3 ioday2

题解中，两点间的距离指两点间路径的边权和。

3.1 算法 1

暴力枚举每个点是否出现，然后暴力求最小生成树。

如果你使用 Kruskal 算法，时间复杂度 $\mathcal{O}(n^2 2^n \log n)$。可以通过 Subtask1，获得 10 分。

3.2 算法 2

对于一条链的情况，显然答案就是选中的点之间树上距离最远的点对的树上距离。

枚举距离最远的点对，不在这两个点之间的链上的点一定不能被选中，在两点之间的链上的点可被选中也可不被选中，这两个点必须被选中。算一下概率，乘以这两点之间的距离加进答案即可。

时间复杂度 $\mathcal{O}(n^3)$。可以通过 Subtask2，获得 20 分。

3.3 算法 3

对于 $p_i \in \{0, 1\}$ 的情况，只需求出新图的最小生成树即可。

~~好像这个 Subtask 暴力就能过。~~

下面我们介绍一种求新图的最小生成树的算法。

任取一个点作根，设点 i 与根的距离为 d_i。

按与根的距离从小到大的顺序把选中的点排序，从第二个点开始依次遍历这些点，遍历到一个点 u 时，取 u 前面与 u 距离最近的点作新图上 u 的父亲。这样得到的一个新图的有根生成树一定是新图的一个最小生成树。

证明写在了题解最后。

用这个算法求最小生成树，根据询问两点间距离时求 lca 的算法，时间复杂度为 $\mathcal{O}(n^2 \log n)$ 或 $\mathcal{O}(n^2)$。

3.4 算法 4

把所有点按与根的距离排序，考虑算法 3 的过程，根据期望的线性性，只需要计算每个点到父亲的距离的期望，加起来即可。对于每个点 u，把它之前的点按到 u 的距离从小到大排序，分别计算 u 的父亲是每个点的概率（u 的父亲是某一个点的概率即是这个点被选中，且它前面的点都没有被选中的概率），乘以距离加进答案即可。

根据实现的不同，时间复杂度 $\mathcal{O}(n^2)$ 或 $\mathcal{O}(n^2 \log n)$，可以通过所有子任务，获得 100 分。

3.5 算法 3 的证明

假设把 $1 \dots n$ 按 d_i 排序后，点 i 出现在位置 p_i。

设被选中的点中到根距离最小的点为 r。

如果存在一个新图的最小生成树，取 d_r 最小的点 r 做根后，对于任意点 $u \neq r$，有 $p_{father(u)} < p_u$，那么算法 3 显然能够求出新图的一个最小生成树。

只需证明存在一个这样的最小生成树。

任取一个新图的最小生成树，设以 r 为根，u 在这个最小生成树上的父亲是 $father(u)$。找到 p_u 最大的满足 $p_{father(u)} > p_u$ 的点 u。设 $v = father(u), w = father(father(u))$，因为我们取的是 p_u 最大的 u，而 $p_v > p_u$，所以 $p_w < p_v$。我们对 u, v, w 三个点在树上的位置关系作讨论。

