接预处理出每个询问的答案。
- 注意到上一个算法里最后的计数实际上可以转化成两个多项式的卷积，因此用 FFT 优化这个过程即可。
- 对于每次计算，假设一共有 $size$ 个点，对于某个 s，有 t 个点，那么处理这个 s 的时候，多项式的次数上界不超过 t，因此每层分治的复杂度为 $O(size \log size)$。
- 时间复杂度 $O(n \log^2 n)$。

残缺的字符串

很久很久以前，在你刚刚学习字符串匹配的时候，有两个仅包含小写字母的字符串 A 和 B，其中 A 串长度为 m，B 串长度为 n。

残缺的字符串

很久很久以前，在你刚刚学习字符串匹配的时候，有两个仅包含小写字母的字符串 A 和 B，其中 A 串长度为 m，B 串长度为 n。

当你现在再次碰到这两个串时，这两个串已经老化了，每个串都有不同程度的残缺，用 * 表示。

残缺的字符串

很久很久以前，在你刚刚学习字符串匹配的时候，有两个仅包含小写字母的字符串 A 和 B，其中 A 串长度为 m，B 串长度为 n。

当你现在再次碰到这两个串时，这两个串已经老化了，每个串都有不同程度的残缺，用 * 表示。

请回答，对于 B 的每一个位置 i，从这个位置开始连续 m 个字符形成的子串是否可能与 A 串完全匹配。

残缺的字符串

很久很久以前，在你刚刚学习字符串匹配的时候，有两个仅包含小写字母的字符串 A 和 B，其中 A 串长度为 m，B 串长度为 n。

当你现在再次碰到这两个串时，这两个串已经老化了，每个串都有不同程度的残缺，用 * 表示。

请回答，对于 B 的每一个位置 i，从这个位置开始连续 m 个字符形成的子串是否可能与 A 串完全匹配。

例子：aa**b 可以匹配 a*cd*。

残缺的字符串

很久很久以前，在你刚刚学习字符串匹配的时候，有两个仅包含小写字母的字符串 A 和 B，其中 A 串长度为 m，B 串长度为 n。

当你现在再次碰到这两个串时，这两个串已经老化了，每个串都有不同程度的残缺，用 * 表示。

请回答，对于 B 的每一个位置 i，从这个位置开始连续 m 个字符形成的子串是否可能与 A 串完全匹配。

例子：aa**b 可以匹配 a*cd*。

- $1 \leq m \leq n \leq 300000$。

残缺的字符串

很久很久以前，在你刚刚学习字符串匹配的时候，有两个仅包含小写字母的字符串 A 和 B，其中 A 串长度为 m，B 串长度为 n。

当你现在再次碰到这两个串时，这两个串已经老化了，每个串都有不同程度的残缺，用 * 表示。

请回答，对于 B 的每一个位置 i，从这个位置开始连续 m 个字符形成的子串是否可能与 A 串完全匹配。

例子：aa**b 可以匹配 a*cd*。

- $1 \leq m \leq n \leq 300000$。
- Source： BZOJ 4259

Solution

- 假设字符串是从第 0 位开始的，那么对于两个长度都为 n 的字符串 A, B，定义距离函数：

$$dis(A,B) = \sum_{i=0}^{n-1} (A_i - B_i)^2 [A_i \neq' *'][B_i \neq' *']$$

Solution

- 假设字符串是从第 0 位开始的，那么对于两个长度都为 n 的字符串 A, B，定义距离函数：

$$dis(A,B) = \sum_{i=0}^{n-1} (A_i - B_i)^2 [A_i \neq' *'][B_i \neq' *']$$

- 若把 * 号都设置为 0，那么有：

$$dis(A,B) = \sum_{i=0}^{n-1} (A_i - B_i)^2 A_i B_i$$

Solution

- 假设字符串是从第 0 位开始的，那么对于两个长度都为 n 的字符串 A, B，定义距离函数：

$$dis(A,B) = \sum_{i=0}^{n-1} (A_i - B_i)^2 [A_i \neq' *'][B_i \neq' *']$$

- 若把 * 号都设置为 0，那么有：

$$dis(A,B) = \sum_{i=0}^{n-1} (A_i - B_i)^2 A_i B_i$$

- 如果 $dis(A,B) = 0$，那么 A 和 B 完全匹配。

Solution

- 枚举 B 的末尾位置 i，设 $f_i = dis(A, B[i-m+1, i])$，那么 B 的这一个子串与 A 完全匹配，有：

Solution

- 枚举 B 的末尾位置 i，设 $f_i = dis(A, B[i-m+1, i])$，那么 B 的这一个子串与 A 完全匹配，有：
- $$f_i = \sum_{j=0}^{m-1} (A_j - B_{i-m+1+j})^2 A_j B_{i-m+1+j} = 0$$

