Covering Planar Metrics (and Beyond): O(1) Trees Suffice

Hsien-Chih Chang* Jonathan Conroy† Hung Le‡ Lazar Milenkovic§
Shay Solomon¶ Cuong Than‖

Abstract

While research on the geometry of planar graphs has been active in the past decades, many properties of planar metrics remain mysterious. This paper studies a fundamental aspect of the planar graph geometry: covering planar metrics by a small collection of simpler metrics. Specifically, a *tree cover* of a metric space (X, δ) is a collection of trees, so that every pair of points u and v in X has a low-distortion path in at least one of the trees.

The celebrated "Dumbbell Theorem" [ADM+95] states that any low-dimensional Euclidean space admits a tree cover with $O(1)$ trees and distortion $1+\varepsilon$, for any fixed $\varepsilon \in (0,1)$. This result has found numerous algorithmic applications, and has been generalized to the wider family of doubling metrics [BFN19]. Does the same result hold for planar metrics? A positive answer would add another evidence to the well-observed connection between Euclidean/doubling metrics and planar metrics.

In this work, we answer this fundamental question affirmatively. Specifically, we show that for any given fixed $\varepsilon \in (0,1)$, any planar metric can be covered by $O(1)$ trees with distortion $1+\varepsilon$. Our result for planar metrics follows from a rather general framework: First we reduce the problem to constructing tree covers with *additive distortion*. Then we introduce the notion of *shortcut partition*, and draw connection between shortcut partition and additive tree cover. Finally we prove the existence of shortcut partition for any planar metric, using new insights regarding the grid-like structure of planar graphs. To demonstrate the power of our framework:

- We establish additional tree cover results beyond planar metrics; in particular, we present an $O(1)$-size tree cover with distortion $1+\varepsilon$ for *bounded treewidth* metrics;
- We obtain several algorithmic applications in planar graphs from our tree cover.

The grid-like structure is a technical contribution that we believe is of independent interest. We showcase its applicability beyond tree cover by constructing a simpler and better embedding of planar graphs into $O(1)$-treewidth graphs with small additive distortion, resolving an open problem in this line of research.

*Department of Computer Science, Dartmouth College. Email: `hsien-chih.chang@dartmouth.edu`.
†Department of Computer Science, Dartmouth College. Email: `jonathan.conroy.gr@dartmouth.edu`
‡Manning CICS, UMass Amherst. Email: `hungle@cs.umass.edu`
§Tel Aviv University
¶Tel Aviv University
‖Manning CICS, UMass Amherst. Email: `cthan@cs.umass.edu`

1 Introduction

Research on the structure of planar graphs has provided many algorithmic tools such as separators [LT79] and cycle separators [Mil86], r-divisions [Fed87], sphere-cut decomposition [DPBF09], abstract Voronoi diagram [Cab18], and strong product theorem [DJM+20], to name a few. These structural results are rooted in the simple topology of planar graphs. Another line of important and complementary research from an algorithmic point of view is to understand the geometry of planar graphs and, more precisely, the metric spaces induced by shortest-path distances in planar graphs. Such metric spaces are called *planar metrics*. Understanding the properties of metric spaces in general—the main subject of the metric embedding literature—has led to surprising algorithmic consequences [LLR95, Bar96, FRT04]. One may naturally expect that the simple topology of planar graphs would help in understanding planar metrics, from which we could significantly extend our algorithmic toolkit. Indeed, there have been a few such successful attempts, such as padded decompositions with $O(1)$ padding parameter [KPR93], or embedding into ℓ_2 with $O(\sqrt{\log n})$ distortion [Rao99] which has a matching lower bound [NR02], and several other results [GNRS04, AFGN22]. However, many basic questions remain open. A notable example is the ℓ_1 embedding conjecture: can we embed a planar metric into ℓ_1 with $O(1)$ distortion [GNRS04]? More generally, what is the distortion for embedding into ℓ_p for any $p \geq 1, p \neq 2$? (See [GNRS04, BC05, AFGN22, Fil20a], and references therein for a host of other related questions.) These suggest that understanding the geometry of planar graphs is very challenging. On the other hand, a deep understanding of the geometry of planar graphs often leads to remarkable algorithmic results. For example, a constant approximation for the sparsest cut problem in planar graphs would follow from a positive resolution of the ℓ_1 embedding conjecture; the best-known algorithm achieving such constant factor approximation without relying on the unproven conjecture runs in quasi-polynomial time [CGKL21].

Embedding is one important aspect of the geometry of planar metrics, but it might not be the only telling one. As planar graphs are defined by drawing in the Euclidean plane, can we relate planar metrics to the 2D (or more generally, low-dimensional) Euclidean metric? Embedding is not illuminating in this respect: there exists an (unweighted) planar metric of n points that requires distortion $\Omega(n^{2/3})$ for any embedding into $\mathbb{R}^2$ [BDHM07]. More generally, as simple as the unweighted star graph of n vertices, any embedding into an $O(1)$-dimensional Euclidean space requires distortion $n^{\Omega(1)}$ by a volume argument. This motivates us to look into the *covering* aspect, namely, covering metrics by simpler metrics. Here *tree metrics* are of special interest due to their simplicity and algorithmic applicability; in addition, Euclidean/doubling metrics were known to have a covering of constant size, as defined next.

We say that an edge-weighted tree T with shortest-path distance d_T is a *dominating tree* of a metric space (X, δ_X) if X is a vertex subset of T, and for every two points x and y in X, one has $\delta_X(x, y) \leq d_T(x, y)$. For any given parameter $\alpha \geq 1$, we say that a collection of trees, denoted by $\mathcal{F}$, is a *α-tree cover* of (X, δ_X) if every tree in $\mathcal{F}$ is dominating, and for every two points $x \neq y$ in X, there is a tree T in $\mathcal{F}$ such that $d_T(x, y) \leq \alpha \cdot \delta_X(x, y)$. The *size* of the tree cover is the number of trees in $\mathcal{F}$. Parameter α is called the *distortion* of the tree cover. When $\alpha = 1$, we say that the tree cover is an *exact tree cover*. The notion of tree covers, and its variants, were studied in the past by many researchers [AP92, AKP94, ADM+95, GKR01, BFN19, FL22a, KLMS22].

More than two decades ago, Arya et al. [ADM+95] showed that any set of n points in $\mathbb{R}^2$ admits a $(1 + \varepsilon)$-tree cover with a constant number of trees for any fixed $\varepsilon \in (0, 1)$.[1] This result indeed holds for any Euclidean space of constant dimension and can be extended to any metric of constant doubling dimension [BFN19]. These results naturally motivate the following question:

[1]In this work, we consider ε to be a fixed constant. We only spell out the precise dependency on ε in theorem statements.

Question 1.1. *Can planar metrics be covered by a constant number of trees with a multiplicative distortion* $(1+\varepsilon)$ *for any fixed* $\varepsilon \in (0,1)$*?*

A positive answer to Question 1.1 would imply that planar metrics are similar to Euclidean/doubling metrics in the tree-covering sense. Furthermore, the tree cover serves as a bridge to transfer algorithmic results from Euclidean/doubling metrics to planar graphs/metrics; for example, routing, spanners, emulators, distance oracles, and possibly more. In their pioneering work [BFN19], Bartal, Fandina, and Neiman constructed a $(1+\varepsilon)$-tree cover of size $O(\log^2 n)$ for planar metrics; they left Question 1.1 as an open problem.

1.1 Our contributions

Tree cover for planar graphs and related results. Our main result is a positive answer to Question 1.1.

Theorem 1.2. *Let* G *be any edge-weighted undirected planar graph with* n *vertices. For any parameter* $\varepsilon \in (0,1)$*, there is a* $(1+\varepsilon)$*-tree cover* $\mathcal{F}$ *for the shortest path metric of* G *using* $O(\varepsilon^{-3} \cdot \log(1/\varepsilon))$ *trees.*

We note a few related known results about tree covers. If we allow the distortion to be a rather large constant C, then Bartal, Fandina, and Neiman [BFN19] showed that it is possible to construct a C-tree cover with $O(1)$ size. Gupta, Kumar, and Rastogi [GKR01] constructed a tree cover with distortion 3 using $O(\log n)$ trees. For distortion 1 (exact tree cover), $O(\sqrt{n})$ trees is sufficient [GKR01]; furthermore, $\Omega(\sqrt{n})$ trees are necessary for some planar graphs if each tree in the tree cover must be a spanning tree of G. The main takeaway is that either the distortion is too high for a constant number of trees, or the number of trees have to depend on the number of vertices. Our tree cover constructed in Theorem 1.2 simultaneously has $1+\varepsilon$ distortion and has no dependency on the size of the graph.

Our proof of Theorem 1.2 follows from a *general framework* that we introduce to construct tree covers. Our framework applies to minor-free graphs, which is a much broader class than planar graphs. At a high level, the framework has three steps. First, we devise a reduction of a tree cover with *multiplicative distortion* to the construction of tree covers with *additive distortion*. Second, inspired by the scattering partition defined by Filtser [Fil20b], we introduce the notion of *shortcut partition*, and show that the existence of a shortcut partition suffices to get a tree cover with additive distortion and a constant number of trees. The final step involves constructing shortcut partitions for graphs of interest. Applying our framework as is, we obtain $(1+\varepsilon)$-tree covers for planar metrics using $O_\varepsilon(1)$ trees, with dependency exponential in $1/\varepsilon$. Surprisingly, we manage to identify a new structural result by leveraging the fact that planar graphs are *grid-like* in a formal sense, and then proceed to construct shortcut partition with additional properties. The additional structure allows us to reduce tree-cover size to a polynomial in $1/\varepsilon$, as stated in Theorem 1.2. To keep our construction simple, we do not attempt to optimize the dependency on $1/\varepsilon$; determining the exact dependency on $1/\varepsilon$ for the size of the tree cover is an interesting question that we do not pursue in this work.

The (weighted) planar grids are often used as canonical examples in developing structural and algorithmic results for planar graphs. In many cases, the planar grids are "hard": for example, the worst-case bound on the separator/treewidth of planar graphs with n vertices is realized by a $\sqrt{n} \times \sqrt{n}$ planar grid—this fact plays a central role in the Robertson-Seymour graph minor theory [RS86]. On the other hand, the planar grids also have a simple regular structure that often serves as a starting point for algorithmic developments (e.g. [GKR01, CK15, FKS19]). Thus, our new grid-like structure for planar graphs may be used to leverage insights developed for planar grids to solve problems on general planar graphs. In addition to our tree cover result (Theorem 1.2), we showcase another application: embedding planar graphs into bounded treewidth graphs with small *additive distortion.*

More formally, given a weighted planar graph G of diameter Δ, we want to construct an embedding $f : V(G) \to V(H)$ into a graph H such that $\delta_G(x, y) \le d_H(f(x), f(y)) \le \delta_G(x, y) + \varepsilon\Delta$ and the treewidth of H, denoted by $\mathrm{tw}(H)$, is minimized. The work of Fox-Epstein, Klein, and Schild [FKS19] was the first to show that $\mathrm{tw}(H) = O(\varepsilon^{-c})$ for some constant c. However, the constant c they obtained is very big—a rough estimate[2] from their paper gives $c \ge 58$—and the proof is extremely complicated, with several reduction steps to what they called the *cage instances*, which themselves require another level of technicality to handle. Followup work [CFKL20, FL22b] provided simpler constructions with a linear dependency on $1/\varepsilon$, at the cost of an extra $O((\log\log n)^2)$ factor in the treewidth. It remains an open problem how to construct an embedding that has the best of both: a *simpler construction* such that the treewidth has a reasonable dependency on $1/\varepsilon$ but no dependency on n. We exploit the aforementioned grid-like structure to resolve this problem.

Theorem 1.3. *Let G be any given edge-weighted planar graph with n vertices and diameter Δ. For any given parameter $\varepsilon \in (0, 1)$, we can construct in polynomial time an embedding of G into a graph H such that the additive distortion is $\varepsilon\Delta$ and $\mathrm{tw}(H) = O(1/\varepsilon^4)$.*

Beyond planar graphs. We also obtain several other results as corollaries of the framework and its technical construction. In particular, we show that bounded treewidth graphs admit a tree cover of constant size as well.

Theorem 1.4. *Let G be any graph of treewidth t with n vertices. For any given parameter $\varepsilon \in (0, 1)$, there is a $(1+\varepsilon)$-tree cover $\mathcal{F}$ for the shortest path metric of G using $2^{(t/\varepsilon)^{O(t)}}$ trees.*

Theorem 1.4 improves upon the tree cover construction by Gupta, Kumar, and Rastogi [GKR01] who obtained a tree cover of size $O(\log n)$ for constant treewidth t and constant ε. Again, our tree cover has no dependency on the number of vertices in G.

We also obtain the first non-trivial result for *exact* tree covers in *unweighted* minor-free graphs G with small diameter. Graphs of small diameter have been central in distributed computing. Specifically, the structure of unweighted *planar graphs* of constant diameter has recently attracted attention from distributed computing community [GH15, GH16, HIZ16, LP19]. We obtain an *exact* tree cover of constant size for such graphs. Furthermore, our tree cover is *spanning* (in the sense of metric embedding literature [AKPW95, AN12]); that is, every tree in the tree cover is a subgraph of G. Having a spanning tree cover is important: For example, it is useful in distributed computing, as messages can only be sent along the edges of the input graph. We believe our result for the exact tree cover is of independent interest.

