# 浅谈一类树上范围相关问题

朱屹帆

长沙市长郡中学

2024 年 1 月

# 前言

树上问题作为一类经典的问题，在信息学竞赛中经常出现，而树上范围相关问题是其中比较重要且具有广泛运用的一类问题。笔者对树链剖分，点分树，基础 Top Tree 进行研究，详细分析其与维护树上范围相关的一些性质，并结合例题进行总结。

# 前言

树上问题作为一类经典的问题，在信息学竞赛中经常出现，而树上范围相关问题是其中比较重要且具有广泛运用的一类问题。笔者对树链剖分，点分树，基础 Top Tree 进行研究，详细分析其与维护树上范围相关的一些性质，并结合例题进行总结。
本文中在通常情况下定义树的大小为 $O(n)$ 级别。

# 树链剖分

重链剖分是树链剖分的一种类型，在处理树上范围相关问题中，重链剖分有着重要的作用。我们默认读者已经掌握重链剖分相关的若干定义和性质。

# 重链剖分的一种理解：树上链分治

我们尝试从类似点分治的角度来理解重链剖分——树上链分治，点分治可以将树上的一条路径唯一对应到点分治中经过一个分治重心的两条路径，我们希望用类似的方式将树上的一条路径唯一对应到重链剖分上的一条重链区间上，将一条路径的唯一对应到的重链区间设置为路径中深度最小的节点的所在重链与路径的交即可。

# 重链剖分的一种理解：树上链分治

我们尝试从类似点分治的角度来理解重链剖分——树上链分治，点分治可以将树上的一条路径唯一对应到点分治中经过一个分治重心的两条路径，我们希望用类似的方式将树上的一条路径唯一对应到重链剖分上的一条重链区间上，将一条路径的唯一对应到的重链区间设置为路径中深度最小的节点的所在重链与路径的交即可。

点分树可以处理一些涉及单点范围信息的维护，因此我们不妨尝试使用树链剖分来处理一些涉及路径范围信息的维护。

# 例题

给定一个 $n$ 个节点的树，树边权均为 $1$ ，每个节点有点权，初始点权均为 $0$ ，有 $m$ 次操作，需要支持两种操作：

1. 给定路径端点 $u, v$ ，范围 $d$ 和权值 $k$ ，将所有满足与路径 $(u, v)$ 上的任意节点的最短距离 $\leq d$ 的节点点权增加 $k$ 。
2. 给定单点 $u$ ，查询点 $u$ 的权值。

数据范围： $1 \leq n, m \leq 10^5$ 。

# 重链剖分的一种理解：树上链分治

在点分树中，对于当前分治层上分治重心的标记，是作用在以分治重心为根的子树上的；因此考虑在树链剖分结构上进行类似的标记，即节点 $x$ 的标记作用在节点 $x$ 的所有轻儿子的子树中，修改时分析路径在重链剖分上的标记位置及其影响，查询时对单点本身的标记和单点的祖先上所有轻边对应的父亲节点的标记进行查询。

# 重链剖分的一种理解：树上链分治

在点分树中，对于当前分治层上分治重心的标记，是作用在以分治重心为根的子树上的；因此考虑在树链剖分结构上进行类似的标记，即节点 $x$ 的标记作用在节点 $x$ 的所有轻儿子的子树中，修改时分析路径在重链剖分上的标记位置及其影响，查询时对单点本身的标记和单点的祖先上所有轻边对应的父亲节点的标记进行查询。

先考虑直接维护贡献，对原树进行重链剖分，对于一次修改操作，将路径 $(u, v)$ 经过的所有重链区间取出，并对每一个重链区间进行标记，形如对于一个在重链区间内的点 $x$，其所有轻儿子的子树内与点 $x$ 距离在范围 $d$ 内的所有节点点权增加 $k$，注意受点 $x$ 的标记影响的节点为点 $x$ 及其轻儿子子树内的所有节点。

# 重链剖分的一种理解：树上链分治

但注意到这样会有三个部分的贡献需要处理：

# 重链剖分的一种理解：树上链分治

但注意到这样会有三个部分的贡献需要处理：
第一个部分是重链区间之后的部分的贡献，形如与重链区间右端点距离在范围 $d$ 内的右端点重子树内的所有节点点权增加 $k$；

