NOIP2022 模拟赛 Round 2 题解

重庆市巴蜀中学 郭雨豪

2021 年 9 月 29 日

目录

1 T1 开挂(openhook)

1.1 算法 1

暴力枚举或分类讨论，可通过测试点 $1 \sim 5$，期望得分 20。

1.2 算法 2

我一看，这个特殊性质 B 比特殊性质 A 简单了 114514 倍!

直接贪心，对于每一个 i，存在一个数被移动了 $i-1$ 次，让小的 b_i 移动的尽量多即可。

结合算法 1，可通过测试点 $1 \sim 5, 14 \sim 16$，期望得分 32。

1.3 算法 3

只看特殊性质 A，我们希望总步数最少，将变化 a 看作给每一个 a_i 分配一个最终的位置，中我们可以看作我们有一个变量 cnt，表示当前还没有确定位置的数的个数，令 c_x 表示 $a_i = x$ 的 i 的个数，可以从小到大枚举 x，并执行如下操作：

1.$cnt+=c_x$

2.$cnt = \max(0, cnt-1)$。

3.$ans+=cnt \times b_1$。

第一个操作表示将当前权值为 x 的数加入没有确定位置的集合，第二个操作表示我们可以将一个数留在原地，第三个操作表示我们要让剩下的数都加一需要的代价。

时间复杂度 $O(n + \max a_i)$，结合以上算法可以通过测试点 $1 \sim 9, 14 \sim 16$，期望得分 48。

1.4 算法 4

还是特殊性质 A，我们发现很多时候其中的 $cnt = 0$，不会造成影响，实际上有值的点只会有 $2n$ 个，遇到 0 是可直接跳到下一个 c_x 不为 0 的位置，时间复杂度 $O(n \log n)$（需要排序）。

1.5 算法 5

考虑结合算法 2,4，将 cnt 这个变量换成一个桶，其中装下了所有还未安排的数，由于我们要让出现次数最多的数出现次数尽量少，即让所有数的出现次数尽量不平均，我们可以维护一个栈，其中栈顶的 a_i 更大，我们每次选 a_i 最大的一个数放在该位置，$cnt-1$ 则看做一次弹栈操作，时间复杂度 $O(n \log n)$，直接使用优先队列应该也能通过。

如果没有想到算法 4 在算法 3 基础上的优化则无法通过测试点 $21 \sim 25$。

2 T2 叁仟柒佰万(clods)

2.1 算法 1

搜索，枚举划分，时间复杂度 $O(n2^{n-1})$，可通过测试点 1，期望得分 10。

2.2 算法 2

枚举区间 mex，进行 DP，f_i 表示最后一个分界点为第 i 个点的方案数，枚举有一个 n，状态有一个 n，转移有一个 n，时间复杂度 $O(n^3)$，可通过测试点 $1,2,3$，期望得分 30。

2.3 算法 3

如果您写出了算法 2，并输出中间值，您会发现只有一个 mex 是会有答案的，这个 mex 就是全局 mex，于是可以直接变成 n^2。

证明 令全局 mex 为 K，则说明 $0 \sim K-1$ 的数都存在于数列中而 K 不存在。
假设我们最后每个区间的 mex 均为 X：
若 $X < K$，由于序列中为 X 的数存在，X 必定在其中一个区间中，与所有区间 $mex = X$ 矛盾。
若 $X > K$，由于不存在 K，显然不合法。
于是只能是 $n = k$。

直接转移可以通过测试点 $1 \sim 5$，期望得分 50。

2.4 算法 4

直接讲 $O(n)$ 的做法：

转移方程为 $f_i = \sum f_j[mex([i,j]) = K]$。

对于同一个右端点其对应的左端点的 mex 是单调的，我们可以直接用一个双指针+桶，维护区间的 mex 是否为 K，时间复杂度 $O(n)$。

测试点 6 是给特判的。

测试点 $7 \sim 9$ 对应着不同复杂度的优化转移。

3 T3 超级加倍(charity)

3.1 算法 1

枚举点 x，dfs 一遍，找到所有合法的 y，时间复杂度 $O(n^2)$，期望得分 15。

3.2 算法 2

对于菊花图，分为过根和不过根两种路径分类讨论即可，时间复杂度 $O(n)$，期望得分 25。

3.3 算法 3

对于链，假设我们选出的 x 在 y 左边，可以维护 L_i, R_i 表示 i 左边第一个 $> p_i$ 的数，以及 i 右边第一个 $< p_i$ 的数可以单调栈预处理，两个点合法当且仅当 $R_x > y$ 且 $L_y < x$，可以看成一个三维偏序问题做到 $O(n\log^2 n)$，但是可以忽略 $x < y$ 这个条件，这样会把所有的 $x > y$ 统计为合法，最后减去即可，时间复杂度 $O(n\log n)$，结合之前的算法期望得分 50，如果其中一些步骤做复杂了时间复杂度退化至 $O(n\log^2 n)$ 仅能获得 40 分。