Theorem 1.5. *Let G be any unweighted K_r-minor-free graph (for any constant k) with n vertices and diameter Δ. There is an exact spanning cover $\mathcal{F}$ for the shortest path metric of G using $2^{O(\Delta)}$ trees.*

Gupta, Kumar, and Rastogi [GKR01] showed that there exists an n-vertex planar graph such that any spanning tree cover of the graph must have size $\Omega(\sqrt{n})$. Our Theorem 1.5 circumvents their lower bound when the diameter of the graph is small.

[2] In page 1084 of [FKS19], the treewidth bound is at least $\varepsilon^{-3} \cdot h(\varepsilon^{-11})$ with $h(x) = O(x^5)$ determined by Proposition 7.7.

Applications. One application of our tree cover result (Theorem 1.2) is to the design of $(1+\varepsilon)$-approximate distance oracle for planar graphs. The goal is to construct a data structure of small space $S(n)$ for a given planar graph, so that each distance query can be answered quickly in time $Q(n)$ and the returned distance is within $1+\varepsilon$ factor of the queried distance. Ideally, we want $S(n) = O(n)$ and $Q(n) = O(1)$. For doubling metrics, such a data structure was known for a long time [HPM06], but for planar metrics, attempts to obtain the same result were not successful for more than two decades [Tho04, Kle02, WN16] until the recent work by Le and Wulff-Nilsen [LWN21]. Our Theorem 1.2 provides a simple reduction to the same problem *in trees*: given a distance query, query the distance in each tree and then return the minimum. This illustrates the power of our tree cover theorem in mirroring results in Euclidean/doubling metrics to planar metrics.

The distance oracle constructed by our tree cover theorem has other advantages over that of Le and Wulff-Nilsen. First, their algorithm is very complicated, with many steps using the full power of the RAM model to pack $O(\log n)$ bits of data in $O(1)$ words to guarantee $O(n)$ space. In many ways, bit packing can be viewed as "abusing" the RAM model. Second, in weaker models, such as the *pointer machine model*—a popular and natural model in data structures—where bit packing is not allowed, their construction does not give anything better (and sometimes worse) than the older constructions. The best oracle in the pointer machine model was by Wulff-Nilsen [WN16], with $O(n\,\mathrm{poly}(\log\log n))$ space and $O(\mathrm{poly}(\log\log n))$ query time.

We instead reduce to querying distances on trees, which we further reduce to the lowest common ancestor (LCA) problem. LCA admits a data structure with $O(n)$ space and $O(1)$ time in the RAM model [HT84, BFC00], and $O(n)$ space and $O(\log\log n)$ query time in the pointer machine model [Van76]. (Harel and Tarjan [HT84] showed a lower bound of $\Omega(\log\log n)$ for querying LCA in the pointer machine model.) As a result, we not only recover the result by Le and Wulff-Nilsen [LWN21], but also obtain the best known distance oracle in the pointer machine model with $O(n)$ space and $O(\log\log n)$ query time.

Theorem 1.6. *Suppose that any n-vertex tree admits a data structure for querying lowest common ancestors with space $S_{LCA}(n)$ and query time $Q_{LCA}(n)$. Then given any parameter $\varepsilon \in (0,1)$, and any edge-weighted undirected planar graphs with n vertices, we can design a $(1+\varepsilon)$-approximate distance oracle with space $O(S_{LCA}(O(n)) \cdot \tau(\varepsilon))$ and query time $O(Q_{LCA}(O(n)) \cdot \tau(\varepsilon))$, where $\tau(\varepsilon) = \varepsilon^{-3}\log(1/\varepsilon)$. Consequently, we obtain:*

- *In the word RAM model with word size $\Omega(\log n)$, our oracle has space $O(n \cdot \tau(\varepsilon))$ and query time $O(\tau(\varepsilon))$.*
- *In the pointer machine model, our oracle has space $O(n \cdot \tau(\varepsilon))$ and query time $O(\log\log n \cdot \tau(\varepsilon))$.*

In Section 9, we also discuss other applications of our tree cover theorems. We construct the first $(1+\varepsilon)$-emulator of planar graphs with linear size. We also obtain improved bounds for several problems in constructing low-hop emulators and routing studied in prior work [GKR01, CKT22a, KLMS22].

1.2 Techniques

Previous constructions of exact or $(1+\varepsilon)$-multiplicative tree covers rely on *separators* of planar graphs. In particular, Gupta, Kumar, and Rastogi [GKR01] constructed an exact tree cover of size $O(\sqrt{n})$ by first finding a separator of size $O(\sqrt{n})$, creating $O(\sqrt{n})$ shortest path trees each rooted at a vertex in the separator, and recursing on the rest of the graph. The $O(\log^2 n)$-size construction of Bartal, Fandina, and Neiman [BFN19] follows the same line, but uses *shortest-path separators* instead. More precisely, they find a balanced separator consisting of $O(1)$ many shortest paths. Then for each shortest path, they construct $O(\log n)$ trees by randomly "attaching" remaining vertices (via new edges) to the $O(1/\varepsilon)$

portals (à la Thorup [Tho04]) along the shortest path, and recurse. As the recursion depth is $O(\log n)$, they obtain a $(1+\varepsilon)$-tree cover of size $O(\log^2 n)$. It is possible to remove the $O(\log n)$ factor in the random attachment step by a more careful analysis, but $\Omega(\log n)$ is the barrier to the number of trees for recursive constructions using balanced separators. (Indeed, many other constructions in planar graphs suffer from the same $\log n$ barriers [Kle02, Tho04, EKM13, CGH16, CFKL20], some of which were overcome by very different techniques [FKS19, LWN21, FL22b, CKT22a].)

Here we devise a new technical framework to overcome the $\log n$ barrier in the construction of tree covers. The first step in the framework is a reduction to a tree cover with additive distortion. Given $\beta > 0$, we say that a set of trees $\mathcal{F}$ is a tree cover with *additive distortion* $+\beta$ if every tree in $\mathcal{F}$ is dominating and for any $x, y \in V(G)$, there exists a tree $T \in \mathcal{F}$ such that $d_T(x, y) \le \delta_G(x, y) + \beta$. We say that the tree cover $\mathcal{F}$ is *Δ-bounded* if the diameter of every tree in $\mathcal{F}$ is at most Δ. In Section 8, we show that the reduction to tree cover with additive distortion $+\varepsilon\Delta$ only incurs tiny loss of $O(\log(1/\varepsilon))$ factor on the size of the tree cover.

Lemma 1.7 (Reduction to additive tree covers). *Let (X, δ_X) be a K_r-minor-free metric (for any constant r) with n points. For any parameter $\varepsilon \in (0,1)$, suppose that any K_r-minor-free submetric induced by a subset $Y \subseteq X$ with diameter Δ has an $O(\Delta)$-bounded tree cover $\mathcal{F}$ of size $\tau(\varepsilon)$ of additive distortion $+\varepsilon\Delta$. Then (X, δ_X) has a tree cover of size $O(\tau(O(\varepsilon)) \cdot \log(1/\varepsilon))$ with multiplicative distortion $1+\varepsilon$.*

To prove Lemma 1.7, we introduce the notion of a *hierarchical pairwise partition family* (HPPF), and show that the reduction follows from an HPPF. An HPPF is a collection of hierarchies of partitions, where each partition at level i of a hierarchy in the family is a partition of the planar metric into clusters of diameter roughly $\Theta(1/\varepsilon^i)$. The HPPF has a special property called the *pairwise property*: for every pair $(x, y) \in X$, there is a partition at some level in a hierarchy in the family such that both x and y are contained in some cluster C of the partition, and the diameter of C is roughly $\Theta(\delta_X(x, y))$. We show that HPPF can be constructed from a hierarchical partition family studied in previous works [KLMN05, BFN19].

Next we focus on constructing a tree cover for planar graphs of diameter Δ with additive distortion $+\varepsilon\Delta$. Our goal is to construct a clustering where each cluster has small diameter $O(\varepsilon\Delta)$. After contracting all these clusters, we obtain a cluster graph $\check{G}$, which we would like to treat as an unweighted graph. Furthermore, the clusters we constructed will ensure that $\check{G}$ has a small unweighted diameter, independent to the size of the original graph. We formalize the properties we need through the notion of an *(ε, h)-shortcut partition*: every cluster in the partition has diameter at most $\varepsilon\Delta$ (where Δ denotes the diameter of G), and (loosely speaking) the hop diameter of $\check{G}$ is at most h; see Definition 2.1 for a more formal definition. Now for this special case when cluster graph $\check{G}$ has constant diameter $h = O_\varepsilon(1)$, we devise an inductive construction that runs in h rounds, where in the i-th round we only preserve paths in $\check{G}$ with distances up to i. The inductive construction not only gives us an exact tree cover of constant size (as claimed in Theorem 1.5), but also has other attractive properties. One property, which then plays a crucial role in our construction of a tree cover with additive distortion, is what we call the *root preservation property* (see Section 2.2): every tree in the tree cover can be decomposed into a forest, where each tree in the forest is a BFS tree, and the distance between any two vertices is preserved by a path going through the root of a tree in some forest. Our construction only makes use of the fact that any minor of the input graph has bounded *degeneracy*; hence the result can be extended to any minor-free graphs. Furthermore, the tree cover we get is a *spanning* tree cover, which means that the trees are subgraphs of the input graph. We emphasize again that our construction is the first that does not use balanced separators.

Here comes another issue: if we construct (an exact) tree cover of $\check{G}$, we must somehow turn it into a tree cover for G. A simple idea is to take a tree, say $\check{T}$, in a tree cover of $\check{G}$, and *expand* each (contracted) vertex $\check{c}$ in $\check{T}$ with the corresponding cluster C in G: attach every vertex $v \in C$ to $\check{c}$ by an edge of length roughly $\varepsilon\Delta$ (the diameter of C). However, the shortest path from u to v in G could intersect multiple clusters, and simply expanding clusters as described above would incur a very large additive distortion (remember that we can only tolerate up to $+\varepsilon\Delta$ distortion). This is where the root preservation property comes to the rescue: we can replace every rooted tree with a (Steiner) star centered at the root, where the star edges are weighted *according to distance on* G. The root preservation property implies the only relevant paths in the tree are those that pass through the root—and transforming trees into stars preserves these distances up to $+O(\varepsilon\Delta)$ distortion. As a result, we are able to show a black-box reduction from (ε, h)-shortcut partition to tree covers with additive distortion. This reduction works for any minor-free graph.

Theorem 1.8. *Let G be an (undirected) weighted minor-free graph with diameter Δ. Suppose that for any $\varepsilon > 0$ there is an $(\varepsilon, f(\varepsilon))$-shortcut partition for G for some function $f(\varepsilon)$ depending only on ε, Then G admits a tree cover of size $2^{O(f(\varepsilon))}$ with additive distortion $+O(\varepsilon\Delta)$.*

The construction of tree cover now is reduced to constructing an $(\varepsilon, f(\varepsilon))$-shortcut partition. For graphs with treewidth t, we provide a construction with $f(\varepsilon) = (t/\varepsilon)^{O(t)}$; see Section 6 for details. This, together with Lemma 1.7 and Theorem 1.8, implies our $O(1)$-size tree cover for bounded-treewidth graphs (Theorem 1.4). For planar graphs, constructing an $(\varepsilon, f(\varepsilon))$-shortcut partition is much more difficult. Our key insight here is that planar graphs are *grid-like*.

Informal discussion of grid-like structure of planar graphs. A planar grid graph can be decomposed into an ordered collection of columns, where each column is a collection of vertices. We identify two important properties of grid graphs:

- Each column is a shortest path.
- Every edge goes between two vertices either in the same column or in consecutive columns. (In particular, this implies that every path from a column C_1 to a column C_2 passes through every column in between C_1 and C_2.)

We would like to say that every planar graph admits a partition into clusters of diameter $\varepsilon\Delta$, such that if we contract all clusters into supernodes, the contracted graph satisfies the above properties (for example, $\check{G}_1$ in Figure 1). However, this may not always be the case.

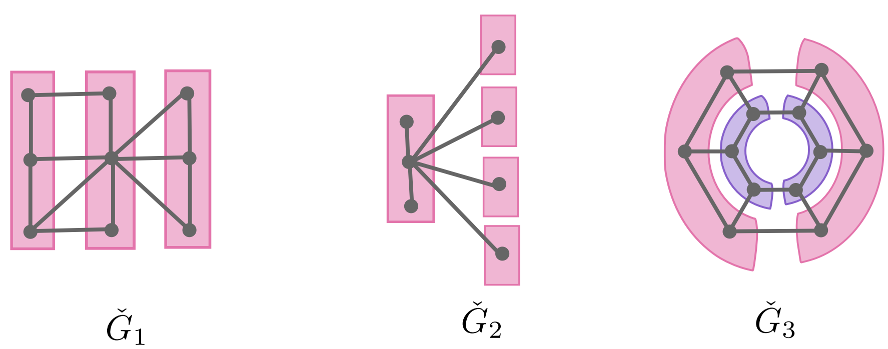

Figure 1. Three planar graphs, with "columns" highlighted in pink. Graph $\check{G}_1$ looks like a grid; graph $\check{G}_2$ requires tree-like column ordering; graph $\check{G}_3$ requires column nesting. The nested columns in $\check{G}_3$ are highlighted in purple.

Issue 1 (Trees): If the contracted graph is a star, it does not look like a grid (for example, $\check{G}_2$ in Figure 1). There is no way to partition the vertices into columns and assign an ordering such that edges only occur between consecutive columns. As such, we relax the guarantee. Instead of assigning a total order to the columns, we assign a tree structure to the columns such that edges occur only between columns that are adjacent in the tree.

