# 沃罗诺伊图还能为平面图中的直径做些什么?

**AmirAbboud** 以色列
魏茨曼科学研究所**Shay**
**Mozes** 以色列雷希曼大学
**OrenWeimann** 以色
列海法大学

arXiv:2305.02946v3 [cs.DS] 4 Jul 2023

**摘要**

Cabello 在 [SODA'17] 中引入沃罗诺伊图技术，以在次二次时间内计算平面图的直径，这一技术彻底改变了平面图距离计算领域。我们在静态、容错和部分动态的无向无权平面图中，提出了该技术的新应用，同时也给出了一些新的局限性。

- 在静态情形下，对于直径为 $D$ 的平面图，我们给出了运行时间为 $n^{3+o(1)}/D^2$ 和 $\tilde{O}(n \cdot D^2)$ 的直径计算算法。当 $D < n^{1/3}$ 或 $D > n^{2/3}$ 时，这些算法比当前最佳结果 $\tilde{O}(n^{5/3})$ [SODA'18] 更快。在容错设置下，对于每条边 $e$，我们给出了一个运行时间为 $n^{7/3+o(1)}$ 的算法，用于计算 $G \setminus \{e\}$ 中的直径（替换直径问题）。这应与朴素的 $\tilde{O}(n^{8/3})$ 时间算法进行比较，后者对每条边运行一次静态算法。在增量设置中，即希望在添加边时维护直径，我们提出了一种总运行时间为 $n^{7/3+o(1)}$ 的算法。这应与朴素的 $\tilde{O}(n^{8/3})$ 时间算法进行比较，后者在每次更新后运行静态算法。我们给出了一个以下界（基于 SETH），排除了以摊销 $O(n^{1-\varepsilon})$ 更新时间维护带权 *weighted* 平面图中直径的可能性。即使对于增量或减量更新，该下界仍然成立。
-

我们的上界是通过对沃罗诺伊图进行新颖的运用和操作得到的。这些方法包括：在删除图中的边时维护 Voronoi 图；允许 Voronoi 图的站点位于 BFS 树层上（而非 $r$-划分的边界上）；以及一种从增量直径归约到增量距离预言机 的新方法，这种方法可能对平面图以外的领域也有意义。我们的下界是首个基于 SETH 条件的动态图问题下界。

**2012 年 ACM 学科分类** 计算理论 → 最短路径；计算理论→ 动态图算法

**关键词与短语** 平面图、直径、动态图、容错

**经费支持** *Amir Abboud*：获 Alon 奖学金及魏茨曼科学研究所新科学家中心研究基金支持。*Shay Mozes*：以色列科学基金会基金 810/21。*Oren Weimann*：以色列科学基金会基金 810/21。

# 1 引言

该直径 问题要求计算图中的最大距离。它是图算法文献中最基础、研究最广泛的问题之一；此外，它在精细复杂性领域也十分重要，并推动了创新困难性归约的发展 [1,4,5,9,11,17,29,36,67]。假设强指数时间假设（SETH）成立，那么对于树宽为 直径 [5,67]的无向无权图，也不存在真正的次二次算法 $\Omega(\log n)$。对于有界树宽的图，可以在近线性时间内计算直径 [5]（另见 [41,50]，其中给出的算法运行时间取决于 $D$）。许多其他受限图族也已有近线性时间算法，例如 [8,14,31–34,40,43,49,66]。

尽管图问题的算法和条件下界在过去十年取得了重大进展，但仍有一个悬而未决的重要问题：平面图中的直径能否在近线性时间内求解？直到 2017 年，人们所知的改进也仅限于在自然的 $O(n^2)$ 界（即计算全点对最短路径，APSP）之上取得对数级提升 [23,72]。普遍看法是，真正的次二次时间是不可能实现的，因此学界的重点是证明其困难性，例如基于 SETH。此后，Cabello [22] 在一篇广受赞誉的论文中给出了一种次二次 $\tilde{O}(n^{11/6})$ 时间算法，之后该算法又得到改进，达到目前最优的 $\tilde{O}(n^{5/3})$ 界 [45]。

Cabello 的工作的突破之处 [22] 在于他创造性地运用了沃罗诺伊图（VD）来设计平面图算法。这一新工具彻底改变了平面图距离计算问题领域，并带来了多项突破 [26,28,35,47,63]，包括一个令人意外且近乎最优的距离预言机——这是一个此前一直通过不同技术逐步改进的问题，在精确 [10,21,26,30,35,39,42、47、57、63–65,73] 以及近似 [24,48,54–56,69,74] 情形下皆是如此。因此，占据平面图算法研究者心头的核心元问题是：沃罗诺伊图还能为我们做些什么？

## 1.1 动态平面直径

人们自然会期待 VD 在动态平面图领域带来突破。支持对图进行更新和查询的动态数据结构，在理论上（作为静态算法的子程序）和实践中（用于处理不断变化的输入）都有卓越的应用。过去几十年里，人们为动态平面图中的基本问题设计了许多巧妙的算法，包括连通性、距离和割 [6,18,19,25,28、37、42、51–53,55,57,59,62,68,69]，但与下界 [3]相比，仍存在很大的（多项式级）差距。这些研究中只有少数 [27,28] 使用了 VD，而且方式有限（每次更新后都从头重新计算 VD）。显然，若能以有意义的方式动态地维护 VD 机制，就有望取得重大进展。本文聚焦于直径问题，探究这种可能性。

当前最先进的算法在每次更新后都以 $\tilde{O}(n^{5/3})$ 的时间从头重新计算直径。这并不令人意外，因为针对直径（在静态图中）唯一有用的技术是基于 VD 的，而我们不知道如何使 VD 动态化。

首先想到的问题是：假设我们乐观地认为，可以让 VD 尽可能动态化；我们希望得到怎样的时间界？显然，在打破静态图的 $\tilde{O}(n^{5/3})$ 界之前，我们不可能将每次更新的时间降至$O(n^{2/3-\varepsilon})$。此外，Abboud 和 Dahlgaard 的归约还给出了一个条件性的 $n^{2/3-o(1)}$ 下界（基于全点对最短路径或在线矩阵向量猜想）。那么，动态 VD 或许会带来一个匹配的 $O(n^{2/3})$ 上界？我们的第一个结果通过一个

## 2 沃罗诺伊图还能为平面图中的直径问题做些什么?

$n^{1-o(1)}$ SETH 下的下界排除了这种可能性。

▶**定理 1** (动态直径下界)。 如果能以每次权重变更 $O(n^{1-\varepsilon})$ 的摊还时间维护一个含有 $n$ 个节点的动态无向平面图的直径，那么 *SETH* 就是错误的。即使动态算法可以用 $poly(n)$ 时间预处理初始图，这一结论仍然成立；在权重只增或只减的部分动态设置中也成立。

值得注意的是，这是首个基于 SETH（而非其他猜想）的动态平面图问题下界；若静态平面图问题也计算在内，这只是第二个此类结果静态平面图问题的例子 [2, 46]。

**迈向动态 Voronoi 图。** 尽管有了我们的下界，仍存在一个很大的差距 $n^{2/3}$，如果我们确实能够实现 VD 的动态维护，这一差距很可能可以弥合。[1] 在本文中，我们朝这一目标迈出一小步（但可以说是第一步）：我们给出一种高效算法，用于在删除图中的一条边后更新 VD，其速度远快于从头重新计算。（概述及全部细节请参见第5 节。）这小小的一步已经带来了有趣的应用。该算法适用于一般的（带权）平面图，但我们发现的应用只有在无权平面图中才能带来优势。

一个具体应用是更快地计算替换直径：给定图 $G$，返回对所有边 $e$，移除边 $e$ 后得到的图 $G\setminus\{e\}$ 的直径。这个问题的朴素算法会对每条边调用一次静态直径算法，共调用 $O(n)$ 次，运行时间为 $\tilde{O}(n^{8/3})$。我们利用对 VD 的高效更新，并结合其他同样基于 VD（但不采用动态方式）的技巧，将这一上界提升了 $n^{1/3}$ 倍，达到 $n^{7/3+o(1)}$。

▶ **定理2** (替换直径)。 给定一个无权无向平面图 $G=(V,E)$，存在一个运行时间为 $n^{7/3+o(1)}$ 的算法，对每条边 $e\in E$ 输出图 $G^e=(V,E\setminus\{e\})$ 的直径。

另一项新成果是在 增量式 场景下更快地计算直径：从空图开始，在添加 $O(n)$ 条边的同时（且不破坏平面性），需要持续维护图的直径。朴素算法会在每次更新后重新计算直径，总运行时间为 $\tilde{O}(n^{8/3})$，我们将其改进为 $n^{7/3+o(1)}$。

▶ **定理 3** （增量直径）。 存在一种算法，能够在边插入操作下维护无权无向平面图的直径，总计用时为 $n^{7/3+o(1)}$。

该结果基于一个优雅的归约：将增量直径归约为增量距离预言机，这一归约可能对平面图以外的情形也有意义。其分析依赖于近期关于二分独立集查询的研究，这类查询由 Beame 等人提出。 [13]。

---

[1] 人们很容易认为定理 6 意味着存在一种更新时间为 $\tilde{O}(n^{1.6})$ 的动态直径算法；使用一个 $r$-划分，并为每个区域维护 DDG 和垂直平分线。在区域 $P$ 中的一条边发生更新后，重新计算 $P$ 的 DDG（使用 MSSP）以及 $P$ 的垂直平分线（使用定理 6）。对于图中的每个顶点，使用 FR-Dijkstra 重新计算所有加性权重，并使用定理 6 计算每个区域中的最远顶点。问题在于，当直径路径的两个端点属于同一区域时（不一定是 $P$），这种方法无法正确处理这种情况。原因是，VD 机制只能处理至少经过一个边界节点的路径。

## 1.2 静态平面直径

回到静态图中的直径问题，我们还能从 VD 中获得什么？当然，最大的开放问题是， $n^{5/3}$ 界能否改进为 $n^{1+o(1)}$，或者能否证明一个超线性的下界。为了回答这一问题，我们希望理解困难/简单情形，而一个自然的考量参数是 $D$ ——直径本身。

本文的一项主要算法贡献对于前述上界至关重要：当 $D$ 较大（处于范围 $[n^{2/3+\varepsilon}, n]$内）时，该算法能够击败 $n^{5/3}$ 。值得注意的是，这意味着任何试图证明紧确条件下界的人都不能采用直径极大的构造。

