NOIP 2019 模拟赛 Day 2

diamond_duke

题目名称	字符串	散步	树
可执行文件名	`string`	`walk`	`tree`
输入文件名	标准输入	标准输入	标准输入
输出文件名	标准输出	标准输出	标准输出
时间限制	1s	2s	4s
内存限制	512MB	512MB	512MB
子任务个数	4	4	6
题目类型	传统型	传统型	传统型

请注意： 评测时开启 O2 优化和 C++11 编译选项，栈空间限制同空间限制。

1 字符串

首先 T 必须得要是 S 的子序列，不然不存在好的下标序列，因此一定无解。

考虑判断一个串 T 是不是 S 子序列的贪心做法：每次从没有匹配的位置中，选择第一个和 T_i 一样的与 T_i 进行匹配。设这样得到的下标序列是 $p_1, p_2, \cdots, p_m$，则显然这是一个好的下标序列。

从刚刚贪心的过程中，我们可以发现，p_1 是所有可能的位置中最小的，p_2 是在满足 p_1 最小的情况下最小的，p_3 是在满足 p_1, p_2 都最小的情况下最小的……则这样得到的序列是所有好的下标序列中，字典序最小的那个。

我们接下来考虑调整这个好的下标序列，使它变成优秀的。我们按照从后往前的顺序依次考虑 S 中所有满足相邻两个字母不同的位置，设为 i 和 $i+1$。根据题意，我们要求 i 和 $i+1$ 中至少有一个在这样的下标序列中，这样的下标序列才会是优秀的。容易证明，对于所有这样的 i，都满足 i 和 $i+1$ 中的至少一个在序列中的好的下标序列一定是优秀的。

如果 i 或者 $i+1$ 已经在序列 p 中了，那么已经符合条件了。否则，我们考虑调整：因为 p 已经是字典序最小的序列了，所以我们无法把某个 p_j 变得更小。因此，我们可以移动的位置一定是满足 $p_j < i$ 的一段前缀。

因为 $S_i \neq S_{i+1}$，所以 $T_j = S_i$ 或 $T_j = S_{i+1}$ 之一一定满足。于是，我们一定可以把 p_j 调整为 i 或 $i+1$ 之一，从而使得 i 满足条件。如果有多个 j 满足这一条件，则我们应当选择最大的那个，容易证明这样一定不劣。如果不存在这样的 j，根据 p 是字典序最小的好的序列，其他序列一定也不存在这样的 j，因此一定无解。

时间复杂度：$\Theta(n)$。

2　散步

2.1　子任务 2

二分答案，这样问题就转化为了判断是否能找到一条路径，使得连续的 `A` 或 `B` 的长度不超过 T。

对于这个问题，我们可以把原图的每个点拆成 $2T$ 个，表示当前已经有连续 k 个 `A` 或 k 个 `B` 在路径的末尾，转移比较显然。

这样问题就转化成了判断图的连通性，直接 DFS 即可。

时间复杂度：$\Theta(n^2 \log n)$。

2.2　子任务 3

上面算法的问题在于把这个问题看得过于一般了，没有用到每个点只有一条 `A` 出边和 `B` 出边的性质。

实际上，在之前的图中转移的时候，如果是 k 个 `A` 转移到 $k+1$ 个 `A`，那么只有一种转移方向。因此，我们可以只考虑从 k 个 `A` 到 1 个 `B` 这样形式的转移，即每个点只保留往 `A` 走的和 `B` 走的两种状态，而转移是往这个方向走 k 步所经过的所有节点。

这样的图有一种经典的优化建图方式：倍增优化建图，这样建出来的图的节点数和边数都是 $\Theta(n \log n)$ 级别的。再加上二分的复杂度，因此总复杂度为 $\Theta(n \log^2 n)$。

由于常数较大，可能无法通过最后一个子任务。

2.3　子任务 4

注意到在上一个做法中，我们只需要找到未访问的点去更新。

因此，可以把问题转化成每次标记一个点，然后取出所有未标记的、从某个点出发沿 `A` 或 `B` 走 k 步所能到的位置。

这个问题如果 `A` 的边或 `B` 的边形成了一棵树，可以用带权并查集解决：如果当前点未访问，则令这个点的父亲等于自己，否则等于不断往 `A` 走走到的第一个未访问的点，并记录下距离即可。

对于基环树的情况其实也可以类似地考虑，只是要注意不能在环上绕多圈。所以在一个环被全部访问完之后，只需要把环断开成链即可。

时间复杂度：$\Theta(n \log^2 n)$。由于并查集的 $\log$ 常数非常小，所以可以获得满分。

3 树

对于一个询问，设路径 (u,v) 经过的所有边的 gcd 为 g，这可以倍增求出。

考虑 g 的所有质因子 $p_1,p_2,\cdots,p_k$，因为 $g\le 10^6$，所以 $k\le 7$。

则最终的路径的 gcd 为 1，等价于对于每个 $1\le i\le k$，存在至少一条路径上的边不是 p_i 的倍数。我们要求 l 的最小值，即等价于对于每个 $1\le i\le k$，计算出最长的不满足条件的 l'，则最终答案即为所有 i 对应的 l' 的最大值加一（无解的情况除外）。

考虑对于某个 p_i 而言，我们如何求出这样的 l'。我们考虑将所有满足 $p_i\mid w$ 的边拿出来，并只保留这些边。则 l' 等价于在这样得到的森林中，经过 (u,v) 的最长路径。使用简单的树形 DP 即可求出某个点向子树方向以及向祖先方向延伸的最长路径，分类讨论即可对于每个 (u,v) 求出对应的 l'。

接下来考虑无解的情况。事实上，无解等价于刚刚求出的某个 l' 和经过 (u,v) 的最长路径相同。经过 (u,v) 的最长路径和刚刚是同样的问题，直接对整棵树都做一遍树形 DP 即可。

接下来考虑复杂度。求出每个询问的 gcd 的复杂度为 $\Theta(q\log_2 n\log_2 w)$，而求出最长路的部分是与边数成线性的，而每条边至多出现 7 次，因此该做法的总复杂度即为 $\Theta(q\log_2 n\log_2 w+n\omega(w))$。