Issue 2 (Nesting): If the contracted graph is a circular grid graph, it also does not look like a grid (for example, $\check{G}_3$ in Figure 1). We cannot partition the vertices into columns and assign a valid ordering to the columns. To deal with this, we allow some subgraphs to "sit in between" two adjacent columns. These subgraphs (recursively) satisfy the grid-like structure with nesting. We guarantee that there are few layers of nesting.

We show that these are essentially the only ways in which planar graphs can violate the grid-like property. To deal with nesting, we decompose the graph into a tree structure called *gridtree hierarchy*, such that each node of the hierarchy is associated with a grid-like structure called *gridtree*. The hierarchy has depth $O(1/\varepsilon)$; i.e., there are few layers of nesting. Each gridtree is a tree, such that each node is associated with a subgraph of G we call a *column*. Each column contains vertices of distance at most $\varepsilon\Delta$ from a shortest path within; we say the width of the hierarchy is $\varepsilon\Delta$. The gridtrees in the hierarchy are reminiscent of the recursive structures built for sparse covers on planar graphs in the work of Busch, LaFortune, and Tirthapura [BLT14]. We show that, given a gridtree hierarchy with width $\varepsilon^2\Delta$, we can construct an $(\varepsilon, O(1/\varepsilon^2))$-shortcut partition of G. Again with Lemma 1.7 and Theorem 1.8 we get a tree cover of size $2^{O(1/\varepsilon^2)}$ out of the box using our framework. Perhaps surprisingly, we can improve upon the exponential size bound by working with the gridtree directly, using techniques inspired by our earlier reduction to tree cover in low-diameter graphs. Roughly speaking, shortcut partitions enable us to reduce to the special case of bounded-diameter planar graphs; the extra properties of the gridtree enable us to reduce (at least in spirit) to the special case of bounded-diameter planar grids. Using the gridtree hierarchy, we construct a tree cover with additive distortion $+\varepsilon\Delta$ and size $O(1/\varepsilon^3)$. Thus, by Lemma 1.7, we obtain a tree cover of size $O(\varepsilon^{-3} \cdot \log(1/\varepsilon))$ as claimed in Theorem 1.2.

Embedding into bounded treewidth graphs with additive distortion. To demonstrate the power of our new grid-like structure, we prove that any planar graph G of diameter Δ can be embedded into a bounded treewidth graph with additive distortion $+O(\varepsilon\Delta)$. We construct a shortcut partition $\mathfrak{P}$ and tree cover $\mathcal{F}$ (which actually is a collection of *spanning forests* with the root preservation property) for G with the following extra property:

- [Disjointness.] No two trees in any forest F in $\mathcal{F}$ contain vertices from the same cluster in $\mathfrak{P}$.

Our embedding has three steps. First, we contract each cluster of $\mathfrak{P}$ into a supernode and obtain $\check{G}$. The low-hop property of shortcut partition $\mathfrak{P}$ implies that $\mathrm{tw}(\check{G}) = O(1/\varepsilon)$. Second, for each vertex $\check{c}$ in $\check{G}$, let C be the corresponding cluster in $\mathfrak{P}$. We attach (a copy of) each vertex v in C to $\check{c}$ via a single edge. At this point, treewidth of $\check{G}$ remains $O(1/\varepsilon)$, but the distortion could be large. Third, we augment $\check{G}$ by adding more edges: For each rooted tree T in each forest F in $\mathcal{F}$, we add an edge from (the copy of) the root of T, say r, in $\check{G}$ to (the copy of) every other vertex of T, say v, by an edge of weight $\delta_G(r, v)$. Let $\hat{G}$ be the resulting graph. The root preservation property of tree cover $\mathcal{F}$ implies that the distortion is $+O(\varepsilon\Delta)$, as required by Theorem 1.3. However, it is unclear why $\hat{G}$ has small treewidth. Here we use the disjointness property, which intuitively implies that the augmentation only happens at "disjoint local areas" of $\hat{G}$. We formalize this intuition via a key lemma (Lemma 7.3) showing that $\mathrm{tw}(\hat{G}) = O(|\mathcal{F}| \cdot \mathrm{tw}(\check{G})) = O(\varepsilon^{-3} \cdot \varepsilon^{-1}) = O(\varepsilon^{-4})$. Theorem 1.3 now follows.

1.3 Related work

A closely related notion of tree cover is a *Ramsey tree cover*. In a Ramsey t-tree cover $\mathcal{F}$, each vertex v is associated with a tree $T_{\mathrm{home}(v)} \in \mathcal{F}$, such that the distance from v in $T_{\mathrm{home}(v)}$ to every other vertex is an approximation of the original distance up to a factor of t. Thus, Ramsey tree covers have stronger guarantees than ordinary tree covers. Both tree covers and Ramsey tree covers for general metrics were studied in the past [BLMN05, TZ01, BFN19]. In general metrics, bounds obtained for tree covers and Ramsey tree covers are almost the same. It is possible to construct a Ramsey tree cover with tradeoff between size k and distortion $\tilde{O}(n^{1/k})$ [BFN19], or of size $\tilde{O}(n^{1/t})$ and distortion $O(t)$ [MN06]; these bounds are almost optimal [BFN19].

However, Ramsey tree covers with a constant number of trees and constant distortion do not exist in planar and doubling metrics. In particular, for metrics which are *both planar and doubling* and for any distortion $\alpha \geq 1$, Bartal *et al.* [BFN19] showed that the number of trees in the Ramsey tree cover must be $n^{\Omega(1/(\alpha \log(\alpha)))}$. This sharply contrasts with our result for (non-Ramsey) tree covers in Theorem 1.2.

2 Tree cover for graphs with good shortcut partition

A *tree cover* $\mathcal{F}$ of an edge-weighted planar graph G is a collection of trees, so that every pair of vertices u and v in G has a *low-distortion* path in at least one of the trees in $\mathcal{F}$. Specifically,

- tree cover $\mathcal{F}$ has *α-multiplicative distortion* if there is a tree T in $\mathcal{F}$ satisfying

$$\delta_G(u,v) \leq \delta_T(u,v) \leq \alpha \cdot \delta_G(u,v).$$

- tree cover $\mathcal{F}$ has *β-additive distortion* if there is a tree T in $\mathcal{F}$ satisfying

$$\delta_G(u,v) \leq \delta_T(u,v) \leq \delta_G(u,v) + \beta.$$

Sometimes it is easier to describe the construction in terms of *forest covers*: that is, instead of a collection of trees, we allow a collection of *forests* to be in the cover of G. Let Δ be the diameter of G. Recall that a tree cover $\mathcal{T}$ is *Δ-bounded* if every tree of $\mathcal{T}$ has diameter at most Δ; we say that a forest cover $\mathcal{F}$ is Δ-bounded if every tree in every forest of $\mathcal{F}$ is Δ-bounded. We remark that one can easily construct an $O(\Delta)$-bounded tree cover from an $O(\Delta)$-bounded forest cover: Simply connect the tree components within each forest into a single tree in a star-like way, assigning weight Δ to the newly added star edges. As the diameter of the graph is Δ, each newly constructed tree is a dominating tree.

Our main result of this section is to show that in order to construct a tree cover of constant size with $(1+\varepsilon)$-multiplicative distortion for arbitrary weighted planar graph G with diameter Δ, it is sufficient to do the following: (1) Reduce the problem to constructing tree covers of constant size with $\varepsilon\Delta$-additive distortion. (2) Find a shortcut partition for G into *clusters* such that every cluster has diameter $\varepsilon\Delta$, and contract all clusters into supernodes to form the *cluster graph* $\check{G}$, such that there is a shortest path between any two nodes that has at most $O_\varepsilon(1)$ edges. (3) Construct a constant-size tree cover for the cluster graph $\check{G}$, which has bounded hop-diameter.

After introducing some terminologies (Sections 2.1 and 2.2), we first prove that the above reduction strategy works in Section 2.3 (Theorem 2.2), then in Section 2.4 we prove the existence of tree cover for planar graph with bounded hop-diameter (Theorem 2.5). (Surprisingly, the tree cover constructed preserves the distance *exactly* without distortion.) In Sections 3 and 4 we prove the existence of shortcut partitions for planar graphs. We construct tree cover for planar graphs whose size is polynomially dependent on $1/\varepsilon$ in Section 5. Next, in Section 6 we construct shortcut partition for bounded-treewidth graphs. In Section 7 we prove that any planar graph embeds into a bounded-treewidth graph with additive distortion. Finally, in Section 8 we present the full details of the reduction from multiplicative to additive distortion for tree covers. We conclude the paper with some applications (Section 9).

2.1 Shortcut partitions

Throughout this section and the rest of the paper, let ε and Δ be fixed parameters, with $\Delta > 1$ and $0 < \varepsilon < 1$. (Later on in the paper we might recurse on some subgraph H of G. In such case we will consistently use Δ for the diameter of G; in particular, the diameter of H might increase and be bigger than Δ.) Let G be an (undirected) weighted planar graph with diameter Δ. When a graph H is a subgraph of G, we write $H \subseteq G$. For any subgraph $H \subseteq G$, let $G[H]$ denote the subgraph of G induced by the vertices of H. A *cluster* C is a subset of vertices in G such that the induced subgraph $G[C]$ is *connected*. A *clustering* of a planar graph G is a partition of the vertices of G into clusters $\mathcal{C} := \{C_1, \ldots, C_m\}$. Let $\check{G}$ be the *cluster graph* of G with respect to $\mathcal{C}$, where each cluster in $\mathcal{C}$ is contracted to a *supernode*. We always treat $\check{G}$ as an unweighted graph; to emphasize this, we use the terms *hop-length*, *hop-distance*, and *hop-diameter* as opposed to length, distance, and diameter when referring to $\check{G}$. An *h-hop path* is a path with h edges.

Our general goal is to construct a clustering for G such that the *diameter* of each cluster is small. One can measure the diameter of a cluster in two ways:

- a cluster C has *strong diameter* D if $\delta_{G[C]}(u, v) \leq D$ for any two vertices u and v in C;
- a cluster C has *weak diameter* D if $\delta_G(u, v) \leq D$ for any two vertices u and v in C.

Notice that cluster C has strong diameter D implies it has weak diameter D as well; the main difference is whether a shortest path between u and v is within the cluster C itself, or within the whole graph G.

Definition 2.1. *An (ε, h)-shortcut partition*[3] *is a clustering $\mathcal{C} = \{C_1, \ldots, C_m\}$ of G such that:*

- [Diameter.] *the strong diameter of each cluster C_i is at most $\varepsilon\Delta$, where Δ is the diameter of G;*
- [Low-hop.] *for any vertices u and v in G, there is an approximate shortest-path π between u and v in G with length at most $(1 + \varepsilon) \cdot \delta_G(u, v)$, and there is a path $\check{\pi}$ in the cluster graph $\check{G}$ between the clusters containing u and v such that (1) $\check{\pi}$ has hop-length at most h, and (2) $\check{\pi}$ only contains clusters that have nontrivial intersection with π.*

Notice that the low-hop property does *not* guarantee that path π between u and v intersects at most h clusters. Rather, it guarantees that there is some h-hop path between the two clusters containing u and v respectively in the cluster graph; see Figure 2. We remark that the cluster graph obtained by contracting every cluster in an (ε, h)-shortcut partition has hop-diameter at most h.

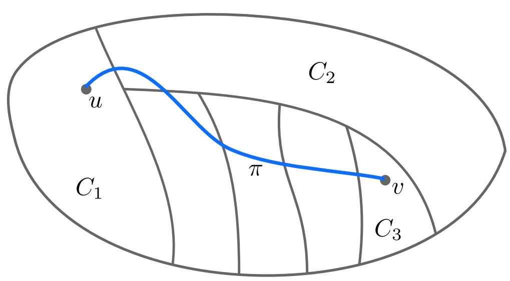

Figure 2. A graph partitioned into clusters. There is a path π between vertices u and v. Path π intersects 6 clusters. In the cluster graph, there is a 2-hop path $\check{\pi} = (C_1, C_2, C_3)$ that only contains clusters intersecting π.

[3] Arnold Filtser [Fil20b] introduced the notion of *scattering partition*. In a scattering partition it is required that *every shortest path* of length $\alpha \cdot \varepsilon\Delta$ intersects at most $O(\alpha)$ clusters, which is stronger than shortcut partition. Scattering partition is conjectured to exist for any minor-free graphs [Fil20b, Conjecture 1].

The following theorem summarizes our framework of constructing (additive) tree covers for graphs with shortcut partitions; it is a restatement of Theorem 1.8 with more details. The proof can be found in Section 2.3 after we introduce the necessary terminologies.

