NOIP2024 模拟赛　题解

龟和贼达不留锐锡

2024 年 11 月 11 日

目录

A 大战波特（bot）

1. 算法一

我会 DP！

记 $DP(x,y,i)$ 表示从 (x,y) 走到 (n,m)，路径上有 i 个格子不确定的方案数，直接转移。

复杂度 $\mathcal{O}(n^3)$，期望得分 64。

2. 算法二

差不多得了，这是送分题！我会跟牛的 DP！

记 $DP(x,y)$ 表示从 (x,y) 走到 (n,m)，所有路径权值的和，记 x 为路径上不确定的格子数，则权值是 3^{-x}，直接转移。

复杂度 $\mathcal{O}(n^2)$，期望得分 100。

B 决胜数论（math）

1. 算法一

我会特殊性质！

若 $M_{0,0}=1$，则必有 $x=0$，直接判断一下，期望得分 23。

2. 算法二

首先特判 $M_{0,0}=1$。

一定有 $M_{i,i}=0, M_{i,j}=M_{j,i}$，特判掉。

考虑 x 和 y 不互质必有 $\gcd(x,y)>1$，相当于存在 $d>1$ 满足 $d\mid x, d\mid y$。也就是说，存在 $p\in\mathbb{P}$，满足 $p\mid x, p\mid y$。

那么一个质数 p 的贡献相当于在 $x+1,\ldots,x+n$ 中，将 p 的倍数挑出来，将它们形成的子矩阵标为 0。然后要将矩阵标成 M 的样子，这样就不难发现，只有 $p<n$ 的质数是有用的。

实际上，选择一个 x 的实质是给每个 p 选取一个起点，后面就是一个 CRT 的问题了。考虑对于每个质数 p，枚举起点 $0\le i<p$，观察 $M_{i,i+p}$，我们发现，这已经限定了 $x+i$ 是否是 p 的倍数。那么如果起点有大于一个选择就寄了；如果没有选择，我们发现，一定要有 $2p>n$，否则无解。

这样子，我们就通过部分信息推出来了唯一可能的 M'，只需要判断一下和输入的 M 是否相等就行了。

复杂度 $\mathcal{O}(\sum\limits_p \dfrac{n^2}{p^2})\approx\mathcal{O}(n^2)$，期望得分 100。

C 未来都市（metro）

1. 算法一

假设最优路径为 $S = t_0, \dots, t_{i-1}, t_i, t_{i+1}, \dots, t_k = x$。

引理一：$t_i \notin P(t_{i-1}, t_{i+1})$。证明：考虑肯定有一部分保留了 mx，又 $mn' \times mx \geq mn \times mx$，所以非常不优。

记 $\min(P) = \min\limits_{x \in P} a_x$，$\max(P)$ 同理，记 $M(u,v) = \max P(u,v)$。

引理二：a_{t_i} 一定是 $\min P(t_{i-1}, t_i)$ 或 $\min P(t_i, t_{i+1})$。证明：考虑不然的话你缩回最小值的位置肯定不劣，如果缩不到就是已经缩到了 $P(t_{i-1}, t_{i+1})$ 上，那可以直接把这个点删掉。

引理三：若 $a_{t_i} = \min P(t_{i-1}, t_i)$，则有 $i = 1$。证明：显然 $M(t_i, t_{i-2}) \leq M(t_i, t_{i-1}) + M(t_{i-1}, t_{i-2})$，那么 $a_{t_i} M(t_i, t_{i-2}) \leq a_{t_i}(M(t_i, t_{i-1}) + M(t_{i-1}, t_{i-2}))$，很难不发现 $a_{t_i} \leq a_{t_{i-1}}$，所以 $a_{t_i} M(t_i, t_{i-2}) \leq a_{t_i} M(t_i, t_{i-1}) + a_{t_{i-1}} M(t_{i-1}, t_{i-2})$。同理有：若 $a_{t_i} = \min P(t_i, t_{i+1})$ 则 $i = k - 1$。

