arXiv:2603.10432v1 [cs.DS] 11 Mar 2026

使用线性对数次最短路径距离查询重构有界树长图

Chirag Kaudan* Amir Nayyeri†‡

2026年3月12日

摘要

我们研究以下图重构问题：给定一个无权连通图 $G=(V,E)$、其可见顶点集 V以及一个预言机，该预言机接收两个顶点 $u,v\in V$并返回 u与 v之间的最短路径距离，需要多少次查询才能完成重构E?

具体而言，我们研究有界度 Δ 且有界树长 tl 的连通图，并证明使用确定性算法可以通过 $O_{\Delta,\mathrm{tl}}(n\log\ n)$ 次查询完成重构。对于这一类图，该结果比已知最佳算法（无论是确定性的还是随机的）提高了一个 $\log n$ 因子，并且达到了有界弦度图这一图类的已知下界，而有界弦度图是有界树长图的子类。

1 引言

在图重构问题中，只有隐藏图 $G=$ （V,E）的顶点集 V是可见的，目标是通过查询预言机来重构边集 E 。图重构算法旨在以尽可能少的自适应选择查询来发现边集。一旦存在与所有查询答案相一致的唯一图，重构任务即被视为完成。针对图重构已研究了多种查询模型，包括边存在性查询、边计数、有效电阻查询和最短路径查询 [RS07, AC04, HMMT18, ABK+04, AA05, BBNW26]。

在最短路径（SP）距离查询模型中，我们可以访问一个距离预言机：对于任意两个顶点 u, $v\in V$，一次查询返回 G 中 u 与 v 之间最短路径的长度，若不存在这样的路径则返回 ∞ 。

利用最短路径查询进行网络重构的研究，其动机来自互联网拓扑重构和进化树重构等应用 [Hei89, WSSB77, BEE+05, DAHB+06, ACKM09]。

任何（无权）图都可以通过 $\binom{n}{2}$ 最短路径查询轻松重构：只需查询每一对顶点，检查它们是否相邻即可。不幸的是，一旦允许无界度 [RS07]，即使对于树，这个上界也是紧的。

*俄勒冈州立大学，kaudanc@oregonstate.edu
†俄勒冈州立大学，amir.nayyeri@oregonstate.edu
‡作者获得了美国国家科学基金会项目 CCF-1941086 和 CCF-2311180 的资助。

此外，如果允许非连通图，这个界也是紧的——重构一个有 n 个顶点和 1 条边的图需要 $\Omega(n^2)$ 次查询。鉴于这些观察，所有非平凡的图重构算法研究都聚焦于连通有界度图。

Mathieu 等人 [MZ13] [KMZ18] 证明，一般的有界度图可以通过随机化算法，使用期望 $\widetilde{O}_\Delta(n^{3/2})$ 次查询进行重构。[1]。这仍是目前已知的一般上界。是否存在一种确定性算法，能够通过 $\widetilde{O}_\Delta(n^{3/2})$ 次查询重构有界度图，尚不得而知。Mathieu 等人还提出了一种随机化算法，仅使用 $O_\Delta(n\log^3 n)$ 次查询即可重构弦图。在此基础上，Bastide 和 Groenland [BG24a] 进一步改进了这一结果，提出了一种重构算法，可使用 $O_{\Delta,\,k}(n\log\, n)$ 次查询，重构有界度 k-弦图（另请参阅 Rong 等人 [RLYW21] 提出的其他近弦图近线性重构算法）。[2]

在一项较新的研究中，Bastide 和 Groenland [BG24b] 提出了一种随机化 $O_{\Delta,\,\tau}(n\log^2 n)$ 次查询算法，用于重构树长至多为 τ 的图。请注意，有界弦度意味着有界树长 [DG07], , 但反之则不成立。因此，后一个结果适用于范围更广的图类，尽管该算法稍慢且采用随机化方法。

本文提出一种确定性的 $O_{\Delta,\tau}(n\log\, n)$ 次查询算法，用于重构最大度有界且树长至多为 τ 的图。与 Bastide 和 Groenland [BG24b] 相比，我们的算法具有确定性，效率提高了 $\log n$ 倍。

定理 1. 给定一个隐藏图 G 和一个整数 $\tau \geq \mathrm{tl}(G)$，存在一种算法，使用至多 $O(\Delta^{3\tau+2}\cdot n\log\, n)$ 次最短路径查询来重构 G，其中 n 是 G 中顶点的数量，Δ 是其最大度。

特别地，我们的结果意味着一种确定性的 $O_{\Delta,k}(n\log\, n)$ 次查询算法，可用于重构有界弦度的有界度图，其结果与重构有界 k-弦度图的最优结果相匹配 [BG24a]。请注意，该定理假设参数 τ 已知。