Theorem 2.2. *Let G be an (undirected) weighted minor-free graph with diameter Δ. Suppose that (1) for any $\varepsilon > 0$ there is an $(\varepsilon, f(\varepsilon))$-shortcut partition for G for some function $f(\varepsilon)$ depending only on ε, and (2) the cluster graph $\check{G}$ with respect to the shortcut partition has a forest cover of $2^{O(f(\varepsilon))}$ size, satisfying the root preservation property defined in Section 2.2. Then G admits an $O(\Delta)$-bounded forest cover with additive distortion $+O(\varepsilon\Delta)$, with $2^{O(f(\varepsilon))}$ forests.*

2.2 Exact spanning tree cover for small diameter minor-free graphs

Let G be an *unweighted*, undirected K_r-minor-free graph with diameter Δ for some constant r. For every $k = 1, \ldots, \Delta$, we will construct a set of forests $\mathcal{F}_k$ (each of which is a *subgraph* of G) such that if $\delta_G(u,v) \le k$, then there is some forest $F \in \mathcal{F}_k$ such that $\delta_F(u,v) = \delta_G(u,v)$. In other words, the shortest path between u and v in F is also a shortest path between u and v in G. As a base case, we use the fact that the edges of a K_r-minor-free graph G can be partitioned into $O_r(1)$ star forests (the *arboricity* and *star arboricity* of K_r-minor-free graphs are both $O(r\sqrt{\log r})$ [Kos82]; this set of forests is $\mathcal{F}_1$. Roughly speaking, we construct $\mathcal{F}_{k+1}$ from $\mathcal{F}_k$ by "expanding" all trees in $\mathcal{F}_k$ to include one more edge. As a first attempt at implementing this idea, we might try replacing every tree $T \in \mathcal{F}_k$ with $T \cup N_G(T)$, where $N_G(T)$ denotes the set of all edges incident to T in G. Expanding all trees in this way would clearly satisfy the distance-preserving guarantee for $k+1$, but the expanded trees might not be vertex-disjoint (i.e., expanding a forest this way might produce a graph that is not a forest).

Perhaps surprisingly, we demonstrate that the suggested approach isn't far off, and the expansion can indeed be simulated with a set of $O(1)$ forests (see Lemma 2.6). This idea gives a forest cover for $O(1)$-diameter minor-free graphs. In fact, the forest cover will have a very specific structure: each tree in each forest will be a BFS tree in the original graph G, and the distance between any two vertices in G is preserved in the forest cover by a path going through the root of some tree. Notice that this implies that the forest cover is *spanning*: in other words, it uses no Steiner points or edges. The spanning property is helpful for later applications.

Root expansions. Let G be a planar graph with vertices $V(G)$ and edges $E(G)$, and H be a subgraph of G. Let $\delta_H(u,v)$ denote the distance in H between vertices u and v in $V(H)$. If $T \subseteq G$ is a tree rooted at r, we say that T is a *BFS tree* if it is a BFS tree on $G[T]$ (that is, if $\delta_T(r,v) = \delta_{G[T]}(r,v)$ for every vertex v of T. In general, $\delta_{G[T]}(r,v)$ may still be larger than $\delta_G(r,v)$.) We say that a forest $F \subseteq G$ is a *BFS forest* if every tree in F is a BFS tree. Let $T \subseteq G$ be a BFS tree rooted at r, and let π be a path in G. The BFS tree T *preserves* π if (i) every vertex in π is in $V(T)$, and (ii) π passes through r.

Observation 2.3. *If tree T preserves a shortest path π in G between vertices u and v, then $\delta_T(u,v) = \delta_G(u,v)$, even though π might not be contained in T.*

We say that a BFS forest F preserves π if F contains a BFS tree that preserves π; a set $\mathcal{F}$ of BFS forests preserves π if $\mathcal{F}$ contains a BFS forest that preserves π. If G is a graph with diameter Δ and $\mathcal{F}$ is a forest cover for G, we say that $\mathcal{F}$ has the *root preservation property* if $\mathcal{F}$ is a set of BFS forests of G that preserves every path in G of length at most Δ.

We now formalize the key idea of "expanding" a tree, as described at the beginning of the section. For any path π in G, we define the *prefix* of π to be the path containing all but the last edge of π.

Definition 2.4. *Let $T \subseteq G$ be a BFS tree with root r. A root expansion of T is a set of BFS trees $\mathcal{T}$ such that for every path π in G whose prefix is preserved by T, the set $\mathcal{T}$ preserves π.*

A *root expansion* of a BFS forest F is a set of BFS forests $\mathcal{F}$, each consisting of vertex-disjoint BFS trees, such that the set of all trees in $\mathcal{F}$ forms a root expansion for each tree in F. The *size* of the expansion is the number of forests in the set. We emphasize that the trees in the same BFS forest have to be vertex-disjoint from each other, while trees in different BFS forests may not be vertex-disjoint.

Lemma 2.6. *For every BFS forest $F \subseteq G$, there is a root expansion of F of size $O(1)$.*

Postponing the proof of Lemma 2.6 to Section 2.4, we first prove our main result of this section.

Theorem 2.5. *Let G be an unweighted planar graph with diameter Δ. Then there is a set of BFS forests $\mathcal{F}$ of size $2^{O(\Delta)}$, such that $\mathcal{F}$ preserves every path in G of length at most Δ. Consequently, for every $u, v \in V(G)$, there is some forest $F \in \mathcal{F}$ where $\delta_F(u,v) = \delta_G(u,v)$.*

Proof: As G is K_r-minor-free, the edges of G can be covered by $O_r(1)$ forests by Nash-Williams theorem [NW64]. These forests can be converted into $O_r(1)$ star forests. Each star is a BFS tree in G. Let $\mathcal{F}_1$ be such a set of BFS forests. For each $k \in 2, \dots, \Delta$, let $\mathcal{F}_k$ be the set containing all BFS forests from the root expansions of each BFS forest in $\mathcal{F}_{k-1}$ provided by Lemma 2.6. Return $\mathcal{F}_\Delta$.
By induction, each $\mathcal{F}_k$ has size $O(1)^k$; setting $k = \Delta$, we find that $\mathcal{F}_\Delta$ has size $2^{O(\Delta)}$. Again by induction, we have that for every path π of length at most k, there is some forest in $\mathcal{F}_k$ that preserves π, by definition of the root expansion. (The base case when $k = 1$ is immediate since each length-1 path belongs to one of the stars of $\mathcal{F}_1$.) As G has diameter Δ, all shortest paths in G are preserved by $\mathcal{F}_\Delta$. □

2.3 Proving Theorem 2.2

Before we proceed to prove Lemma 2.6, first we prove the correctness of our reduction to bounded hop-diameter case.

Proof (of Theorem 2.2): By assumption (1) of Theorem 2.2, there is an $(\varepsilon, f(\varepsilon))$-shortcut partition for G. As each cluster is connected, the cluster graph $\check{G}$ obtained by contracting each cluster is still minor-free. For each subset $\check{S}$ of $\check{G}$, there is a corresponding set of vertices in $V(G)$ associated with $\check{S}$, which we will naturally denoted as S.

Construction. Treat $\check{G}$ as an unweighted graph. By assumption (2) of Theorem 2.2, there is a forest cover $\check{\mathcal{F}}$ for $\check{G}$ of size $2^{O(f(\varepsilon))}$ that preserves all paths in $\check{G}$ with hop-length $O(f(\varepsilon))$.

For each tree $\check{T}$ in each forest $\check{F}$ in $\check{\mathcal{F}}$ rooted at some supernode C, perform the following *transformation*: Let r be an arbitrary vertex in C; construct a *star S_r* centered at r connected to every vertex in T, where the weight of the edge[4] from r to v is set to be $\delta_G(r,v)$. Applying this transformation to every tree in a forest on $V(\check{G})$ produces a forest on $V(G)$: the fact that clusters in the shortcut partition are vertex-disjoint implies that two trees $\check{T}_1$ and $\check{T}_2$ in forest $\check{F}$ are vertex-disjoint if and only if the two corresponding subsets T_1 and T_2 of G are vertex-disjoint. Return $\mathcal{F}$, the set of forests produced.

[4]Note that the star S_r uses Steiner edges. Unlike in Theorem 2.5, the tree cover we construct for Theorem 2.2 is not a spanning tree cover.

Distortion guarantee. We claim that for every pair u and v in G, there is a path in some forest $F \in \mathcal{F}$ such that $\delta_F(u,v) \le \delta_G(u,v) + O(\varepsilon\Delta)$. Let u and v be two vertices in G. By assumption (1), there is a $f(\varepsilon)$-hop path $\check{\pi}$ in the cluster graph induced by the subset of clusters that has nontrivial intersections with some shortest path $\pi(u,v)$ between u and v in G. By construction, $\check{\mathcal{F}}$ contains some BFS tree $\check{T}$ in a forest $\check{F}$ that preserves $\check{\pi}$. Let C be the cluster that is the root of $\check{T}$. Then $\mathcal{F}$ contains some star S_r that is rooted at $r \in C$ connecting to every vertex in T. As $\check{\pi}$ is preserved by $\check{T}$, T contains both u and v because $\check{\pi}$ starts at the cluster containing u and ends at the one containing v. Thus, $\delta_{S_r}(u,v)$ is the length of a shortest path between u and v in G that passes through r; i.e. $\delta_{S_r}(u,v) = \delta_G(r,u) + \delta_G(r,v)$. Further, because C is guaranteed to intersect $\pi(u,v)$ by the property of shortcut partition, there is a short path in G between u and v through r—walk from u along $\pi(u,v)$ until reaching some vertex $c \in C$, walk from c to r and then back to c (in G but not necessarily in C_r), and then finish traveling along $\pi(u,v)$. As C has diameter $\varepsilon\Delta$, this path has length $\delta_G(u,v) + O(\varepsilon\Delta)$. Thus, the star S_r in $\mathcal{F}$ satisfies $\delta_{S_r}(u,v) \le \delta_G(u,v) + O(\varepsilon\Delta)$. We remark that each star has radius Δ, so our forest cover is $O(\Delta)$-bounded. □

2.4 Expanding a forest

Lemma 2.6. *For every BFS forest $F \subseteq G$, there is a root expansion of F of size $O(1)$.*

Proof: We give an algorithm for constructing a root expansion. Starting with the graph G, treat each tree in F as a cluster, and create the cluster graph $\check{G}$. Find a vertex-coloring of $\check{G}$ with $O_r(1)$ colors (the *chromatic number* of a K_r-minor-free graph is upper-bounded by its *degeneracy*, which is asymptotically equivalent to arboricity [NW64, Die17]), and let F_c denote the set of trees in F that are colored with color c. Starting with G, contract each tree of F_c into a supernode, and call the resulting graph $\check{G}_c$. One important property is that the other endpoint of any edge of $\check{G}_c$ incident to a supernode must be an *ordinary* vertex not in F_c. (See Figure 3.)

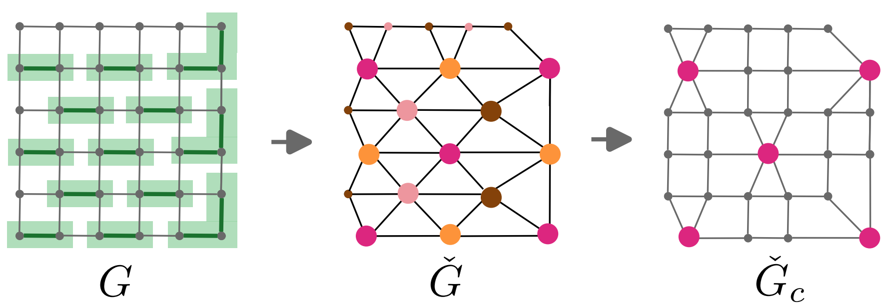

Figure 3. Graph G with forest F colored in green; contracted graph $\check{G}$ with an $O(1)$ vertex-coloring; and contracted graph $\check{G}_c$.

Using the minor-freeness of $\check{G}_c$, find an edge decomposition of $\check{G}_c$ by $O_r(1)$ star forests $\{\check{F}_{c,1}, \dots, \check{F}_{c,O(1)}\}$. For each star $\check{S}$ in $\check{F}_{c,t}$, define $\textsc{Uncompress}(\check{S})$ as follows:

> If the center of $\check{S}$ is a supernode (corresponding to some BFS tree in F_c), let $r \in V(G)$ be the root of that tree. Otherwise, let $r \in V(G)$ be the center of the star $\check{S}$.
>
> Let S denote the set of vertices in G that belong to $\check{S}$ (both as an ordinary vertex in G, or as a vertex in some supernode of $\check{F}_{c,t}$.) Return the BFS tree on $G[S]$ rooted at r.

For every star forest $\check{F}_{c,t}$, let $F_{c,t}$ be the union of $\textsc{Uncompress}(\check{S})$ over every star $\check{S}$ in $\check{G}_{c,t}$. By construction $F_{c,t}$ is a subset of edges in G. Because the stars in $\check{F}_{c,t}$ are vertex-disjoint in $\check{G}$ and the vertex sets in G corresponding to any two nodes in $\check{G}$ are disjoint, $F_{c,t}$ is a disjoint union of BFS trees in G, which is a forest. Return the set of $O_r(1)$ BFS forests $\mathcal{F} := \{F\} \cup \bigcup_{c,t}\{F_{c,t}\}$.

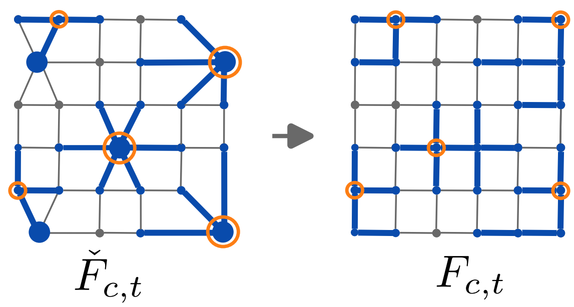

Figure 4. Star forest $\check{F}_{c,t}$ and the uncompressed forest $F_{c,t}$. The root of each tree is circled in orange.

Preservation guarantee. Let π be a path in G between vertices u and v, whose prefix is preserved by F. Let T be a tree in F that preserves the prefix of π.