一种更简单的做法是先一遍单调栈求出 L_i，第二次在单调栈中维护所有的当前的后缀最小值，在这个单调栈上进行二分，可以做到 $O(n\log n)$，期望得分 50。

3.4 算法 4

对于树的问题，可以考虑点分治，每次处理过分治中心的点对，一个点合法需要满足是到根的最小值/最大值，另外需要求出路径最小值/最大值来和路径的另外一边合并，合并时仍存在二维偏序的限制，必定会带一个 log，无法优化，时间复杂度 $O(n\log^2 n)$，结合之前的算法期望得分 75，如果写丑了写成 $n\log^3 n$ 仅能获得 65 分。

3.5 算法 5

对比算法 4 和算法 3，发现毫不相关，且点分治本身并没有利用到太多性质，回到算法 3，$R_x > y$ 这类性质可以联想到笛卡尔树，而笛卡尔树在树上的一个很类似的形态是 Kruskal 重构树，按点权从小到大建树建成 T_1，从大到小建成 T_2（满足任意两点 x, y 在 T_1 中的 lca 是路径最小值，在 T_2 中是路径最大值），以上部分可以用并查集实现，那么两点 x, y 满足条件当且仅当 x 在 T_1 中是 y 的祖先，y 在 T_2 中是 x 的祖先，这是一个朴素的偏序问题，在 T_1 中求出 DFS 序，在 T_2 上 DFS，用一个树状数组维护可行答案，时间复杂度 $O(n\log n)$，由于复杂度瓶颈在树状数组，常数非常小，当然，算法 3 也可以用同样的方法建出笛卡尔树然后转化为相同的问题，同样也是 $O(n\log n)$ 的，期望得分 100。

4 T4 欢乐豆(happybean)

4.1 算法 1

使用 Floyed 算法，时间复杂度 $O(n^3)$，可以通过测试点 $1,2,3$，期望得分 12。

4.2 算法 2

对于 $m=0$，发现 $x \to y$ 的最短路一定是直接花费 a_x 从 x 走到 y，即答案为 $\sum a_i \times (n-1)$，直接计算即可，结合算法 1 期望得分 20。

4.3 算法 3

对于特殊性质 A，可以分类讨论，一个点 x 走的方式一定是直接走 x 或者走一条 $x \to y$ 的边再走一个 a_y，分类讨论即可，可通过测试点 $4 \sim 8$。

4.4 算法 4

发现在整张大图中，除了存在改变边权的点，还有很多没有任何改动的点，这启发我们对有改变边权的这些点的导出子图单独考虑。

将每一条改变边权的边视为无向边，得到一些连通块。

考虑 (x,y) 两点间的最短路：

若 (x,y) 不属于同一连通块，那么 $x \to y$ 的路径可以看作：x 在自己的连通块走，走到一个点 z，此时 z 再往外走，由于走的点不是同一连通块，那么出边的边权一定是 a_z，此时直接走到 y 一定最优。此时可以发现找到的点 z 仅需要满足 $dis(x,z)+a_z$ 最小，其中 $dis(x,z)$ 表示同一连通块两点间的距离。

也就是说我们把问题转化为了求出所有同一连通块的 $dis(x,y)$。

对于边数 m 只有 500，其涉及的最大的连通块大小只有 501，可以直接 Floyed 求解，结合之前的算法可以通过测试点 $1 \sim 12$，期望得分 48。

注意这里有一个小细节，一条边 $x \to y$ 可能在边权被修改之后边权非常大，此时可以先从 x 走到 z，再从 z 走到 y，其中 z 是连通块外部的一点，可以发现选取的一定是 a 最小的 z。

4.5 算法 5

再看求的这个问题，可以看做如下形式：

有一个 n 个点 m 条边的有向图，$n,m \le 3001$，每条有向边有边权，若一条边 $x \to y$ 不存在于图上，则视为存在一条 $x \to y$ 的有向边，边权为 a_x，也就是说，在这个图上，所有点的出边边权基本相同，除了不超过 m 条边，我们需要求的是全源最短路。

朴素的最短路算法是无法解决的，但我们可以用数据结构优化。

由于边权非负，可以模拟 Dijkstra 算法，我们需要执行两种操作：

1. 取出最全局最小值。
2. 用边权更新权值。

使用线段树，动态维护每个点的 dis，用 x 出边更新时将所有的出边指向的点 y 排序，看成给所有的 y_i 单点取 min，给所有的区间 (y_i, y_{i+1}) 区间取 min，这两部分不重叠所以不会出现重复转移，由于边数总共是 m，所以涉及到的单点操作和区间操作都是 $O(m)$ 级别的，最后在将 x 在线段树上单点修改为 inf，由于线段树的所有操作都是 $O(\log m)$ 的，所以总复杂度 $O(m^2 \log m)$。

特殊性质 B 是给的是同样用优化的最短路算法但没有处理好重复转移的问题，也就是说直接用 a_x 去转移 x 的所有出边，然后再用每一条特殊边去转移，这样可能存在更好写的写法。

最后加上算法 4 中的分类讨论，总时间复杂度为 $O(n + m^2 \log m)$，期望得分 100。