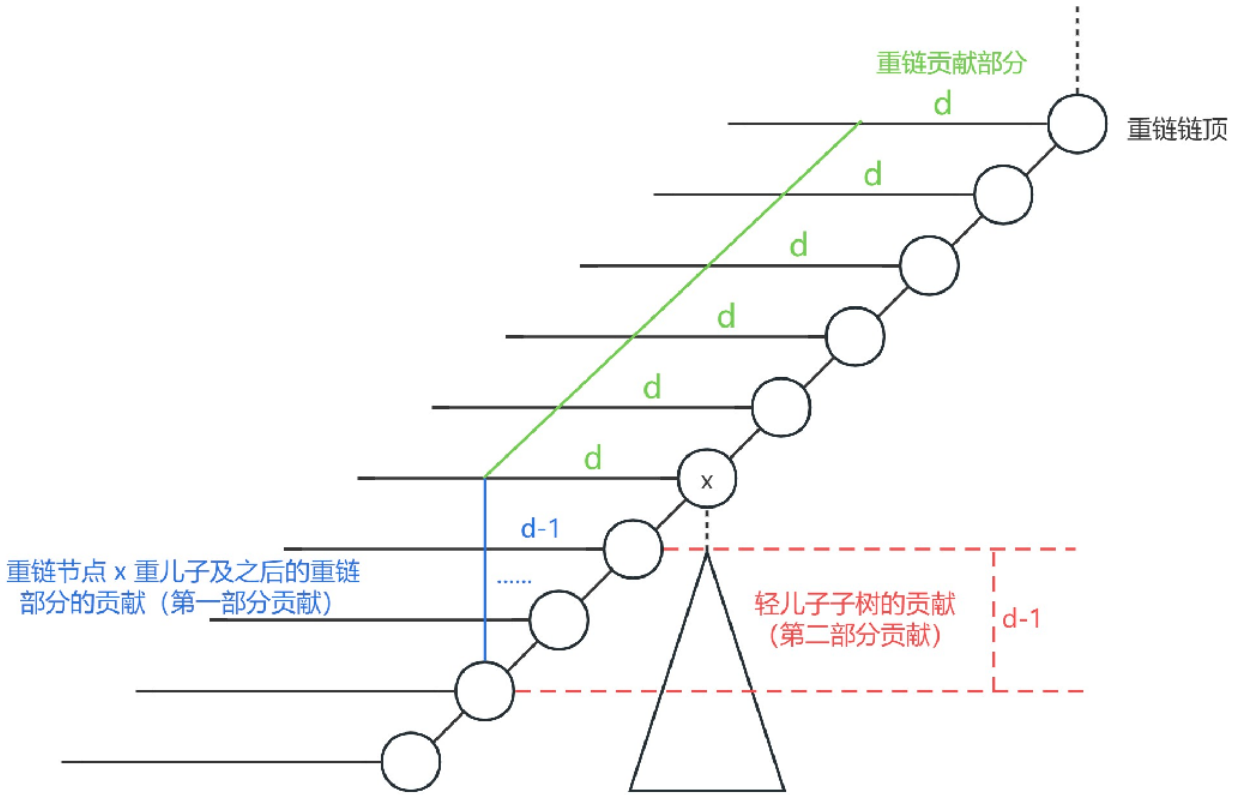
# 重链剖分的一种理解：树上链分治

但注意到这样会有三个部分的贡献需要处理：
第一个部分是重链区间之后的部分的贡献，形如与重链区间右端点距离在范围 $d$ 内的右端点重子树内的所有节点点权增加 $k$；
第二个部分是通过轻边跳到一条新的重链时，对重链区间上轻儿子子树的操作会影响到跳上来时的轻儿子子树，原来的轻儿子的贡献需要去掉重复计算部分；

# 重链剖分的一种理解：树上链分治

但注意到这样会有三个部分的贡献需要处理：
第一个部分是重链区间之后的部分的贡献，形如与重链区间右端点距离在范围 $d$ 内的右端点重子树内的所有节点点权增加 $k$；
第二个部分是通过轻边跳到一条新的重链时，对重链区间上轻儿子子树的操作会影响到跳上来时的轻儿子子树，原来的轻儿子的贡献需要去掉重复计算部分；
第三个部分是跳到路径深度最小的节点之后其上部分需要添加贡献。

# 重链剖分的一种理解：树上链分治




图：重链部分贡献，第一部分贡献，第二部分贡献

# 重链剖分的一种理解：树上链分治

考虑一条重链，对于一个节点的标记为 $d$，可以发现：直接维护贡献部分的标记形如重链区间上一段标记为 $d$；第一个部分设重链区间右端点为 $r$，形如重链以 $r$ 为端点的一段后缀上的点 $x$ 的标记为 $d-\mathrm{dist}(r,x)$；第三个部分设路径深度最小的节点为 $lca$，形如重链以 $lca$ 到该重链的轻边父节点 $y$ 为端点的一段前缀上的点 $x$ 的标记为 $d-\mathrm{dist}(lca,y)-\mathrm{dist}(x,y)$。对每条重链维护区间所有相同标记信息，区间左端点向右标记递减标记信息，区间右端点向左标记递减标记信息。

# 重链剖分的一种理解：树上链分治

考虑一条重链，对于一个节点的标记为 $d$，可以发现：直接维护贡献部分的标记形如重链区间上一段标记为 $d$；第一个部分设重链区间右端点为 $r$，形如重链以 $r$ 为端点的一段后缀上的点 $x$ 的标记为 $d-\mathrm{dist}(r,x)$；第三个部分设路径深度最小的节点为 $lca$，形如重链以 $lca$ 到该重链的轻边父节点 $y$ 为端点的一段前缀上的点 $x$ 的标记为 $d-\mathrm{dist}(lca,y)-\mathrm{dist}(x,y)$。对每条重链维护区间所有相同标记信息，区间左端点向右标记递减标记信息，区间右端点向左标记递减标记信息。

