《蜘蛛爬树》解题报告

孟煜皓

1 题意简述

给定一棵 n 个点的树，第 i 条边连接 u_i 和 v_i，边权为 w_i。节点从 1 开始编号。

将这棵树复制 m 份，第 k 棵树的节点 i 被重编号为 $(k-1)n+i$。对于任意 $1 \le k < m, 1 \le x \le n$，将第 k 棵树的节点 x 与第 $k+1$ 棵树的节点 x 连边权为 a_x 的边。

定义一条路径的长度为，该路径经过的边的边权之和。其中重复经过的边会被计算多次。

q 次询问，第 j 次询问新图中 s_j 到 t_j 的最短路径长度。

2 数据范围

$1 \le n, q \le 2 \times 10^5, 1 \le m \le 10^9, 1 \le a_x \le 10^9, 1 \le w_i \le 10^{12}$。

3 子任务

- 子任务 1（3 分）：$n \le 100, m \le 100, q \le 1000$。
- 子任务 2（5 分）：$n, q \le 5000$。
- 子任务 3（11 分）：$m \le 20$。
- 子任务 4（12 分）：树是一条链。
- 子任务 5（9 分）：树是一个菊花。
- 子任务 6（22 分）：$s_j = 1$。
- 子任务 7（18 分）：$n, q \le 5 \times 10^4$。
- 子任务 8（20 分）：无附加限制。

4 解题思路

4.1 子任务 1

按照题意建出新图。对于每次询问，使用堆优化 Dijkstra 算法求出单源最短路径即可。

时间复杂度 $O(qnm\log(nm))$。

4.2 子任务 2

为方便描述，我们规定一些记号：

- (k,x) 表示编号为 $(k-1)n+x$ 的节点。
- $\to$ 表示仅经过树上的边的简单路径，$\Rightarrow$ 表示仅经过相邻两棵树之间的边的简单路径。
- $\mathrm{dist}(s,t)$ 表示原树上 s 到 t 的简单路径长度。

设起点和终点分别为 $(i,s),(j,t)$。有引理：

引理 4.2.1. *存在 $1\le x\le n$，使得 $(i,s)\to(i,x)\Rightarrow(j,x)\to(j,t)$ 为 (i,s) 到 (j,t) 的最优路径。*

证明. 考虑任意一条 (i,s) 到 (j,t) 的路径 $(k_0,x_0)\to(k_0,x_1)\Rightarrow(k_1,x_1)\to(k_1,x_2)\Rightarrow(k_2,x_2)\to(k_2,x_3)\Rightarrow\cdots\to(k_{p-1},x_p)\Rightarrow(k_p,x_p)\to(k_p,x_{p+1})$，其中 $k_0=i,x_0=s,k_p=j,x_{p+1}=t$。该路径的长度为

$$\sum_{i=0}^{p}\mathrm{dist}(x_i,x_{i+1})+\sum_{i=1}^{p}|k_i-k_{i-1}|a_{x_i} \tag{1}$$

令 $x_{\min}$ 为 $x_1,x_2,\ldots,x_p$ 中满足 a_{x_i} 最小的 x_i。我们可以将该路径调整为 $(k_0,x_0)\to(k_0,x_{\min})\Rightarrow(k_p,x_{\min})\to(k_p,x_{p+1})$，则路径长度为

$$\mathrm{dist}(x_0,x_{\min})+\mathrm{dist}(x_{p+1},x_{\min})+|k_p-k_0|a_{x_{\min}} \tag{2}$$

由于 w_i 与 a_x 均非负，所以有

$$\sum_{i=0}^{p}\mathrm{dist}(x_i,x_{i+1})\ge\mathrm{dist}(x_0,x_{\min})+\mathrm{dist}(x_{p+1},x_{\min})$$
$$\sum_{i=1}^{p}|k_i-k_{i-1}|a_{x_i}\ge|k_p-k_0|a_{x_{\min}}$$

所以 (1) 式不小于 (2) 式。也就是说，(i,s) 到 (j,t) 的任意一条路径都可以调整为 $(i,s)\to(i,x)\Rightarrow(j,x)\to(j,t)$ 的形式，且长度不增。 □

根据上述引理，令 $k=|i-j|$，则 (i,s) 到 (j,t) 的最短路径长度为

$$\min_{1\le x\le n}(\mathrm{dist}(s,x)+\mathrm{dist}(x,t)+a_xk) \tag{3}$$

预处理原树上所有点对的最短路，每次询问枚举 x 即可。时间复杂度 $O(n^2+nq)$。

4.3 子任务 3

对于每一个 k，使用换根 DP 预处理

$$f_p=\min_{1\le x\le n}(2\cdot \mathrm{dist}(p,x)+a_xk)$$

(3) 式等价于

$$\mathrm{dist}(s,t)+\min_{p\in \mathrm{path}(s,t)} f_p$$

其中 $\mathrm{path}(s,t)$ 表示树上 s 到 t 的简单路径经过的点集，包含 s 和 t。
使用倍增算法，或者树链剖分后使用线段树维护，即可快速求出上式的值。
时间复杂度 $O(nm\log n+q\log n)$、$O(nm+q\log^2 n)$ 或 $O(nm+q\log n)$。

