最近一年 THU 命题组出过的数学题选讲

n+e

Tsinghua University

2017 年 7 月 9 日

- 给定 N, M，和 M 条边连接 x_i 和 y_i 节点。求用剩下的边连成一个连通图有几种方法，要求使用边数最少。答案对 P 取模。
- $N, M \leq 10^5$

- 给定 N, M，和 M 条边连接 x_i 和 y_i 节点。求用剩下的边连成一个连通图有几种方法，要求使用边数最少。答案对 P 取模。
- $N, M \leq 10^5$

e.g. $N = 4, M = 1, 1 - 2$

- Answer=8

- 缩点！缩点！

- 缩点！缩点！
- 直接基尔霍夫矩阵？

- 缩点！缩点！
- 直接基尔霍夫矩阵？
- Prüfer 编码

- In order to add the least roads, we can build a spanning tree of each connected component.
- In Prüfer coding, we map a tree to a vector with $n-2$ elements whose elements are integers from 1 to n:
- First, we put all of the numbers from 1 to n which is not shown in the vector into a set A. Each time we pop the first element u of the vector and the element v with the smallest number in set A, build an edge between u and v.
- Then, if u is the last time shown in the vector, add u to set A. After we add $n-2$ edges, the vector will be empty and the set A will only contain 2 elements. Add the last edge between these two elements.

- Considering every element is a connected component instead of a vertex in this problem, we should determine one vertex in both connected components when we add an edge. For connected component from the vector, we have c_u choices when the element from the vector is u, where c_u is the number of vertices in connected component u. Also, we have c_v choices for v from set A.
- Because each element of the vector is arbitrary and each connected component will be added to set A exactly once, the answer for k connected components is

$$(\sum_{u=1}^{k} c_u)^{k-2} \prod_{v=1}^{k} c_v = n^{k-2} \prod_{v=1}^{k} c_v$$

- $m \times m$ 的 01 矩阵 A，其加法为按位异或。给定长度为 m 的 01 列向量 b，n 次询问 $A^{k_i}b$
- $n \leq 100, m \leq 1000, k \leq 10^9$

- 特征多项式的那套理论：

$$p(\lambda) = |\lambda I - A| = \lambda^m - k_1\lambda^{m-1} - \cdots - k_m\lambda^0$$

- 有：$p(A) = 0$，和 $A^k \equiv A^k \bmod p(A)$
- 所以求出 $A^0 \cdots A^m$ 即可。
- 国冬上 AKF 已经讲过了：求特征多项式是 $O(m^4)$ 的

- 随机一个叫做 c^T 的向量，使得 $\lambda c^T \neq c^T A$，然后式子就是

$$c^T A^m - w_1 c^T A^{m-1} - \cdots - w_m c^T A^0 = 0$$

使用高斯消元就能够解出所有的 w，这一步是 $O(m^3)$，总复杂度是 $O(m^3 + nm^2 \log k)$，乘上使用 bitset 之后的常数是可以通过的。

- 假设直线 L 和 L' 相交于原点 O。假设 $S=\{s_1,s_2,\ldots,s_n\}$ 是平面上的 n 个点。
- 你打算找四个点 A,B,A',B' 满足如下条件：
  1. $A\in L$ 而 $A'\in L'$。
  2. B,B' 都属于 S；即 $B\in S$ 且 $B'\in S$。
  3. A,A' 的中点与 B,B' 的中点重叠。这意味着 $ABA'B'$ 是一个平行四边形（或者退化的平行四边形）。
  4. 平行四边形 $ABA'B'$ 的面积最大。
- 直线方程为 $L:ax+by=0,L':a'x+b'y=0$，$n\le 10^6$