第二个部分注意到重复计算的部分恰好为一个重链的顶部节点的子树内的范围为 $d-1$ 内的节点，在每条重链顶部维护标记。

# 重链剖分的一种理解：树上链分治

考虑一条重链，对于一个节点的标记为 $d$，可以发现：直接维护贡献部分的标记形如重链区间上一段标记为 $d$；第一个部分设重链区间右端点为 $r$，形如重链以 $r$ 为端点的一段后缀上的点 $x$ 的标记为 $d-\mathrm{dist}(r,x)$；第三个部分设路径深度最小的节点为 $lca$，形如重链以 $lca$ 到该重链的轻边父节点 $y$ 为端点的一段前缀上的点 $x$ 的标记为 $d-\mathrm{dist}(lca,y)-\mathrm{dist}(x,y)$。对每条重链维护区间所有相同标记信息，区间左端点向右标记递减标记信息，区间右端点向左标记递减标记信息。

第二个部分注意到重复计算的部分恰好为一个重链的顶部节点的子树内的范围为 $d-1$ 内的节点，在每条重链顶部维护标记。

查询单点时，对单点本身标记和单点祖先上所有轻边对应父亲节点的标记进行查询，对每条重链顶部的标记进行查询即可。

# 重链剖分的一种理解：树上链分治

考虑一条重链，对于一个节点的标记为 $d$，可以发现：直接维护贡献部分的标记形如重链区间上一段标记为 $d$；第一个部分设重链区间右端点为 $r$，形如重链以 $r$ 为端点的一段后缀上的点 $x$ 的标记为 $d-\mathrm{dist}(r,x)$；第三个部分设路径深度最小的节点为 $lca$，形如重链以 $lca$ 到该重链的轻边父节点 $y$ 为端点的一段前缀上的点 $x$ 的标记为 $d-\mathrm{dist}(lca,y)-\mathrm{dist}(x,y)$。对每条重链维护区间所有相同标记信息，区间左端点向右标记递减标记信息，区间右端点向左标记递减标记信息。

第二个部分注意到重复计算的部分恰好为一个重链的顶部节点的子树内的范围为 $d-1$ 内的节点，在每条重链顶部维护标记。

查询单点时，对单点本身标记和单点祖先上所有轻边对应父亲节点的标记进行查询，对每条重链顶部的标记进行查询即可。

使用线段树维护每条重链的标记，时间复杂度 $O(n\log^3 n)$，使用全局平衡二叉树维护每条重链的标记，时间复杂度 $O(n\log^2 n)$。

# 基于调整搜索顺序的重标号树链剖分

树链剖分的性质非常丰富且非常广泛，在一些题目中我们需要维护一些复杂的信息，树链剖分的其中一些性质非常适合维护这类信息，但受到其结构的影响会导致一些信息不太方便很好的合并，此时可以考虑通过改变其结构来保留其中一些有用的性质并更好的维护信息。

# 例题：数据结构

小 Z 有一棵 $n$ 个节点构成的树，这棵树以点 $1$ 为根，边权均为 $1$ ，每个节点 $x$ 有点权 $a_x$ ，初始所有节点的权值为 $0$ 。
小 Z 想对这棵树进行总共 $m$ 次操作，并且小 Z 希望能够支持以下四种类型的操作：
1. 给定 $x, y, k, v$ ，对满足节点 $i$ 到树上唯一的最短路径 $(x, y)$ 的最短距离 $\leq k$ 的所有节点 $i$ 的权值增加 $v$ ，即 $a_i \leftarrow a_i + v$ 。
2. 给定 $x, v$ ，将点 $x$ 的子树内的所有节点 $i$ 的权值增加 $v$ ，即 $a_i \leftarrow a_i + v$ 。
3. 给定 $x, y, k$ ，查询所有满足节点 $i$ 到树上唯一的最短路径 $(x, y)$ 的最短距离 $\leq k$ 的所有节点 $i$ 的权值的和。
4. 给定 $x$ ，查询点 $x$ 的子树内的所有节点 $i$ 的权值和。

由于答案会比较大，输出对 $2^{64}$ 取模即可。数据范围：
$1 \leq n, m \leq 2 \times 10^5$ ，$1 \leq x, y \leq n$ ，$1 \leq v \leq 10^9$ ，$0 \leq k \leq 3$ 。

# 基于调整搜索顺序的重标号树链剖分

考虑到当 $k=0$ 时，我们可以依据重链剖分的性质，由于路径至多跳 $\log n$ 条轻边，因此在 $\log n$ 条重链上进行处理，而重链的标号具有连续性，因此可以支持快速对点进行修改和查询，这启发我们尝试在保持重链剖分考虑 $\log n$ 条重链进行处理的性质的基础上，通过稍微调整重链的标号，使得重链上与重链上节点距离较小的节点标号大致连续，方便快速修改。

# 基于调整搜索顺序的重标号树链剖分

考虑到当 $k=0$ 时，我们可以依据重链剖分的性质，由于路径至多跳 $\log n$ 条轻边，因此在 $\log n$ 条重链上进行处理，而重链的标号具有连续性，因此可以支持快速对点进行修改和查询，这启发我们尝试在保持重链剖分考虑 $\log n$ 条重链进行处理的性质的基础上，通过稍微调整重链的标号，使得重链上与重链上节点距离较小的节点标号大致连续，方便快速修改。