▶ **定理 4** (静态大直径)。 对于直径为 $D$ 的无权无向平面图，可以在 $n^{3+o(1)}/D^2$ 时间内计算其直径。

我们的新算法以一种新颖的方式应用 VD，其中 VD 站点位于 BFS 树层上，而非位于某个 $r$-划分中各区域的边界上。

虽然我们的结果首次处理了大 $D$ 的情形，但另一极端——小 $D$ ——早已得到研究。Eppstein [41] 提出了首个针对常数 $D$ 的近线性时间算法，其对 $D$ 的依赖呈指数级。后来，作为针对直径 [15,24,41,70] 的新型（$1+\varepsilon$）近似算法的副产品，这一依赖得到了改进。当前最佳结果为 $\tilde{O}$（$n\cdot D^5$），采用 Chan 和 Skrepetos[24] 的（$1+\varepsilon$）近似算法 $\tilde{O}$（$n\cdot(1/\varepsilon)^5$）时间算法，并满足 $\varepsilon = 1/D$。本文的最终结果是将上界改进为 $\tilde{O}$（$nD^2$），从而将能够击败该 $n^{5/3}$ 界的范围从 $D < n^{2/15-\varepsilon}$ 扩大到 $D < n^{1/3-\varepsilon}$。

▶ **定理 5** (静态小直径)。 在直径为 $D$ 的无权无向平面图上，直径可在 $\tilde{O}(n\cdot D^2)$ 时间内计算得出。

与 Chan 和 Skrepetos 的算法相比，我们的算法以一种更加自然的方式利用 VD [24]，前提是我们的目标是精确求解较小的 $D$ 情形（请回想一下，他们关注的是近似算法）。关于 $\tilde{O}(n\cdot(1/\varepsilon)^5)$时间复杂度的近似算法能否得到改进，仍然是一个值得研究的开放问题。这与另一个挑战有关，即如何比精确 VD 更快地计算近似 *VD*，而这并非本文所讨论的内容。

## 2 预备知识

平面图 $G$ 的递归分解树 $G$ 通过递归地用大小为 $G$ 的分隔器分隔 $\sqrt{|G|}$，在线性时间内得到。$\mathcal{T}$ 是一棵二叉树，其节点对应于 $G$ 的子图（称为 区域），根节点对应整个 $G$，叶节点对应常数大小的片段。我们将每个片段 $P$ 与 $\mathcal{T}$中代表它的节点等同起来（因此可以滥用记号，写作 $P \in \mathcal{T}$），并将其与边界 $\partial P$（即递归分解过程中用于得到 $P$的某个分隔器中的顶点）等同起来。对我们而言，一个重要性质（例如参见 [47, 引理 3.1]）是，所有兄弟节点对 $P, P\prime$在 $\mathcal{T}$中的 $|P|\cdot|\partial P'|$ 之和为 $\tilde{O}(n^{1.5})$。

平面图 $G$ 的 $r$-分解 [44] 是将 $G$ 分解为 $\Theta$ 个（$n/r$）片段，其中每个片段包含 $O$ 个（$r$）顶点和 $O$ 个（$\sqrt{r}$）边界顶点（即与其他片段共享的顶点）。可以在 $O$（$n$）时间内计算出一个 $r$-分解 [58]，并具有一项实用性质：每个片段的边界顶点都位于该片段的常数个面上（称为孔洞）。

区域 $P$ 的稠密距离图 (DDG)是 $P$ 的边界顶点构成的完全图。$P$ 的 DDG 中边 $uv$ 的长度等于 $P$ 内 $u$ 到 $v$ 的距离。请注意， $P$ 的 DDG 是非平面的。$r$-划分的 DDG 是该 $r$-划分中所有区域的 DDG 的并集。因此，DDG 中的顶点总数为 $O(n/\sqrt{r})$，且 _

˜边的总数为 $O(n)$。一个 $r$-division 的 DDG 可使用 MSSP 算法在 $O(n)$ 时间内˜计算得出 [57]。Fakcharoenphol 和 Rao [42] 描述了一种在 DDG 上运行的 $O(n/\sqrt{r})$ 时间算法 Dijkstra 实现（昵称为 FR-Dijkstra）。

计算直径时，困难的情况是最远顶点对分别位于不同的区域中。考虑一个位于某个区域 $P$之外的源顶点 $s$。对于 $P$ 的每个边界顶点 $u$，令 $d(u)$ 表示在 $G$ 中从 $s$ 到 $u$ 的距离。相对于 $d(\cdot)$ 的 $P$ 加权沃罗诺伊图，是将 $P$ 的顶点划分成两两不相交的集合（沃罗诺伊单元）；每个单元都对应一个唯一的边界顶点（站点）$u$。单元 Vor($u$) 中的顶点，是 $P$ 中满足以下条件的所有顶点 $v$：$u$ 是 $P$ 中从 $s$ 到 $v$ 的最短路径上最后一个边界顶点。换句话说，$P$ 中的每个站点 $u$ 都具有加性权重 $d(u)$；从站点 $u$ 到 $P$ 中顶点 $v$ 的加性距离，定义为 $d(u)$ 加上 $P$ 内从 $u$ 到 $v$ 的最短路径长度；单元 Vor($u$) 则包含 $P$ 中所有相较于 $S$ 中任何其他站点、到 $u$ 更近（按加性距离计）的顶点 $v$。单元 Vor($u$) 的边界 $\partial$Vor($u$) 由 $P$ 中恰有一个端点位于 Vor($u$) 内的所有边组成。例如，在仅有两个站点 $u$ 和 $v$ 的沃罗诺伊图中，单元 Vor($u$) 的边界是一条 $uv$-割，因此也是对偶图中的一个环。这个环称为 $uv$-平分线。沃罗诺伊单元 Vor($u$) 的复杂度 $|\partial\text{Vor}(u)|$，是指 $P$ 中同时包含 Vor($u$) 的顶点以及至少另外两个沃罗诺伊单元顶点的面的数量。因此，对于每个源点 $s$，要计算 $P$ 中距 $s$ 最远的顶点，只需对每个站点 $u$，求出 Vor($u$) 中距 $u$ 最远的顶点（按加性距离计），然后返回所有站点 $u$ 对应的最大值。

▶定理 *6*（[45]）。设 $P$ 是一个边带权平面图，具有 $r$ 个顶点。设 $S$ 是一组 $b$ 个站点，它们位于 $\tilde{O}$ 个（*1*）[2] 面的边界上。所有站点对 $u$、$v \in S$ 的 $uv$-平分线以及所有可能的加性权重 $d$（$u$） $d$（$v$）均可在 $\tilde{O}$（$rb^2$）时间和空间内计算并表示。给定任意分配给 $S$ 的加性权重 $d$（·），可在 $O$（$|S|$）时间内构造相对于这些权重的 *Voronoi˜* 图表示。利用这一表示，对于任意站点 $u \in S$，我们可以查询从 $u$ 到 *Vor(u)* 中某个顶点的最大距离，查询时间为 *˜O*（$|\partial$*Vor(u)*$|$）。

# 3 静态直径

## 3.1 一种 $n^{3+o(1)}/D^2$ 算法

在本小节中，我们证明定理4，该定理指出，在直径为$n^{3+o(1)}/D^2$ 的无权无向平面图上，可以在 $D$时间内计算直径。我们首先介绍一种随机化的 $\tilde{O}$（$n^4/D^3$）时间算法，随后说明如何将其改进为$n^{3+o(1)}/D^2$。接着，我们将说明如何对这两种算法进行去随机化。首先，我们给出关于直径为 $\geq D$时 BFS 层的两个简单观察。

▶ **观察 7.** 设 $s$ 是一个直径为 $\geq D$ 的图中的任意节点。那么，以 $s$ 为根的广度优先搜索树中，至少有一个位于中间的 $D/2$ 层，其大小为 $O(n/D)$。

▶观察 *8. </style}*设 $s$ 是 $G$ 中的任意一个节点，$L_i$ 是以 $s$ 为根的广度优先搜索树中位于第 $i$ 层的节点集合。设 $G_i$ 是由 $\bigcup_{j\geq i} L_j$ 诱导的 $G$ 子图。那么，对于 $G_i$ 的每个连通分量 $C$，$L_i \cap C$ 中的节点都位于同一个面上。

[2] 定理 1.1 中 [45] 表述的是面数 d（称为孔洞）为常数的情况。不过，正如 [45], 脚注 8 所指出的，运行时间对孔洞数量的依赖是多项式级的，因此该定理也适用于孔洞数量为多对数级的情况。

**证明。** 为说明 $L_i \cap C$ 的顶点都位于 $G_i$的同一个面上，考虑分量 $C$ 在 $G_i$ 中的嵌入，该嵌入继承自 $G$的嵌入。将 $C$ 视为通过删除边和顶点从 $G$ 得到的图，可以从 $L_i$ 中的任意顶点出发，沿着一条仅经过已删除边和顶点的平面曲线前进，直到抵达广度优先搜索树的根 $s$ 。因此， $L_i$ 的所有顶点都位于 $C$的同一个面上，因而也位于 $G_i$的同一个面上。◀

**一种随机化算法。**我们首先用 $O(n)$ 时间计算出 $D$ 的一个 2-近似（下界） $\tilde{D}$：计算一棵广度优先搜索树，并将 $D$ 设为从根到叶子的最远距离。然后，我们重复以下过程 $\theta(n\log n/D)$ 次，并返回找到的最大距离：

1. 随机选取一个源点 $s$，并计算其广度优先搜索树。令 $D'$ 为这棵树的深度。注意，$D \geq D' \geq D/2$。令 $S = L_i$ 为满足 $D'/4 < i < 3D'/4$ 和 $|S| = O(n/D') = O(n/D)$ 的第 $i$ 层节点集合。根据观察 7，这样的集合存在。令 $G_i$ 为由 $\bigcup_{j\geq i} L_j$ 诱导的 $G$ 子图。2. 对所有 $v \in G$ 和所有 $b \in S$，计算 $d(v,b)$。 3. 对 $G_i$ 的每个连通分量 $C$：a. 计算 $C$ 中站点 $C \cap S$ 的所有垂直平分线（根据观察 8，这些站点位于同一个面上）。b. 对每个属于 $G \setminus G_i$ 的节点 $v$，计算 $C$ 关于加性权重 $d(v,b)$ 的 Voronoi 图，并计算从 $v$ 到该 Voronoi 图每个 Voronoi 单元中最远顶点的距离。