$O(n^2)$：固定 B 和 B'，$O(1)$ 求解

- 也就是说，当 B 和 B' 确定，A 和 A' 也随之确定。
- 怎么优化？

- 定义有向距离：

$$dist_L(P) = \frac{ax_P + by_P}{\sqrt{a^2 + b^2}}$$

$$dist_{L'}(P) = \frac{a'x_P + b'y_P}{\sqrt{a'^2 + b'^2}}$$

- 设 L 和 L' 的夹角为 θ

$$Area(\Diamond(B, B')) = \frac{|dist_L(B) \cdot dist_{L'}(B) - dist_L(B') \cdot dist_{L'}(B')|}{\sin\theta}$$

- for 一遍就没了。就没了。
- 还有这种操作.jpg

```
def f(x, y):
    c = 0
    while y > 0:
        c += 1
        t = x % y
        x = y
        y = t
    return c * x * x
```

- 给出三个正整数 n, m, p，你需要计算：

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\left\lfloor \frac{i \times j}{f(i, j)} \right\rfloor$$

对 p 取模的结果。$n, p \leq 666,666,666, m \leq 666$

- 这是在干啥？
- 直接分析：x 最后是输入两数的 gcd，c 是计算 gcd 所用的次数

- 这是在干啥？
- 直接分析：x 最后是输入两数的 gcd，c 是计算 gcd 所用的次数
- 设 $t(x,y)$ 表示计算 gcd 所用的次数，则

$$\left\lfloor \frac{i \cdot j}{f(i,j)} \right\rfloor = \left\lfloor \frac{\frac{i}{gcd(i,j)} \cdot \frac{j}{gcd(i,j)}}{t(i,j)} \right\rfloor$$

其中分子是整数

- 此外，已知了 j 和 $i \bmod j$ 就能确定 $gcd(i,j)$ 和 $t(i,j)$

- 欲求 $\sum_{i=1}^{n}\sum_{j=1}^{m}\left\lfloor \frac{\frac{i}{gcd(i,j)}\cdot\frac{j}{gcd(i,j)}}{t(i,j)} \right\rfloor$
- 枚举 j 和 $i \bmod j$，即可转化为求 $\sum_i \left\lfloor \frac{C_1\cdot i}{C_2} \right\rfloor$，其中 i 取某个等差数列，C_1, C_2 与 i 无关。
- 或者，设 $\left\lfloor \frac{i-1}{j} \right\rfloor = x$，就是求 $\sum_{x=0}^{C_3} \left\lfloor \frac{C_4x+C_5}{C_6} \right\rfloor$
- 这个式子可以使用经典方法，在 $O(\log n)$ 的时间求出
- 也可以发现，式子里的除数实际上就是 gcd 的次数，这个数是 $O(\log m)$ 的，故拆成这么多个等差数列暴力计算亦可
- 总时间复杂度 $O(m^2 \log m)$

- 有一种卡牌对战游戏。游戏总有 n 张卡牌。最开始小 W 和小 M 各会拥有其中的一部分。每一局游戏，小 W 和小 M 都需要从自己的卡牌中选一张进行对战，获胜的一方会获得双方选出的两张卡牌。游戏会一直进行下去，直到其中一个人获得了所有的卡牌。获得所有卡牌即赢得胜利。
- 对于一对特定的卡牌 i 和 j,i 战胜 j 的概率为 $P_{i,j}$ 。此概率与其他事件独立。每次对战一定会决出胜负，因此有 $P_{i,j}+P_{j,i}=1$。
- 小 W 和小 M 采取了同一套看起来合理的选牌方式:
  1. 对于自己的卡牌 i，计算出这张卡牌能赢得对方每张卡牌的概率之和 $S_i=\sum_{j\text{是对方的卡牌}} P_{i,j}$;
  2. 令自己选出卡牌 i 的概率正比于 S_i，即选出 i 的概率为 $S_i/\sum_{k\text{是自己的卡牌}} S_k$ 。
- 小 W 想知道，对于给出的 m 种初始状态，小 M 最终获胜的概率是多少。$n\le 15, m\le 2^n$