笔者在下文中将这种在保持树链剖分部分性质的基础上，通过调整标号顺序便于维护信息的方式称为“重标号树链剖分”，这里介绍本题的一种标号方式：

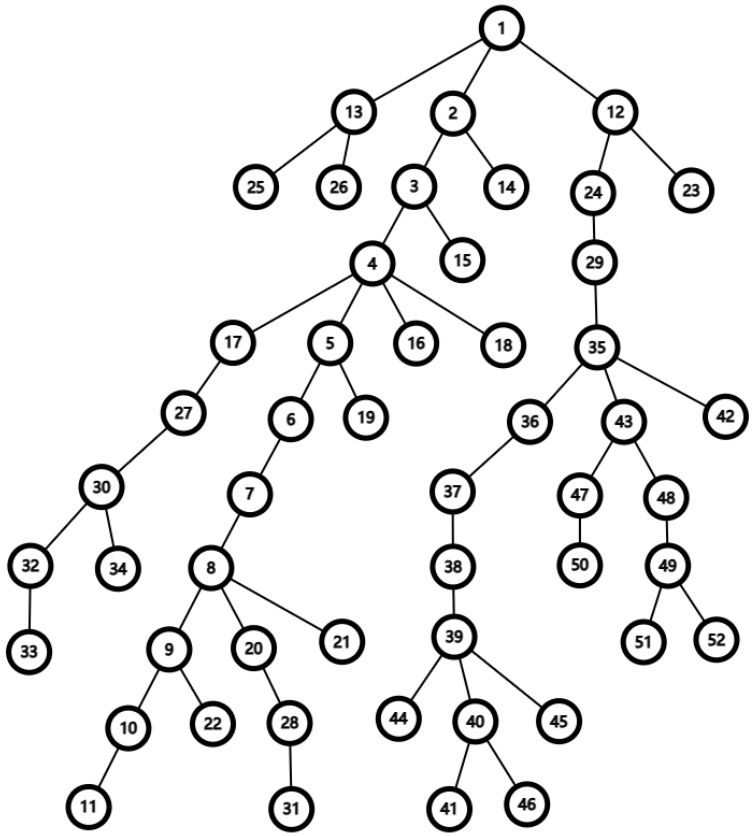
# 基于调整搜索顺序的重标号树链剖分

按照重链中深度最小的点的 dfs 顺序来处理每一条重链，考虑对于一条重链，先将这条重链上的所有点按照深度从小到大排序进行标号，然后依次枚举距离 $d$ 为 $1$ 和 $2$ 和 $3$ ，每次将这条重链上的所有点按照深度从小到大处理每个点，处理到点 $x$ 时将点 $x$ 的所有轻儿子子树中的与点 $x$ 距离 $=d$ 的所有点进行标号。注意到有可能处理一条重链时与重链顶部节点的距离 $\leq 3$ 的节点已经被标号，此时已经被标号的点不会再进行标号。

# 基于调整搜索顺序的重标号树链剖分

按照重链中深度最小的点的 dfs 顺序来处理每一条重链，考虑对于一条重链，先将这条重链上的所有点按照深度从小到大排序进行标号，然后依次枚举距离 $d$ 为 $1$ 和 $2$ 和 $3$ ，每次将这条重链上的所有点按照深度从小到大处理每个点，处理到点 $x$ 时将点 $x$ 的所有轻儿子子树中的与点 $x$ 距离 $=d$ 的所有点进行标号。注意到有可能处理一条重链时与重链顶部节点的距离 $\leq 3$ 的节点已经被标号，此时已经被标号的点不会再进行标号。
可以发现这样的结构保留了树链剖分的许多性质，比如保证了至多在 $\log n$ 条重链上进行处理的性质，并且这样的标号方式使得树上路径的小范围信息具有了一些连续性。

# 基于调整搜索顺序的重标号树链剖分




图：进行上述标号方式后，该树的一种合法的标号

# 基于调整搜索顺序的重标号树链剖分

对于每一个节点 $x$，考虑维护线段集合 $s0_{x,d}$ 表示与点 $x$ 距离 $=d$ 的在点 $x$ 的子树内的所有节点编号区间（当 $d=4$ 时描述的是距离 $>4$ 的点），线段集合 $s1_{x,d}$ 表示与点 $x$ 距离 $=d$ 的在点 $x$ 的轻儿子的子树内的所有节点编号区间（当 $d=4$ 时描述的是距离 $>4$ 的点），可以发现转移是简单的。

# 基于调整搜索顺序的重标号树链剖分

对于每一个节点 $x$，考虑维护线段集合 $s0_{x,d}$ 表示与点 $x$ 距离 $=d$ 的在点 $x$ 的子树内的所有节点编号区间（当 $d=4$ 时描述的是距离 $>4$ 的点），线段集合 $s1_{x,d}$ 表示与点 $x$ 距离 $=d$ 的在点 $x$ 的轻儿子的子树内的所有节点编号区间（当 $d=4$ 时描述的是距离 $>4$ 的点），可以发现转移是简单的。