- **Case 1:** $v \in V(T)$**.** In this case, T preserves the entirety of π, not just the prefix. As $F \in \mathcal{F}$, the path π is preserved by $\mathcal{F}$.
- **Case 2:** $v \notin V(T)$**.** Let c be the color of the forest F_c where T is in. As $v \notin V(T)$, there is an edge $e_\pi \in E(\check{G}_c)$ that corresponds to the last edge of π. The edge e_π is covered by a star $\check{S}$ in some $\check{G}_{c,t}$, connecting a vertex in T with v. There are two subcases.
  - **Case 2a: T is the center of Š.** As the prefix of π passes through the root r of T, clearly π passes through r. Further, the vertices of π are all in S, as e_π is in $E(\check{S})$. Thus, the BFS tree UNCOMPRESS($\check{S}$) preserves π.
  - **Case 2b: T is not the center of Š.** Here the root of the star $\check{S}$ is v, and T is a leaf. Clearly π passes through v, and the vertices of π are all in S. Thus, UNCOMPRESS($\check{S}$) preserves π. $\square$

3 A grid-like clustering for planar graphs

We show that planar graphs can be partitioned into clusters, such that the clusters interact with each other in a manner similar to a grid graph. Using this structure, we show that every planar graph admits an $(\varepsilon, O(1/\varepsilon^2))$-shortcut partition. In this section, we define the grid-like structure and show that it leads naturally to a shortcut partition. In Section 4, we show that every planar graph admits a grid-like clustering. In Section 5, we use this structure to directly construct tree covers for planar graphs.

3.1 Definition of gridtree

Let G be a graph and H be a connected induced subgraph of G, and let w be a fixed parameter. Throughout the rest of the paper, when G is a planar graph, we assume that G has a fixed drawing in the plane; all references to the external face of (a subgraph of) G refer to this fixed drawing. All subgraphs of G inherit the drawing of G.

Definition 3.1. *Let H be a connected graph with a disjoint vertex partition into subsets, some of which are columns and some of which are leftover sets. A width-w gridtree $\mathcal{T}$ (for short, a w-gridtree) is a tree in which there is a one-to-one correspondence between columns and nodes of $\mathcal{T}$, and between leftover sets and edges of $\mathcal{T}$. The gridtree $\mathcal{T}$ satisfies the following properties:*

- [Column adjacency.] *Let (u,v) be an edge of H. Either (1) the endpoints u and v belong to the same subset (either a column or a leftover set) in the partition, or (2) u and v belong to columns that are adjacent in $\mathcal{T}$, or (3) u and v belong to a column and a leftover set that are incident in $\mathcal{T}$.*
- [Column width.] *Let η be a column in $\mathcal{T}$. Notice that every column (other than η) and every edge in $\mathcal{T}$ is either above or below η in $\mathcal{T}$. If some column or edge containing a vertex v is above (resp. below) η, we say that v is above (resp. below) η. If a is a vertex in η that is adjacent (in H) to a vertex above η, and b is a vertex below η, and P is a path in H between a and b, then we say P passes through η. Every path that passes through η has length at least w.*
- [Column shortcut.] *Let η be a column in $\mathcal{T}$, and let H_η denote the subgraph induced by all columns below η, together with all leftover sets below η or incident to η. There is a shortest path π_η in H_η such that all vertices in η are within distance $2w$ to π_η with respect to the induced subgraph $H[\eta]$.*

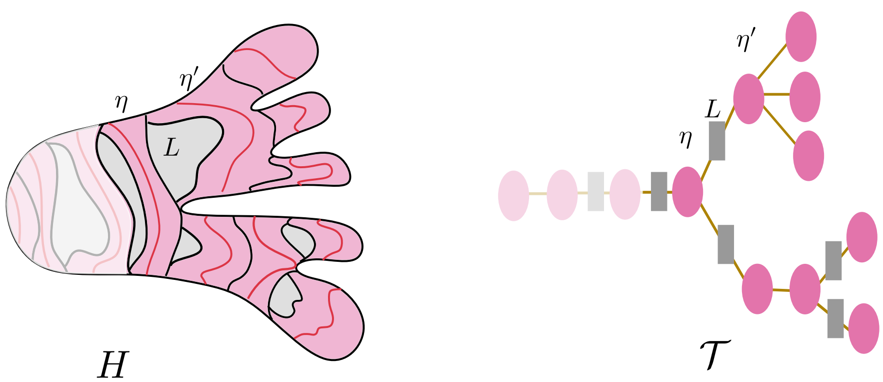

Figure 5. Graph H partitioned into columns and leftover sets, and a gridtree $\mathcal{T}$. Columns η and η' are adjacent, and the leftover set L lies between them. Column η is above L and η. Each column contains a path π (from column shortcut property), marked in red. The subgraph H_η is marked in a darker color.

Lemma 3.2. *For any $w > 0$, any planar graph H has an w-gridtree $\mathcal{T}$ in which every vertex that is at most w away from an external vertex (in a given planar drawing of H) belongs to a column of $\mathcal{T}$.*

Definition 3.3. *A width-w gridtree hierarchy $\mathcal{H}$ of G is a tree in which each node is a pair $\mu = (H^\mu, \mathcal{T}^\mu)$, where H^μ is a connected subgraph of G, and $\mathcal{T}^\mu$ is a w-gridtree for H^μ. The root of $\mathcal{H}$ is associated with the entire graph G. The parent node of μ is denoted by $\mathrm{Pa}(\mu)$. The hierarchy $\mathcal{H}$ satisfies the following:*

- [Layer nesting.] *The children of every node $(H^\mu, \mathcal{T}^\mu)$ are in one-to-one correspondence with the components of subgraphs induced by the leftover sets of $\mathcal{T}^\mu$, together with their gridtrees.*
- [Layer width.] *For every node $(H^\mu, \mathcal{T}^\mu)$, we say that a vertex v in H^μ is an outer vertex of H^μ if v is adjacent to some vertex in a column of $\mathcal{T}^{\mathrm{Pa}(\mu)}$. (The root node of $\mathcal{H}$ has no outer vertices.) For every outer vertex v in H^μ, every vertex u in H^μ with $\delta_{H^\mu}(u,v) \le w$ belongs to some column of $\mathcal{T}^\mu$. In other words, every vertex that is at most distance w away from any outer vertex is covered by columns of $\mathcal{T}^\mu$.*

We remark that, when we construct a gridtree hierarchy for planar graphs in Section 4, the outer vertices of H^μ will be the vertices on the external face of H^μ.

Define *level* of a column η in gridtree $\mathcal{T}$ to be the distance between η and the root of $\mathcal{T}$. Define *layer* of a pair $(H^\mu, \mathcal{T}^\mu)$ in gridtree hierarchy $\mathcal{H}$ to be the distance between μ and the root of $\mathcal{H}$. A gridtree hierarchy $\mathcal{H}$ has *depth* d if the number of layers in $\mathcal{H}$ is at most d. We will reiterate that there are

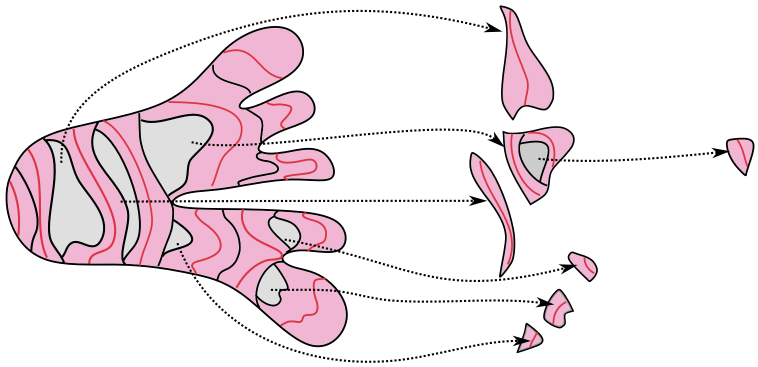

Figure 6. A gridtree hierarchy with depth 3.

two tree structures at play here: the *gridtrees* $\mathcal{T}^\mu$ associated with each instance $(H^\mu, \mathcal{T}^\mu)$ of the gridtree construction, and the *gridtree hierarchy* $\mathcal{H}$ which represents the recursive nature of the construction where every component of the leftover sets of $\mathcal{T}^\mu$ becomes a child of μ. We will be consistent when it comes to the term *levels* and *layers*, where the former is within a specific grid tree $\mathcal{T}^\mu$, and the latter is for the whole hierarchy $\mathcal{H}$. Notice that all columns and leftover sets in a gridtree are pairwise vertex-disjoint, and thus so are the subgraphs H^μ associated with nodes in the same layer of $\mathcal{H}$.

Lemma 3.4. *Let G be a planar graph with diameter Δ, and let ε be a parameter in $(0,1)$. Then G has an $\varepsilon\Delta$-gridtree hierarchy $\mathcal{H}$.*

We will prove Lemma 3.2 and Lemma 3.4 in Section 4.

3.2 Gridtree hierarchy gives shortcut partition

For the rest of the section, we prove that every gridtree hierarchy for a graph G gives rise to a shortcut partition for G. In particular, the hierarchy constructed by Lemma 3.4 produces a shortcut partition. This suggests that planar graphs admit a stronger form of shortcut partition; using this partition, we will prove (cf. Section 5) that planar graphs have $O_\varepsilon(1)$-size tree cover where the constant depends polynomially on $1/\varepsilon$.

Recall that a *clustering* of a planar graph G is a partition of the vertices of G into *clusters* $\mathcal{C} := \{C_1, \ldots, C_m\}$. Let $\check{G}$ be the *cluster graph* of G with each cluster in $\mathcal{C}$ contracted to a *supernode*. Denote $\|P\|$ to be the length of the path P; in notation, if P starts at vertex s and ends at t, then $\|P\| := \delta_P(s,t)$.

Definition 3.5. *The cost of a path P in G with respect to clustering $\mathcal{C}$, denoted as $\text{cost}_\mathcal{C}(P)$, is equal to the minimum hop-length over all paths $\check{P}$ in $\check{G}$ where (1) the endpoints of $\check{P}$ are the clusters containing u and v, and (2) $\check{P}$ only touches supernodes that correspond to clusters that P passes through. For any t between 0 and 1 and for any vertex pair (u,v), we define the cost with $(1+t)$ distortion with respect to clustering $\mathcal{C}$, denoted $\text{cost}_{t,\mathcal{C}}(u,v)$, to be the minimum $\text{cost}_\mathcal{C}(P)$ across every approximate shortest path P between u and v whose length is at most $(1+t)\cdot\delta_G(u,v)$.*

When $\mathcal{C}$ is clear from context, we omit it from the subscript and simply write $\text{cost}(P)$ and $\text{cost}_t(u,v)$. We now introduce a slight generalization of shortcut partitions.

Definition 3.6. *Let G be a graph with diameter Δ. An (ε, h)-shortcut partition with $(1+t)$ distortion for G is a clustering $\mathcal{C} = \{C_1, \ldots, C_m\}$ of G such that:*

- [Diameter.] *The strong diameter of each cluster C_i is at most $\varepsilon\Delta$;*

- [Low-hop.] *For any vertices u and v in G, we have* $\text{cost}_{t,\mathcal{C}}(u,v) \le h$.

Notice that an (ε, h)-shortcut partition with $(1+\varepsilon)$ distortion (as defined above) is an (ε, h)-shortcut partition as defined in Section 2.1. Given a planar graph G with diameter Δ along with a $(t\varepsilon\Delta)$-gridtree hierarchy (where t and ε are between 0 and 1), we will find an $(O(\varepsilon), O\left(\frac{1}{t\varepsilon}\right))$-shortcut partition with $(1+O(t))$ distortion for G. In fact, the partition constructed will satisfy an extra property:

- [Cluster ordering.] For every node $(H, \mathcal{T})$ in the hierarchy $\mathcal{H}$ and for every column η in $\mathcal{T}$, there is an ordering on the *cluster centers* $c_1, \ldots, c_m$ in η, one from each cluster containing vertices of η, such that for every pair $i, j \in \{1, \ldots, m\}$, we have $\delta_{H_\eta}(c_i, c_j) \ge |i-j| \cdot \varepsilon\Delta$.

3.2.1 Clustering a column

```
ClusterColumn(η):
  ⟨⟨select a set of cluster centers⟩⟩
  π′ ← π_η
  i ← 1
  while π′ is nonempty:
      c_i ← first vertex on π′
      π′ ← longest suffix of π′ that is ≥ εΔ shorter than π′
      initialize cluster C_i ← {c_i}
      increment i

  ⟨⟨assign vertices in π_η using closest center⟩⟩
  for each v in V(π_η):
      i* ← arg min_{i∈[m]} δ_η(c_i, v), breaking ties by choosing the smallest i
      assign v to cluster C_{i*}

  ⟨⟨assign vertices in η using closest vertex in π_η⟩⟩
  for each v in η:
      v* ← arg min_{v∈V(π_η)} δ_{H[η]}(v*, v), breaking ties using some fixed ordering of V(π_η)
      assign v to the cluster containing v*
  return the clusters {C_1, ..., C_m}
```

First we describe how to create clusters for a single column in any $(t\varepsilon\Delta)$-gridtree in the hierarchy $\mathcal{H}$. To simplify the presentation, we assume $t \in (0, 1/8]$; indeed, if $t \in (1/8, 1)$ then we can scale down t by a factor of 8 without affecting our results by more than a constant factor. Let $(H, \mathcal{T})$ be a node in $\mathcal{H}$. Let η be a column in $\mathcal{T}$ and let π_η be the shortest path in H_η guaranteed by the column shortcut property (Definition 3.1) for gridtree. In particular, every vertex in η is within distance $2t\varepsilon\Delta$ to π. Then we can cluster η according to the procedure ClusterColumn(η). It is immediate from the construction of the clustering that the cluster centers satisfy the cluster ordering property. Notice that while the column η has width $t\varepsilon\Delta$, the diameter of each cluster constructed is $O(\varepsilon\Delta)$, not $O(t\varepsilon\Delta)$.