2 背景与记号

本文中的所有图均为简单、连通、无权无向图。对于图 $G=(V,E)$ 及其顶点 $u,v\in V(G)$，我们用 $d_G(u,v)$ 表示 u 与 v 在 G 中的最短路径长度。对于图 $G=(V,E)$ 及顶点 $v\in V(G)$，我们用 $N_G(v)$ 表示图 G 中 v 的所有邻居构成的集合，并用 $N_G[v]=N_G(v)\cup\{v\}$ 分别表示 v 的开邻域和闭邻域。类似地，对于图 $G=(V,E)$ 及集合 $S\subseteq V(G)$，我们令 $N_G[S]=\{u\in V(G)\ \exists a\in S, d_G(a,u)\ \leq 1\}$ 和 $N_G(S)=N_G[S]\setminus S$ 分别表示 S 在 G 中的闭邻域和开邻域。对于图 $G=(V,E)$ 及 G 中的路径 $\pi=v_0v_1\ldots v_k$，我们用 $\pi[v_i,v_j]$ 表示 π 中从 v_i 开始、到 v_j 结束的子路径，其中 $0\leq i\leq j\leq k$。此外，对于图 G 中的路径 $\pi=\pi_0,\ldots,\pi_n$ 和 $\rho=\rho_0,\ldots,\rho_m$，若 $\pi_n=\rho_0$，我们将它们的拼接记为 $\pi\circ\rho=\pi_0,\ldots,\pi_n,\ldots,\rho_n$，从而得到 G 中一条从 π_0 到 ρ_n 的游走。

树分解与树长。 $G=(V,E)$的**树分解**是一个有序对$(T,\{V_t\mid t\in V(T)\})$，其中 T是一棵树，且 $V_t\subseteq V(G)$是与每个顶点相关联的子集

[1]我们用 $\widetilde{O}(f(n))$ 简写为 $O(f(n)\cdot\mathrm{polylog}(n))$。此外，对于不同的参数 p，我们使用 $O_p(\cdot)$ 表示隐藏常数依赖于 p。
[2]如果一个图的所有诱导圈长度都至多为 k，则称该图为 k-弦图；弦图是 3-弦图。

属于 T。树 T以及子集族 $\{V_t \mid t \in V(T)\}$（通常称为树分解的**袋**）必须满足以下性质。

- 的每个顶点都包含在至少一个袋 V_t 中，即 $V(G) = \bigcup_{t \in V(T)} V_t$.
- 对于 G的每条边 e ，都存在一个袋 V_t ，其中包含 e的两个端点。
- 设 t, t'和 t''是 T的三个顶点，且 t'位于 T中从 t到 t''的唯一一条路径上。那么，如果 G 的顶点 v 同时属于 V_t 和 $V_{t''}$，它也属于 $V_{t'}$。

图 $G = (V, E)$的**树长**记为 tl(G)，它是满足以下条件的最小整数 j：存在一个树分解$(T, \{V_t \mid t \in V(T)\})$，其中对于所有 $u, v \in V_t$， $t \in V(T)$均满足 $d_G(u, v) \leq j$。也就是说，树分解中同一个袋内的任意两个顶点之间的距离至多为 j。

例如，完全图的树长为1，因为可以将所有顶点放入同一个包中。更一般地，弦图的树长为1，而 k-弦图的树长为 $O(k)$。

分层树。我们的算法所使用的一个关键工具，是Chepoi与Dragan [CD00]所采用的分层树概念。类似概念曾以合并树、连接树或分裂树等不同名称出现在图论、《计算几何》和拓扑学的文献中（参见这些参考文献：[CSA03, WWW14]）。

设 $G = (V, E)$为一个连通图，并设 $s \in V$ 为任意顶点。我们将广度优先搜索（BFS）**层** L_i 定义为到 s的最短路径距离为 i的顶点集合，即 $L_i = \{v \mid d_G(s, v) = i\}$。特别地， $L_0 = \{s\}$。我们使用记号 $L_{\leq k} := L_0 \cup \ldots \cup L_k$，以及$L_{\geq k} := L_k \cup \ldots \cup L_{n-1}$。

对于任意 $i \geq 0$，令 $S_i = \{S_i^1, \ldots, S_i^{s_i}\}$ 为 $G \setminus L_{\leq i-1}$ 的连通分量集合，并注意到 $L_{-1} = \emptyset$。反过来，对于任意 $1 \leq j \leq s_i$，令 $P_i^j = S_i^j \cap L_i$ 为 S_i^j 中属于 L_i 的顶点集合。我们将 $\mathcal{P}_i = \{P_i^1, \ldots, P_i^{s_i}\}$ 的元素称为 G 在第 i 层的**部分**。由此可知，这些部分构成 L_i 的划分。此外，所有层中的全部部分构成的集合 $\mathcal{P} = \bigcup_{0 \leq i} \mathcal{P}_i$，划分了 V（G）。