- 直接列式子。$f_0 = 0, f_{all} = 1$

$$f_S = \sum_{i \in S, j \notin S} \frac{Sm_i}{Sm_{tot}} * \frac{Sw_j}{Sw_{tot}} * (p_{i,j} * f_{S \cup j} + p_{j,i} * f_{S \setminus i})$$

- 直接 Gauss 会 T？

- 直接列式子。$f_0 = 0, f_{all} = 1$

$$f_S = \sum_{i \in S, j \notin S} \frac{Sm_i}{Sm_{tot}} * \frac{Sw_j}{Sw_{tot}} * (p_{i,j} * f_{S \cup j} + p_{j,i} * f_{S \setminus i})$$

- 直接 Gauss 会 T？
- 有个东西叫做马尔可夫链。随机设定初始值，直接迭代就能收敛。

- 你以为这样就能过？Naive!

返回试题列表

提交清单

提交编号	试题名称	提交时间	代码长度	编程语言	评测结果	得分	时间使用	空间使用
360627	卡牌游戏	06-24 22:29	1.307KB	C++	错误	95	2.89s	9.156MB
360625	卡牌游戏	06-24 22:27	1.434KB	C++	正确	100	2.921s	9.156MB
360623	卡牌游戏	06-24 22:25	1.403KB	C++	错误	95	2.906s	9.156MB
360621	卡牌游戏	06-24 22:25	1.403KB	C++	错误	95	2.75s	9.156MB
360619	卡牌游戏	06-24 22:24	1.403KB	C++	运行超时	95	运行超时	9.156MB
360616	卡牌游戏	06-24 22:19	1.335KB	C++	运行超时	95	运行超时	9.156MB
360615	卡牌游戏	06-24 22:18	1.335KB	C++	错误	85	3.046s	9.156MB
360614	卡牌游戏	06-24 22:17	1.335KB	C++	运行超时	95	运行超时	9.156MB
360613	卡牌游戏	06-24 22:17	1.335KB	C++	运行超时	95	运行超时	9.156MB
360608	卡牌游戏	06-24 21:50	1.215KB	C++	运行超时	95	运行超时	11.66MB
360607	卡牌游戏	06-24 21:49	1.215KB	C++	错误	85	3.015s	11.66MB
360606	卡牌游戏	06-24 21:49	1.215KB	C++	运行超时	95	运行超时	11.66MB
360605	卡牌游戏	06-24 21:37	1.161KB	C++	运行超时	95	运行超时	11.66MB
360602	卡牌游戏	06-24 21:16	1.112KB	C++	运行超时	90	运行超时	11.66MB

- $f_s = 1 - f_{\neg s}$，常数/2
- 所有计算 f 的东西能预处理就预处理，计算 f 的时候不需要做除法
- register(不过效果不是很明显？)
- 设定迭代精度 $1e-8$，卡时
- 初始值选得好就能少迭代好多次。

$$f_S = \frac{S\text{中 1 的个数}}{n}$$

- 输入 n, 输入 X, Y, Z，分别是 2,3,5 的次幂，同时 $1 \le n \le 1000, 1 \le XYZ \le 2000$。
- 输入四个长度为 n 的数组 $\{a_i\}, \{b_i\}, \{c_i\}, \{r_i\}(0 \le a_i, b_i, c_i, r_i \le 1000000000)$
- 对于 (u, v, w) 求有多少组解 $\{x_i\}, \{y_i\}, \{z_i\}$ 满足对于所有的 i, 有 $a_i \le x_i, b_i \le y_i, c_i \le z_i, r_i \ge x_i - a_i + y_i - b_i + z_i - c_i$，并且 $\left(\sum_{i=1}^{n} x_i\right) \bmod X = u, \left(\sum_{i=1}^{n} y_i\right) \bmod Y = v, \left(\sum_{i=1}^{n} z_i\right) \bmod Z = w$,
- 设解的个数为 $F(u, v, w)$，输出

$$\operatorname*{xor}_{\substack{0 \le u < X \\ 0 \le v < Y \\ 0 \le w < Z}} ((uYZ + vZ + w) \times (F(u, v, w) \bmod 466560001))$$