Lemma 3.7. *The algorithm* ClusterColumn(η) *returns a clustering of η such that (i) each cluster has strong diameter at most $O(\varepsilon\Delta)$, and (ii) for any pair of points u and v in η, we have* $\text{cost}_{8t}(u,v) \le \frac{\delta_{H_\eta}(u,v)}{\varepsilon\Delta} + 4$.

Proof: **(i)** By assumption, every vertex v in η is within distance $2t\varepsilon\Delta$ of some vertex p_v on π_η such that v is assigned to the same cluster as p_v. By choice of cluster centers $\{c_i\}$, vertex p_v is within distance $\varepsilon\Delta$ (not $t\varepsilon\Delta$) of some c_i and is assigned to same cluster as c_i. Thus, every cluster has radius at most

$(1+2t)\varepsilon\Delta < 2\varepsilon\Delta$ and diameter at most $4\varepsilon\Delta$. Every point on a shortest path between v and p_v belongs to the same cluster, because the construction breaks ties by a fixed ordering of $V(\pi_\eta)$. Similarly, every point on a shortest path between p_v and c_i belongs to the same cluster. We conclude that every clusters is connected, and the $4\varepsilon\Delta$ bound applies to strong diameter.

(ii) Let P be a shortest path in H_η with endpoints u and v in η. Let C_u and C_v denote the clusters containing u and v, respectively. If $C_u = C_v$ or if C_u and C_v are adjacent in the ordering, then $\text{cost}_{8t}(u,v) \le 1$ and we are done. Otherwise, we note that P may walk outside of the column η into other columns or leftover sets. However, by the column shortcut property (Definition 3.1), we can find an approximate shortest path P' that is contained within η: From vertex u, walk (along a shortest path in C_u) to the point p_u in $\pi_\eta \cap C_u$ that is closest to u, then walk along π_η to the point p_v in $\pi_\eta \cap C_v$ that is closest to v, and then walk (along a shortest path in C_v) to v. Denote the subpath of P' on π_η from p_u to p_v as $P'[\pi_\eta]$; see Figure 7.

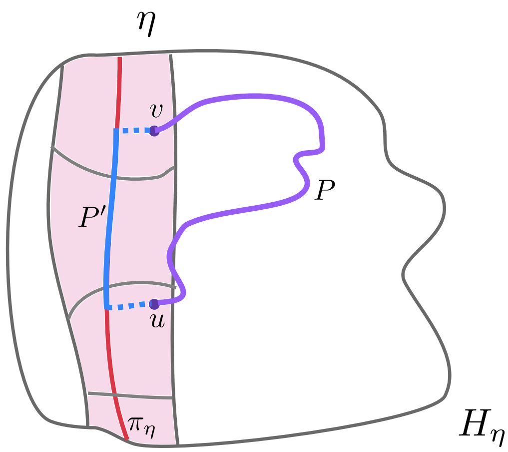

Figure 7. A partition of the column η into clusters, the graph H_η, and vertices u and v in η. The path P (in purple) is a shortest path between u and v in H_η. The path P' (in blue) described in the proof of Lemma 3.7; the subpath $P'[\pi_\eta]$ is solid, and the subpaths $P' \setminus P'[\pi_\eta]$ are dotted.

Cost of P'. First observe that, because of the way vertices not in π_η were assigned, we can walk from u to p_u (and from p_v to v) while remaining within a single cluster. This implies that $\text{cost}(P') = \text{cost}(P'[\pi_\eta])$. Because of the cluster ordering, path $P'[\pi_\eta]$ fully walks past every cluster that it intersects (other than C_u and C_v); that is, $\text{cost}(P'[\pi_\eta]) \le \frac{\|P'[\pi_\eta]\|}{\varepsilon\Delta} + 2$. Now observe that, because $P'[\pi_\eta]$ is a shortest path in H_η, we have $\|P'[\pi_\eta]\| \le \|P\| + 2t\varepsilon\Delta$. We conclude that $\text{cost}(P') \le \frac{\|P\|}{\varepsilon\Delta} + 4$.

Length of P'. By assumption u and v are not in adjacent clusters, so p_u and p_v are also not in adjacent clusters. This implies that $\|P'[\pi_\eta]\| \ge \varepsilon\Delta$. As every vertex is within distance $2t\varepsilon\Delta$ of π, we have $\|P\| \ge \|P'[\pi_\eta]\| - 4t\varepsilon\Delta$. Using the fact that $t \le 1/8$, we have $\|P\| \ge \varepsilon\Delta/2$, and we conclude that $\|P'\| \le \|P'[\pi_\eta]\| + 2t\varepsilon\Delta \le \|P\| + 4t\varepsilon\Delta \le (1+8t)\cdot\|P\|$.

Combining the two claims, we get that $\text{cost}_{8t}(u,v) \le \text{cost}(P') \le \frac{\|P\|}{\varepsilon\Delta} + 4$. □

3.2.2 Clustering a gridtree hierarchy

Given a planar graph G (with fixed parameters ε, t, and Δ) and a $(t\varepsilon\Delta)$-gridtree hierarchy $\mathcal{H}$ for G, we use the recursive algorithm ClusterHierarchy$(G, \mathcal{H})$ to create a shortcut partition of G. Let $\mathcal{C}$ be the resulting clustering. For every node $(H^\mu, \mathcal{T}^\mu)$ in $\mathcal{H}$, we assign clusters to each column η of $\mathcal{T}^\mu$ by calling ClusterColumn(η). Let $\mathcal{C}^\mu$ denote the set of clusters associated with columns of $\mathcal{T}^\mu$. We have

```
CLUSTERHIERARCHY(G, ℋ):
  ⟨⟨Cluster each column η⟩⟩
  𝒞 ← ∅
  for each node (H^μ, 𝒯^μ) in ℋ:
      for each column η of 𝒯^μ:
          𝒞 ← 𝒞 ∪ CLUSTERCOLUMN(η)
  return 𝒞
```

$\mathcal{C} = \bigcup_\mu \mathcal{C}^\mu$. We want to show that $\mathcal{C}$ is indeed a shortcut partition by proving each of the properties. The diameter and cluster ordering properties follow directly from the correctness of CLUSTERCOLUMN.

To prove the low-hop property that $\text{cost}_{8t}(u, v) \le f(\varepsilon)$ for any u and v, we will look at an arbitrary shortest path P between u and v: Find the highest node $(H^\mu, \mathcal{T}^\mu)$ in the hierarchy $\mathcal{H}$ such that H^μ contains a vertex in P. Chop up P into parts, alternating between parts that are covered by the clusters of the columns in $\mathcal{T}^\mu$ and parts that are in the leftover sets. Inductively the parts in the leftover sets have low costs; we just have to prove that the parts covered by $\mathcal{T}^\mu$ passes through at most $O\left(\frac{1}{t\varepsilon}\right)$ many columns in the worse case, using the column width property. We define the notation $P[u, v]$ to be the subpath of P that starts from vertex u and ends at vertex v for any path P.

Lemma 3.8. *Let $(H, \mathcal{T})$ be node in the hierarchy $\mathcal{H}$ that is not the root. Let u and v be two vertices in H such that u is an outer vertex, and H contains a shortest path P (with respect to G) between u and v. Then $\text{cost}_{8t}(u, v) \le 85 \cdot \frac{\|P\|}{t\varepsilon\Delta} + 16$.*

Proof: We proceed by induction on layers of nodes of $\mathcal{H}$. For the base case, we prove the claim when $(H, \mathcal{T})$ is a leaf in the deepest layer. In the inductive case, we assume the claim is true for all nodes in deeper layers. We begin with the inductive case.

Let η be the lowest column in $\mathcal{T}^\mu$ such that H_η contains P. Notice that η contains some vertex p of P (as otherwise, column adjacency property would imply that there is some child η' of η where $H_{\eta'}$ contains P). We split P into two paths: $P^{(u)}$, the subpath starting at p and ending at u; and $P^{(v)}$, the subpath starting at p and ending at v. (Notice that both $P^{(u)}$ and $P^{(v)}$ contain p and are therefore not vertex-disjoint; this is the only time in this section where we split a path into subpaths that are not vertex-disjoint.)

We chop $P^{(u)}$ into ℓ_u vertex-disjoint subpaths $P^{(u)} = P_1 \circ Q_1 \circ \ldots \circ P_{\ell_u} \circ Q_{\ell_u}$, where each P_i (except for P_1 and P_{ℓ_u}) *passes through* a column (as defined in the column width property), and each Q_i (possibly empty) is contained within a single leftover set. We do this as follows: Initialize $P^* \leftarrow P^{(u)}$ and $i \leftarrow 1$. Define η_i to be the column containing the first vertex of P^*. Define P_i to be the maximal prefix of P^* that ends at some vertex in η_i. Remove prefix P_i from P^*, and define Q_i to be the maximal prefix of P^* that touches no column. (Notice that Q_i may be empty.) Remove prefix Q_i from P^*, and increment i. (Notice that by maximality of Q_i, the first vertex on P^* is in some column.) The process terminates when P^* is empty. Using an identical process, we chop $P^{(v)}$ into ℓ_v vertex-disjoint subpaths $P^{(v)} = P_{\ell_u+1} \circ Q_{\ell_u+1} \circ \ldots \circ P_{\ell_u+\ell_v} \circ Q_{\ell_u+\ell_v}$; see Figure 8.

We now argue that there are few paths, i.e. $\ell_u + \ell_v = O\left(\frac{\|P\|}{t\varepsilon\Delta}\right)$. For every i, let (p_i, p_i') be the endpoints of P_i, and let (q_i, q_i') be the endpoints of Q_i if Q_i is nonempty. For every P_i that is a subpath of $P^{(u)}$, let P_i^+ denote the path P_i concatenated with the next vertex in $P^{(u)}$. That is, $P_i^+ := P^{(u)}[p_i, q_i]$ if q_i exists, and $P_i^+ := P^{(u)}[p_i, p_{i+1}]$ if q_i does not exist and $i \neq \ell_u$, and $P_i^+ := P_i$ if q_i does not exist and $i = \ell_u$. We similarly define P_i^+ for subpaths P_i of $P^{(v)}$, and define Q_i^+ for subpaths Q_i of $P^{(u)}$ and $P^{(v)}$. Notice that $\|P\| = \sum_i \left(\|P_i^+\| + \|Q_i^+\|\right)$, and that $\|P\| \ge \sum_i \left(\|P_i^+\| + \|Q_i\|\right)$.

By choice of η and p, the path P_1^+ is contained within $H_\eta = H_{\eta_1}$ (and a similar claim holds for $P_{\ell_u+1}^+$). Column adjacency property implies that path P_1^+ ends at a vertex below η_1 in $\mathcal{T}$ (unless $\ell_u = 1$). An easy

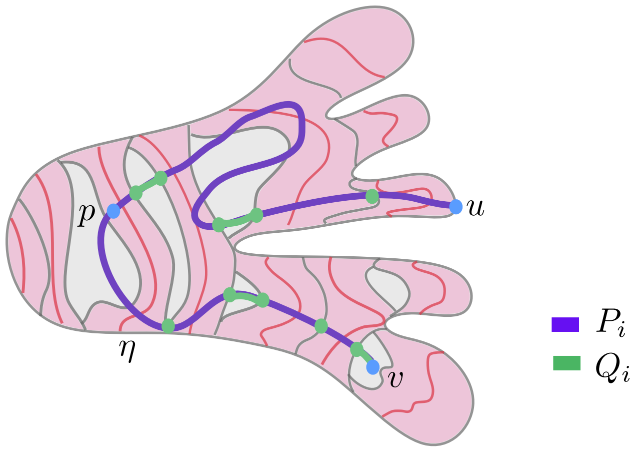

Figure 8. The path P from u to v, chopped into subpaths P_i and Q_i. The column η is the lowest node in the gridtree such that H_η contains P. Vertex p is in $\eta \cap P$.

inductive argument generalizes this statement: For all i, path P_i^+ is contained in H_{η_i}; if $i \notin \{\ell_u, \ell_u + \ell_v\}$, then the path P_i^+ ends at a vertex below η_i in $\mathcal{T}$; and if $i \notin \{1, \ell_u + 1\}$, then the subpath P_i^+ begins at a vertex above η. This implies that, if $i \notin \{1, \ell_u, \ell_u + 1, \ell_u + \ell_v\}$, the subpath P_i^+ *passes through* column η_i. By the column width property of the $(t\varepsilon\Delta)$-gridtree $\mathcal{T}$, every such P_i^+ has length at least $t\varepsilon\Delta$. We are now ready to bound $\text{cost}_{8t}(u,v)$. We can write $\text{cost}_{8t}(s,t)$ in terms of the costs of (the endpoints of) the subpaths P_i and Q_i:

$$\operatorname*{cost}_{8t}(u,v) \le \sum_i \operatorname*{cost}_{8t}(p_i, p_i') + \sum_{i:Q_i \ne \varnothing} \operatorname*{cost}_{8t}(q_i, q_i')$$

We bound the cost of P_is and Q_is separately.