运行时间。第一步通过计算并遍历 $s$ 的广度优先搜索树，需要 $O(n)$ 时间。第二步通过从 $S$ 的每个顶点执行一次广度优先搜索，需要 $O(n^2/D)$ 时间，每次搜索耗时 $O(n)$。开销最大的步骤是 3a。根据定理 6，可以在 $\tilde{O}(|C|\cdot|C\cap S|^2)$ 时间内计算连通分量 $C$ 的所有垂直平分线。对所有连通分量求和，总耗时为 $\tilde{O}(n\cdot(n/D)^2)$（因为各个 $C$ 互不相交，且总和为 $n$，而各个 $C \cap S$ 也互不相交，且总和为 $O(n/D)$）。最后，在步骤 3b 中，对于每个顶点 $v$，根据定理 6，计算 $v$ 的 VD 及每个单元中的最远 $\tilde{}$ 顶点需要 $O(|C \cap S|)$ 时间。对所有连通分量求和，总耗时为 $\tilde{O}(n/D)$，因此对所有顶点 $v$ 求和后，耗时为 $\tilde{O}(n^2/D)$。因此，整个过程的总运行时间为 $\tilde{O}(n\cdot(n/D)^2)$；由于我们重复执行该过程 $\tilde{O}(n/D)$ 次，最终得到 $\tilde{O}(n^4/D^3)$。

正确性。我们还需证明，以高概率返回的距离确实是直径。设 $x, y$ 是直径的两个端点（即 $D = d(x,y)$）。那么，随机源点 $s$ 满足 $d(s,x) \leq D'/4$ 且 $d(s,y) \geq 3D'/4$ 的概率至少为 $D'/4n$ （因为当 $s$ 是从 $x$ 到 $y$ 的路径上最前面的 $D'/4$ 个节点之一时，就会出现这种情况）。因此，对于我们选取的源点 $s$，至少有一个以高概率满足这一条件。对于这个 $s$，我们有 $x \in G \setminus G_i$，而 $y \in G_i$（不可能有 $y \in G \setminus G_i$，因为那样一来，经过 $s$ 的 $x$-到-$y$ 路径就会比 $D$ 更短），于是我们找到的最大距离必定是 $d(x,y)$。

**去随机化。**注意，要对算法进行去随机化，只需将源点的采样替换为确定性地选出一个大小为 $O(n/D)$ 的源点集合$\mathcal{S}$，使得某个直径端点 $x$ 到至少一个源点 $s \in \mathcal{S}$的距离为 $\leq D'/4$。

要构造 $\mathcal{S}$，任意选择一个源点 $s$，并计算其深度为 $D' \leq D$ 的广度优先搜索树 $T$。找到一个层 $L_i$，其中只有 $O(n/D')$ 个 $= O(n/D)$ 节点和 $0.4D' \leq i \leq 0.5D'$。类似地，找到一个层 $L_j$，其中只有 $O(n/D)$ 个节点和 $0.8D' \leq j \leq 0.9D'$。源点集合随后为 $\mathcal{S} = \{s\} \cup L_i \cup L_j$。容易验证，图中的每个顶点 $v$ 在 $T$ 中都有一个祖先或后代属于 $\mathcal{S}$，且与 $v$ 的距离至多为 $D'/4 \leq D/4$。

**一种更快的算法。** 接下来，我们将运行时间优化至 $n^{3+o(1)}/D^2$。同样，我们将先从一种随机化算法入手，然后再将其去随机化。令 $B_\rho(v)$ 表示以顶点 $v$ 为中心、半径为 $\rho$ 的球。回顾一下，我们的目标是以高概率在 $B_{\tilde{D}/4}(x)$ 中采样一个顶点 $s$（无需知道 $x$），其中 $x$ 是一个直径端点。

令 $\rho = D/4$。为了以高概率从 $B_\rho(x)$ 中采样一个顶点 $s$，只需随机采样一个包含 $O(n/|B_\rho(x)|)$ 个顶点的集合（而不是像上述方法那样采样 $O(n/\rho)$ 个顶点）。然后，对于每个采样到的顶点 $s$，我们可以在 $s$ 的广度优先搜索树中找到一个层 $L_i$，使得$\rho < i \le 2\rho$ $|L_i| < |B_{2\rho}(s)|/\rho$（而不是像上述方法那样使用 $n/\rho$）。然后，对某个特定的$s$ 执行上述方法（即执行上述 $\tilde{O}(n^4/D^3)$ 算法的步骤 2–3），计算所有垂直平分线需要 $\tilde{O}(n(|B_{2\rho}(s)|/\rho)^2)$ 的时间，计算所有加性权重需要 $\tilde{O}(n|B_{2\rho}(s)|/\rho)$ 的时间，构造上方所有顶点的 Voronoi 图需要 $\tilde{O}(|B_{2\rho}(s)|^2/\rho)$ 的时间。由此可见，如果 $|B_\rho(x)|$ 较大，我们就能受益，因为需要采样的顶点更少；如果 $|B_{2\rho}(s)|$ 较小，我们也能受益，因为处理每个采样顶点所需的工作量会减少。

要使这一方法奏效，我们需要（1）估计 $|B_\rho(x)|$，以及（2）建立 $|B_\rho(x)|$ 和$|B_{2\rho}(s)|$之间的关系。为解决（1），我们只需枚举所有二的幂 $2^k$，从而估计 $|B_\rho(x)|$，其中 $0 \le k \le \log n$。为解决（2），注意到 $|B_\rho(x)| < |B_{2\rho}(s)| < |B_{3\rho}(x)|$，并且必定存在某个 $j \in \{1, 2, \ldots, \sqrt{\log_3 n}\}$，使得 $|B_{3\rho 3^{-j}}(x)|/|B_{\rho 3^{-j}}(x)| < 3^{\sqrt{\log_3 n}} = n^{o(1)}$（否则，$|B_\rho(x)| > n$，矛盾）。

因此，算法如下：对每个 $1 \le j \le \sqrt{\log_3 n}$，令 $\rho_j = 3^{-j}\rho$。对每个$0 \le k < \log n$，我们采样（$n \log n$）$/2^k$ 个顶点 $s$（这反映了我们的假设，即 $B_{\rho_j}(x) \le 2^k$）。对于每个采样顶点 $s$，如果 $|B_{2\rho_j}(s)| > 2^k 3^{\sqrt{\log_3 n}}$，那么，由于 $|B_\rho(x)| < |B_{2\rho}(s)| < |B_{3\rho}(x)|$，必定有 $s \notin B_{\rho_j}(x)$或 $|B_{\rho_j}(x)| > 2^k$或 $|B_{\rho_{j-1}}(x)|/|B_{\rho_j}(x)| > 3^{\sqrt{\log_3 n}}$（这些条件并非互斥）。因此，在这种情况下，我们舍弃 $s$ 并继续处理下一个采样顶点。否则，$|B_{2\rho_j}(s)| \le 2^k 3^{\sqrt{\log_3 n}}$，我们就能在以 $s$ 为根的广度优先搜索树中找到一个层 $L_i$，满足 $\rho_j < i < 2\rho_j$，然后按照先前算法的步骤2–3继续执行。总运行时间为

$$\sum_{j=0}^{\sqrt{\log_3 n}} \sum_{k=0}^{\log n} \tilde{O}\left(\frac{n}{2^k}\left(n(2^k 3^{\sqrt{\log_3 n}}/\rho_j)^2 + n2^k 3^{\sqrt{\log_3 n}}/\rho_j + (2^k 3^{\sqrt{\log_3 n}})^2/\rho_j\right)\right) = n^{3+o(1)}/D^2.$$

为论证正确性，注意对于满足 $|B_{\rho_{j-1}}(x)|/|B_{\rho_j}(x)| \le 3^{\sqrt{\log_3 n}}$ 的 $j$，以及满足 $2^{k-1} \le |B_{\rho_j}(x)| \le 2^k$ 的 $k$，采样 $(n \log n)/2^k$ 个顶点后，以高概率会得到一个顶点 $s \in B_{\rho_j}(x)$，且该顶点不会被丢弃。这个 $s$ 满足 $d(s, x) \le \rho_j$，并且满足 $d(s, y) \ge 2\rho_j$，因此，按照慢速算法正确性证明中的相同论证，可以保证该 $s$ 找到的最大距离为 $d(x, y)$。

**去随机化。** 我们使用 Busch、Lafortune 和 Tirthapura 的稀疏邻域覆盖[20] 对该算法进行去随机化。图 $G$ 的 $\rho$-邻域覆盖 $Z$ 是一组称为簇的连通子图，其并集为图 $G$ 的顶点集，并且对于每个节点 $v \in G$，都存在一个包含 $B_\rho(v)$ 的簇 $C \in Z$。覆盖 $Z$ 的半径是 $Z$ 中各簇半径的最大值。覆盖 $Z$ 的度是 $G$ 中一个节点所属簇的最大数量。Busch 等人给出了一种确定性算法，可在 $(n \log n)$ 时间内，对于任意 $\rho > 0$ 和任意连通平面图，计算半径为 $16\rho$、度为 18 的 $\rho$- 邻域覆盖。另请参见 [60]，其中给出了一个时间复杂度为 $O(n)$ 的算法。

为调整论证，我们重新定义 $\rho_j = \rho 33^{-j}$ for $j = 1, \ldots, \sqrt{\log_{33}(n)}$，并利用这样一个事实：对于某个 $j$, $|B_{\rho_{j-1}}|/|B_{\rho_j}| < 33^{\sqrt{\log_{33} n}}$。为了避免在算法中进行采样，对于

每种 $j, k$选择，我们都会计算一个 $\rho_j$-邻域覆盖 $Z$。我们从 $Z$ 中满足 $|C| > 2^k$ 条件的每个簇 $C$ 里任意选取一个顶点 $s$。由于 $Z$ 的度为 18，我们选取的顶点数 $s$ 至多为 $18n/2^k$。

如果 $2^k < |B_{\rho_j}(x)| > 2^{k+1}$，那么包含 $B_{\rho_j}(x)$ 的簇 $C$ 将有 $|C| > 2^k$ 个顶点，而我们将选择一个顶点 $s \in C$。由于 $Z$ 的半径为 $16\rho_j$，$d(s,x) \leq 16\rho_j$。如果 $|B_{17\rho_j}(s)| > 2^{k+1}33^{\sqrt{\log_{33} n}}$，我们就丢弃 $s$。由于 $B_{17\rho_j}(s)$ 包含于 $B_{33\rho_j}(x) = B_{\rho_{j-1}}(x)$ 中，因此可以保证至少有一个 $s$ 不会被丢弃。对于这样的 $s$，我们在以 $s$ 为根的广度优先搜索树中找到一层 $L_i$，使得 $16\rho_j < i < 17\rho_j$，且 $|L_i| < 2^{k+1}33^{\sqrt{\log_{33} n}}/\rho_j$。$x$ 在广度优先搜索树中的层数至多为 $16\rho_j$；又由于 $\rho_j < \rho < D/4$，满足 $d(x,y) = D$ 的顶点 $y$ 在广度优先搜索树中的层数大于 $i$。因此，执行第 5 节中该算法过程的第 2–3 行将报告距离 $D$。运行时间的分析与随机化版本相同，因为我们确保了去随机化过程中选择的顶点数量至多为随机化算法中采样顶点数量的某个固定常数倍。