- 每个 i 看成一个三元多项式，求这些多项式乘积模 x^X-1, y^Y-1, z^Z-1。
- 由于取模的特殊性，应该先求值再插值。
- 46650001 的原根是 13。
- 这样 x, y, z 分别有 X, Y, Z 个取值，求值一个的复杂度为 $O(XYZ)$，需要进行 n 次。
- 最后插值的复杂度为 $O((XYZ)^2)$，总复杂度为 $O((n+XYZ)XYZ)$

- 北大街，在中国是一个非常常见的地名，比较著名的有上海北大街，西安北大街，成都北大街，太原北大街，中关村北大街等。
- 我们都知道，北的意思是自由民主，大的意思是兼收并蓄，所以住在北大街的人也性情迥异，我们假设北大街住了 n 个人。
- 有人向住在北大街的这 n 个人提了 $n-1$ 个问题，比如
  - “用不用筷子？”
  - “吃不吃红烧肉？”
  - “写代码用 tab 还是 space”
  - “大括号换不换行？”
  - “……”
- 根据每个人的回答，他会被分配一个 $n-1$ 维的零一坐标，也就是一个点。这样 n 个点可以恰好构成一个 $n-1$ 维空间中的凸包。

- 北大街的居民认为，在这个多面体内，便是华夏；多面体之外，便是蛮夷。我们可以很容易的计算出华夏部分的广义凸包体积。
- 有一天，清华路的 B 君来北大街玩，听说了这个故事觉得很有趣，于是也试着给出了这 $n-1$ 个问题的答案，清华路的 B 君，当然认为自己属于华夏，但是北大街表示在 $n-1$ 维空间中如果有 $n+1$ 个点的话，华夏部分的体积难以计算。
- 这下子气氛突然江化。所以这个问题就留给你了，输入 $n-1$ 维度空间中的 $n+1$ 个点，求广义凸包的体积。
- 由于这个体积可能不是整数，你只需要输出体积乘以 $n-1$ 的阶乘，然后对 1000000007 取模的结果。
- 一句话：输入 $n+1$ 个 $n-1$ 维的点，求凸包体积。$n \leq 35$

- 答案是任取 n 个点的体积的和除以二。
- $O(n)$ 枚举 n 个点，$O(n^3)$ 计算体积。一共 $O(n^4)$

- 小葱面前有 N 只大葱怪。
- 大葱怪很厉害，第 i 只大葱怪攻击力为 a_i，防御力为 d_i。
- 小葱的攻击力为 A，防御力为 D。
- 小葱打掉第 i 只大葱怪的代价是 $A \cdot d_i + D \cdot a_i$。
- 小葱需要选择出 R 只大葱怪，使得自己能够从神葱那里得到的评价最高。

$$\max_{S \subseteq [N], |S|=R} \left[\min_{A,D \in \mathbb{Z}^+} \frac{\max_{i \in R} (A \cdot d_i + D \cdot a_i)}{\max_{i \in [N]} (A \cdot d_i + D \cdot a_i)} \right]$$

- $1 \le R \le N \le 10^3$, a_i, d_i 均为正整数。

- 答案与 A, D 具体是多少无关，只与其形成的角度有关。
- 令 $(A, D) = (x, 1-x)$，则 $A \cdot d_i + D \cdot a_i = (d_i - a_i)x + a_i$，即一条直线答案具有可二分性，对于某个答案 k，判断其能否成立即是将原来所有直线的 y 坐标变为 $k \cdot y$，然后问能否在原来的直线中找出 R 条覆盖新的直线。
- 即判断一个半平面交是否包含另外一个半平面交。
- 由于本题的特殊性，可以将所有直线投影到 x 轴上，转换为找出 R 个区间覆盖整个大区间的动态规划问题。复杂度 $O(n^2 \log n)$，使用数据结构可优化到 $O(n \log n)$