- Path P_i is a shortest path in H_η, so Lemma 3.7 implies that $\text{cost}(p_i, p_i') \le \frac{\|P_i\|}{\varepsilon\Delta} + 4$. As $t \le 1$ and $\|P_i\| \le \|P_i^+\|$, we have $\text{cost}(p_i, p_i') \le \frac{\|P_i^+\|}{t\varepsilon\Delta} + 4$. If $i \notin \{1, \ell_u, \ell_u + 1, \ell_v + \ell_v\}$, we have $\|P_i^+\| \ge t\varepsilon\Delta$ and so $\text{cost}(p_i, p_i') \le 5 \cdot \frac{\|P_i^+\|}{t\varepsilon\Delta}$. Thus, we have

$$\sum_i \operatorname*{cost}_{8t}(p_i, p_i') \le 16 + 5 \cdot \sum_i \frac{\|P_i^+\|}{t\varepsilon\Delta}. \tag{1}$$

- Path Q_i lies in some component of a leftover set, which corresponds to a child μ in the hierarchy $\mathcal{H}$. The starting vertex q_i is an outer vertex of H^μ (as it is adjacent to a vertex in a column of $\mathcal{T}$), so induction hypothesis implies $\text{cost}_{8t}(q_i, q_i') \le 85 \cdot \frac{\|Q_i\|}{t\varepsilon\Delta} + 16$. If $i \notin \{1, \ell_u, \ell_u + 1, \ell_u + \ell_v\}$, subpath Q_i is accompanied by an occurrence of P_i^+ where $\|P_i\| \ge t\varepsilon\Delta$. For these i, we have $\text{cost}_{8t}(q_i, q_i') \le 85 \cdot \frac{\|Q_i\|}{t\varepsilon\Delta} + 16 \cdot \frac{\|P_i^+\|}{t\varepsilon\Delta}$. Thus, we have

$$\sum_{i:Q_i \ne \varnothing} \operatorname*{cost}_{8t}(q_i, q_i') \le 64 + \sum_{i:Q_i \ne \varnothing} \left(85 \cdot \frac{\|Q_i\|}{t\varepsilon\Delta} + 16 \cdot \frac{\|P_i^+\|}{t\varepsilon\Delta}\right). \tag{2}$$

We now combine the costs bounds for the P_is and Q_is. We consider two cases.

- If there are no nonempty Q_i, then we have

$$\operatorname*{cost}_{8t}(u,v) = \sum_i \operatorname*{cost}_{8t}(p_i, p_i) \le 6 \cdot \frac{\|P\|}{t\varepsilon\Delta} + 16$$

  and the claim is satisfied.

- If there is some nonempty Q_i, then the layer width property implies that $\sum_i \|P_i^+\| \geq t\varepsilon\Delta$, as every vertex within $t\varepsilon\Delta$ distance of u is assigned to some column. We have

$$\begin{aligned}
\underset{8t}{\text{cost}}(u,v) &= \sum_i \underset{8t}{\text{cost}}(p_i,p_i') + \sum_{i:Q_i\neq\varnothing} \underset{8t}{\text{cost}}(q_i,q_i') \\
&\leq 21\cdot\sum_i \frac{\|P_i^+\|}{t\varepsilon\Delta} + 85\cdot\sum_i \frac{\|Q_i\|}{t\varepsilon\Delta} + 80 && \text{by Equations (1) and (2)} \\
&\leq 21\cdot\frac{\sum_i \|P_i^+\|}{t\varepsilon\Delta} + 85\cdot\frac{\|P\| - \sum_i \|P_i^+\|}{t\varepsilon\Delta} + 80 \\
&\leq 85\cdot\frac{\|P\|}{t\varepsilon\Delta} + 16.
\end{aligned}$$

The last inequality holds because $\frac{\sum_i \|P_i^+\|}{t\varepsilon\Delta} \geq 1$.

This completes the inductive case. The base case, when $(H,\mathcal{T})$ is a leaf at the deepest layer of $\mathcal{H}$, is identical except that all Q_i are empty and so there is no need to appeal to the inductive hypothesis. $\square$

Lemma 3.9. *Let u and v be two vertices in G (where u is not necessarily an outer vertex), and let P be a shortest path (with respect to G) between u and v. Then $\text{cost}_{8t}(u,v) \leq 85\cdot\frac{\|P\|}{t\varepsilon\Delta} + 80$.*

Proof: Let $(H,\mathcal{T})$ be the lowest node in the hierarchy $\mathcal{H}$ such that H fully contains P. There is some vertex along P in a column of $\mathcal{T}$. This means that, following Lemma 3.8, we can chop P into subpaths $P_1 \circ Q_1 \circ \ldots \circ P_{\ell_1+\ell_2} \circ Q_{\ell_1+\ell_2}$. Letting (p_i,p_i') and (q_i,q_i') denote the endpoints of P_i and Q_i, respectively, we have

$$\underset{8t}{\text{cost}}(u,v) \leq \sum_i \underset{8t}{\text{cost}}(p_i,p_i') + \sum_{i:Q_i\neq\varnothing} \underset{8t}{\text{cost}}(q_i,q_i').$$

To bound $\text{cost}_{8t}(p_i,p_i')$, we may use identical analysis to that in the proof of Lemma 3.8 to show that Equation (1) still holds. To bound $\text{cost}_{8t}(q_i,q_i')$, we follow a similar proof to that of Equation (2). There is one minor change. To prove that $\text{cost}_{8t}(Q_i) \leq 85\cdot\frac{\|Q_i\|}{t\varepsilon\Delta} + 16$, the earlier proof of Equation (2) appealed to an inductive hypothesis. We no longer have this inductive hypothesis; instead, we can directly apply the statement of Lemma 3.8. The rest of the proof follows verbatim, and so Equation (2) holds in this setting. Combining these bounds, we conclude:

$$\underset{8t}{\text{cost}}(u,v) \leq 21\cdot\sum_i \frac{\|P_i^+\|}{t\varepsilon\Delta} + 85\cdot\sum_i \frac{\|Q_i\|}{t\varepsilon\Delta} + 80 \leq 85\cdot\frac{\|P\|}{t\varepsilon\Delta} + 80.$$

$\square$

Lemmas 3.7 and 3.9 imply the following theorem.

Theorem 3.10. *Let ε be a number in $(0,1)$, let t be a number in $(0,1/8]$, and let G be a graph with diameter Δ that has a $(t\varepsilon\Delta)$-gridtree hierarchy $\mathcal{H}$. Then the set $\mathcal{C}$ of clusters produced by* ClusterHierarchy(G) *satisfies the following: (i) each cluster in $\mathcal{C}$ has strong diameter $O(\varepsilon\Delta)$, and (ii) for any $u,v\in V(G)$, we have $\text{cost}_{8t,\mathcal{C}}(u,v) = O\left(\frac{1}{t\varepsilon}\right)\cdot\frac{\delta_G(u,v)}{\Delta} + O(1)$.*

In particular, planar graphs have $(t\varepsilon\Delta)$-gridtree hierarchies for any $t,\varepsilon\in(0,1)$ (by Lemma 3.4), and so they have $(O(\varepsilon), O\left(\frac{1}{t\varepsilon}\right))$-shortcut partitions with $(1+O(t))$ distortion. By choosing $t = O(\varepsilon)$ or choosing $t = O(1)$, we conclude:

Corollary 3.11. *Any planar graph has an $(\varepsilon, O(\varepsilon^{-2}))$-shortcut partition with $(1+O(\varepsilon))$ distortion, satisfying the diameter, low-hop, and cluster ordering properties.*

Corollary 3.12. *Any planar graph has an $(\varepsilon, O(\varepsilon^{-1}))$-shortcut partition with t distortion for any constant $t > 1$, satisfying the diameter, low-hop, and cluster ordering properties.*

4 Constructing gridtree hierarchy for planar graphs

In this section we prove the existence of an $\varepsilon\Delta$-gridtree hierarchy $\mathcal{H}$ for any planar graph G (Lemma 3.4). Throughout this section, we assume that G has a fixed drawing in the plane; all references to the external face of (a subgraph of) G refer to this fixed drawing. All subgraphs of G inherit the drawing of G.

To construct a gridtree hierarchy for G, we proceed in rounds. In each round we will construct a gridtree $\mathcal{T}$ for a subgraph H of G using the algorithm GRIDTREE(H), iteratively partitioning the vertices in H into columns and leftover sets, such that every vertex within distance $\varepsilon\Delta$ from the exterior face of H is assigned to a column. We then recursively construct a gridtree hierarchy on each leftover component. Within a single round, we adapt an algorithm by Busch, LaFortune, and Tirthapura for constructing a sparse cover of a planar graph [BLT14]. Their algorithm crucially relies on recursively selecting a set of "far apart" paths. We follow the same algorithm for selecting paths, and define the columns to be $O(\varepsilon\Delta)$-neighborhoods around each path.

4.1 Constructing a single gridtree

We begin by defining the algorithm SELECTPATHS$_H(H', \pi)$, which appeared implicitly in [BLT14]. It takes as input a subgraph H' of H (which itself is a subgraph of G) and a path π that is a shortest path on H' between external vertices of H. (We allow π to be a path of length 0 — that is, π may be a single external vertex of H.)

```
SELECTPATHS_H(H', π) :
  ⟨⟨Cut away εΔ-neighborhood of π⟩⟩
  N ← εΔ-neighborhood of π in H'
  H'_1, ..., H'_m ← components of H' \ N containing at least an external vertex of H

  ⟨⟨Recurse⟩⟩
  for each H'_i ← H'_1, ..., H'_m:
      Y_i ← set of external vertices of H'_i connected to N by some external edge
      ⟨⟨by [BLT14], 1 ≤ |Y_i| ≤ 2⟩⟩
      if |Y_i| = 1:
          π_i ← Y_i
      else:
          π_i ← shortest path in H'_i connecting both vertices in Y_i
      SELECTPATHS_H(H'_i, π_i)
```

We initialize the process by selecting an arbitrary external vertex π_0 of H, and call SELECTPATHS$_H(H, \pi_0)$. There is a tree $\mathcal{T}^*$ naturally associated with the recursion of SELECTPATHS$_H(H, \pi_0)$. Each node η of $\mathcal{T}^*$ contains the following:

- An induced subgraph $H^*_\eta \subseteq H$, containing at least one external vertex of G. We say that a vertex v *appears* in the subgraph of η if $v \in V(H^*_\eta)$.
- A path π_η which is a shortest path on H^*_η.
- A neighborhood N_η defined to be the set of vertices $v \in V(H^*_\eta)$ where $\delta_{H^*_\eta}(v, \pi_\eta) \leq \varepsilon\Delta$
- A "leftover" subgraph L_η, defined to be the union of all connected components of $H^*_\eta \setminus N_\eta$ that do *not* contain any external vertices of H.

The children of η are associated with components of $V(H^*_\eta) \setminus (V(L_\eta) \cup N_\eta)$. This recursion tree was discussed in [BLT14], where the authors implicitly prove the following lemma (cf. [BLT14, Lemma 9]):

Lemma 4.1. *Let η be a node of the recursion tree $\mathcal{T}^*$. Let u be a vertex in H^*_η that is adjacent in H to some vertex in $V(H^*_{\mathrm{Pa}(\eta)})$, where $\mathrm{Pa}(\eta)$ is the parent of η in $\mathcal{T}^*$. Let $v \in V(H^*_\eta) \setminus (V(L_\eta) \cup N_\eta)$ be a vertex that appears in the subgraph of some child of η. Then every path P in H^*_η between u and v intersects π_η.*

We can transform the recursion tree $\mathcal{T}^*$ into a gridtree $\mathcal{T}$ for H. The columns (resp. leftover sets) of $\mathcal{T}$ are in one-to-one correspondence with the nodes (resp. edges) of $\mathcal{T}^*$: the column associated with node $\eta \in V(\mathcal{T}^*)$ is N_η, and the leftover set associated with the edge $(\eta, \eta') \in E(\mathcal{T}^*)$ is $L_{\eta'}$. In the proofs below, we will slightly abuse the notation and use η to refer simultaneously to the column of $\mathcal{T}$ and to the corresponding node of $\mathcal{T}^*$; for example, the column η denotes the same set as N_η. Notice that the subgraph H_η defined in the column shortcut property (Definition 3.1) is the same as H^*_η.

Lemma 4.2. *Tree $\mathcal{T}$ is an $\varepsilon\Delta$-gridtree for H, and every external vertex of H is assigned to some column.*

Proof: We prove that $\mathcal{T}$ satisfies the three required properties.

[Column adjacency.] Let (u, v) be an edge in H. Let η_u and η_v be the nodes of $\mathcal{T}^*$ such that $u \in N_{\eta_u} \cup L_{\eta_u}$ and $v \in N_{\eta_v} \cup L_{\eta_v}$. If $\eta_u = \eta_v$, then the edge (u, v) satisfies the column adjacency property. Otherwise, notice that η_u and η_v are in an ancestor-descendent relationship, as SELECTPATHS recurses on (maximal) connected components in each iteration. Assume without loss of generality that η_u is an ancestor of η_v. For the sake of simplifying notation, we write $\eta := \eta_u$. Let η' denote the child of η such that v appears in the subgraph of η'. We claim that v does not appear in the subgraph of any child of η'. Indeed, if it did, then Lemma 4.1 would imply that the edge (u, v) is incident to $\pi_{\eta'}$; as $u \notin V(H^*_{\eta'})$, this implies that $v \in \pi_{\eta'} \subseteq N_{\eta'}$ and thus does not appear in the subgraph of any child of η'. We conclude that $v \in L_{\eta'} \cup N_{\eta'}$, and so (u, v) satisfies the column adjacency property.

