Spring Test Day1
Solution

Miaomiao

February 27, 2018

1 Alice

1.1 100%

枚举矩阵行的下边界，每次把这一行所有的光点进行处理。

对于每一列，我们记一下当前这个位置到最近的光点有多少距离，这样就可以得到一个序列。

这个序列对于答案的贡献我们用$treap$维护，里面用序列值作小根堆排序，这样我们就可以这样维护答案：

$$sum[o] = sum[lc] + sum[rc] + (siz[o] + 1) * siz[o] * (val[o] - val[fa])$$

因为数据随机，所以效率为$O(qlogn)$。

2 Good

2.1 100%

首先这题就是求随机点分治的期望时间复杂度。

分析点对(x,y)，考虑点x对点y作的贡献，就是如果在x至y这条链上，点x在点y之前被选，并且是这条链中第一个被选的点。这个概率是$\frac{1}{dis_{x,y}}$。于是我们要求的东西可以转化为：

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{1}{dis_{i,j}}$$

然后我们点分治求每种距离的点对数就可以了，中间用FFT处理一下。

还有一点要注意，就是我们不能把子树的信息逐个合并，这样效率是平方级别的，因为你每次都要FFT一遍。应该要把自己自乘一下，然后减掉每个子树自乘的，这样算复杂度才对。

效率$O(nlog^2n)$。

标程写的常数巨无霸大就不要理辣鸡出题人了。

应该都跑得比标程快多了。（逃

3 Night

3.1 1 - 5

数据狭长。

n, m对称，不妨只考虑n很小，m很大的情况。

枚举两行，于是我们可以分块或者莫队。

分块：$O(n^2(m\sqrt{m}logm + klogk))$，莫队：$O(n^2(m\sqrt{m}logm))$

耐心等一会儿，跑得出来的。

3.2 6

满足$A[x, y] < min(A[x-1, y], A[x, y-1])$。

任意满足$1 \le x_1 \le x_2 \le n$，$1 \le y_1 \le y_2 \le m$的均为逆序对，除非$x_1 = x_2$且$y_1 = y_2$。

令$n = u_2 - u_1 + 1$，$m = v_2 - v_1 + 1$

再令$c = \frac{n(n+1)}{2}$，$d = \frac{m(m+1)}{2}$

得$Ans = c * d - n * m$。

3.3 7

间隔形矩阵：$A[i, j] = (i + j)\ mod\ 2$

递推计算。

$c_x[n, m]$：左上角是x的$n * m$间隔形矩阵的0的个数。

$f_x[n, m]$：左上角是x的$n * m$间隔形矩阵的逆序对个数。

$f_1[n, m] = c_x[n, m] + f_0[n-1, m] + f_0[n, m-1] - f_1[n-1, m-1]$

$f_0[n, m] = f_1[n-1, m] + f_1[n, m-1] - f_0[n-1, m-1]$

效率$O(nm)$。

3.4 8 - 9

所有询问(u_1, v_1, u_2, v_2)满足$u_1 = 1, u_2 = n$。
统计所有$f[i, j]$表示第i列与第j列的逆序对个数。
效率$O(m^2nlogn)$

记$ans[i, j]$表示第i列与第j列围成的子矩阵内的逆序对个数。
$ans[i, j] = ans[i, j - 1] + \sum_{a=i}^{j} f[a, j]$
效率$O(m^2)$

可以$O(1)$回答每个询问。

3.5 10

随机数据。
令$a[x_1, y_1, x_2, y_2]$：$A[x_1, y_1] > A[x_2, y_2]$是否满足。
令$b[x_1, y_1, x_2, y_2]$：从(x_1, y_1)出发的子矩阵内的逆序对个数。
令$c[x_1, y_1, x_2, y_2]$：子矩阵内的逆序对个数。

$$\begin{aligned} b[x_1, y_1, x_2, y_2] = {} & b[x_1, y_1, x_2 - 1, y_2] + b[x_1, y_1, x_2, y_2 - 1] \\ & - b[x_1, y_1, x_2 - 1, y_2 - 1] + a[x_1, y_1, x_2, y_2] \end{aligned}$$

$$\begin{aligned} c[x_1, y_1, x_2, y_2] = {} & c[x_1 + 1, y_1, x_2, y_2] + b[x_1, y_1 + 1, x_2, y_2] \\ & - b[x_1 + 1, y_1 + 1, x_2, y_2] + b[x_1, y_1, x_2, y_2] \end{aligned}$$

效率$O(n^2m^2)$

感谢@Easy友情加工插图。