我们将相对于 s 的 G 的**分层树** $\mathcal{T}$ =（$\mathcal{P}$， $\mathcal{E}$）定义为这样一个图：其顶点是 G 的部分。对于两个部分 P， $P' \in \mathcal{P}$，（P， P'）当且仅当存在 $u \in P$ 和 $u' \in P'$，使得（u， u'） $\in E$（G）。

在本文中，部分既指分层树的顶点，也指 $V(G)$ 的子集；不过根据上下文，应该很容易判断我们指的是哪一种。我们的算法需要考虑由 $\mathcal{P}$ 的子集诱导出的子树。我们用 $\mathcal{T}_k$ 表示诱导子树 $\mathcal{T}[\bigcup_{i \leq k-1} \mathcal{P}_i]$，即由 $\mathcal{T}$ 中第 0、1、.....、$k-1$ 层上的部分诱导出的子树，而 不包括 k。

此外，我们将分层树的**长度** 定义为其任一部分的最大直径，记作 $\ell(\mathcal{T})$，即 $\ell(\mathcal{T}) = \max_{P \in \mathcal{P}} (\max_{u,\ v \in P}(d_G(u,\ v)))$。通过简单的计数论证，可以界定分层树中每个部分的大小。

引理 1. 设 $G = (V, E)$ 是一个最大度为 Δ 的图，且 $\mathcal{T} = (\mathcal{P}, \mathcal{E})$ 是它的任意一棵分层树。$\mathcal{P}$ 中每个部分的大小至多为 $\frac{\Delta^{\ell(\mathcal{T})+1}-1}{\Delta-1} \leq \Delta^{\ell(\mathcal{T})+1}$ ————

我们的分层树定义等价于 Dourisbourne 和 Gavoille [DG07]– 所使用的定义：在他们的定义中， L_i 中的两个顶点属于同一部分，当且仅当它们之间存在一条 $G[L_{\geq i}]$中的路径。他们在论文 [DG07], 的定理 8 中证明了以下关于 G 的分层树长度与其树长之间的界。

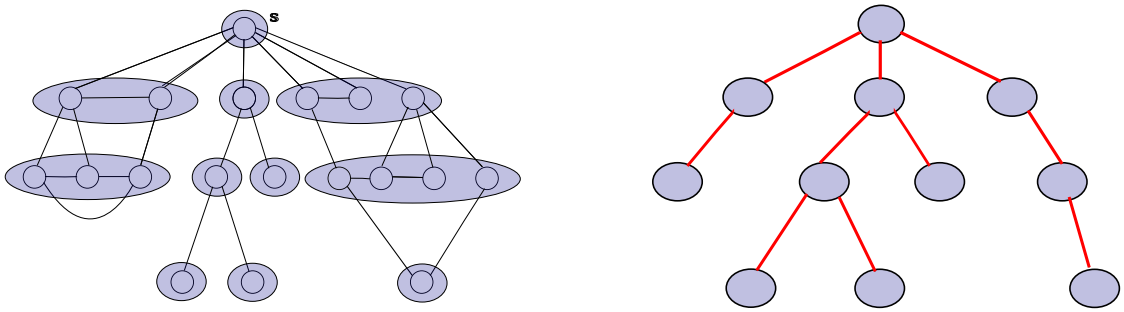


图 1：左：根据以 s 为参照进行的 BFS 分层得到的图的各个部分，以紫色阴影标出；右：分层树，其顶点与这些部分一一对应。

引理 2 (Dourisbourne 和 Gavoille [DG07], 定理 8)。设 $G = (V, E)$ 是一个树长为 $\mathrm{tl}(G)$ 的连通图，并令 $\mathcal{T} = (\mathcal{P}, \mathcal{E})$ 为其分层树（相对于任意顶点 s）。那么 $\ell(\mathcal{T}) \leq 3\mathrm{tl}(G)$。

Chepoi 和 Dragan [CD00] 证明了，可以在线性时间内计算图的分层树。本文关注的是查询复杂度，而非时间复杂度。然而，不难证明，当最大度和树长视为常数时，我们算法的渐近时间复杂度与其查询复杂度相同。

3 重构算法

设 $G = (V, E)$ 是一个隐藏图，其可见顶点集的基数为 n，并且具有最短路径距离预言机。我们的算法首先任意选取一个顶点 $s \in V(G)$，并使用 $n-1$ 次查询找出所有 $v \in V(G) \setminus \{s\}$对应的 $d_G(s, v)$。基于这些信息，我们的算法可以构建 BFS 层 $L_0, L_1, \ldots, L_{n-1}$。

接下来，我们的算法逐层重构 G。具体而言，对于每个顶点 $v \in L_i$，我们的算法使用最短路径查询来寻找 v 在 $L_i \cup L_{i-1}$中的邻居。