根据标号的连续性，可以发现 $s1_{x,d}(0\le d\le 4)$ 均满足线段数量是 $O(1)$ 的，具体来说均满足线段数量 $\le 2$，因为一个节点的与其距离 $\le 3$ 的距离相同的轻儿子一定会在同一时刻被连续标号，但由于可能并不是处理当前节点所在重链时对其的标号（即与其距离 $<3$ 的祖先方向的节点与其并不在一条重链上，并且会将这一层的节点进行标号），因此这个节点的重儿子的编号可能会将连续编号划分成两段，因此线段数量 $\le 2$；而当距离 $\ge 4$ 时显然编号是连续的，因此线段数量 $\le 1$。

# 基于调整搜索顺序的重标号树链剖分

同时也可以发现 $s0_{x,d}(0 \leq d \leq 4)$ 均满足线段数量是 $O(k)$ 的，具体来说满足线段数量 $\leq 4$ ，因为一个节点的与其距离 $\leq 3$ 的距离相同的所有节点中，对于相同的重儿子来说，每一层一定都是连续的，因此即使每一层之间都是独立的，线段数量仍然满足 $\leq 4$ 。

# 基于调整搜索顺序的重标号树链剖分

同时也可以发现 $s0_{x,d}(0 \leq d \leq 4)$ 均满足线段数量是 $O(k)$ 的，具体来说满足线段数量 $\leq 4$ ，因为一个节点的与其距离 $\leq 3$ 的距离相同的所有节点中，对于相同的重儿子来说，每一层一定都是连续的，因此即使每一层之间都是独立的，线段数量仍然满足 $\leq 4$ 。

因此如果我们需要对于单点修改其 $\leq d$ 的轻儿子子树的范围内的所有点，则线段总数为 $O(k)$ ，如果我们需要对于单点修改其 $\leq d$ 的子树范围内的所有点，则线段总数为 $O(k^2)$ 。

# 基于调整搜索顺序的重标号树链剖分

考虑一次链邻域修改操作：考虑每次对同一条重链上的点 $x$ 和点 $y$ 之间的信息进行处理，首先对重链上的顶部的至多三个节点进行特殊处理，需要对其轻儿子子树信息进行处理；重链的尾部节点需要对其子树信息进行处理；中间的部分可以通过编号的连续性进行处理；重链贡献重复部分对其子树信息进行处理；总共需要处理 $O(\log n)$ 条重链，因此该部分时间复杂度为 $O(k^2 \log^2 n)$ ；最后需要对最近公共祖先之上的部分进行处理；因此单次修改总时间复杂度为 $O(k^2 \log^2 n)$ 。

# 基于调整搜索顺序的重标号树链剖分

考虑一次链邻域修改操作：考虑每次对同一条重链上的点 $x$ 和点 $y$ 之间的信息进行处理，首先对重链上的顶部的至多三个节点进行特殊处理，需要对其轻儿子子树信息进行处理；重链的尾部节点需要对其子树信息进行处理；中间的部分可以通过编号的连续性进行处理；重链贡献重复部分对其子树信息进行处理；总共需要处理 $O(\log n)$ 条重链，因此该部分时间复杂度为 $O(k^2 \log^2 n)$ ；最后需要对最近公共祖先之上的部分进行处理；因此单次修改总时间复杂度为 $O(k^2 \log^2 n)$ 。

考虑一次链邻域查询操作：可以发现和修改操作是类似的，同样对重链顶部上至多三个节点进行特殊处理；重链尾部节点进行子树处理；中间部分通过编号连续性来处理；重链贡献重复部分进行子树处理；最后对最近公共祖先之上的部分进行处理；因此单次查询总时间复杂度为 $O(k^2 \log^2 n)$ 。

# 基于调整搜索顺序的重标号树链剖分

子树修改和子树查询是简单的，由于我们按照 dfs 顺序给每条重链标号，因此我们可以扫描节点 $x$ 的子树信息进行处理，单次修改和查询的总时间复杂度为 $O(k^2 \log n)$ 。

# 基于调整搜索顺序的重标号树链剖分

子树修改和子树查询是简单的，由于我们按照 dfs 顺序给每条重链标号，因此我们可以扫描节点 $x$ 的子树信息进行处理，单次修改和查询的总时间复杂度为 $O(k^2 \log n)$ 。
该做法还可以支持查询链上范围 $\leq k(0 \leq k \leq 3)$ 的点权最值和子树的点权最值，做法和查询点权和基本相同。

# 基于调整搜索顺序的重标号树链剖分

子树修改和子树查询是简单的，由于我们按照 dfs 顺序给每条重链标号，因此我们可以扫描节点 $x$ 的子树信息进行处理，单次修改和查询的总时间复杂度为 $O(k^2 \log n)$ 。
该做法还可以支持查询链上范围 $\leq k (0 \leq k \leq 3)$ 的点权最值和子树的点权最值，做法和查询点权和基本相同。
时间复杂度 $O(mk^2 \log^2 n)$ ，常数较大，可以通过本题。

# 总结

通过树链剖分的树上链分治和调整标号的角度，我们可以得到许多关于树上范围相关问题的思考和启发。
树链剖分具有非常丰富和广泛的性质，包含重链剖分、长链剖分等多种类型，还可以深入发掘其中更具有拓展性的思想。