## 3.2 一个 $\tilde{O}(n \cdot D^2)$ 算法

在本小节中，我们证明定理 5，该定理指出，对于直径为 $D$的无权平面图，可以在 $\tilde{O}(n \cdot D^2)$ 时间内计算其直径。我们首先介绍一些预备知识，涉及使用最短路径分隔器进行递归分解。

预备知识。平面图 $G$ 的最短路径分隔器是一个无向环 $C(G)$，由一条从 $s$ 到 $u$ 的最短路径、一条从 $s$ 到 $v$ 的最短路径以及一条单独的边 $uv$ 组成，使得该环的内部和外部各自所包含的面数均不超过总面数 $G$ 的 $2/3$。这样的分隔器可以在 [61] 时间内找到，耗时为 $O(n)$。通过使用最短路径分隔器递归地分隔$G$（当达到大小为 $\leq D$ 的子图时停止递归），我们得到分解树 T。T 的根对应于整个图 $G$。对应于子图 $P$ 的节点（我们也将其称为节点 $P$）有两个子节点，其子图分别对应于分隔器 $C(P)$ 的内部和外部。

注意，对于每个节点 $P \in$ T，最短路径分隔器 $C(P)$ 的大小为 $O(D)$。这是因为 $C(P)$ 由两条最短路径组成，每条路径的长度至多为 $D$。此外，$P$（$P$ 中与不属于 $P$ 的顶点有相邻边的顶点）的边界包含在所有 $C(P')$ 的并集中，其中 $P'$ 是 $P$ 的祖先，因此其大小为 $O(D \log n)$，并且位于 $P$ 的 $O(\log n)$ 个面上。我们为每个节点（子图）$P \in \tilde{\text{T}}$ 计算 DDG（即，计算一个数据结构，使其能在 $O(1)$ 时间内报告图 $P$ 中 $P$ 的任意一对边界顶点之间的距离），方法是在 $P$ 上执行 $O(\log n)$ 次 MSSP。这样，在整个 T 上总共需要 $O(n)$ 时间。现在，给定子图 $P$ 中的任意顶点 $v$，我们可以使用 FR-Dijkstra，在 $O(D)$ 时间内计算 $G$ 中从 $v$ 到 $P$ 的所有边界顶点的距离。具体来说，我们将 $P$ 的 $O(D)$ 个边界顶点初始化为它们在图 $P$ 中距 $v$ 的距离（通过 MSSP 查询获得），然后在 $P$ 的 DDG 与所有满足以下条件的 $P'$ 的 DDG 的并集上运行 FR-Dijkstra：$P'$ 是 $P$ 的某个祖先的兄弟节点。

该算法。对于每个非叶节点 $P \in$ T，我们计算一对最远顶点 $u, v \in P$，其中 $u$ 位于 $C(P)$ 内部，而 $v$ 位于 $C(P)$ 外部。注意，距离必须在整个图 $G$ 中计算，因为最短的 $u$到$v$ 路径可能会延伸到 $P$之外。为此，我们预先计算图 $P \in$ T 中的所有平分线，站点为 $P$的 $O(D)$ 个边界顶点。根据定理 6，这需要 $\tilde{O}(|P| \cdot D^2)$ 时间（其中 $|P|$ 表示子图 $P$的大小），因此对所有 T 来说总共需要 $\tilde{O}(n \cdot D^2)$ 时间。（注意，这里我们使用定理 6 时，站点位于 $O$个面上，数量为 $O(\log n)$。据我们所知，在此前所有

对定理 6 的应用中，站点都位于 O( $O$) 个面上）。然后，对于每个顶点 $v \in P$，我们使用 FR-Dijkstra，在 $O(D)$ 时间内计算 $G$ 中从 $v$ 到 $P$ 所有边界顶点的距离，具体方法如上所述。接着，我们将这些距离作为加性权重，并应用定理 6，在 $P$中找出距 $v$ 最远的顶点。这同样需要 $O(D)$ 时间，因此总计为 $O(n \cdot D)$。

我们显式处理叶节点 $P \in \mathcal{T}$（回想一下 $|P| \leq D$）。对于每个叶节点 $P$，我们计算 $G$ 中任意两个顶点 $u, v \in P$ 之间的全点对最短路径（APSP）。具体做法是：在图 $P$ 上，从每个 $v \in P$ 运行戴克斯特拉标准算法，并将 $P$ 的边界顶点初始化为它们在 $G$ 中距 $v$ 的距离（这些距离已作为 $v$ 的加性权重计算得出）。每个 $v$ 需要 $\tilde{O}(D)$ 时间，因此每个 $P$ 需要 $\tilde{O}(D^2)$ 时间；对所有叶节点 $P$ 而言，总计需要 $\tilde{O}(D^2 \cdot n/D) = \tilde{O}(nD)$ 时间。

## 4 动态直径的一个下界

本节证明定理1。具体而言，我们给出一个条件下界，排除在一系列边权更新过程中维护 $O(n^{1-\varepsilon})$ 摊销更新时间的可能性；这里维护的是带权平面图的直径。

该证明的灵感来自 [3], ；不过，与 [3] 中的归约相比，这里有不少变化：归约对象是全点对最短路径问题（而非 SETH），目标是动态距离预言机（而非动态直径），并且排除了 $O(n^{0.5-\varepsilon})$ 更新时间（而非 $O(n^{1-\varepsilon})$）。我们的归约针对以下问题；它只是用图的语言重新表述正交向量问题。

▶ **定义 9** (图正交向量问题)。**给定一个无向三部图** $G$，**其部分为** $A, C, B$，其中$|A| = |B| = n$，且中间层的大小为 $|C| = O(\log n)$。所有边都属于 $A \times C$和$C \times B$。判断是否存在一对 $a_i \in A$, $b_j \in B$，使得 $d_G(a_i, b_j) > 2$。

已知，若能在 $O(n^{2-\varepsilon})$ 时间内解决该图正交向量问题，则 SETH 将被推翻 [67, 71]。此外，对于满足 $|A| = n^{\alpha}$ 且 $|B| = n^{\beta}$ 的任意常数$\alpha$，$\beta > 0$ 我们知道，若存在一个 $O(n^{\alpha+\beta-\varepsilon})$ 时间算法，则 SETH 将被推翻。

归约结构。给定图正交向量问题的一个实例 $G$，我们构造一个动态平面图 $H$。图 $H$ 由两个网格组成，分别为左网格和右网格，每个网格的尺寸为 $|C| = O(\log n) \times |A| = n$。两个网格的列均以 $A$ 的节点为索引，因此左（分别为右）网格中第 $i^{th}$ 列的顶端节点称为 $a_i$（分别为 $a_i'$）。网格的行对应于 $C$ 中的节点，因此左（分别为右）网格中第 $k^{th}$ 行最右侧（分别为最左侧）的节点称为 $c_k$（分别为 $c_k'$）。两个网格中的所有水平边的权重均为 $2|C|$。在左网格中，第 $i$ 列的竖直边权重为 $2i$；在右网格中，第 $i$ 列的竖直边权重为 $2(n-i)$。在左网格中，对于每个 $i$ 和 $k$，若 $G$ 中存在边 $(a_i, c_k)$，则添加一条从顶点 $(k-1, i)$ 指向顶点 $(k, i+1)$ 的对角边 $e_k$，其权重为 $2i + 2|C| - 1$。我们称这样的边为捷径边（与先经过一条竖直边、再经过一条水平边的替代路径相比，它短 1）。两个网格通过 $|C|$ 条边相连：对于每个 $k$，我们都添加一条从 $c_k$ 到 $c_k'$、权重为 $2n|C| - 2nk$ 的边。这些 $|C|$ 条边是 $H$ 中唯一在整个归约过程中权重会发生变化的边，其余边的权重都保持不变。我们添加一个单独的节点 $x$，并将其连接到左网格顶行中的所有节点以及右网格顶行中的所有节点。我们将每条边 $(a_i, x)$ 的权重设为 $i \cdot 4|C|$，并将每条边 $(x, a_j')$ 的权重设为 $(n-j) \cdot 4|C|$。

**动态更新**。构造出上述初始图 $H$ 后，对于每个 $j$ =1、……、$n$，我们通过对 $H$执行以下更新来得到一个图 $H_j$：对于每个 $k = 1$、……、$|C|$，如果边（$c_k$，$b_j$）存在于 $G$ 中，则将 $H$ 中边（$c_k$，$c_k'$）的权重减 1（我们将这类边（$c_k$，$b_j$）称为减权边）。

下面的主引理表明， $H_j$ 的直径揭示了是否存在某个 $a_i \in A$ ，使得 $d_G(a_i, b_j) > 2$。需要特别注意的是，我们可以仅使用 $O(n \log n)$ 次更新，依次生成所有图 $H_1$、……、$H_n$，因为$H_i$ 与 $H_{i-1}$ 仅有 $O(\log n)$ 条边的权重不同。在 SETH 假设下，我们无法在 $O(n^{2-\varepsilon})$ 时间内维护这一序列中的直径。因此，每次更新都不可能在 $O(n^{1-\varepsilon})$ 摊还时间内完成，从而证明了全动态情形下的定理 1。为了证明增量情形（其中边权只能减小），我们可以进行如下操作（减量情形是对称的）。重新定义这 $O(\log n)$ 条边的权重，使其在整个序列中只减小：在 $H_i$ 中将 2（$n-i$）加到它们的权重上，使得它们的权重在 $H_1$ 中最大、在 $H_n$ 中最小。这样，整个图序列仅需通过 $O(n \log n)$ 次减权更新即可生成。$H_i$ 的直径恰好增加 2（$n-i$），因此相同的分析仍然成立。为简化起见，本节后续证明继续采用全动态情形下的构造。

▶ **引理 10.** 对于任意 $j$， $H_j$ 的直径大于 $4n|C|-2$，当且仅当存在 $a_i \in A$ s使得 $d_G(a_i, b_j) > 2$。

在本节余下部分中，我们证明上述引理。首先注意，根据我们对边权重的选择， $H_j$ 的直径对应于某条最短的 $a_i$-到-$a'_\ell$ 路径。下面的断言表明，事实上它是一条 $a_i$-到-$a'_i$ 路径。