令 $G_i = G[L_{\leq i-1}]$ 表示由 G 的前 i 层诱导的子图。还记得，$\mathcal{T}$是以 s为根的分层树，并令 ℓ 为其长度的一个上界，即 $\ell \geq \ell(\mathcal{T})$。类似地，令 $\mathcal{T}_k$表示 $\mathcal{T}$中由 $\mathcal{T}$（即分层树）的前 $k = i - \ell - 2$ 层诱导的子树。我们的算法交替地将 G_i 扩展至 G_{i+1}，并将 $\mathcal{T}_k$ 扩展至 $\mathcal{T}_{k+1}$。具体而言，给定 G_i，我们无需最短路径查询即可构造 $\mathcal{T}_k$。然后，给定 $\mathcal{T}_k$和 G_i，我们使用 $|L_i| \cdot \Delta^{O(\ell)} \cdot \log n$次最短路径查询来构造 G_{i+1}。后一个算法由 $\mathcal{T}_k$ 中的一种二分搜索型算法，以及在 k 到 i层内进行的穷举搜索组成，其思想与 Bastide 和 Groenland [BG24a]对 k-弦图的重构类似。

3.1 扩展分层树

我们首先证明，给定 G_i，可以重构 $\mathcal{T}$的前 $k = i - \ell - 2$ 层，记为 $\mathcal{T}_k$。根据分层树的定义，如果在 $G[L_{\geq k}]$ 中的两个顶点 $u, v \in L_k$之间存在一条路径，则 u 和 v 属于第 k层的同一部分；特别地，如果存在一条 u-v 路径位于

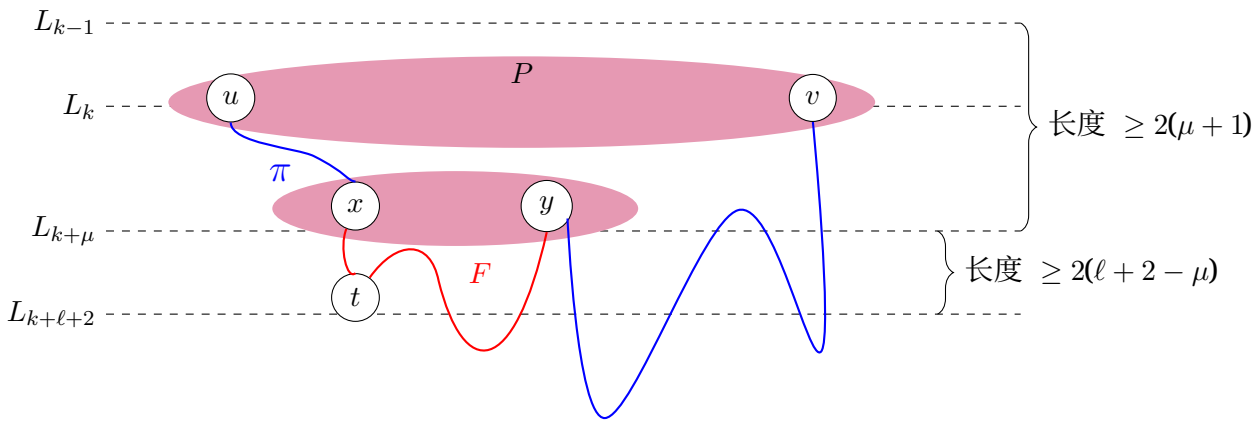


图2： $G \setminus L_{\leq k-1}$ 中一条最短的 u-v 路径 π 及其 x-y 子路径 F（红色），见引理 3 的证明。部分由阴影椭圆表示。

接下来的 $O(\ell)$层 G中，则 u 和 v 属于第 k层的同一部分。下述引理是我们这一构造的关键结构性见解，表明其逆命题同样成立，即证明：若两个顶点属于第 k 层的同一部分，则在接下来的$O(\ell)$层 G中存在一条连接它们的路径。因此，该引理表明，要构造 $\mathcal{T}_k$，我们只需知道 G的前$k + O(\ell)$层。

引理 3。设 $G = (V, E)$，设 $\mathcal{T} = (\mathcal{P}, \mathcal{E})$ 是 G 的分层树，并设 $\ell \geq \ell(\mathcal{T})$。假设$u$、$v \in V(G)$ 在第 k 层属于 $\mathcal{T}$ 的同一部分 P。那么，G 中存在一条连接 u 和 v 的路径，其顶点都属于 $L_{\leq k+\ell+1} \cap L_{\geq k}$，即 u 和 v 在 $G \setminus (L_{\leq k-1} \cup L_{\geq k+\ell+2})$ 中连通。