# 点分树

点分树在处理树上范围相关问题中具有广泛应用。我们默认读者已经掌握点分树的相关内容和基本应用。

# 点分树与树上范围传递性相关问题


长沙市长郡中学 朱屹帆

# 例题：地雷

给定一棵 $n$ 个节点的树，边 $i$ 连接 $u_i$ 和 $v_i$，边权为 $w_i$。点 $i$ 有半径 $r_i$ 和点权 $a_i$，定义 $\mathrm{dist}(i,j)$ 表示点 $i$ 和点 $j$ 在树上的最短路径长度。当点 $i$ 被激活时，所有满足 $\mathrm{dist}(i,j) \leq r_i$ 的尚未被激活的点 $j$ 将会被激活，反复进行该过程，直到没有节点被激活为止。

对于每一个节点 $i$，计算如果初始时激活了点 $i$，最终所有被激活的点的点权之和。

数据范围：$1 \leq n \leq 10^5, 1 \leq a_i \leq 10^9, 0 \leq r_i \leq 10^{18}, 1 \leq u_i, v_i \leq n, 1 \leq w_i \leq 10^{12}$。

# 点分树与树上范围传递性相关问题

定义点 $i$ 直接可达点 $j$ 表示 $\mathrm{dist}(i,j) \le r_i$ ，定义点 $i$ 可达点 $j$ 表示点 $i$ 直接可达点 $j$ 或存在一个点 $k$ 满足点 $i$ 可达点 $k$ 且点 $k$ 可达点 $j$ 。

# 点分树与树上范围传递性相关问题

定义点 $i$ 直接可达点 $j$ 表示 $\mathrm{dist}(i,j) \le r_i$ ，定义点 $i$ 可达点 $j$ 表示点 $i$ 直接可达点 $j$ 或存在一个点 $k$ 满足点 $i$ 可达点 $k$ 且点 $k$ 可达点 $j$ 。

注意到点 $i$ 直接激活的点在树上点 $i$ 的半径为 $r_i$ 的范围，建立点分树。定义 $rt$ 为当前点分树中的一个子树的根。

# 点分树与树上范围传递性相关问题

定义点 $i$ 直接可达点 $j$ 表示 $\mathrm{dist}(i,j) \le r_i$ ，定义点 $i$ 可达点 $j$ 表示点 $i$ 直接可达点 $j$ 或存在一个点 $k$ 满足点 $i$ 可达点 $k$ 且点 $k$ 可达点 $j$ 。

注意到点 $i$ 直接激活的点在树上点 $i$ 的半径为 $r_i$ 的范围，建立点分树。定义 $rt$ 为当前点分树中的一个子树的根。

设 $f_{rt,x}$ 表示初始激活点 $x$ 后，最终能给根 $rt$ 提供的激活的范围余量，即设点 $y$ 最终被激活，则 $f_{rt,x} = \max_y\{r_y - \mathrm{dist}(y,rt)\}$ ；

# 点分树与树上范围传递性相关问题

定义点 $i$ 直接可达点 $j$ 表示 $\mathrm{dist}(i,j) \le r_i$ ，定义点 $i$ 可达点 $j$ 表示点 $i$ 直接可达点 $j$ 或存在一个点 $k$ 满足点 $i$ 可达点 $k$ 且点 $k$ 可达点 $j$ 。

注意到点 $i$ 直接激活的点在树上点 $i$ 的半径为 $r_i$ 的范围，建立点分树。定义 $rt$ 为当前点分树中的一个子树的根。

设 $f_{rt,x}$ 表示初始激活点 $x$ 后，最终能给根 $rt$ 提供的激活的范围余量，即设点 $y$ 最终被激活，则 $f_{rt,x} = \max_y\{r_y - \mathrm{dist}(y,rt)\}$ ；

设 $g_{rt,x}$ 表示至少需要初始时 $rt$ 提供的激活范围 $g_{rt,x}$ 能够使得最终点 $x$ 被激活，即存在点 $y$ 满足 $\mathrm{dist}(rt,y) \le g_{rt,x}$ 且点 $y$ 可达点 $x$ 。

# 点分树与树上范围传递性相关问题

首先考虑给 $f$ 和 $g$ 添加点分树子树的限制，即在上述定义的基础上限制访问节点的范围为 $rt$ 的子树，按深度由深到浅依次处理出 $f$ 和 $g$ 。

# 点分树与树上范围传递性相关问题

首先考虑给 $f$ 和 $g$ 添加点分树子树的限制，即在上述定义的基础上限制访问节点的范围为 $rt$ 的子树，按深度由深到浅依次处理出 $f$ 和 $g$ 。

对于一个根 $rt$ ，考虑逐渐增大其初始提供的激活范围 $r'$ ，由于其子树内的 $f$ 和 $g$ 更新完毕，因此当点 $x$ 被激活时，设 $rt'$ 为点分树上 $rt$ 子树内且在点 $x$ 祖先中的节点，则对于点 $y$ 满足 $f_{rt',x} \geq g_{rt',y}$ 时，点 $y$ 将被激活，按照上述过程，可以处理出当前点分树子树的 $g$ 的信息。

# 点分树与树上范围传递性相关问题

首先考虑给 $f$ 和 $g$ 添加点分树子树的限制，即在上述定义的基础上限制访问节点的范围为 $rt$ 的子树，按深度由深到浅依次处理出 $f$ 和 $g$ 。