- 现有一个长度为 n $(1 \le n \le 2 \times 10^5)$ 的非负整数数组 $\{a_i\}$。小 L 定义了一种神奇变换：

$$f_k = \sum_{i=1}^{n} {a_i}^k \pmod{998244353}$$

- 小 L 计划用变换生成的序列 f 做一些有趣的事情，但是他并不擅长算乘法，所以来找你帮忙，希望你能帮他尽快计算出 $f_1 \sim f_n$。
- $n \le 200000$

- “虽然我还不太会做这道题，但我知道一定是 FFT！”
- 能想到这一点……说明你已经做出来一半了

- “虽然我还不太会做这道题，但我知道一定是 FFT！”
- 能想到这一点……说明你已经做出来一半了
- 观察一番：假设有 3 个数 a, b, c

$$(a+b+c)^2 = f[2] - 2ab - 2bc - 2ac$$

$$(a+b+c) \times f[2]$$

$$= a^3 + b^3 + c^3 - ab^2 - bc^2 - ca^2 - ba^2 - cb^2 - ac^2$$

$$= f[3] - f[2] \times (ab + bc + ac) + 3 \times abc$$

- 对于等于 4 的情况也可以类似地推导

- 我们发现，如果令 $g[i]$ 表示任意 i 个不同的数的乘积的和，那么我们有：

$$f[1] = g[1]$$

$$f[2] = f[1] \times g[1] - 2 \times g[2]$$

$$f[3] = f[2] \times g[1] - f[1] \times g[2] + 3 \times g[3]$$

$$f[4] = f[3] \times g[1] - f[2] \times g[2] + f[1] \times g[3] - 4 \times g[4]$$

- 把 f[] 和 g[] 的生成函数写出来，记为 F、G
- 经过一番推导，发现由 G 可以多项式求逆得到 F
- 使用分治 FFT 计算 G，多项式求逆得到 F
- 时间复杂度 $O(n \log^2 n)$

- 定义多边形 P 与区间 $[l, r]$ 的作用

$$P \circ [l, r] := \{a \cdot p : a \in [l, r], p \in P\}.$$

- 现在有一个 n 个顶点的凸多边形 P 和 m 个形如 $s_i := [x_i, 2x_i]$ 的区间，从中选出 k 个区间使得最大化 $P \circ s_{a_1}, P \circ s_{a_2}, \ldots, P \circ s_{a_k}$ 的面积并。
- $3 \leq n \leq 100, 1 \leq k \leq m \leq 100, x_i > 0$

- 多边形求交、多边形求面积

- 多边形求交、多边形求面积
- sort 一下线段，记对应生成的多边形为 P_i
- dp[i][j] 表示 $P_1 \sim P_i$ 取 j 个，并且强制 P_i 取的最大面积
- 预处理所有的 $P_i - P_j$ 的面积即可。

- 两个人同时从 $(0,0)$ 出发，只能向下或者向右走到 (n,m)，路径除了起点与终点之外不能有任何交点，求不同的方案数 $\bmod 3486784401$
- $n,m \le 10^{18}$

$$2 \times (C_{n+m-2}^{n-1} C_{n+m-2}^{m-1} - C_{n+m-2}^{n-2} C_{n+m-2}^{m-2})$$

- $3486784401 = 3^{20}$，还要手写一个 $3^t \times s$ 的运算……

听说我不会经典套路（

- 第一步肯定是一个人向右一个人向下，汇合的时候两个人分别是从上边和左边过来的
- 相当于计算从 $(1,0)$ 和 $(0,1)$ 到 $(n, m-1)$ 和 $(n-1, m)$ 的不相交路径条数