证明。假设 $G \setminus (L_{\leq k-1} \cup L_{\geq k+\ell+2})$ 中不存在 u-v 路径。由于 $u, v \in P$，根据定义， $G \setminus L_{\leq k-1}$ 中存在一条 u-v 路径。设 π 是这样一条最短的 u-v 路径。那么 π 在 $L_{k+\ell+2}$ 中有一个顶点 t。设 $\mu = \lfloor \frac{\ell}{2} \rfloor$，并设 x 是子路径 $\pi[u, t]$ 上属于 $L_{k+\mu}$ 的最后一个顶点， y 是子路径 $\pi[t, v]$ 上属于 $L_{k+\mu}$ 的第一个顶点。注意， x 和 y 都在 $L_{k+\mu}$ 中，并且 $G \setminus L_{\leq k+\mu-1}$ 中存在一条由 $\pi[x, t]$ 和 $\pi[t, y]$ 组成的路径 $\pi[x, y]$，二者都在 $G \setminus L_{\leq k+\mu-1}$ 中。因此， x 和 y 属于 $\mathcal{T}$ 的同一部分。由此可得， $d_G(x, y) \leq \ell(\mathcal{T}) \leq \ell$，因为每个部分的直径至多为 ℓ。

根据 π 的定义，路径 $F := \pi[x, t] \circ \pi[t, y]$ 是一条最短的 x-y 路径，其顶点均位于 $L_{\geq k}$ 中。因此，任何 x-y 路径的长度要么至少与 F 的长度相同，要么包含一个位于第 L_{k-1} 层的顶点。也就是说， $d_G(x, y) \geq \min\{2(\ell + 2 - \mu), 2(\mu + 1)\} \geq \ell + 1$，这与假设矛盾。 □

我们使用引理 3 根据 G_i 构造 $\mathcal{T}_k$。注意，要构造 $\mathcal{T}_k$，只需能够检查一对顶点 u， $v \in L_k$ 在图 $G \setminus L_{\leq k-1}$ 中是否连通。根据引理 3，要检查这种连通性，我们只需考虑 G_i。

推论 1. 设 $G = (V, E)$，并设 $\mathcal{T} = (\mathcal{P}, \mathcal{E})$ 为 G 的分层树。对于任意 $i \geq \ell(\mathcal{T}) + 2$ 和 $k \leq i - \ell(\mathcal{T}) - 2$，存在一种算法，无需任何最短路径查询，仅凭 G_i（不需要关于 G 的其他信息）即可计算出 $\mathcal{T}_k$。

3.2 扩展图重构

接下来，我们将证明，给定 G_i 和 $\mathcal{T}_k$，可以针对 $k = i - \ell - 2$ 重构 G_{i+1}。也就是说，我们可以找出属于 G_{i+1} 但不属于 G_i 的所有边，即两个端点都在 L_{i+1} 中的边，或一个端点在 L_i 中、另一个端点在 L_{i+1} 中的边。

我们的算法分两步完成这一重构。首先，对于每个顶点 $v \in L_i$，它会在 $G \setminus L_{\leq k-1}$ 中找到 v 的连通分量；更具体地说，是位于第 k 层、且为包含 v 的部分的祖先（在部分构造的分层树 $\mathcal{T}_k$ 中）的那一部分。这一步需要 $\Delta^{O(\ell)} \cdot \log n$ 次最短路径查询。找到该连通分量后，我们的算法会穷尽搜索该连通分量中位于第 L_i 层和第 L_{i-1} 层、与 v 相邻的所有顶点。这一步对 L_i 中的每个顶点需要 $\Delta^{O(\ell)}$ 次查询。

为了更正式地描述我们的方法，我们给出几个定义：

- 对于顶点 $v \in V(G)$，我们将 **部分(v)** 定义为包含 v 的分层树 $\mathcal{T}$ 中的部分。注意，对于任意顶点 $v \in L_{\leq k-1}$，包含 v 的 $\mathcal{T}_k$ 中的顶点等价于 part(v)，因此，对于此类顶点 v，即使可能尚未重构 $\mathcal{T}$，我们仍将使用 part(v)。

- 对于顶点 $v \in L_i$ 和整数 $k \in \{0, 1, \ldots, i\}$，我们将**anc($v, k$)**定义为 $\mathcal{T}$ 中这样一部分：它是 part(v) 在深度 k 处的祖先部分。例如，anc(v, i) = part(v)，而 anc(v,0) = part(s) = $\{s\}$，其中 s 是广度优先搜索树的根。

- 对于 $\mathcal{T}$ 的一个部分 P，我们将**comp(P)**定义为包含在 P 中的顶点以及 P 在 $\mathcal{T}$ 中的所有后代所组成的子集。也就是说，这是 $V(G)$ 中包含在以 $\mathcal{T}$ 为根的子树中的顶点子集。注意，如果 P 是第 k 层的一个部分，那么 comp(P) 就是 $G \setminus L_{\leq k-1}$ 中包含 P 中顶点的连通分量。