对于一个根 $rt$ ，考虑逐渐增大其初始提供的激活范围 $r'$ ，由于其子树内的 $f$ 和 $g$ 更新完毕，因此当点 $x$ 被激活时，设 $rt'$ 为点分树上 $rt$ 子树内且在点 $x$ 祖先中的节点，则对于点 $y$ 满足 $f_{rt',x} \geq g_{rt',y}$ 时，点 $y$ 将被激活，按照上述过程，可以处理出当前点分树子树的 $g$ 的信息。

对于一个点 $x$ ，我们可以处理出其在 $rt$ 对应儿子节点子树的范围内能够提供给 $rt$ 的最大激活范围，即设 $rt'$ 为点分树上 $rt$ 子树内且在点 $x$ 祖先中的节点，则对于点 $y$ 满足 $f_{rt',x} \geq g_{rt',y}$ 时，点 $y$ 可以给 $f_{rt,x}$ 提供 $r_y - \mathrm{dist}(y, rt)$ 的贡献。

# 点分树与树上范围传递性相关问题

首先考虑给 $f$ 和 $g$ 添加点分树子树的限制，即在上述定义的基础上限制访问节点的范围为 $rt$ 的子树，按深度由深到浅依次处理出 $f$ 和 $g$ 。

对于一个根 $rt$ ，考虑逐渐增大其初始提供的激活范围 $r'$ ，由于其子树内的 $f$ 和 $g$ 更新完毕，因此当点 $x$ 被激活时，设 $rt'$ 为点分树上 $rt$ 子树内且在点 $x$ 祖先中的节点，则对于点 $y$ 满足 $f_{rt',x} \geq g_{rt',y}$ 时，点 $y$ 将被激活，按照上述过程，可以处理出当前点分树子树的 $g$ 的信息。

对于一个点 $x$ ，我们可以处理出其在 $rt$ 对应儿子节点子树的范围内能够提供给 $rt$ 的最大激活范围，即设 $rt'$ 为点分树上 $rt$ 子树内且在点 $x$ 祖先中的节点，则对于点 $y$ 满足 $f_{rt',x} \geq g_{rt',y}$ 时，点 $y$ 可以给 $f_{rt,x}$ 提供 $r_y - \mathrm{dist}(y, rt)$ 的贡献。

对 $rt$ 的 $f$ 和 $g$ 排序后维护前缀信息即可转移。

# 点分树与树上范围传递性相关问题

接下来考虑去掉给 $f$ 和 $g$ 添加的点分树子树范围限制，此时注意到点分树上的根（全局重心）的信息是正确的，因此在点分树上换根处理，按深度由浅到深依次处理 $f$ 和 $g$ 。

# 点分树与树上范围传递性相关问题

接下来考虑去掉给 $f$ 和 $g$ 添加的点分树子树范围限制，此时注意到点分树上的根（全局重心）的信息是正确的，因此在点分树上换根处理，按深度由浅到深依次处理 $f$ 和 $g$ 。

对于一个根 $rt$ ，其点分树上的祖先的 $f$ 和 $g$ 是正确信息，考虑如何更新其 $f$ 和 $g$ 的信息。

# 点分树与树上范围传递性相关问题

接下来考虑去掉给 $f$ 和 $g$ 添加的点分树子树范围限制，此时注意到点分树上的根（全局重心）的信息是正确的，因此在点分树上换根处理，按深度由浅到深依次处理 $f$ 和 $g$ 。

对于一个根 $rt$ ，其点分树上的祖先的 $f$ 和 $g$ 是正确信息，考虑如何更新其 $f$ 和 $g$ 的信息。

对于一个点 $x$ ，考虑其激活后首次激活到子树外的点，激活到一个子树外的点 $y$ ，且 $x$ 和 $y$ 路径上对应点分树祖先为 $rt'$ ，枚举 $rt'$ ，则当 $f_{rt',x} \geq g_{rt',y}$ 时，点 $y$ 可以给 $f_{rt,x}$ 提供 $r_y - \mathrm{dist}(y, rt)$ 的贡献；同样的，对于一个点 $x$ ，考虑其最终被激活前最后一次由子树外的点激活到子树内的点，由一个子树外的点 $y$ 激活，且 $x$ 和 $y$ 路径上对应点分树祖先为 $rt'$ ，枚举 $rt'$ ，则当 $f_{rt',y} \geq g_{rt',x}$ 时，点 $y$ 可以给 $g_{rt,x}$ 提供 $\mathrm{dist}(rt, y)$ 的贡献。

# 点分树与树上范围传递性相关问题

接下来考虑去掉给 $f$ 和 $g$ 添加的点分树子树范围限制，此时注意到点分树上的根（全局重心）的信息是正确的，因此在点分树上换根处理，按深度由浅到深依次处理 $f$ 和 $g$ 。