设 $l_1 = lca(u, v), l_2 = lca(v, w)$，点 u 到点 v 的距离是 $dis(u, v)$。

情况 1 $l_1 = l_2$ 或 l_1 是 l_2 的祖先。

考虑进行如下操作。

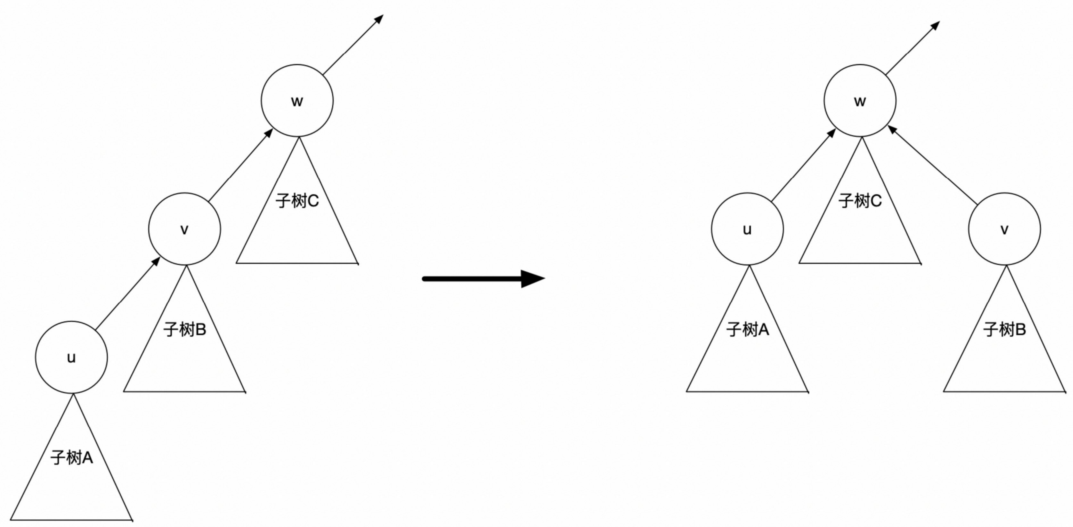

Figure 1: 情况 1 的操作

这一操作只改变了 u 的父亲。$dis(u,w) = d_u + d_w - 2d_{l_1} \leq d_u + d_v - 2d(l_1) = dis(u,w)$。所以这个操作不会使生成树上边权和变大，操作后得到的是另一个最小生成树。

情况 2 $l_1 \neq l_2$ 且 l_2 是 l_1 的祖先。

考虑进行如下操作。

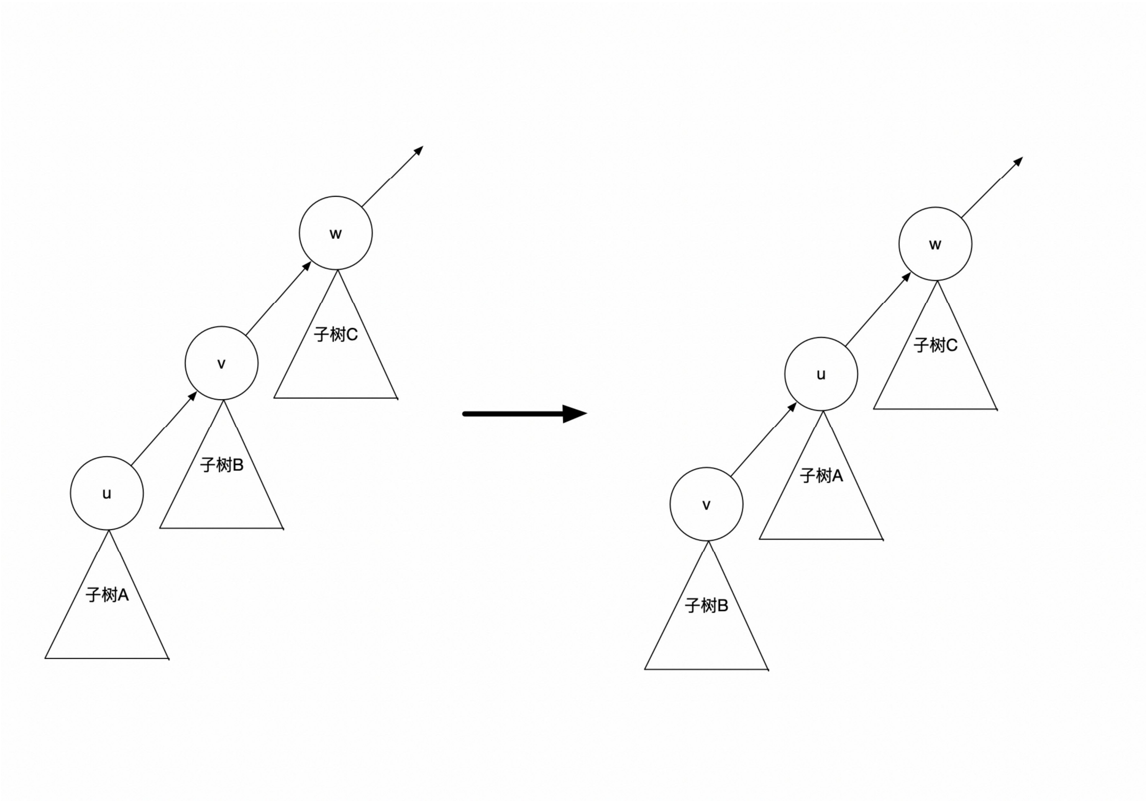

Figure 2: 情况 2 的操作

这一操作只改变了 u,v 的父亲。$dis(v,u) + dis(u,w) \leq dis(u,v) + d_u + d_w - 2d_{l_2} \leq dis(u,v) + d_v + d_w - 2d_{l_2} = dis(u,v) + dis(v,w)$。所以这个操作也不会使生成树上边权和变大，操作后得到的是另一个最小生成树。

容易发现，这两种操作都只会使满足 $p_{father(u)} > p_u$ 的点 u 的个数减少。只要任取一棵最小生成树，不断进行操作，一定能在有限步内变成不存在 $p_{father(u)} > p_u$ 的点 u 的最小生成树。这就完成了证明。