连通分量的对数搜索。令 $v \in L_i$，并令 $k \in \{0, 1, \ldots, i\}$。我们算法的第一步是通过对数搜索找到 anc($v, k$)。为此，我们使用以下关于树的顶点分隔集的经典结果。图 G 的 **(1/2)-割顶点**或**质心**是指这样一个顶点：移除它后得到的连通分量大小均至多为 $\frac{1}{2}|V(G)|$。

引理 4. 每棵树都有一个质心。（参见 *[Die17，第 1] 章*）

引理 5. 设 $G = (V, E)$，设 $\mathcal{T} = (\mathcal{P}, \mathcal{E})$ 是 G 的一棵分层树，并设 $\ell \geq \ell(\mathcal{T})$。此外，设 $i \geq k \geq 0$，并假设 $\mathcal{T}_k$ 和 G_i 已知。对于任意 $x \in L_i$，可以通过 $O(\Delta^{\ell+2} \cdot \log n)$ 次查询找到 $\mathrm{anc}(x, k)$。

证明。由于 $\mathcal{T}_k$ 是一棵树，根据引理 4，它包含一个 (1/2)-割点。证明构造了 $\mathcal{T}_k$ 的一系列子树，即 $(\mathcal{T}_k^j)_{j=0}^f$，以及与这些子树对应的一系列 (1/2)-割点 $(P_j)_{j=0}^{f-1}$，使得 (1) $f \leq \lceil \log n \rceil$，且 (2) $\mathcal{T}_k^f$ 只有一个顶点：anc(x, k)。初始时，$\mathcal{T}_k^0 = \mathcal{T}_k$。树 $\mathcal{T}_k^{j+1}$ 定义为 $T_k^j \setminus \{P_j\}$ 中包含 anc(x, k) 的连通分量。

由于 $P_j \in V(\mathcal{T}_k^j)$ 是一个分隔点，$\mathcal{T}_k^{j+1}$ 是 $\mathcal{T}_k^j$ 的子树，其中包含 P_j 的邻居，而该邻居包含 $N_G(P_j)$ 中所有顶点里距离 x 最近的顶点；$N_G(P_j)$ 是 $V(G) \setminus P_j$ 中所有在 P_j 有邻居的顶点组成的集合。因此，要找到 $\mathcal{T}_k^{j+1}$，只需查询 x 到 $N_G(P_j)$ 中所有顶点的距离。注意，anc(x, k) 是 $\mathcal{T}_k$ 的一个叶节点，因此我们总能避免将它或其他任何叶节点选作序列 $(P_j)_{j=0}^{f-1}$ 中的顶点；这也保证了对于序列中的所有 P_j，$N_G(P_j) \subseteq G[V(\mathcal{T}_k)]$。

根据引理 1， P_j中至多有 $\Delta^{\ell+1}$ 个顶点。由于 $N_G(P_j)$ 中的每个顶点在 P_j 中至少有一个邻居，且 G 的最大度为 Δ，我们有

$$|N_G(P_j)| \leq \Delta \cdot |P_j| = \Delta^{\ell+2}$$

由于 $\mathcal{T}_k$ 至多有 n 个部分，且每一步至少消去一半的部分，因此我们知道这次搜索至多需要 $\lceil \log_2 n \rceil$ 步，所以总查询复杂度为 $\Delta^{\ell+2} \cdot \lceil \log_2 n \rceil = O(\Delta^{\ell+2} \cdot \log n)$

□

根据上述引理，我们可以计算 $G \setminus L_{\leq k-1}$中每个 $v \in L_i$ 的连通分量。

引理 6. 设 $G = (V, E)$，设 $\mathcal{T} = (\mathcal{P}, \mathcal{E})$ 是 G 的分层树，并设 $\ell \geq \ell(\mathcal{T})$。此外，设$i \geq k \geq 0$，并假定已知 $\mathcal{T}_k$ 和 G_i。存在一种算法，可以通过 $O(|L_i| \cdot \Delta^{\ell+2} \cdot \log n)$ 次查询，计算所有深度为 k 的 P 的 $\mathrm{comp}(P) \cap L_{\leq i}$。

证明。 要计算深度为 k的 $\mathcal{T}$中所有 P的 comp$(P) \cap L_{\leq i}$，只需找到所有$u \in L_k \cup \ldots \cup L_i$对应的 anc$(u, k)$。由于我们有 G_i，它等于 $G[L_{\leq i-1}]$，因此对于每个 $u \in L_k \cup \ldots \cup L_{i-1}$，其到 L_k 的最短路径都作为广度优先搜索树的一部分；所以我们可以通过检查深度为 k的哪个部分与 $G \setminus L_{\leq k-1}$中的 u 连通，来找到 anc(u, k)。由于 G_i已给出，此步骤不需要任何最短路径查询。对于顶点 $u \in L_i$，我们使用引理 5，通过 $O(\Delta^{\ell+2} \cdot \log\ n)$ 次查询找到 anc(u, k)。由于第 i层有 $|L_i|$ 个顶点，查询总数如引理所述。