由引理三，我们能推出一个重要的事实：$k \leq 2$。

2. 算法二

我会特殊性质 B！

很难不发现至少存在一种方案是 $S \to 1 \to x$，所以答案不超过 $2n$。预处理 $k = 1$ 的情况，$k = 2$ 的情况暴力更新，更新次数不超过 $\sum \frac{n}{i}$，复杂度 $\mathcal{O}(n \ln n)$。

3. 算法三

难点在于处理 $k = 2$ 的情况。

记 $f_u = \max P(S,u) \times \min P(S,u)$，$g_u$ 为答案。

由引理二，我们发现我们枚举的中间点一定均是两部分路径上的最小值。事实上，我们可以钦定 u 时最小值，因为错解不优。那么我们的所有更新都形如 $g_u \leftarrow f_v + a_v \times \max P(u,v)$。

路径最大值不好处理，考虑建出点权 Kruskal 重构树，那么 $\max P(u,v)$ 相当于 $a_{\mathrm{LCA}(u,v)}$。那么现在更新形如 $g_u \leftarrow f_v + a_v \times a_{\mathrm{LCA}(u,v)}$，考虑在 $\mathrm{LCA}(u,v)$ 出统计贡献，那么实际上相当于查询 $\max\limits_{v \in \mathrm{sub}(u)} f_v + a_v \times a_u$，其中 $\mathrm{sub}(u)$ 为 u 的子树。不难发现，这相当于一次函数最大值，可以用李超线段树维护，由于需要统计子树答案，所以直接线段树合并就可以了（你想写 DSU on tree 也可以 /hsh）。

总时间复杂度 $\mathcal{O}(n \log V)$，空间复杂度 $\mathcal{O}(n \log V)$。

D 数据结构（ds）

1. 算法一

我会特殊性质 B！

可以用吉司机线段树维护。

我会特殊性质 C！

发现修改均不影响 a 的单调性，那么操作一相当于后缀覆盖，操作二可以直接打标记，复杂度 $\mathcal{O}(n\log n)$。

2. 算法二

发现特殊性质 C 的做法非常有前途！

可以发现，感性上修改操作都是让序列往单调的方向发展的。那么可以看成是，序列最开始时不是单调的，但是随着操作的进行，序列的一些部分逐渐变的单调了。

于是一个直接的思路就是，记 A 为原序列，我们将 A 分为两部分 B、$A\setminus B$，分别维护，满足 B 是单调的。B 初始时为空，在操作进行的过程中，会有一些 $A\setminus B$ 的部分加入到 B 中，所以每个点什么时候加入 B 是我们所关心的。

一个观察是：一操作实质上是后缀推平，也就是说，它会影响 B 的一个后缀，并且将它们推平到 v。那么如果某个 $x\in A\setminus B$ 也受到了这个一操作的影响，那么意味着结束时它的值也会变成 v，那么它也就能自然的融入 B 受影响的后缀了。

也就是说，我们关注的时间戳实际上就是每个点第一次受一操作影响的时间戳，记 t_i 为 i 的时间戳。而另外一个观察是：若 $p<q$ 且 $t_p>t_q$，那么从 t_q 这个节点开始，a_p 恒 $\le a_q$。这个就是由于在 t_q 这个时刻，我们能确定 $a_p<a_q$，而修改不改变单调性，所以这个性质会一直保持下去。

那么这会不会存在问题呢？考虑如果 $p\in A\setminus B, q\in B, p<q$，且 p 受到影响但 q 不受影响，也就是说 $a_p>v\ge a_p$，这与观察二矛盾，所以这种冲突是不存在的。

那么我实际上只要求出 $t_{1,\dots,n}$ 就完事了！

记 b_i 表示：第 i 个一操作之前，有多少个二操作。那么 t_i 相当于是最小的 p 满足 $t_i+i\times b_p>v_p$，略微移项就变成了：$t_i>v_p-i\times b_p$。考虑二分，用持久化李超线段树维护最小值，复杂度 $\mathcal{O}(n\log^2 n)$，可以通过。

考虑更近一步，我们发现 b_p 具有单调性，这启发我们维护 (b_p,v_p) 组成的凸包查询最小值。顺序扫 i，那么切点显然可以用指针移动维护，复杂度 $\mathcal{O}(n\log n)$。用整体二分不难将空间做到线性。