▷ 断言 11. 对所有 $i \neq \ell$, $d_{H_j}(a_i, a'_\ell) < 4n|C|-2$。

**证明**。 如果 $\ell > i$，那么路径 $a_i - x - a'_\ell$ 由两条边组成，其代价为 $(n-(\ell-i))\cdot 4|C| < 4n|C|-2$。否则 $\ell < i$，那么 $a_i$-到-$a'_\ell$ 仅使用水平边的路径，其代价为$2|C|(n-i+\ell)$ $+2|C| = 2n < 4n|C|-2$。 ◀

以下断言完成了证明。

▷ 断言 12. 对任意 $i$, $d_{H_j}(a_i, a'_i) > 4n|C|-2$ 当且仅当 $d_G(a_i, b_j) > 2$。

**证明**。注意，由两条边组成的路径 $a_i - x - a'_i$ 的代价为 $4n|C|$。然而，可能存在一条更短的 $a_i$到$a'_i$ 路径经过网格。根据我们对边权的选择（与 [3]类似），这条最短路径必定以左侧网格中的一段 $a_i$到$c_k$ 前缀开头（对于某个$k \le |C|$），然后使用$(c_k, c'_k)$ 边，接着在右侧网格中先走 $i$ 步水平步，再走 $k$ 步垂直步。此外， $a_i$到$c_k$ 前缀先走 $k-1$ 步垂直步，然后使用一条捷径边 $e_k$ （如果存在；否则先走一步水平步，再走一步垂直步），接着走 $n-i-1$ 步水平步，直到到达 $c_k$。

首先假设完全没有快捷边和缩短边。那么这样的 $a_i$-到-$a'_i$ 路径长度将会是

$$d_H(a_i, a'_i) = k\cdot 2i + (n-i)\cdot 2|C| + (2n|C| - 2nk) + i\cdot 2|C| + k\cdot 2(n-i) = 4n|C|.$$

注意，该长度（$4n|C|$）与 $k$ 和 $i$ 无关，始终保持不变。因此， $a_i$-到-$a'_i$ 的路径若要比 $4n|C|$ 更短，唯一的方式就是使用捷径边和缩短边。然而，它至多只能使用一条捷径边 $e_k$ 和一条缩短边 $(c_k, c'_k)$。因此，当且仅当存在一个 $k$，使得捷径 $e_k$ 存在（即 $(a_i, c_k) \in E(G)$），且边 $(c_k, c'_k)$ 被缩短（即 $(c_k, b_j) \in E(G)$）时，其长度才为 $4n|C|-2$，这又等价于 $d_G(a_i, b_j) \le 2$。 ◀

▶ 备注 13. 通过细分所有边，上述归约表明，不可能以 $O(n^{1/2-\varepsilon})$ 的更新时间维护无权平面图的直径。

**10 平面图中的直径问题还能从沃罗诺伊图中获得什么?**

## 5 递减式沃罗诺伊图与替换直径

**概述：迈向动态 Voronoi 图。** Voronoi 图对于静态平面图中的直径计算和距离查询很有用，因此自然会想到，能否在动态场景下高效维护 Voronoi 图的某种有用表示。这似乎颇具挑战，因为只要一条边发生变化，或一个加性权重发生变化，就可能导致整个 Voronoi 图彻底改变。例如，降低某条边的权重，可能会使某个站点 $\mathrm{Vor}(b)$ 的 Voronoi 单元 $b$ 扩张，$\mathrm{Vor}(b)$ 从而挤占其他所有 Voronoi 单元的空间，甚至包括变化前并非 $\mathrm{Vor}(b)$邻居的单元。降低 $b$的加性权重也会产生同样的效果。事实上，据我们所知，少数尝试在动态平面图中使用 Voronoi 图的方法， [27, 28], 都会在每次更新时从头重新计算 Voronoi 图。

我们通过开发一种在递减式场景下更新沃罗诺伊图的机制，朝动态 Voronoi 图迈出了一小步。在我们看来，这是本工作中最具创新性的技术贡献。删除图中某个部分的一条边，只会导致某些站点的加性权重增加。当站点$b$ 的加性权重增加时，其 Voronoi 单元会缩小。也就是说，增加之前属于 Vor($b$) 的某些顶点，在增加之后将属于其他站点的 Voronoi 单元。关键的观察是，在这一过程中，唯一相关的站点是 $b$ 以及原始 Voronoi 图中与 Vor($b$) 相邻的单元所对应的站点。因此，所得更新过程所需的时间与 $b$的单元度成正比，而非与 Voronoi 图中的站点总数成正比。遗憾的是，一般而言， $b$ 的单元度可能与站点数一样大。尽管如此，这一过程对替换直径问题仍然有用，因为我们可以界定每个站点受某次边删除影响的次数。

**表示沃罗诺伊图。**设 $P$ 是一个边长为实数的平面图。设 $S$ 为位于 $O$ 个面上的顶点（站点）集合，这些面称为孔洞。回顾一下，每个站点 $s \in S$ 都有一个对应的加性权重 $d(s)$。

考虑 $P$ 的沃罗诺伊图，其站点为 $S$，加性权重为 $d(s)$。设 $P^*$ 为 $P$ 的平面对偶。设 $\mathrm{VD}_0$ 为 $P^*$ 的子图，它由以下边 $(u, v)$ 的对偶组成： $u$ 和 $v$ 位于不同的 Voronoi 单元中。通过反复收缩 $\mathrm{VD}_0$ 中任意一个度为 2 的顶点所关联的边，消去所有度为 2 的顶点后，得到图 VD。VD 的顶点称为沃罗诺伊顶点，边称为沃罗诺伊边。注意，每条沃罗诺伊边都对应于某两个站点之间一条平分线的连续线段。注意，VD 可能不连通，即它是一个平面地图，并且该平面地图的面的边界可能不连通。VD 的每个面都对应某个站点 $s \in S$ 的单元 Vor($s$)。因此，VD 中至多有 $|S|$ 个面。文献 [45] 证明了 VD 的边、顶点和面的总数为 $O(|S|)$。下文中，若称我们计算一个沃罗诺伊图 VD，意指使用定理 6 中的算法，该算法会计算上述平面地图 VD 的一种表示。VD 的每条沃罗诺伊边都对应一段平分线。

### 5.1 在加性权重增加时维护 Voronoi 图。

考虑将一组 $B \subseteq S$ 站点的加性权重增加。这样的增加只会改变 $v$ 位于$B$中各站点的 Voronoi 单元内的顶点的最短路径（相对于加性权重）。对于这样的 $v$，其最短路径要么保持不变，但长度会增加与该站点加性权重的变化量；要么 $v$ 成为另一个站点的 Voronoi 单元的一部分。在后一种情况下，由于每条最短路径都完全包含于单个 Voronoi 单元中，平面性

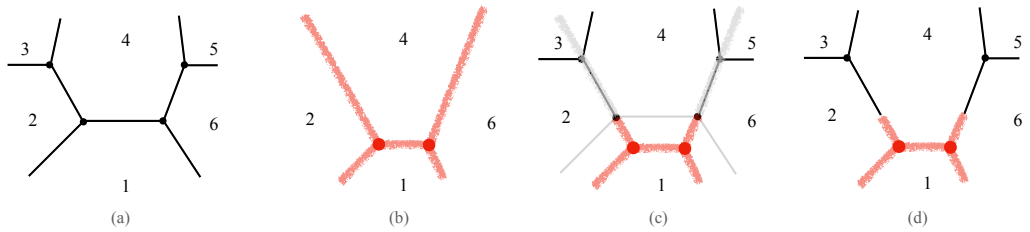


**图1** 示意了站点 1 的加性权重增加时，如何计算一个包含 6 个站点的区域的 Voronoi 图。(a) VD，即权重增加前所有 6 个站点的 Voronoi 图。(b) VD′，即权重增加后，仅由权重增加的站点 (1) 及其邻居 (2, 4, 6) 构成的 Voronoi 图。(c) 将 VD 和 VD′ 叠加，并以灰色显示从 VD 和 VD′ 中删除的边。注意，VD 中出现的邻居 (2,4,6) 的单元之间的所有平分线线段，在 VD′中也会出现。(d) 拼接的 Voronoi 图。

要求新站点必须是 $B$中某个站点的邻居。我们将 $B$ 中各站点的邻居集合 $N(B)$定义为这样一个站点集合：这些站点要么位于 $B$ 中，要么其 Voronoi 单元与 $B$中各站点的 Voronoi 单元相邻。注意，$|N(B)| = O(\sum_{b\in B}$ 单元度$(b))$。由上述讨论可知，因这种增加而导致 Voronoi 单元可能发生变化的唯一站点，就是 $N(B)$中的站点。

为了计算新的 Voronoi 图，我们首先计算仅包含以下站点的 $P$ $N(B)$的 Voronoi 图。根据定理 6，这需要 $O(\sum_{b\in B}$ 单元度$(b))$ 时间。令 VD′ 表示该 Voronoi 图，并令 VD 表示变化前 $P$ 的 Voronoi 图。需要强调的是，VD′ 的加性权重是增加后的权重，而 VD 的加性权重是增加前的权重。要获得变化后 $P$ 的 Voronoi 图，我们按如下方式将VD′ 和 VD 的部分“拼接”在一起。示意见图 1 。

回想一下，VD 是一个（可能不连通的）平面地图，其边对应于 VD 中站点对的平分线线段。这些线段的端点是 VD 的 Voronoi 顶点。首先，我们从 VD 中删除所有对应于至少涉及一个 $B$ 中站点的平分线的 Voronoi 边。对于每个与 $B$ 中某个站点的 Voronoi 单元相邻的 Voronoi 顶点 $v$，如果与 $v$ 相邻的所有 Voronoi 边都已删除，那么我们也删除 $v$。令 E 表示 VD 中满足以下条件的 Voronoi 边 $e$： $e$ 与某个 Voronoi 顶点 $v$ 相邻，$e$ 未被删除，但在 $v$ 周围边的循环顺序中，$e$ 的前一条或后一条 Voronoi 边已被删除。每条 Voronoi 边 $e \in$ E 都对应于一条线段 $\beta$，它是两个站点 $s_1$、$s_2 \in N(B) \setminus B$ 之间的平分线的一部分。由于这些站点的加性权重不变，线段 $\beta$ 必须由 VD′ 中的一条 Voronoi 边 $e'$ 表示。注意，$\beta$ 可能是对应于 $e'$ 的平分线线段 $\beta'$ 的子线段。还要注意，在计算 VD′ 时，很容易识别 $e'$ 与 $e$，且不会产生渐近时间开销。[3]对于每条 Voronoi 边 $e \in$E（属于 VD），我们将其对应的 Voronoi 边 $e'$（属于 VD′）分成两条边 $e'_1$、$e'_2$，方法是在 $v$ 处将 $\beta'$ 分成两个子线段。假设 $e'_2$ 是对应的平分线线段包含 $\beta$ 的那条边。注意，如果 $v$ 是 $e'$ 的一个端点（即，如果 $v$ 同时也是 VD′ 的 Voronoi 顶点），那么 $e'_1$ 是 $s_1$-$s_2$ 平分线上的一个平凡空线段。我们从 VD′ 中删除 $e'_2$，并将 VD 中的 Voronoi 边 $e$ 与 VD′ 中的 Voronoi 边 $e'_1$ 合并为一条边