□

通过暴力搜索查找邻居。到目前为止，我们已为深度为 k 的所有 P 计算出 comp$(P) \cap L_{\leq i}$。我们对这些顶点集的大小进行界定。

引理 7。设 $G = (V, E)$，设 $\mathcal{T} = (\mathcal{P}, \mathcal{E})$ 是 G 的分层树，并设 $\ell \geq \ell(\mathcal{T})$。此外，设 $P \in \mathcal{P}$ 是 $\mathcal{T}$ 的一个部分，其深度为 k。再设 $i \geq k$。那么 $|\mathrm{comp}(P) \cap L_{\leq i}| \leq \Delta^{\ell+i-k+2}$.

证明。comp$(P) \cap L_{\leq i}$ 中的每个顶点到 P 中某个顶点的距离至多为 $i - k$。由于该图的最大度为 Δ，因此这样的顶点至多有 $|P| \cdot \Delta^{i-k+1}$ 个。根据引理 1， $|P| \leq \Delta^{\ell+1}$，这就得到了引理中的界。

□

结合引理 6 和引理 7，我们的算法根据 G_i 和 $\mathcal{T}_k$ 构造出 G_{i+1}。

引理 8。 设 $G = (V, E)$，设 $\mathcal{T} = (\mathcal{P}, \mathcal{E})$ 是 G 的分层树，并设 $\ell \geq \ell(\mathcal{T})$。此外，设$k \geq 0$和 $i = k + \ell + 2$。假设给定了 G_i和 $\mathcal{T}_k$。存在一种算法，可用$O(|L_i|(\Delta^{\ell+2} \log n + \Delta^{4\ell+8}))$ 次查询计算 G_{i+1}。

证明。 根据引理 6，对于深度为 k 的所有 P，都可以通过 $O(|L_i| \cdot \Delta^{\ell+2} \cdot \log n)$ 次查询计算 comp$(P) \cap L_{\leq i}$。要计算 L_i 与 L_{i-1} 之间以及 L_i 内部的边，只需检查每个顶点 $v \in L_i$ 是否与 $G \setminus L_{\leq k-1}$ 中和 v 属于同一连通分量的顶点相邻。为此，我们检查集合 comp(anc(v, k)) $\cap L_{\leq i}$，以寻找 v 可能的邻居。根据引理 7，这些集合的大小至多为 $\Delta^{\ell+i-k+2} = \Delta^{2\ell+4}$。总的来说，后一类查询至多有 $|L_i| \cdot \Delta^{4\ell+8}$ 次。因此，查询总次数为 $O(|L_i|(\Delta^{\ell+2} \log n + \Delta^{4\ell+8}))$ □

3.3 主结果的证明

最后，我们可以证明主要结果了。

定理 *1* 的证明。 首先，我们在 $V(G)$ 中任取一个顶点 s，并查询 s 到每个 $v \in V(G) \setminus \{s\}$ 的距离，以构建各层 $L_0, \ldots, L_{n-1}$；这需要 $n-1$ 次查询。令 $\mathcal{T}$ 为以 $\{s\}$ 为根的 G 的分层树（注意，我们尚未计算 $\mathcal{T}$）。根据引理 2，我们知道 $\ell(\mathcal{T}) \leq 3\mathrm{tl}(G)$。我们设 $\ell := 3\tau \geq 3\mathrm{tl}(G)$，其中 τ 来自定理陈述。因此 $\ell \geq \ell(\mathcal{T})$。

我们计算 G $\ell+2$，方法是查询 $L_{\leq \ell+1}$ 中每一对顶点，最多需要 $\Delta^{2\ell+4}$ 次查询。

对于 $i \geq \ell+2$，我们分两步从 G_i 计算 G_{i+1}。首先，根据推论 1 中的算法，无需查询即可计算 $k = i-(\ell+2)$ 的 $\mathcal{T}_k$。然后，根据引理 8，我们使用至多 $|L_i|(\Delta^{\ell+2} \cdot \log\ n + \Delta^{4\ell+8})$ 次查询计算 G_{i+1}。对所有剩余层求和，该数量至多为 $n(\Delta^{\ell+2} \cdot \log n + \Delta^{4\ell+8})$。因此，我们至多使用

$$\begin{aligned} &n - 1 + \Delta^{2\ell+4} + n(\Delta^{\ell+2} \cdot \log n + \Delta^{4\ell+8}) \\ &\quad = O(\Delta^{\ell+2} \cdot n \log n) = O(\Delta^{3\tau+2} \cdot n \log n) \end{aligned}$$

