NOIP2024 模拟赛题解

CDQZ

2024 年 11 月 18 日

1. T1

原题：AT_abc173_f

1.1 part 1

$n \leq 200$，考虑可以直接枚举 l,r，然后看编号在 $l \sim r$ 之间的点暴力跑并查集，时间复杂度是 $\mathcal{O}(n\alpha)$ 的。总复杂度 $\mathcal{O}(n^3\alpha)$

1.2 part 2

一条编号连续为 $1 \sim n$ 的链，答案就是 n^2 ，因为编号连续的在原图上一定是一条链。

1.3 part 3

菊花图，考虑编号区间包不包含菊花的中心点，包含中心点答案就是 1 ，不包含答案就是 $r-l+1$。这里提示了去考虑每个点的贡献，并用统计代表元素的方法使得每个连通快只被计算一次。具体方法就是：假设中心点编号为 x ，那么所有包含中心点的 (l,r) 都在中心点处计算答案，计算到的的区间 l 在 $1 \sim x$ 中取值，r 在 $x \sim n$ 中取值，答案为 $x \times (n-x+1)$，所有不包含中心点的区间内，每个点都要做一次贡献，所以假设一个点的编号为 a ，那如果 $a < x$ 它的贡献为 $a \times (x-1-a)$,$a > x$ 它的贡献为 $(a-x) \times (n-a+1)$

1.4 part 4

$n <= 4000$ ，这部分在算法 1 的基础上优化。考虑固定 l ，对 r 做扫描线，并查集连边，动态维护连通快的变化：设 $(l,r-1)$ 的连通快个数为 res ，考虑加一个点就先把答案 $+1$ ，然后去枚举它的所有连边，如果加边使得连通块合并，那么 l,r 连通快个数在 res 的基础上 -1 ,然后并查集

合并这两个点。如果这条边连接的另一端点不在 (l,r) 区间内，就忽略这条边。对于每个 l ，每条边只被枚举到 2 次，所以时间复杂度是 $\mathcal{O}(n^2\alpha)$

1.5 part 5

考虑正解。是 part 3 和 part 4 的结合。考虑贡献。只考虑点不连边的贡献是 $\sum_{i=1}^{n}\frac{i\times(i+1)}{2}$。然后减去边的贡献，发现如果两边的点都在 $(l,r-1)$ 每条边会减去一个连通快的贡献，所以一条边 $(u,v),u<v$ 会使所有包含这条边两个点的编号区间 -1 ，总共减去 $u\times(n-v+1)$ 的贡献。

2. T2

2.1 $\mathcal{O}(n^2m^2)$

这个复杂度，无论采用建图还是DP，核心式子就是

$$f_{i,j}=\min_{i',j',a_{i',j'}=a_{i,j}-1}f_{i',j'}+|i-i'|+|j-j'|$$

2.2 $\mathcal{O}(nm\sqrt{nm})$，常数较小

上面那种做法的复杂度，实际上是 $\mathcal{O}(\sum\limits_{i\in[1,p)}w_{i-1}\cdot w_i)$，考虑到 w_p 的总和为 $n\cdot m$，数学很好的你又受到良心搬题人子任务3的启发，尝试根号算法。

- 如果 $w_kw_{k+1}\le nm$。考虑 $\sum w_k\le nm$，由均值不等式有 $\sum\sqrt{w_kw_{k+1}}\le nm$。不妨设 $\sqrt{w_kw_{k+1}}=a_k$。那么也就是求 $\sum a_k^2$ 大概是多少。我们假设 a_k 的最大值为 v。由于 $v^2\le mn$，$v\le\sqrt{nm}$。那么 $\sum a_k^2\le v\sum a_k\le mn\sqrt{mn}$。也就是复杂度上限也是 $\mathcal{O}(nm\sqrt{nm})$ 的。（节选自题解）。
- 如果 $w_k\cdot w_{k+1}>n\cdot m$，我们使用多源BFS。

2.3 $\mathcal{O}(n\log n)$

考虑拆绝对值，现在在考虑点 p，则我们只需要分讨从 $\begin{matrix}1 & 2\\ 3 & 4\end{matrix}$ p 4个方向转移，每个方向的点就可以一视同仁了。这个二维问题，直接上扫描线，这时又出现了一个新的问题：如果我向四个方向分别加入矩形（这里我采用的理解方式是：以所在行编号为 y，所在列编号为 x，建平面直角坐标系，纵向从低到高扫），那么就要加入八根线段，也就是对于一个区间取 $\min$，单点查询的线段树，我居然要支持撤销？其实好办，考虑分从下到上，从上到下扫两次，就可以了。

3. T3

3.1 Subtask 1,2

直接 $\mathcal{O}(n^3)$ 暴力做即可。

3.2 Subtask 3

每次钦定一个根，然后从根到叶子一次统计 mex，每次 dfs 均摊是 $\mathcal{O}(n)$，那么总的复杂度就是 $\mathcal{O}(n^2)$。