[3] 例如，可以通过在构造算法中使用的平分线线段二叉搜索树表示中添加一个布尔标记来实现（参见 [45]），用于标记 $\mathcal{E}$ 中的边。然后，我们可以遍历 VD′ 中的所有 Voronoi 边，并列出每条边对应的已标记边 $e \in \mathcal{E}$，前提是与该 Voronoi 边对应的平分线线段中存在这样的边。

Voronoi 边，其对应线段是 $\beta$ 上的线段与 $e_1'$ 上的线段的拼接，其中前者属于 $e$ 。

对边 $e \in \mathcal{E}$ 执行此操作，实际上会将 VD′ 中的相关部分“拼接”到 VD中，替换我们之前删除的 VD 部分。 [45] 用于从预计算的垂直平分线构造沃罗诺伊图的算法会执行类似的拼接操作，而用于表示沃罗诺伊图和平分线的数据结构支持˜所有必要操作，且每次操作耗时 $O(1)$ 。因此，整个过程的时间复杂度与 $B$中各站点的沃罗诺伊单元的沃罗诺伊顶点数量成正比（忽略对数因子），即 $O(\sum_{b\in B}$单元度$(b))$。

## 5.2 替换直径

现在，我们将介绍如何使用用于维护加性权重递减时 Voronoi 图的新算法，从而获得更快的替换直径算法。该算法首先计算图 $G$ 的完整递归分解树 T。对于 T 中的每个节点（区域）（对应于 $G$ 的一个子图），我们计算其所有平分线。根据定理 6，这一步在整个 T 上耗时 $\tilde{O}(n^2)$。然后，对于每个顶点 $s \in V$，我们在 $G$ 中计算 $s$ 的广度优先搜索树 $T_s$，并计算 ˜ 的容错单源距离预言机。Baswana 等人提出的 [12] 算法用于处理 $G$ 中的 $s$ 。该预言机可通过 $G$ 在 $O(n)$ 时间内构建，并且对于任意 $s$、$t \in V$ 和 $e \in E$，可在 $O(1)$ 时间内报告图 $G^e = (V, E\setminus\{e\})$ 中从 $s$ 到 $t$ 的距离。总体而言，这一步也耗时 $\tilde{O}(n^2)$。对于 T 中的每个区域 $P \in$ ，以及其中的每个边界顶点 $b \in \partial P$，我们通过标记 $P$ 的所有顶点及其所有最近公共祖先，并收缩端点未标记的边，从 $T_b$ 中构造诱导树 $T_b^P$。所得 $T_b^P$ 的大小为 $O(|P|)$。对于过程中收缩的 $G$ 中的每条边 $e$，我们存储 $e$ 被收缩到的 $T_b^P$ 中的边。由于 T 中所有区域的边界节点总数 ˜ 与区域大小总和为 $O(n)$，构造所有这些诱导树的总耗时为 $\tilde{O}(n^2)$。对于 T 中的每个区域 $P \in$，令 $P'$ 为 T 中 $P$ 的兄弟节点。令 $b_1$、$b_2$、..... 为 $\partial P'$ 的顶点，顺序任意。对于每个顶点 $s \in P$，我们计算 $s$ 关于 $P'$、站点为 $\{b_i\}$ 且加性权重为 $d(s, b_i)$ 的加权 Voronoi 图。我们还为 $s$ 存储一个关于 $b_1$、$b_2$、.....的二叉搜索树（BST），其中树中的节点 $i$ 存储 ˜ 从 $s$ 到 Vor$(b_i)$ 中最远顶点的距离。对于所有 $P \in$ T 和所有 $s \in P$，这一步总共耗时 $O(n\sqrt{n})$。对于每个包含顶点 $v_1$、$v_2$、.....的区域 $P$（顺序任意），我们存储一个关于 $\{v_i\}$ 的 BST，其中节点 $i$ 存储 $P'$ 中距 $v_i$ 最远的顶点。对于每个 $v_i$，通过查询 $v_i$ 的 BST 中存储的最大距离，即可在 $O(1)$ 时间内找到该顶点 ˜。

对于每条边 $e \in E$，我们需要计算图$G^e = (V,\ E\setminus\{e\})$中的最远点对。对于一条边 $e \in E$和两个顶点 $u,\ v \in G$，如果 $e$ 位于 $T_u$中从根到$v$ 的路径上，我们就称点对 $u,\ v$ 在 $G^e$ 中受到影响。核心思路是利用这样一个事实：特定点对 $u,\ v$ 至多在 $D$ 个图 $G^e$ 中受到影响（而不是 $n$个），因为 $G$ 中从$u$到$v$ 的最短路径至多包含 $D$ 条边。

对于每个受影响的顶点对 $(u, v)$，都存在一对兄弟片段 $(P, P')$，使得 $u \in P$and$v \in P'$。我们的策略是遍历 T 中的兄弟片段对 $(P, P')$，并按如下方式对每一对 $(P, P')$ 的所有受影响顶点对一并处理。不失一般性，假设 $e \notin P'$。对于每个 $b \in \partial P'$，我们在 $T_b^P$ 中枚举 $T_b^P$ 的边上所有后代顶点；该边被收缩到 $e$ 中（若 $e \notin T_b$，则该集合可能为空）。这样，我们就能找出所有形如 $(u, b)$ 的受影响顶点对，其中 $u \in P$and $b \in \partial P'$。对于每个这样的受影响顶点对，我们查询 Baswana 等人的预言机，获取 $u$ 到 $b$ 在 $G^e$ 中的距离。对于每个 $u \in P$，令 $B$ 为满足 $(u, b)$ 是受影响顶点对的边界顶点集合。对于满足 $|B| \geq 1$ 的每个顶点 $u \in P$，我们使用递减式 Voronoi 图过程更新 $u$ 相对于 $P'$ 的 Voronoi 图；该过程详见第 5.1 小节。此过程会更新 VD（以及距每个站点最远的顶点）

相对于新的加性权重，所需时间为 $\sum_{b\in B}$单元度$(b)$ 其中，单元度是与原始 VD 中 Vor() 单元相邻的 Voronoi 单元数量[4] （即删除 $e$ 之前）。利用更新后的 VD，我们将 BST 中对应于每个 $b \in B$ 的节点更新为 Vor($b$) 中新的最远顶点。令 $d$ 为 $u$ 的整个 BST 中存储的最大距离。我们将 $P$ 的 BST 中对应于 $u$ 的节点更新为值 $d$。以这种方式处理完所有满足 $|B| \geq 1$ 的 $u \in P$ 后， $P$ 的整个 BST 中存储的最大值，就是 $G^e$ 中满足 $u \in P$ and $v \in P'$ 的任意一对顶点 $(u, v)$ 之间的最大距离。对 T 中所有兄弟片段对 $(P, P\prime) \in$ 取最大值，即得到 $G^e$ 的直径。

计算兄弟节点 $(P, P')$ 的最远点对的总运行时间分析如下。瓶颈在于更新 VD 所需的时间。每当一对 $u, b$ （其中 $u \in P$ 且 $b \in \partial P'$）受到影响时，我们会花费 $O$(单元度($b$)) 的时间来更新 $u$ 的 VD。由于每对最多会受到删除 $D$ 条边的影响，因此，为 $(P, P')$ 更新 VD 所耗费的总时间上界为 $\sum_{u\in P, b\in\partial P'} D \cdot$ 单元度$(b) = |P|D\sum_b$单元度$(b)$，也就是 $\tilde{} O(|P|D \cdot |\partial P'|)$，因为 VD 中各单元的单元度之和与该 VD 的站点数同阶。对所有兄弟片段对求和可得，总时间为 $\sum_{(P,P')\in}$T $\tilde{O}(|P|D \cdot |\partial P'|) = \tilde{O}(n^{1.5}D)$。因此，计入预处理后，整个替代直径算法的总时间为 $\tilde{O}(n^2 + n^{1.5}D)$。

我们注意到，当 $D \geq n^{5/6}$ 时，针对每次边失效，直接使用第 $n^{3+o(1)}/D^2$ 节中的静态 3.1 时间算法更好。因此，替换直径问题可在 $\min(n^{3+o(1)}/D^2, \tilde{O}(n^2 + n^{1.5}D)) = n^{7/3+o(1)}$ 时间内求解。

## 6 增量直径

在本节中，我们证明定理3。具体而言，我们给出一种通用归约，说明如何利用以下两个组成部分，高效求解增量图中的直径问题：（1）适用于增量图的距离预言机；（2）当直径较大时速度相对较快的静态图直径算法。结合 Das 等人的增量距离预言机 [37] 和第 3.1 节中的静态算法，我们得到一个总运行时间为 $n^{7/3+o(1)}$ 的算法，优于 $\tilde{O}(n^{8/3})$这一朴素界。本节的新算法进一步逼近了带权图在第 $n^{2-o(1)}$ 4 节中给出的下界（无权图的最佳下界为 $n^{1.5-o(1)}$）。

本节其余部分将用于证明这一 theorem。我们首先介绍一般的归约（不假设图具有平面性，也不假设边无权重），然后说明如何将其与现有的平面图算法结合起来，从而得到该定理。

**从直径到 $s,t$-最短路径的归约。** 在增量图中，直径会随着时间推移而减小，从某个 $D \leq n$ 开始（否则图不连通，且可以轻松高效地检查），最终到达某个 $D \geq 1$。归约的思路很简单：我们希望只在直径减小时重新计算直径，而不是在每次 $n$ 更新后都重新计算。虽然直径可能会减小 $\Omega(n)$ 次，从$n$ 到 1，但关键在于，直径较大时重新计算的效率很高（得益于第 3.1 节的 $n^{3+o(1)}/D^2$ 算法），而当直径小于 $D$ 时，重新计算只会发生 $O(D)$ 次。

我们的增量算法如下：

- **步骤 1 - 采样一个新的直径对：** 令 $P = \{(s, t) \mid d(s, t) = \Delta(G)\}$ 为实现当前直径 $\Delta(G)$ 的点对集合。从中均匀采样一对 $(s', t')$，