4.4 子任务 4

仍然使用 (3) 式计算答案。分别考虑 x 在 s 到 t 之间、s 前面和 t 后面的情况。
对于 x 在 s 到 t 之间的情况，(3) 式等价于

$$\mathrm{dist}(s,t)+k\min_{x\in \mathrm{path}(s,t)} a_x$$

将 a 按链上的顺序排列后，维护区间最小值即可。
对于 x 在 s 前面的情况，设链头为 r，记 $d_u=\mathrm{dist}(r,u)$。此时 (3) 式等价于

$$\mathrm{dist}(s,t)+2d_s+\min_{x\in \mathrm{path}(r,s)}(a_xk-2d_x)$$

将询问离线，从前往后扫描，使用李超树维护直线 a_xk-2d_x 构成的上凸壳，并在扫到 s 时在凸壳上查询 k 处的点值即可。
x 在 t 后面的情况与上一种情况类似，不再赘述。
时间复杂度 $O((n+q)\log n)$。

4.5 子任务 5

设 r 为菊花的中心。将 r 设为树的根，记 $d_u=\mathrm{dist}(r,u)$。
对于所有叶子 x，维护直线 a_xk+2d_x 构成的上凸壳。
特殊处理 $x=s$、$x=t$ 和 $x=r$ 的情况，其余情况可以在凸壳上查询 k 处的点值求出。
时间复杂度 $O((n+q)\log n)$。

4.6 子任务 6

下文中，我们在考虑一个询问的答案时，将 (3) 式先减去 $\mathrm{dist}(s,t)$，最后再加上。

以 1 为根，对原树进行树链剖分。记 $d_u=\mathrm{dist}(1,u)$，$\mathrm{top}(u)$ 为 u 所属重链的链头，$\mathrm{fa}(u)$ 为 u 的父亲。

考虑从 t 往根不断跳重链和轻边的过程。

当跳过重链 $p\to\mathrm{top}(p)$ 时，用 $p\to\mathrm{top}(p)$ 每个点及其所有轻子树中的点 x 更新答案。具体地，考虑将跳的过程离线，将当前询问挂在点 p 处。对树进行 DFS，在 DFS 的过程中，假设当前点为 p，对于 $p\to\mathrm{top}(p)$ 的每个点 v 及其轻子树中的点 x，使用李超树维护直线 $a_xk+2(d_x-d_v)$ 构成的上凸壳。通过在凸壳上查询更新挂在 p 处的询问的答案。

当跳过轻边 $p\to\mathrm{fa}(p)$ 时，用 $\mathrm{fa}(p)$ 子树内的所有点 x 更新答案。具体地，仍然将询问离线并将当前询问挂在 $\mathrm{fa}(p)$ 处。在对树进行 DFS 的过程中，假设当前点为 p，对于 p 子树内的点 x，使用李超树合并维护直线 a_xk+2d_x 构成的上凸壳。然后在凸壳上查询，并减去 $2d_p$，更新挂在 p 处的询问的答案。

时间复杂度 $O((n+q)\log^2 n)$。

4.7 子任务 7

点分治，设当前分治中心为 r。我们只保留 s,t 均在当前连通块内的询问。

对于所有经过 r 的询问 $s\to t$，拆成 $s\to r$ 和 $t\to r$，使用与子任务 6 相同的方法解决。

对于不经过 r 的询问 $s\to t$，用当前连通块的所有点 x 更新答案。具体地，记 $d_x=\mathrm{dist}(r,x)$，求出直线 a_xk+2d_x 构成的上凸壳。对于每个询问，在凸壳上查询 k 处的值，并加上 $2d_{\mathrm{LCA}(s,t)}$ 更新答案。其中 $\mathrm{LCA}(s,t)$ 表示当前连通块以 r 为根时 s,t 的最近公共祖先。

时间复杂度 $O(n\log^3 n+q\log^2 n)$。

事实上，由于该做法常数较小，若实现较为优秀，可以直接通过本题。

4.8 子任务 8

沿用子任务 6 的做法。

记 $\mathrm{LCA}(u,v)$ 表示 u,v 的最近公共祖先。

我们先使用类似子任务 6 的做法，将 s,t 跳到 $\mathrm{LCA}(u,v)$ 处。

注意此时最后一条重链不一定能表示成 $p\to\mathrm{top}(p)$，而可能是某条重链的一个区间。但是这样的区间只有一个，我们可以单独使用线段树套李超树处理。为了降低空间复杂度，可以将这些区间再次离线并挂在线段树上，然后在线段树上使用李超树合并求出所有询问的答案。

接下来考虑 $\mathrm{LCA}(s,t)$ 子树外的点对该询问的影响。

对于 $\mathrm{LCA}(s,t)$ 子树外的点 x，其贡献为 $a_x k + 2 \cdot \mathrm{dist}(\mathrm{LCA}(s,t), x)$。注意到该式仍然可以使用类似于子任务 6 的做法进行处理。

具体区别是，对于子任务 6 中的询问点 t，一个点 x 的贡献是 $a_x k + 2d_x - 2d_{\mathrm{LCA}(x,t)}$；在当前问题中，对于询问点 $y = \mathrm{LCA}(s,t)$，一个点 x 的贡献为 $a_x k + 2d_x - 4d_{\mathrm{LCA}(x,y)}$，最后还要加上 $2d_y$。

时间复杂度 $O((n+q)\log^2 n)$，空间复杂度 $O((n+q)\log n)$。

另外，上述算法中的每一部分都可以看成是树链剖分后对一个元素为直线的序列的某个区间构成的凸壳的查询，所以也可以在线段树上维护凸壳实现。时间复杂度仍然为 $O((n+q)\log^2 n)$。

5 参考资料

无。

6 验题

感谢精英培训选手张湫阳、何卓锟，以及王陈然等其他一些本校同学参与验题。