Solution

- 如果把 A 串翻转，并在后面不断补 0 直至和 B 串等长的话，那么有：

$$\begin{aligned} f_i &= \sum_{j=0}^{i} (A_j - B_{i-j})^2 A_j B_{i-j} \\ &= \sum_{j=0}^{i} (A_j^2 - 2A_j B_{i-j} + B_{i-j}^2) A_j B_{i-j} \\ &= \sum_{j=0}^{i} A_j^3 B_{i-j} - 2\sum_{j=0}^{i} A_j^2 B_{i-j}^2 + \sum_{j=0}^{i} A_j B_{i-j}^3 \end{aligned}$$

Solution

- 如果把 A 串翻转，并在后面不断补 0 直至和 B 串等长的话，那么有：

$$\begin{aligned} f_i &= \sum_{j=0}^{i} (A_j - B_{i-j})^2 A_j B_{i-j} \\ &= \sum_{j=0}^{i} (A_j^2 - 2A_j B_{i-j} + B_{i-j}^2) A_j B_{i-j} \\ &= \sum_{j=0}^{i} A_j^3 B_{i-j} - 2\sum_{j=0}^{i} A_j^2 B_{i-j}^2 + \sum_{j=0}^{i} A_j B_{i-j}^3 \end{aligned}$$

- 显然可以分成三段做 FFT 求出所有的 f_i，时间复杂度为 $O(n \log n)$。

航海舰队

给定一个 $n \times m$ 的网格图，有些点是障碍点。

航海舰队

给定一个 $n \times m$ 的网格图，有些点是障碍点。

地图上还有一支固定阵形的船队，问船队能扫过的格子数。

航海舰队

给定一个 $n \times m$ 的网格图，有些点是障碍点。

地图上还有一支固定阵形的船队，问船队能扫过的格子数。

- $1 \leq n \leq 700$。

航海舰队

给定一个 $n \times m$ 的网格图，有些点是障碍点。

地图上还有一支固定阵形的船队，问船队能扫过的格子数。

- $1 \leq n \leq 700$。
- $1 \leq m \leq 700$。

航海舰队

给定一个 $n \times m$ 的网格图，有些点是障碍点。

地图上还有一支固定阵形的船队，问船队能扫过的格子数。

- $1 \leq n \leq 700$。
- $1 \leq m \leq 700$。
- Source：BZOJ 2017 省选十连测第一场

Solution

- 首先抠出包围了阵形的最小矩形。

Solution

- 首先抠出包围了阵形的最小矩形。
- 将地图拉伸成一条链，即将第一行、第二行、第三行按顺序连接。阵形也可以用同样的方法处理。

Solution

- 首先抠出包围了阵形的最小矩形。
- 将地图拉伸成一条链，即将第一行、第二行、第三行按顺序连接。阵形也可以用同样的方法处理。
- 那么问题转化为，给定两个 01 串 S 和 T，问每个 S 中长度为 $|T|$ 的子串是否存在一个点，两个串对应字符都是 1。

Solution

- 首先抠出包围了阵形的最小矩形。
- 将地图拉伸成一条链，即将第一行、第二行、第三行按顺序连接。阵形也可以用同样的方法处理。
- 那么问题转化为，给定两个 01 串 S 和 T，问每个 S 中长度为 $|T|$ 的子串是否存在一个点，两个串对应字符都是 1。
- 将 T 串翻转，那么就变成了卷积的形式，FFT 计算即可。

Solution

- 首先抠出包围了阵形的最小矩形。
- 将地图拉伸成一条链，即将第一行、第二行、第三行按顺序连接。阵形也可以用同样的方法处理。
- 那么问题转化为，给定两个 01 串 S 和 T，问每个 S 中长度为 $|T|$ 的子串是否存在一个点，两个串对应字符都是 1。
- 将 T 串翻转，那么就变成了卷积的形式，FFT 计算即可。
- 在 BFS 求出所有可行的位置之后，对于答案的计算，也是卷积的形式，用 FFT 加速即可。

Solution

- 首先抠出包围了阵形的最小矩形。
- 将地图拉伸成一条链，即将第一行、第二行、第三行按顺序连接。阵形也可以用同样的方法处理。
- 那么问题转化为，给定两个 01 串 S 和 T，问每个 S 中长度为 $|T|$ 的子串是否存在一个点，两个串对应字符都是 1。
- 将 T 串翻转，那么就变成了卷积的形式，FFT 计算即可。
- 在 BFS 求出所有可行的位置之后，对于答案的计算，也是卷积的形式，用 FFT 加速即可。
- 时间复杂度 $O(nm\log(nm))$。