[4] 如果存在一条 $G$ 的边 $e$，且其两个端点分别位于两个单元中，则这两个单元相邻。

随机采样（或者从某种分布中采样，使得每一对被采样的概率至多为 $O(1/|P|)$）。

- **步骤 2——监测采样直径对的距离：**使用增量式距离预言机，在整个边插入序列中监测 $s'$ 与 $t'$ 之间的距离。只要 $d(s', t')$ 不减少，就什么也不做（查询预言机除外）；在这种情况下，它仍然是图的正确直径，并且每当有查询时都可以输出。如果一条新边导致 $d(s', t')$ 减少，则返回步骤 1。

这两个步骤分别涉及我们归约中的两个要素。步骤 2 使用增量式距离预言机，而步骤 1 使用一种静态直径算法，该算法还可以对一个直径对进行采样。在本节末尾，我们将给出一个一般性归约：将后一个近似采样问题归约为寻找图中每个节点的最大距离的问题（即计算所有离心率）。或者，可以注意到，我们将在步骤 1 中采用的直径算法（以及许多其他自然的直径算法）都可以修改为以均匀随机方式对一个直径对进行采样。

**运行时间。** 我们先界定进入步骤 1 的次数，因为这一步需要计算静态直径，代价最高。步骤 2 实际上开销很小，因为我们只需对增量距离预言机执行一次更新和一次查询。

▷ 断言 14. 对于任意不会使图的直径减小的（非自适应）边插入序列，我们的算法期望采样直径对（即进入步骤 1）的次数为 $O(\log n)$。

**证明。** 我们先分析理想情况：在步骤 1 中能够进行真正的均匀采样；然后指出，当我们进行近似均匀采样时，基本上同样的分析仍然成立。

每条新边 $e$ 都会缩短某个点对集合 $X_e \subseteq P$ 中的距离。由于对手完全不知道特殊点对 $(s', t')$，而它正是用来选择边插入序列的，因此 $e$ 导致算法进入步骤 1 的概率恰好为 $|X_e|/|P|$；在这种情况下，新的“直径对”集合变为 $P \setminus X_e$。因此，采样次数的期望上界为：$f(|P|) \leq \max_{0 \leq x \leq |P|} x/|P| + f(|P| - x) = O(\log |P|)$。

如果第 1 步中的采样仅近似均匀，但仍满足以至多 $O(1/|P|)$ 的概率选取一个直径对，那么上述分析仍然成立，只需额外增加一个 $O(1)$ 因子。◀

令 $T^{Diam}(n, D)$ 表示这样一种静态直径算法的运行时间：当图的直径为 $D$ 时，按照第 1 步的方式采样一个直径对。在全部 $O(n)$ 次边插入中，第 1 步的总期望运行时间因此至多为 $\sum_{D=1}^{n} \log n \cdot T^{Diam}(n, D)$。

为了得到我们所声称的上界 $n^{7/3+o(1)}$，我们将在该归约中使用两种直径算法：Section $T^{Diam}(n, D) = n^{3+o(1)}/D^2$ 中的算法 3.1（适用于较大的$D$），以及 $T^{Diam}(n, D) = \tilde{O}(n^{5/3})$ 算法 [45]（适用于较小的 $D$）。（根据 Section 6.1中的归约，这些算法还可以按照第 1 步的要求采样一个近似均匀的直径对。）总期望时间为：

$$\sum_{D=1}^{n} \log n \cdot T^{Diam}(n, D) = \tilde{O}\left(\sum_{D=1}^{n^{2/3}} n^{5/3} + \sum_{D=n^{2/3}}^{n} n^{3+o(1)}/D^2\right) = n^{7/3+o(1)},$$

因为 $\sum_{D=n^{2/3}}^{n} n^{3+o(1)}/D^2 \leq \sum_{i=}^{\log_2 n} \log_2 n^{2/3}\, 2^{i+1} \cdot n^{3+o(1)}/(2^i)^2 \leq \frac{n^{3+o(1)}}{n^{2/3}} \cdot 2\log n$。步骤 2 的额外耗时至多为 $n \cdot \sqrt{n}$，使用 Das 等人的增量距离预言机[37]，其每次更新和查询的耗时为 $O(\sqrt{n})$。

## 6.1 采样一个直径对

本节将介绍增量直径算法的最后一个区域。具体来说，我们将说明如何调整前述静态直径算法，使其能够近似均匀地采样一个直径对。

一种行不通的初步尝试是，为每条边 $e$ 的权重添加一个随机“扰动” $p_e \in (0, \varepsilon)$，其中 $\varepsilon < 1/D$，然后论证：在新图中实现直径的那一对（很可能是唯一的一对）是 $P = \{(s,t) \mid d(s,t) = \Delta(G)\}$ 中均匀随机选出的一对。注意，扰动会使所有点对之间的距离增加 $< 1$，因此非直径对不可能变成直径对。然而，一个问题是，彼此之间存在许多条长度为 $\Delta(G)$ 的路径的点对，比存在此类路径较少的点对更容易被选中。解决这一问题的第二种尝试是对节点添加扰动（例如，为每个节点添加一条私有叶节点边，并在新边上赋予随机权重）。这一思路更接近实际的解决方案，但仍存在相关性问题：参与许多点对的节点，可能比只参与少数点对的节点更少被采样。因此，分配权重时必须考虑这种差异。

要让这些想法行得通，稍微有些复杂。幸运的是，我们可以将当前问题归约到由 Beame 等人提出的二分独立集查询模型 [13]，然后以黑盒方式利用该模型的已有结果 [7, 16, 38]。

▶ **定理 15**。存在一种算法，能够抽样得到一对 $P = \{(s,t) \mid d(s,t) = \Delta(G)\}$，且每一对被抽中的概率至多为 $O(1/|P|)$；对于直径为 $D$ 的无权平面图，该算法的运行时间为 $\tilde{O}(n^{5/3} + n^{3+o(1)}/D^2)$。

证明该定理所需的主引理如下。

▶ **引理 16**。通过调用一个返回所有离心率的算法 $\log^{O(1)} n$ 次，我们可以抽样得到一对 $P = \{(s,t) \mid d(s,t) = \Delta(G)\}$，且每一对被抽中的概率至多为 $O(1/|P|)$

**证明**。考虑一个隐式图 $H$，其中两个节点之间有一条边 $s$，$t$ 当且仅当它们在 $G$ 中构成直径对（即，$(s, t) \in P$）。我们的目标是从 $H$ 中近似均匀地采样一条边。这可以通过向一个预言机发起多对数次查询来实现。该预言机给定两个子集 $L$，$R \subseteq V(H)$，判断 $L \times R \cap E(H)$ 中是否存在边。文献中称之为二部独立集预言机，名称沿袭自 Beame 等人 [13]。因此，我们只需证明，这类查询可以在调用一个计算图中所有节点离心率的算法所需的时间内得到支持。

首先，我们预先计算 $G$ 的直径 $\Delta(G)$。然后，给定查询 $L$，$R \subseteq V$ 我们按如下方式从 $G$ 构造图 $G'$。对于每个节点 $v \in R$，我们添加一个新的“叶”节点 $l_v$，并用一条权重为 1 的边将其连接到 $v$。接下来，我们计算 $G'$ 中所有节点的离心率。最后，当且仅当存在一个 $u \in L$，使得 $u$ 在 $G'$ 中的离心率为 $\Delta(G) + 1$ 时，查询答案为是；这可以在 $O(n)$ 时间内检查。

答案的正确性可由以下观察得出：当且仅当存在一个节点 $v$ 属于 $G$，满足（1）$d_G(u, v) = \Delta(G)$，且（2）一个新的叶节点 $l_v$ 被添加到 $v$ 时，$u$ 在 $G'$ 中的任意节点的离心率均为 $\Delta(G) + 1$。这意味着（1）$(u, v) \in P$ 是 $G$ 中的一个直径对，也就是说，$(u, v) \in E(H)$；并且（2）$v \in L$。由于我们只检查 $u \in R$，因此只要 $L \times R \cap E(H)$ 非空，我们的答案就是正确的。◀

为完成定理 15 的证明，我们只需指出，相关的两种直径算法都已经计算出了所有节点的离心率。

## 16 沃罗诺伊图还能为平面图的直径做些什么?

### 参考文献


1 A. Abboud、K. Censor-Hillel、S. Khoury 和 A. Paz。更小的割，更高的下界。《ACM 算法汇刊》，17(4):30:1–30:40，2021。 2 A. Abboud、V. Cohen-Addad 和 P. N. Klein。平面图问题在 p 中的新困难性结果及一种最稀疏割算法。载于第 52 届 STOC，页码 996–1009，2020。 3 A. Abboud 和 S. Dahlgaard。流行猜想作为动态平面图算法的障碍。载于第 57 届 FOCS，页码 477–486，2016。 4 A. Abboud 和 V. V. Williams。流行猜想蕴含动态问题的强下界。载于 FOCS，页码 434–443，2014。 5 A. Abboud、V. V. Williams 和 J. R. Wang。稀疏图中半径和直径的近似算法与固定参数次二次算法。载于 SODA，页码 377–391，2016。 6 I. Abraham、S. Chechik 和 C. Gavoille。通过禁集距离标签实现平面图的全动态近似距离预言机。载于第 44 届 ACM 理论计算研讨会，页码 1199–1218，2012。 7 R. Addanki、A. McGregor 和 C. Musco。通过二分图独立集查询进行非自适应边计数与抽样。arXiv 预印本 arXiv:2207.02817，2022。 8 A. Aggarwal、M. M. Klawe、S. Moran、P. Shor 和 R. Wilber。矩阵搜索算法的几何应用。Algorithmica，2(1):195–208，1987。 9 B. Ancona、M. Henzinger、L. Roditty、V. V. Williams 和 N. Wein。动态图中直径的算法与困难性。载于第 46 届 ICALP，第 132 卷，页码 13:1–13:14，2019。 10 S. R. Arikati、D. Z. Chen、L. P. Chew、G. Das、M. H. M. Smid 和 C. D. Zaroliagis。平面图稀疏化器以及平面障碍物间的近似最短路径查询。载于第 4 届 ESA，第 1136 卷，页码 514–528，1996。 11 A. Backurs、L. Roditty、G. Segal、V. V. Williams 和 N. Wein。迈向图直径与离心率的紧近似界。《SIAM 计算杂志》，50(4):1155–1199，2021。 12 S. Baswana、U. Lath 和 A. S. Mehta。避免失效节点或链路的平面有向图单源距离预言机。载于第 23 届 SODA，页码 223–232，2012。 13 P. Beame、S. Har-Peled、S. N. Ramamoorthy、C. Rashtchian 和 M. Sinha。利用独立集预言机进行边估计。《ACM 算法汇刊》（TALG），16(4):1–27，2020。 14 B. Ben-Moshe、B. K. Bhattacharya、Q. Shi 和 A. Tamir。仙人掌网络中心问题的高效算法。《理论计算机科学》，378(3):237–252，2007。 15 P. Berman 和 S. Kasiviswanathan。更快的图距离近似。载于第 10 届国际算法与数据结构研讨会（WADS），页码 541–552，2007。 16 A. Bhattachrya、A. Bishnu、A. Ghosh 和 G. Mishra。使用彩色决策预言机的更快计数与抽样算法。莱布尼茨国际信息学会议录（LIPIcs），219，2022。 17 É. Bonnet。4 对 7：稀疏无向无权图直径在时间 n4/3 下是 SETH 困难的。《ACM 算法汇刊》，18(2):11:1–11:14，2022。 18 G. Borradaile、S. Pettie 和 C. Wulff-Nilsen。平面图的连通性预言机。载于第 13 届 SWAT，第 7357 卷，页码 316–327，2012。 19 G. Borradaile 、 P. Sankowski 和 C. Wulff-Nilsen。具有近线性预处理时间的平面图最小 st- 割预言机。载于第 51 届 FOCS，页码 601–610，2010。 20 C. Busch、R. LaFortune 和 S. Tirthapura。平面图以及排除固定禁用 minor 的图的稀疏覆盖。Algorithmica，69(3):658–684，2014。 21 S. Cabello。平面图中的许多距离。Algorithmica，62(1-2):361–381，2012。 22 S. Cabello。平面图直径及两两距离之和的次二次算法。载于第 28 届 SODA，页码 2143–2152，2017。 23 T. M. Chan。 $o(mn)$ 时间内无权无向图的全点对最短路径。《 ACM 算法汇刊》，8(4):34:1–34:17，2012。 24 T. M. Chan 和 D. Skrepetos。平面图中更快的近似直径与距离预言机。Algorithmica，81(8):3075–3098，2019。

