Gena and Second Distance 解题报告

长郡中学 陈胤伯

1 试题来源

http://codeforces.com/contest/442/problem/E

2 试题大意

有一个 $w \times h$ 的矩形区域，左下角坐标为 $(0, 0)$。

矩形区域内有 n 个点，第 i 个点坐标 (x_i, y_i)。你需要在矩形区域内找一个位置 (X, Y)，使得这个位置离给定的所有点的次近点最远。

你只需要输出所求点的次近距离的值即可。

$n \leq 1000$，$0 \leq w, h \leq 10^6$。

3 算法介绍

3.1 二分答案

对于这种类似于“最小值最大”的问题我们有种常用的效果拔群的方法就是二分答案。

设当前二分的答案为 R，我们要判定是否存在 $ans \geq R$。

我们以这 n 个点为圆心分别作半径为 R 的圆，既然要求选择的位置的次近距离不小于 R，那么符合要求的点一定被至多一个圆覆盖。

考虑满足“被至多一个圆覆盖”的区域，假如存在，不妨在区域里面随便取一个点，然后开始乱走，直到碰到了这个区域的边界。为什么不能再跨过一步了呢？只有两种可能，要么走到了矩形的边界；要么走到了某个圆的圆周上，再往前一步就被两个圆覆盖了。进一步的，这样的区域的边界至少有一条是圆周构成的（不然边界就是整个矩形了），所以我们只需要从圆周上找这样的点。

考虑枚举一个圆 c，然后在 c 的圆周上找符合要求的点。这样的点已经被 c 覆盖，就不能再被其他圆覆盖了。注意到每个其他的圆和 c 圆周的交是一段连续的区间，那么我们只需要求出这些区间，排好序，做一次简单的区间覆盖问题即可完成“c 圆周上是否有符合要求的点”的判定。

如果直接这样做的话，复杂度是 $O(n^2 \log n \log v)$的。

3.2 一个小优化

之前我们是先二分答案再枚举圆周的，这样做似乎很难继续优化了。考虑把顺序换过来，先枚举圆周再二分答案。

我们先枚举一个点 c，把点 c 圆周上二分出的最终答案称为点 c 生成的答案。

现在可以做两件事：

1. $O(n \log v \log n)$ 求出这个点生成的答案。
2. 对于任意 R，$O(n \log n)$ 判定这个点生成的答案是否大于 R。

一个优化是，在枚举点 c 时，不急着二分答案，先把之前求出的最优解 cur 带入 $O(n \log n)$ 检查一次，看看点 c 是否能生成出大于 cur 的答案。如果能，再用 $O(n \log n \log v)$ 去二分。

如果以完全随机的顺序处理每个点，复杂度是靠谱的。

先考虑检查所耗费的复杂度：n 个点，每个点用 $O(n \log n)$ 的复杂度检查，总复杂度 $O(n^2 \log n)$。

再考虑二分所耗费的复杂度：如果在枚举第 i 个点时要进入二分，那么第 i 个点生成的答案必然是前 i 个枚举的点中最大的，这个概率为 $O(\frac{1}{i})$。

这样一共期望进入 $O(\sum_{i=1}^{n} \frac{1}{i}) = O(\log n)$ 次二分过程，每次二分 $O(n \log n \log v)$，总复杂度 $O(n \log^2 n \log v)$。

然后就可以通过这道题了。

3.3 总结

事实上这道题提供了一个不错的思路。

考虑有 n 个黑箱，每个黑箱需要花费较高的复杂度产生一个答案，但只需较低的复杂度即可判定其产生的答案是否 $> K$。你想要找出最大的答案。

那么我们以随机顺序访问每个黑箱，先判定，后产生，这样往往能得到更加优秀的复杂度。