查询。 □

参考文献

[AA05] Noga Alon 和 Vera Asodi。学习隐藏子图。*SIAM* 离散数学杂志，18(4):697–712，2005年。[ABK+04] Noga Alon、Richard Beigel、Simon Kasif、Steven Rudich 和 Benny Sudakov。学习隐藏匹配。*SIAM* 计算杂志，33(2):487–501，2004年。[AC04] Dana Angluin 和 Jiang Chen。每条边仅用 o(log n) 次查询学习隐藏图。载于 John Shawe-Taylor 和 Yoram Singer 编，《学习理论》，第 210–223 页，柏林、海德堡，2004年。Springer Berlin Heidelberg。[ACKM09] Dimitris Achlioptas、Aaron Clauset、David Kempe 和 Cristopher Moore。论路由追踪采样的偏差：或正则图中的幂律度分布。《*ACM* 杂志》，56(4)，2009年7月。[BBNW26] Huck Bennett、Mitchell Black、Amir Nayyeri 和 Evelyn Warton。通过有效电阻查询进行图推断。载于第*37*届国际算法学习理论会议论文集，2026年。[BEE+05] Zuzana Beerliova、Felix Eberhard、Thomas Erlebach、Alexander Hall、Michael Hoff- ˇmann、Mat´uˇs Mihal´ak 和 L. Shankar Ram。网络发现与验证。载于 Dieter Kratsch 编，《计算机科学中的图论概念》，第 127–138 页，柏林、海德堡，2005年。Springer Berlin Heidelberg。[BG24a] Paul Bastide 和 Carla Groenland。无长诱导圈图的最优距离查询重构，2024年。

[BG24b] Paul Bastide 和 Carla Groenland。´有界树长图的准线性距离查询重构。载于 Edouard Bonnet 和 Pawel Rzazewski 编，《第十九届参数化与精确计算国际研讨会》，IPEC 2024，2024年9月4日至6日，伦敦大学皇家霍洛威学院，英国埃格姆，第 321 卷 LIPIcs，第 20:1–20:11 页。Schloss Dagstuhl - Leibniz-Zentrum f¨ur In- formatik，2024年。[CD00] Victor Chepoi 和 Feodor Dragan。关于图中距离近似树的一则短札。《欧洲组合学杂志》，21(6):761–766，2000年。[CSA03] Hamish Carr、Jack Snoeyink 和 Ulrike Axen。计算所有维度的等高线树。《计算几何》，24(2):75–94，2003年。第四届 CGC计算几何研讨会特刊。[DAHB+06] Luca Dall'Asta、Ignacio Alvarez-Hamelin、Alain Barrat、Alexei V´azquez 和 Alessan- dro Vespignani。用类traceroute探测探索网络：理论与模拟。《理论计算机科学》，355(1):6–24，2006年。复杂网络。[DG07] Yon Dourisboure 和 Cyril Gavoille。具有小直径袋的树分解。《离散数学》，307(16):2008–2029，2007年。[Die17] Reinhard Diestel。《图论》。Springer Publishing Company, Incorporated，第5版，2017年。[Hei89] Jotun J. Hein。一种从加性距离数据重构树的最优算法。《数学生物学通报》，51(5):597–603，1989年。[HMMT18] Jeremy Hoskins、Cameron Musco、Christopher Musco 和 Babis Tsourakakis。根据基于随机游走的节点相似度推断网络。载于《神经信息处理系统进展》，第31卷，2018年。[KMZ18] Sampath Kannan、Claire Mathieu 和 Hang Zhou。图重构与验证。《ACM 算法汇刊》，14(4)，2018年8月。[MZ13] Claire Mathieu 和 Hang Zhou。通过距离预言机进行图重构。载于 Fedor V. Fomin、R¯usi¸nˇs Freivalds、Marta Kwiatkowska 和 David Peleg 编，《自动机、语言与程序设计》，第 733–744 页，柏林、海德堡，2013年。Springer Berlin Heidelberg。[RLYW21] Guozhen Rong、Wenjun Li、Yongjie Yang 和 Jianxin Wang。通过距离预言机重构和验证弦图。《理论计算机科学》，859:48–56，2021年。[RS07] Lev Reyzin 和 Nikhil Srivastava。利用查询学习和验证图，重点关注边计数。载于 Marcus Hutter、Rocco A. Servedio 和 Eiji Takimoto 编，《算法学习理论》，第 285–297 页，柏林、海德堡，2007年。Springer Berlin Heidelberg。[WSSB77] M.S. Waterman、T.F. Smith、M. Singh 和 W.A. Beyer。加性进化树。《理论生物学杂志》，64(2):199–213，1977年。

[WWW14] Suyi Wang、Yusu Wang 和 Rephael Wenger。连接树和分裂树的 js-图。收录于第三十届计算几何年度研讨会论文集，SOCG'14，第 539–548 页，美国纽约，2014 年。计算机协会。