25 H. Chang、R. Krauthgamer 和 Z. Tan。用于平面图的近线性 $\epsilon$ 模拟器。载于第54届 STOC，1311–1324页，2022年。26 P. Charalampopoulos、P. Gawrychowski、S. Mozes 和 O. Weimann。用于平面图的近乎最优距离预言机。载于第51届 STOC，138–151页，2019年。27 P. Charalampopoulos 和 A. Karczmarz。动态平面图中的单源最短路径与强连通性。《计算机与系统科学杂志》，124：97–111，2022年。28 P. Charalampopoulos、S. Mozes 和 B. Tebeka。具有故障顶点的平面图的精确距离预言机。载于第30届 SODA，2110–2123页，2019年。29 S. Chechik、D. H. Larkin、L. Roditty、G. Schoenebeck、R. E. Tarjan 和 V. V. Williams。改进图直径的近似算法。载于第25届 SODA，1041–1052页，2014年。30 D. Z. Chen 和 J. Xu。平面图中的最短路径查询。载于第32届 STOC，469–478页，2000年。31 V. Chepoi 和 F. F. Dragan。寻找弦图中心顶点的线性时间算法。载于 ESA，159–170页，1994年。32 V. Chepoi、F. F. Dragan、B. Estellon、M. Habib 和 Y. Vaxès。δ-双曲测地空间与图的直径、中心及近似树。载于计算几何研讨会，59–68页，2008年。33 V. Chepoi、F. F. Dragan 和 Y. Vaxès。平面三角剖分与四边形剖分中的中心和直径问题。载于 SODA，346–355页，2002年。34 K. Clarkson 和 P. Shor。随机抽样在计算几何中的应用，第二部分。《离散与计算几何》，4(5)：387–421，1989年。35 V. Cohen-Addad、S. Dahlgaard 和 C. Wulff-Nilsen。用于平面图的快速紧凑的精确距离预言机。载于第58届 FOCS，962–973页，2017年。36 M. Dalirrooyfard、R. Li 和 V. V. Williams。近似直径问题的困难性：现在针对无向图。《第62届 IEEE计算机科学基础年会（FOCS 2021）》，美国科罗拉多州丹佛，2022年2月7日至10日，1021–1032页。IEEE，2021年。37 D. Das、M. P. Gutenberg 和 C. Wulff-Nilsen。平面有向图动态全点对最短路径问题的近乎最优离线算法。载于第33届 SODA，3482–3495页，2022年。38 H. Dell、J. Lapinskas 和 K. Meeks。利用彩色决策预言机对小型见证进行近似计数与抽样。《SIAM 计算杂志》，51(4)：849–899，2022年。39 H. Djidjev。平面有向图中最短路径查询的高效算法。载于第22届图论研讨会，第1197卷，151–165页，1996年。40 F. F. Dragan 和 F. Nicolai。距离遗传图的 LexBFS排序及其在直径点对问题中的应用。《离散应用数学》，98(3)：191–207，2000年。41 D. Eppstein。平面图中的子图同构及相关问题。载于第6届 SODA，632–640页，1995年。42 J. Fakcharoenphol 和 S. Rao。平面图、负权边、最短路径与近线性时间。《计算机与系统科学杂志》，72(5)：868–889，2006年。43 A. M. Farley 和 A. Proskurowski。外平面图中心和直径的计算。《离散应用数学》，2(3)：185–191，1980年。44 G. N. Frederickson。平面图中最短路径的快速算法及其应用。《SIAM 计算杂志》，16(6)：1004–1022，1987年。45 P. Gawrychowski、H. Kaplan、S. Mozes、M. Sharir 和 O. Weimann。平面图上的沃罗诺伊图，以及在确定性 $\tilde{O}(n^{5/3})$ 时间内计算直径。《SIAM 计算杂志》，50(2)：509–554，2021年。46 P. Gawrychowski、S. Mozes 和 O. Weimann。平面负权 k 环。载于第32届 SODA，2717–2724页，2021年。47 P. Gawrychowski、S. Mozes、O. Weimann 和 C. Wulff-Nilsen。平面图精确距离预言机的更优权衡。载于 SODA，2018年。48 Q. Gu 和 G. Xu。用于平面图的常数查询时间 $(1+\varepsilon)$ 近似距离预言机。《理论计算机科学》，761：78–88，2019年。

49 J. Hershberger 和 S. Suri。基于最短路径度量的矩阵搜索。《SIAM 计算杂志》，26(6):1612–1634，1997年。50 T. Husfeldt。以树宽和直径为参数计算图距离。载于 IPEC，页码 16:1–16:11，2016年。51 G. F. Italiano、A. Karczmarz、J. Lacki 和 P. Sankowski。平面有向图中的减量单源可达性。载于第49届 STOC，页码 1108–1121，2017年。52 G. F. Italiano、Y. Nussbaum、P. Sankowski 和 C. Wulff-Nilsen。无向平面图中最小割和最大流的改进算法。载于第43届 STOC，页码 313–322，2011年。53 A. Karczmarz。平面有向图及更一般图中的减量传递闭包与最短路径。载于第29届 SODA，页码 73–92，2018年。54 K. Kawarabayashi、P. N. Klein 和 C. Sommer。适用于平面图、有界亏格图及无特定子式图的线性空间近似距离预言机。载于第38届 ICALP，第6755卷，页码 135–146，2011年。55 K. Kawarabayashi、C. Sommer 和 M. Thorup。用于无向平面图中近似距离的更紧凑预言机。载于第24届 SODA，页码 550–563，2013年。56 P. N. Klein。对无向平面网络进行预处理，以支持快速近似距离查询。载于第13届 SODA，页码 820–827，2002年。57 P. N. Klein。平面图中的多源最短路径。载于 SODA，页码 146–155，2005年。58 P. N. Klein、S. Mozes 和 C. Sommer。平面图的线性时间结构化递归分隔器分解。载于第45届 STOC，页码 505–514，2013年。59 J. Lacki 和 P. Sankowski。平面图中的最优减量连通性。Theory Comput. Syst.，61(4):1037–1053，2017年。60 H. Le 和 C. Wulff-Nilsen。平面图的最优近似距离预言机。载于第62届 IEEE计算机科学基础年会（FOCS 2021），美国科罗拉多州丹佛，2022年2月7日至10日，页码 363–374。IEEE，2021 年。61 R. J. Lipton 和 R. E. Tarjan。平面图的分隔器定理。《 SIAM J. Appl. Math》，36(2):177–189，1979年。62 J. Łącki 和 P. Sankowski。在 $O(n \log \log n)$ 时间内求解平面图中的最小割和最短环。载于第19届 ESA，页码 155–166，2011年。63 Y. Long 和 S. Pettie。具有更优时间—空间权衡的平面距离预言机。载于第32届 ACM-SIAM离散算法研讨会论文集（SODA），页码 2517–2536，2021年。64 S. Mozes 和 C. Sommer。平面图的精确距离预言机。载于第23届 SODA，页码 209–222，2012年。65 Y. Nussbaum。平面图中改进的距离查询。载于第12届 WADS，页码 642–653，2011年。66 S. Olariu。一种用于计算区间图中心的简单线性时间算法。International Journal of Computer Mathematics，34:121–128，1990年。67 L. Roditty 和 V. V. Williams。稀疏图直径和半径的快速近似算法。载于第45届 STOC，页码 515–524，2013年。68 S. Subramanian。用于平面有向图可达性的全动态数据结构。载于 T. Lengauer（编），第1届 ESA，第726卷，页码 372–383，1993年。69 M. Thorup。用于平面有向图可达性和近似距离的紧凑预言机。J. ACM，51(6):993–1024，2004年。70 O. Weimann 和 R. Yuster。以近线性时间近似计算平面图的直径。ACM 算法汇刊，12(1):12:1–12:13，2016年。71 R. Williams。一种用于最优约束满足的新算法及其意义。载于第31届 ICALP，页码 1227–1237，2004年。72 C. Wulff-Nilsen。以亚二次时间计算平面图的 Wiener 指数和直径。技术报告，08-16，哥本哈根大学计算机科学系，2008年。可通过 http://www.diku.dk/OLD/publikationer/tekniske.rapporter/rapporter/08-16.pdf获取。初步版本发表于 EurCG 2009。

73 C. Wulff-Nilsen。 平面图和度量空间中的图的算法。博士论文，哥本哈根大学，2010年。74 C. Wulff-Nilsen。针对平面图的近似距离预言机，具有改进的查询时间—空间权衡。载于 第*27*届 *SODA*，第351–362页，2016年。