NOI 模拟赛 Day1 题解

2025 年 4 月 8 日

目录

1 互质序列

1.1 算法一

设 $dp_{i,j}$ 表示填了 i 个数，第 i 个数为 j 的方案数。转移枚举下一位填的是什么，复杂度 $O(m(d(n))^2+qd(n))$，期望得分 10 分。

1.2 算法二

显然你只需要关心每个质因子是否出现过，状态就从 $O(d(n))$ 变成了 $O(2^{w_n})$，期望得分 20 分。

1.3 算法三

上述 dp 显然可以通过矩阵快速幂优化，复杂度 $O(q2^{3w_n}\log m)$，期望得分 45 分。

1.4 算法四

矩阵快速幂肯定是无法省掉的，也无法使用稀疏矩阵乘法等技巧来优化矩阵快速幂，那么只能考虑继续压缩状态。注意到对于两个质因子 p_1,p_2，如果它们在 n 中的次数是一样的，那么它们是可以不必区分的。

所以可以对于质因子按照它们在 n 中的次数进行分类，对于每一类质因子只需要记录其在上一个数中出现过的数量即可。打表发现状态数 $T\le 255$，复杂度优化为 $O(qT^3\log m)$，期望得分 80 分。

1.5 算法五

最后一步是一个很常见的矩阵快速幂优化技巧。令转移矩阵为 W，考虑预处理 $W,W^2,W^4...W^{2^k}$，在询问时只需要进行 $O(\log m)$ 次向量乘矩阵即可。复杂度优化为 $O(T^3\log m+qT^2\log m)$，期望得分 100 分。

2 树的搜索

2.1 算法一

枚举 x,y 并枚举贡献的位置 u 后计算该情况成立的概率。复杂度 $O(n^4)$，期望得分 $4\sim 12$ 分。

2.2 算法二

算法一可以通过若干调整枚举顺序和拆贡献的技巧做到 $O(n^3)$ 或 $O(n^2)$，由于并不是本体重点所以不再赘述。期望得分 $12\sim 24$ 分。

2.3 算法三

链，菊花都是平凡的，结合算法二期望得分 36 分。

2.4 算法四

上述做法比较劣的主要原因是需要枚举很多东西才能计算贡献，考虑使用 dp 来整体计算贡献。设 $dp_{u,i}$ 表示 $x=u, y\in \text{Subtree}(u)$ 期望有多少个 y 对应的 $f(x,y)=i$。考虑如何进行转移。

考虑 $dp_{v,i}$ 会转移到哪里，第一种情况是起点变成 u 之后最小值位置仍然在 v 子树内取到。令 mn_v 表示 v 子树内的最小值，那么有转移 $dp_{u,i}\leftarrow dp_{v,i}\times\frac{1}{\sum_{x\in \text{Son}(u)[mn_x\le i]}}$。转移系数表示所有 mn_x 比 i 小的子树遍历顺序都必须在 v 之后。

第二种情况是起点变成 u 之后最小值位置在 v 子树以外，令最小值所在子树为 x，这部分又分为两种情况，$mn_v<mn_x$ 和 $mn_v>mn_x$。令 rk_v 表示 mn_v 在所有子树中的排名，显然需要所有 $rk_y<rk_x$ 的 y 访问顺序都在 v 之后，而 x 访问顺序在 v 之前。所以前者转移系数为 $\frac{1}{rk_x(rk_x-1)}$，后者系数为 $\frac{1}{rk_x(rk_x+1)}$。

按照上述进行转移，最后将 i 对 a_u 取 min，复杂度 $O(n^2)$。但随机数据下复杂度为 $O(n\log n)$，也可以通过值域较小的数据，期望得分 56 分。

如果对于 mn_v 相等的情况大小关系判断有问题，也可以通过 a_i 是排列的部分分。

2.5 算法五

考虑优化上述做法，前者显然可以线段树合并，系数可以通过线段树区间乘来处理，本质不同的区间只有 $O(deg_u)$ 个。后者可以将所有儿子按照 mn_v 排序后，分别处理 $mn_v<mn_x$ 和 $mn_v>mn_x$ 的两部分，具体来说可以通过线段树合并临时得到一个前缀或后缀对应的线段树，在上面进行区间查询，并在 u 对应的线段树上单点修改即可。