binomial

给定 n 和 p，对于所有 $0 \leq i < p$，求满足 $C(n, k) \bmod p = i$ 的 k 的个数。

binomial

给定 n 和 p，对于所有 $0 \leq i < p$，求满足 $C(n, k) \bmod p = i$ 的 k 的个数。

答案模 29。

binomial

给定 n 和 p，对于所有 $0 \leq i < p$，求满足 $C(n,k) \bmod p = i$ 的 k 的个数。

答案模 29。

- $1 \leq n \leq p^{10}$。

binomial

给定 n 和 p，对于所有 $0 \leq i < p$，求满足 $C(n, k) \bmod p = i$ 的 k 的个数。

答案模 29。

- $1 \leq n \leq p^{10}$。
- $p = 51061$，是一个质数。

binomial

给定 n 和 p，对于所有 $0 \leq i < p$，求满足 $C(n, k) \bmod p = i$ 的 k 的个数。

答案模 29。

- $1 \leq n \leq p^{10}$。
- $p = 51061$，是一个质数。
- Source：BZOJ 2629

Solution

- 根据 Lucas 定理，等价于在 p 进制下每一位分别求组合数后乘积模 p。

Solution

- 根据 Lucas 定理，等价于在 p 进制下每一位分别求组合数后乘积模 p。
- 因为答案为 0 的并不好算，所以可以考虑用 $n+1$ 减去其它所有的答案。

Solution

- 根据 Lucas 定理，等价于在 p 进制下每一位分别求组合数后乘积模 p。
- 因为答案为 0 的并不好算，所以可以考虑用 $n+1$ 减去其它所有的答案。
- 那么每一位的组合数都不能是 0，那么这就保证了 k 的每一位都不大于 n，所以无需考虑 $k \leq n$ 这个限制。

Solution

- 根据 Lucas 定理，等价于在 p 进制下每一位分别求组合数后乘积模 p。
- 因为答案为 0 的并不好算，所以可以考虑用 $n+1$ 减去其它所有的答案。
- 那么每一位的组合数都不能是 0，那么这就保证了 k 的每一位都不大于 n，所以无需考虑 $k \leq n$ 这个限制。
- 因为 p 是质数，所以存在原根，每个非 0 数都可以表示成原根的幂，可以将乘法看作指标相加。

Solution

- 求出模 p 下每个数的指标。

Solution

- 求出模 p 下每个数的指标。
- 考虑数位 DP，设 $f_{i,j}$ 表示考虑了最后 i 位，组合数的指标之和模 $\varphi(p)$ 为 j 的方案数。

Solution

- 求出模 p 下每个数的指标。
- 考虑数位 DP，设 $f_{i,j}$ 表示考虑了最后 i 位，组合数的指标之和模 $\varphi(p)$ 为 j 的方案数。
- 那么转移就是卷积的形式，FFT 加速即可。

Solution

- 求出模 p 下每个数的指标。
- 考虑数位 DP，设 $f_{i,j}$ 表示考虑了最后 i 位，组合数的指标之和模 $\varphi(p)$ 为 j 的方案数。
- 那么转移就是卷积的形式，FFT 加速即可。
- 时间复杂度 $O(p \log p \log n)$。

课后习题

- Triple Sums (SPOJ TSUM)

课后习题

- Triple Sums (SPOJ TSUM)
- Xavier is Learning to Count (HDU 4093)

课后习题

- Triple Sums (SPOJ TSUM)
- Xavier is Learning to Count (HDU 4093)
- Shell Necklace (HDU 5730)

课后习题

- Triple Sums (SPOJ TSUM)
- Xavier is Learning to Count (HDU 4093)
- Shell Necklace (HDU 5730)
- 二元运算 (BZOJ 4836)

课后习题

- Triple Sums (SPOJ TSUM)
- Xavier is Learning to Count (HDU 4093)
- Shell Necklace (HDU 5730)
- 二元运算 (BZOJ 4836)
- Fuzzy Search (Codeforces #296 Div. 1 D)

课后习题

- Triple Sums (SPOJ TSUM)
- Xavier is Learning to Count (HDU 4093)
- Shell Necklace (HDU 5730)
- 二元运算 (BZOJ 4836)
- Fuzzy Search (Codeforces #296 Div. 1 D)
- Forest Game (2016-2017 National Taiwan University World Final Team Selection Contest)

课后习题

- Triple Sums (SPOJ TSUM)
- Xavier is Learning to Count (HDU 4093)
- Shell Necklace (HDU 5730)
- 二元运算 (BZOJ 4836)
- Fuzzy Search (Codeforces #296 Div. 1 D)
- Forest Game (2016-2017 National Taiwan University World Final Team Selection Contest)
- XM Reserves (HDU 5885)

课后习题

- Triple Sums (SPOJ TSUM)
- Xavier is Learning to Count (HDU 4093)
- Shell Necklace (HDU 5730)
- 二元运算 (BZOJ 4836)
- Fuzzy Search (Codeforces #296 Div. 1 D)
- Forest Game (2016-2017 National Taiwan University World Final Team Selection Contest)
- XM Reserves (HDU 5885)
- [SDOI2015] 序列统计 (BZOJ 3992)

Thank you!