矩阵乘法

问题描述

给你一个 N*N 的矩阵，不用算矩阵乘法，但是每次询问一个子矩形的第 K 小数。

输入格式

第一行两个数 N,Q，表示矩阵大小和询问组数；

接下来 N 行 N 列一共 N*N 个数，表示这个矩阵；

再接下来 Q 行每行 5 个数描述一个询问：x1,y1,x2,y2,k 表示找到以(x1,y1)为左上角、以(x2,y2)为右下角的子矩形中的第 K 小数。

输出格式

对于每组询问输出第 K 小的数。

样例输入

```
2 2
2 1
3 4
1 2 1 2 1
1 1 2 2 3
```

样例输出

```
1
3
```

数据规模和约定

矩阵中数字是 10^9 以内的非负整数；

20%的数据：N<=100,Q<=1000；

40%的数据：N<=300,Q<=10000；

60%的数据：N<=400,Q<=30000；

100%的数据：N<=500,Q<=60000。

题解：
题目描述纯属无聊，以泄撞题悲愤之情.....

【大暴力之术】

有一个很简易并且常数很小的暴力，复杂度为 $O(N^2Q)$，一开始将所有数直接排好序，对于每组询问，线性的按照每个数的数值大小遍历一遍即可得出答案，期望得分 20~40 分。

【分析】

李超神牛之前出过类似的题目，我的做法比较类似，只不过这里我不存在修改操作，所以可以使用离线算法。

这题空间限制比较吃紧，而很多在线算法是需要大量空间的，比如划分树套二维树状数组，所以我们使用离线算法来优化空间。

还是基于划分树的边二分边查询的思想，我们假设所有询问一起二分，一开始他们都会二分同样的一个数值，即查询一个子矩形中，包含有多少个小于等于 mid 的数，这里我们可以直接用二维树状数组进行统计。

二分完成后所有节点便会兵分两路，我们直接把所有询问和矩阵中的数字分为两部分继续递归即可，这里这个算法的复杂度 $O(logN*logN*logQ*(Q+N^2))$，实际上由于二维树状数组常数非常小，用这个算法已经可以顺利的通过这道题了。

【优化】

我们可以发现，这个算法的瓶颈在于在每一层，求一次每一个子矩形中包含了多少个数字，这一步我们的复杂度是$O(logN*logN*Q)$的，实际上我们可以将其优化到$O(logN*Q)$，那就是将所有询问按照 x 坐标排好序，用一维树状数组来查询 y 坐标即可，这是一个经典的降维的优化，总复杂度变为了 $O(logN*logQ*(Q+N^2))$。

但实现以后效果不明显，仅从 1.3s 优化到了 1.1s，因为每次重新递归询问的时候，要通过一次归并来使得保持 x 坐标单调，这个地方成为了瓶颈，让我很是郁闷= =。

不过第二个算法还是有他的优势所在的，它不需要开一个大小为 N*N 的数组，所以如果这道题是一个稀疏矩阵，每个数字的坐标非常的大，对于后面这个算法几乎没有影响，而前面的算法用于空间吃不消而需要使用一些特殊的方法，这样就会使得常数增大的多了。