听说我不会经典套路（

- 第一步肯定是一个人向右一个人向下，汇合的时候两个人分别是从上边和左边过来的
- 相当于计算从 $(1,0)$ 和 $(0,1)$ 到 $(n,m-1)$ 和 $(n-1,m)$ 的不相交路径条数
- 不考虑相交是答案的第一项
- 对于每种相交，不妨让两个人在第一次相交的时候交换身份，那么每种相交的走法对应了一种从 $(1,0)$ 和 $(0,1)$ 到 $(n-1,m)$ 和 $(n,m-1)$ 的路径，并且这个对应显然是一一映射的。

听说我不会经典套路（

- 第一步肯定是一个人向右一个人向下，汇合的时候两个人分别是从上边和左边过来的
- 相当于计算从 $(1,0)$ 和 $(0,1)$ 到 $(n,m-1)$ 和 $(n-1,m)$ 的不相交路径条数
- 不考虑相交是答案的第一项
- 对于每种相交，不妨让两个人在第一次相交的时候交换身份，那么每种相交的走法对应了一种从 $(1,0)$ 和 $(0,1)$ 到 $(n-1,m)$ 和 $(n,m-1)$ 的路径，并且这个对应显然是一一映射的。
- K 个点出发到 K 个点的时候也能做……有一个叫做 LGV 公式的东西
- 不过去年已经有人拿这个出过题了好像并没有火起来（

- 求 L 到 R 这 $R-L+1$ 个数中，一共有多少个子集，满足子集中的数的乘积是完全平方数。
- 多组数据，$1 \le L \le R \le 10^7$，$T \le 100$

- 设 x_i 表示第 i 个数是否需要取，以 $L=1, R=12$ 为例：

对于质数 2 $x_2 \ xor \ x_6 \ xor \ x_8 \ xor \ x_{10} = 0$

对于质数 3 $x_3 \ xor \ x_6 \ xor \ x_{12} = 0$

对于质数 5 $x_5 \ xor \ x_{10} = 0$

对于质数 7 $x_7 = 0$

对于质数 11 $x_{11} = 0$

- 解这个异或方程组，可以发现只有 7 个自由元，因此答案是 $2^7 = 128$。
- 因此可得出结论：若这个异或方程组的秩为 r，则答案为 $2^{R-L+1-r}$。
- 现在问题就转化为如何快速求出 r。如果直接暴力高斯消元的话，是可以拿到 50 分分数的。

- 有一个显然的结论：对于任何一个数 k 而言，至多只有一个素因子大等于 $\sqrt{k}$。
- 利用这个结论，对于最大素因子为 d，并且 $d \geq \sqrt{\max(R_i)}$ 的数，在它们之间取一个数 a，再将所有的数的状态与 a 的状态进行异或，这样异或出的结果一定只有小等于 $\sqrt{\max(R_i)}$ 的数。再将这个新的状态扔进之前的状态中进行高斯消元即可。显然，答案还要扣去 1——因为这些数要被答案是 d^2 的倍数这个条件所约束。
- 这么做就能通过这些测试点。时间复杂度为 $O($ 预处理 $+\sum(R_i - L_i + 1) *$ 小于 $\sqrt{\max(R_i)}$ 的素因子个数 $/32)$，其中除以 32 是因为使用 bitset 进行高斯消元运算。

- 对于测试点 $11, 12$，满足 $R \leq 10^6$ 并且 $R - L \geq 999990$

- 对于测试点 $11, 12$，满足 $R \leq 10^6$ 并且 $R - L \geq 999990$
- 显然 $r = 2 \sim R$ 中质数个数。直接统计就好了。