对于一个根 $rt$ ，其点分树上的祖先的 $f$ 和 $g$ 是正确信息，考虑如何更新其 $f$ 和 $g$ 的信息。

对于一个点 $x$ ，考虑其激活后首次激活到子树外的点，激活到一个子树外的点 $y$ ，且 $x$ 和 $y$ 路径上对应点分树祖先为 $rt'$ ，枚举 $rt'$ ，则当 $f_{rt',x} \geq g_{rt',y}$ 时，点 $y$ 可以给 $f_{rt,x}$ 提供 $r_y - \mathrm{dist}(y, rt)$ 的贡献；同样的，对于一个点 $x$ ，考虑其最终被激活前最后一次由子树外的点激活到子树内的点，由一个子树外的点 $y$ 激活，且 $x$ 和 $y$ 路径上对应点分树祖先为 $rt'$ ，枚举 $rt'$ ，则当 $f_{rt',y} \geq g_{rt',x}$ 时，点 $y$ 可以给 $g_{rt,x}$ 提供 $\mathrm{dist}(rt, y)$ 的贡献。

按照上述方式进行换根处理，维护出所有 $rt$ 的 $f$ 和 $g$ 的信息，查询时从点 $x$ 向点分树上跳父亲节点，去掉点 $x$ 所在儿子节点子树后查询经过该重心的点权和，时间复杂度 $O(n \log^2 n)$ 。

# 总结

点分树可以处理许多较为基础的树上范围相关问题。从上述例题中还可以发现，点分树在处理一类树上范围传递性相关问题中具有较强性质。

点分树由于其优秀的树形结构，在处理一类树上范围相关问题时，能够维护许多范围相关的信息，因此具有广泛应用。

# Top Tree

Top Tree 功能非常强大，本文仅介绍较为基础的 Top Tree 与树上范围相关问题的一些应用。我们默认读者已经掌握 Top Tree 的相关定义，下文中簇的端点表示簇的界点。

# 静态 Top Tree 的一种构建（全局平衡二叉树）

感兴趣的读者可以阅读我的集训队论文。

# 例题：树上邻域数点

给定一棵 $n$ 个节点的树，你可以使用一些 compress 和 rake 操作进行簇运算，可以进行不超过 $M$ 次簇运算的预处理，有 $m$ 次询问，每次询问给出节点 $u$ 和范围 $d$，要求使用至多 $1$ 次簇运算，得到一个点集没有限制，边集恰好为节点 $u$ 的范围为 $d$ 的所有边构成的集合的合法的簇。

数据范围：$1 \leq n \leq 2 \times 10^5$，$1 \leq m \leq 10^6$，预处理的簇运算次数限制 $M = 3 \times 10^7$。

# Top Tree 与树上范围相关的应用

首先构建原树的 Top Tree，考虑当节点 $u$ 为原树的根时的做法，我们在每个簇上对该簇的两个端点，都分别维护出簇的边集中与端点距离为 $d$ 的合法簇信息，由于我们只维护簇内的信息，因此需要维护的 $d$ 的范围上界为该簇的端点对应的最长路径长度，方便起见我们使用每个簇的大小作为 $d$ 的上界。

设簇类型 $f_{x,0/1,d}$ 表示簇内距离簇 $x$ 的左端点或右端点距离 $\leq d$ 的簇的信息，要求信息的一端为对应左右端点，且当 $d \geq$ 簇的左右端点距离时信息两端点分别为该簇的左右端点，发现 compress 和 rake 类型节点可以进行来自左右儿子的信息合并，因此可以不使用簇运算回答每次关于根的询问。

# Top Tree 与树上范围相关的应用

当询问点为原树任意点 $x$ 时，我们选择该点在原树上的任意一条与之相连的边，找到一个满足条件的簇使得该簇是该边在 Top Tree 对应节点的祖先且点 $x$ 的范围为 $d$ 可以完全包含该簇信息，要求该簇在 Top Tree 上的深度最小，此时答案为点 $x$ 到该簇的两个端点再向该范围外进行拓展的簇的信息合并。

设簇类型 $g_{x,0/1,d}$ 表示簇外距离簇 $x$ 的左端点或右端点距离 $\leq d$ 的簇的信息，其中 $d$ 的上界为簇 $x$ 在 Top Tree 上父节点的大小，当簇 $x$ 不为 Top Tree 的根时，父亲节点大小 $\geq d$，因此可以使用类似换根 dp 的方式通过簇自身的 $f$ 和父亲节点对应簇的 $g$ 维护出簇自身的 $g$，然后通过一个簇的 $f$ 和 $g$ 的信息即可在至多 $1$ 次簇运算内快速回答询问。

# Top Tree 与树上范围相关的应用

按照上述方式预处理，由于该 Top Tree 深度为 $O(\log n)$ ，即节点子树大小的和的级别为 $O(n \log n)$ ，因此预处理簇运算次数为 $O(n \log n)$ ，可以通过本题。

# 总结

Top Tree 功能强大，从上述例题中可以发现 Top Tree 在处理相关问题时的特殊的结构和特殊的信息合并方式，进一步提升了对树上范围相关问题的处理能力。

# 总结

本文介绍了树链剖分、点分树和基础 Top Tree 在树上范围相关问题中的运用，事实上很多树上问题经典算法在树上范围相关问题中具有广泛运用的潜力，笔者希望本文能够激发读者思考，启发读者对树上范围相关问题进行更加深入的研究。

# 谢谢大家！

祝大家学业有成！