3.3 树随机

考虑到一条路径的 mex 值不会很大，所以就直接枚举所有答案，统计可能的 (i,j) 的对数。

3.4 正解

首先我们把有序对 (x,y) 转化为无序对 (x,y) 且 $x\neq y$，最后输出 $ans\times 2+1$（$+1$ 是因为有 $(0,0)$）。

然后考虑到 max 可能不太好做，于是考虑 min-max 容斥一下，再改变一下求和顺序，现在就变成了求：

$$\sum_{i=0}^{n-1}\sum_{x=0}^{n-1}\sum_{y=0}^{n-1}[\min_{t=1}^{k}\mathrm{cute}(t,x,y)>i]$$

即对于每一个 i，求有多少个 (x,y)，满足在所有树上 (x,y) 这条路径都包含了 $0\sim i$ 的所有数。

首先 $i=0$ 与 $i>0$ 的做法有些区别，我们分开考虑。对于 $i=0$，我们在 k 棵树都以 0 为根节点 dfs 一遍。然后我们考虑对于任意一棵树，先给根节点 0 赋一个权值 1，然后对于 0 的每一棵子树，分别为其赋上 $2,3\dots$ 的权值，现在每个点 i 在 k 棵树上都有了一个权值 $a_{i,k}$。那么两个点 (i,j) 有贡献当且仅当 $\forall 1\le t\le k,a_{i,t}\neq a_{j,t}$。这个可以再容斥一下，即钦定一个子集里的数全都对应相同，其他的任意，这个可以预处理出来。

然后是 $i>0$，对于每一棵树，我们先随便以 $(0,1)$ 路径中间的一个点为根，那么如果一个点对 (x,y) 符合 x 在子树 0 内，y 满足在子树 1 内，这个点对就合法，然后我们把 0 的子树内权值赋为 0，1 的子树内权值赋为 1（如果 (x,y) 中间没有点也无所谓，重要的是标出点的权值）。现在每个点在 k 棵树中的权值就形如 $(0,0,0),(0,0,1)\dots(1,1,1)$ 这种，一个点对有贡献的条件和上面一样，于是你就可以统计出 $(0,0,0),(0,0,1)\dots$ 每一种的个数，计算答案的话直接算即可。

然后依次枚举 $2,3,4\dots$，每次枚举到一个新的数时先判断这个数能否和之前的数构成一条路径，如果能，就相当于会删去几个点的贡献，每次删掉一个点也是可以快速计算带来的贡献，而总的删除次数一定不超过 n。所以最后的时间复杂度就是 $\mathcal{O}(n2^kk)$，细节比较多，实现起来可能比较麻烦。

4. T4

原题：CF720D Slalom

4.1 测试点 $1,2$

枚举每一步是往上还是往右，统计每个障碍在路径的哪一侧即可。

4.2 测试点 $3,4$

枚举每个障碍物被从哪侧绕过，把合法位置取交，然后看是否存在一条合法路径即可。

4.3 测试点 $5 \sim 9$

考虑在所有等价的合法路径中选出一条代表的路径，并对代表路径计数。对于一条路径，我们不断进行如下操作（如果能保证操作后路径仍合法）直到不能操作：选择路径上一点，将这个点左侧的横线段下移一格。例如路径 $(1,1)(1,2)(2,2)(3,2)(3,3)$ 在选择 $(3,2)$ 后变为 $(1,1)(2,1)(3,1)(3,2)(3,3)$。

显然这种操作不会改变路径位于障碍的哪一侧，且所有等价的路径一直操作下去会变成同一条，而两条本质不同的路径，操作后一定不会变成同一条路径，那么我们就可以选择所有无法操作的路径作为代表，答案即为有多少条无法操作的路径。

我们考虑一个 dp，设 $f_{i,j}$ 表示走到 (i,j) 的代表路径有多少条，注意我们并不考虑 $f_{i,j}$ 对 $f_{i,j+1}$ 的贡献，而只考虑 f_{i-1} 对 f_i 的贡献。首先每个点都可以从左边的点转移过来，即 $f_{i,j} \leftarrow f_{i-1,j}$。

然后我们考虑对于一个点，如果存在一个矩形，满足该点是矩形左上角上面一格的点，就称之为特殊点。对于特殊点还有转移，设 p 表示 $(i-1,j)$ 下方第一个障碍所在的 y 坐标，那么有 $f_{i,j} \leftarrow f_{i-1,k} \quad (p<k<j)$。于是加个前缀和优化，就可以 $\mathcal{O}(n^2)$ 转移了。时间复杂度 $\mathcal{O}(n^2+m)$。

4.4 测试点 $10, 11$

将坐标离散化，然后还是用上一个的 dp 做法。

4.5 测试点 $12 \sim 16$

障碍物无交的特殊性质即为 CF720D。

上面的 dp 其实已经很接近正解了。考虑扫描线，用一个线段树来维护 f_i，再用一个 set 维护当前的障碍物。每次就枚举当前横坐标下所有的特殊点，然后在 set 中二分出下面第一个障碍物的位置，就变成了一个区间求和，单点修改。注意转移应该从上往下考虑所有特殊点，这样转移才不会重复。

4.6 测试点 $17 \sim 20$

如果障碍物有交的话，那么还需要额外判断每个特殊点是否已经被障碍物覆盖了，如果被覆盖了就不能转移。还要注意有可能两个矩形的左上角重合了，就不要转移两次这个特殊点。