总结一下，线段树需要支持的操作为区间乘，单点加，区间查询和，都是可以简单维护的操作。时空复杂度均为 $O(n\log n)$，期望得分 100 分。实现时需要注意处理一下上述几种操作的顺序，注意线段树不要覆盖掉还有用的先前版本。

一些常数更大的做法或 $O(n\log^2 n)$ 的做法也可以获得 76 分。

3 卡牌游戏

3.1 算法一

令 dp_i 表示上一个选择位置为 i 的答案，转移有 $dp_i=\max_{j<i}\{dp_j+w(a_j,a_i)\}$。复杂度 $O(n^2\log n)$。$tp=1$ 的情况显然可以通过正反两边 dp 得到，期望得分 5 分。使用一些技巧预处理后做到 $O(1)$ 查询 gcd，可以做到 $O(n^2)$，期望得分 10 分。

3.2 算法二

容易发现 $w(x,y)=\max_{z\le S,x|z,y|z}\{z\}$，也就是 S 以内 $\text{lcm}(x,y)$ 最大的倍数。在 a_i 随机的情况下，a_i 的倍数期望是 $O(\log S)$ 的，所以可以开一个桶，把 dp_j 贡献到所有 a_j 的倍数上，查询查询所有 a_i 的倍数处最大的 val_k+k 即可。复杂度 $O(n\log S)$。合并正反两边的 dp 值也是容易的，期望得分 20 分。

3.3 算法三

对于 $a_i\le 100$ 的情况，很容易想到记录 g_i 表示 $\max_{a_k=i}\{f_k\}$，转移复杂度 $O(n|a|\log S)$。合并只需要再记录 $g2_i$ 表示上上个 i 出现的位置即可。结合算法二期望得分 30 分。

3.4 算法四

可能存在若干 $O(n\sqrt{n}\log n)$ 甚至 $O(n^{\frac{5}{3}})$ 的做法，我也不是很懂。期望得分 $30\sim50$ 分。

3.5 算法五

考虑将算法二和算法三结合在一起，设立阈值 B，将 $a_i>B$ 的叫做大点，$a_i\le B$ 的叫做小点。对于大点向大点转移的情况，可以沿用算法二，复杂度不会超过 $O(\frac{S}{B})$。对于任意点与小点之间转移的情况，可以枚举小点的值后暴力查询 w，复杂度 $O(B\log S)$。平衡得到 $O(n\sqrt{S\log S})$。

对于合并的情况，两种情况都可以通过记录上一个位置的值和上上个位置的值来解决。而每个合并的格式为对 $[l,r]$ 区间内的 ans 对 x 取 max，逆用 ST 表即可做到 $O(1)$ 修改最后 $O(n\log n)$ 整体查询。总复杂度 $O(n\sqrt{S\log S})$，常数比分块小但可以跑的比较满，期望得分 $50\sim70$ 分。

瓶颈在于 gcd，可以通过一些预处理技巧做到 $O(1)$ 查询 gcd，复杂度 $O(n\sqrt{S})$，期望得分 70 分。

3.6 算法六

进一步观察性质，注意到如果 $a_i,a_j\le\sqrt{S}$，有 $w(a_i,a_j)\ge\frac{S}{2}$。令 $B=\sqrt{S}$，如果 i 和 j 之间间隔了至少 3 个小点，那么肯定不会在 i 和 j 之间转移，因为这样不如将小点全部选择。

这样就可以将有小点参与的转移优化到 $O(1)$，同时合并的时候这部分也可以优化到 $O(1)$。注意到寻址，乘除法的常数瓶颈都在这一部分，所以将这一部分优化为 $O(1)$ 后常数可以大幅减小，复杂度仍为 $O(n\sqrt{S})$，期望得分 100 分。