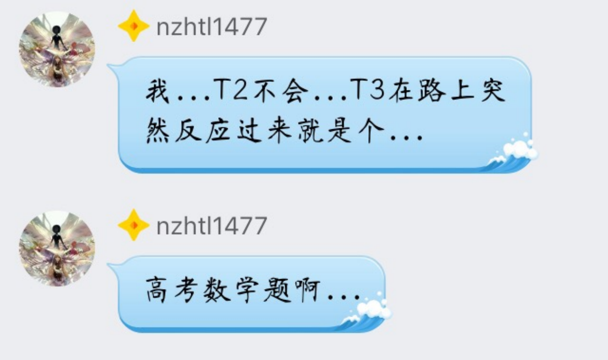
- 瓶颈卡在消元上。这个时候需要仔细分析一下什么时候之前的算法的结论会成立。
- 经过研究，可得出当 $R-L\geq 6000$ 的时候，之前的算法的结论一定成立。这是因为当 $R=10^7$ 时，小于 $\sqrt{R}$ 的质数个数只有不到 500 个。再考虑上素数分布，可以估算出这个结论。
- 因此只要当 $R-L\geq 6000$ 时直接 for 一遍有出现在 $L\sim R$ 之间的素数（而不是 $1\sim R$ 中的素数个数），计算 r 即可。时间复杂度为 $O($ 预处理 $+\sum(R_i-L_i+1)+\gamma(L_i,R_i))$，其中

$$\gamma(L,R)=\begin{cases}0, & \text{if } R-L+1\geq 6000\\ 446/32*(R-L+1) & \text{if } R-L+1<6000\end{cases}$$

- 给一个长度为 n 的环，有 m 种颜色，要求每相邻的 m 个珠子不能恰好出现所有 m 种颜色。
- 旋转认为是本质相同，翻转不是。求方案数 $\bmod 1000000007$
- $n \le 10^9, m \le 7$

- puts("2"); 10 分
- 暴力状压 DP+polya：f[i][j] 表示 $i-m$ 到 $i-1$ 的颜色状态为 j 的方案数
- 默认大家都会 polya 了~~（其实这就涉及到我的知识盲区了）~~
- 如果不考虑循环，打个表出来发现是个线性递推，系数可以直接高斯消元搞出来
- 打表复杂度 $7^7 * n*$ 最小表示法个数
- 听说是个 409 阶递推式？还要用 myy 的集训队论文里面的方法？丧心病狂.jpg

- 在 K 维空间中给出 K 个球，求它们的所有公切面。$K \le 10$。
- 为了降低难度，本题变成了提交答案的形式。选手可通过观察答案/乱搞骗分来得分。
- 可是似乎比 CTSC2017 Day2 T3 的得分率还要低 ??? 抓脑袋



- 根据题目中所给的定义，可以得出在 K 维空间中的超平面方程为 $a_0x_0 + a_1x_1 + \cdots + a_{K-1}x_{k-1} = d$，其中 $a_0, a_1, \cdots, a_{K-1}$ 和 d 为参数。
- 这样的话一共有 $K+1$ 个参数，多出一个自由元，我们只要强行令 $\sum_{i=0}^{K-1} a_i^2 = 1$ 即可。
- 类比高中数学必修 2 上的“点到直线的距离公式”，可以得出在 K 维空间中，点到超平面的距离公式为

$$dis(x, a, d) = \frac{|a_0x_0 + a_1x_1 + \cdots + a_{K-1}x_{K-1} - d|}{\sqrt{a_0^2 + a_1^2 + \cdots + a_{K-1}^2}}$$

- 相切的条件即为一个球的球心到这个平面的距离等于其半径。

- 由于我们强行令 $\sum_{i=0}^{K-1} a_i^2 = 1$，那么距离公式就变成了

$$dis(x, a, d) = |\sum_{i=0}^{K-1}(a_i x_i) - d|$$

- 对于第 i 个圆，需要满足的式子为 $|\sum_{j=0}^{K-1}(a_j(x_i)_j) - d| = r_i$
- 通过枚举正负号，可以把绝对值去掉，因此答案个数最多为 2^K。
- 去掉之后，将所有的 a_i 用 $C_1 d + C_2$ 表示，带入 $\sum a_i^2 = 1$，然后直接解一元二次方程就能直接得出答案。
- 复杂度为 $O(2^K K^3)$

Noi2017 Bless All!