[Column width.] Let η be a column in $\mathcal{T}$. Let P be a path that passes through η. Let P' denote the longest suffix of P that is fully contained in H_η. Because $V(H_\eta)$ includes all vertices below η in $\mathcal{T}$, and P starts at a vertex that is adjacent to a vertex above η, the path P' starts at a vertex that is adjacent to a vertex above η. Column adjacency property implies that P' starts at a vertex that adjacent to some vertex in $V(H^*_{\mathrm{Pa}(\eta)})$. By Lemma 4.1, the path P' intersects π_η. As every vertex in $V(H^*_\eta) \setminus (V(L_\eta) \cup N_\eta)$ has distance (with respect to H^*_η) at least $\varepsilon\Delta$ from π_η, this implies that P has length at least $\varepsilon\Delta$.

[Column shortcut.] Let η be a column in $\mathcal{T}$, defined to be the vertex set N_η from the associated node of $\mathcal{T}^*$. By construction, every vertex in N_η is within distance $\varepsilon\Delta$ (with respect to $H[N_\eta]$) of the path π_η, and π_η is a shortest path with respect to H_η. $\square$

Notice that the definition of an $\varepsilon\Delta$-gridtree (column shortcut property) only requires that each vertex v in a column η is within distance $2 \cdot \varepsilon\Delta$ of the path π_η; however, the proof above gives a better guarantee on the distance.

Observation 4.3. *Let v be a vertex in a column η in the $\varepsilon\Delta$-gridtree $\mathcal{T}$, and let π_η be the path in η guaranteed by the column shortcut property. Then v is within distance $\varepsilon\Delta$ (with respect to η) of π_η.*

The gridtree $\mathcal{T}$ does *not* guarantee that every vertex within $\varepsilon\Delta$ distance of the external face is assigned to a column. It only guarantees that every vertex on the external face itself is assigned to a column. To fix this, we simply assign every vertex within $\varepsilon\Delta$ distance of the external face to the closest column in $\mathcal{T}$. We describe this approach in the GRIDTREE(H) algorithm, which takes as input a planar graph H.

Claim 4.4. *Let v be a vertex in $V(H)$. If v is in a column η in $\mathcal{T}$, then v is in the associated column η^+ in $\mathcal{T}^+$. If v is in some leftover set (η_1, η_2) in $\mathcal{T}$, then in $\mathcal{T}^+$ the vertex v is either in the column η_1^+, the column η_2^+, or the leftover set (η_1^+, η_2^+).*

```
GridTree(H):
  𝒯 ← gridtree associated with SelectPaths(G)

  ⟨⟨Select vertices to add to columns⟩⟩
  X ← ∅
  for each column η of 𝒯:
      X ← X ∪ {v ∈ V(H) | δ_H(v, π_η) ≤ 2εΔ}

  ⟨⟨Assign vertices to closest η⟩⟩
  initialize η⁺ ← η for each column η ∈ V(𝒯)
  for each vertex v in X:
      η_v ← arg min_{η∈V(𝒯)} δ(v, η)
      (breaking ties based on a fixed ordering of V(𝒯))
      η_v⁺ ← η_v⁺ ∪ {v}

  ⟨⟨Define the gridtree⟩⟩
  𝒯⁺ ← 𝒯
  assign each node η ∈ V(𝒯⁺) to the column η⁺
  assign each edge (η, η′) ∈ E(𝒯⁺) to the leftover set (η, η′) ∈ E(𝒯), excluding the vertices in X
  return 𝒯⁺
```

Proof: If v is in a column of $\mathcal{T}$, the claim is immediate from the construction. Suppose v is in a leftover set (η_1, η_2) of $\mathcal{T}$. If v is not in the set X during the call to GRIDTREE(H), then v stays in the associated leftover set (η_1^+, η_2^+) in $\mathcal{T}^+$. Otherwise, v is assigned to column η^+ in $\mathcal{T}^+$, where η is the the closest column in $\mathcal{T}$ to v. Let P be the shortest path from v to η. By the column adjacency property of $\mathcal{T}$, the first vertex on P that leaves the leftover set (η_1, η_2) is either in η_1 or η_2. Thus, the closest column to v is either η_1 or η_2. □

The following lemma is a restatement of Lemma 3.2.

Lemma 4.5. *The tree $\mathcal{T}^+$ returned by* GRIDTREE(H) *is a gridtree for H, in which each vertex within distance $\varepsilon\Delta$ of an external vertex of H is assigned to a column.*

Proof: As every external vertex in H is assigned to a column in $\mathcal{T}$, and (by Observation 4.3) every such vertex is within distance $\varepsilon\Delta$ of a path π_η, triangle inequality implies that every vertex v within distance $\varepsilon\Delta$ of some external vertex is within distance $2\varepsilon\Delta$ of some π_η. Thus, v is assigned to a column in $\mathcal{T}^+$. We now prove that $\mathcal{T}^+$ is a gridtree.

[Column adjacency.] Let $(u, v) \in E(H)$. If both u and v were in columns of $\mathcal{T}$, then Lemma 4.2 implies that u and v are in adjacent columns of $\mathcal{T}$. Claim 4.4 implies that u and v are also in adjacency columns of $\mathcal{T}^+$. Otherwise, suppose that u is in some leftover set associated with the edge (η_1, η_2) of $\mathcal{T}$. By Lemma 4.2, the vertex v is either associated with the leftover set (η_1, η_2) in $\mathcal{T}$, or with the columns η_1 or η_2 in $\mathcal{T}$. By Claim 4.4, the vertices u and v are associated with (η_1^+, η_2^+) or η_1^+ or η_2^+ in $\mathcal{T}^+$. In each of these cases, the edge (u, v) satisfied the column adjacency property in $\mathcal{T}$.

[Column width.] Let η^+ be a column in $\mathcal{T}^+$, corresponding to column η in $\mathcal{T}$. Let P be a path that passes through η^+. Notice that every vertex above (resp. below) η^+ in $\mathcal{T}^+$ is above (resp. below) η in $\mathcal{T}$: this follows from Claim 4.4. Thus, P passes through η in $\mathcal{T}$, and so the column width property of $\mathcal{T}$ implies that it has length at least $\varepsilon\Delta$.

[Column shortcut.] Let η^+ be a column in $\mathcal{T}^+$, corresponding to column η in $\mathcal{T}$. Every vertex v in η^+ is within $2\varepsilon\Delta$ distance of π_η. Because GRIDTREE(H) breaks ties based on a fixed ordering of $V(\mathcal{T})$ when assigning vertices to columns, every vertex in the shortest path between v and π_η is in η^+. We now claim π_η is a shortest path with respect to H_{η^+}. Indeed, Claim 4.4 implies that columns and leftover sets are below η in $\mathcal{T}$ if and only if they are below η^+ in $\mathcal{T}$, from which we can conclude that H_η is a superset of H_{η^+}. By column shortcut property for $\mathcal{T}$, path π_η is a shortest path with respect to H_η, and thus is a shortest path with respect to H_{η^+}. □

4.2 Constructing a gridtree hierarchy

We summarize our multi-round strategy with the recursive algorithm GridtreeHierarchy(G).

```
GridtreeHierarchy(G):
  T ← Gridtree(G)
  set (G, T) to be the root of the hierarchy H
  for each connected component H in a leftover set of T:
      H_H ← GridtreeHierarchy(H)
      attach the root of H_H as a child of the node (G, T) in H
  return H
```

Proof (of Lemma 3.4): We prove that GridtreeHierarchy(G) returns a gridtree hierarchy for G. The layer nesting property is immediate from the construction. We now prove the layer width property. Let $\mu = (H^\mu, \mathcal{T}^\mu)$ be some node in the hierarchy, and let $\mu' = (H^{\mu'}, \mathcal{T}^{\mu'})$ be the parent of μ. If a vertex v in H is adjacent to some vertex u in $H^{\mu'}$, then we claim that v is an external vertex of H^μ. Indeed, the vertices in columns of $\mathcal{T}^{\mu'}$ induce a connected component that includes an external vertex of $H^{\mu'}$. When an external vertex of a planar graph is removed, its neighbors becomes external vertices of the resulting graph. The graph H^μ is a connected component of the graph obtained by removing all columns of $\mathcal{T}^{\mu'}$. We conclude that v is an external vertex of H^μ. □

5 Constructing tree cover from grid-like clustering

Before describing the construction, we prove two lemmas about gridtree hierarchies, which are easy consequences of the layer width and column width properties.

Lemma 5.1. *Any $\varepsilon\Delta$-gridtree hierarchy $\mathcal{H}$ for a graph G with diameter Δ has depth $O(1/\varepsilon)$.*

Proof: We prove by induction that for any node μ of $\mathcal{H}$, every vertex v that has distance at most $\alpha \cdot \varepsilon\Delta$ to an outer vertex of μ belongs to some column of a gridtree at most α layers deeper than μ. To prove the lemma, we substitute $\alpha = 1/\varepsilon$; as G has diameter Δ, every vertex has distance at most Δ from an outer vertex on layer 2 (recall that the root node at layer 1 has no outer vertices). We conclude that gridtree hierarchy $\mathcal{H}$ with depth $O(1/\varepsilon)$ much have every vertex assigned to some column of a gridtree in $\mathcal{H}$.

Consider an arbitrary vertex v in μ and a shortest path π from v to the exterior of μ, whose length is at most $\alpha \cdot \varepsilon\Delta$. By layer width property (guaranteed by Lemma 3.2), every vertex in the prefix of π that has distance at most $\varepsilon\Delta$ to an outer vertex of μ belongs to some column of the gridtree $\mathcal{T}^\mu$. If v belongs to some column of $\mathcal{T}^\mu$ as well then we are done. Otherwise, the suffix of π that lies completely in a leftover component L containing v has length at most $(\alpha - 1) \cdot \varepsilon\Delta$. By induction, v belongs to some column of a gridtree $\alpha - 1$ layers deeper than L. This shows that v belongs to a column of some gridtree at most α layers from μ. □

Claim 5.2. *Let η and η' be two columns in a gridtree $\mathcal{T}$ of a graph H. Suppose that η is an ancestor of η', and that the path between η and η' in $\mathcal{T}$ consists of m columns $\eta_1, \ldots, \eta_m$ strictly between η and η'. Then any path P in H between a vertex u in η and a vertex v in $H_{\eta'}$ has length at least $m \cdot \varepsilon\Delta$.*

Proof: For ease of notation, we refer to η as η_0 and refer to η' as η_{m+1}. By the column adjacency property, P touches at least one vertex in each of $\eta_1, \ldots, \eta_{m+1}$. For every $i \in \{1, \ldots, m+1\}$, let p_i be the first vertex in P that is in column η_i, as P travels from u to v. The path P can be written as a concatenation of subpaths $P[u, p_1] \circ P[p_1, p_2] \circ \ldots \circ P[p_{m+1}, v]$. Column adjacency property implies that every p_i is adjacent (in H) to a vertex above η_i in $\mathcal{T}$, and p_{i+1} is a vertex below η_i in $\mathcal{T}$. From the column width property, we conclude that each of the m subpaths $P[p_i, p_{i+1}]$ has length at least $\varepsilon\Delta$. □

5.1 Construction

Let G be a graph with an $\varepsilon\Delta$-gridtree hierarchy $\mathcal{H}$. Recall from Section 3.2 that G has an $(O(\varepsilon), O(\varepsilon^{-1}))$-shortcut partition $\mathcal{C}$ with $O(1)$ distortion, associated with $\mathcal{H}$, where the clusters satisfy the following properties:

- [Diameter.] There exists some constant $\gamma \geq 1$ such that $\delta_C(u, v) \leq \gamma \cdot \varepsilon\Delta$ for any u and v in the same cluster C.
- [Low-hop.] $\text{cost}_{O(1),\mathcal{C}}(u, v) \leq O(\varepsilon^{-1})$ for any u and v in G.
- [Cluster ordering.] For every node $(H, \mathcal{T})$ in the hierarchy $\mathcal{H}$ and for every column η in $\mathcal{T}$, there is an ordered set of *cluster centers* $c_1, \ldots, c_m \in \eta$, one from each cluster containing vertices of η, such that for every pair $i, j \in \{1, \ldots, m\}$, we have $d_{H_\eta}(c_i, c_j) \geq |i - j| \cdot \varepsilon\Delta$.

We use partition $\mathcal{C}$ to build a forest cover for G. To construct the forest cover, we only use the cluster ordering property. In a later section (cf. Section 7), we show that G can be embedded into a bounded-treewidth graph; this proof will use the fact that a forest cover for G can be built using a partition with the diameter and low-hop properties. In this section, we use the cluster centers associated with $\mathcal{C}$ in an algorithm CoverGridtree$(H, \mathcal{T})$ (see Figure 9), which takes as input a node $(H, \mathcal{T})$ from the hierarchy $\mathcal{H}$, where H is a subgraph and $\mathcal{T}$ is a gridtree on H. We show that this procedure returns a set of $O(\varepsilon^{-2})$ forests (cf. Lemma 5.3), and that those forests that preserve distances for vertices in columns of $\mathcal{T}$ (implicit in Lemma 5.6).

Lemma 5.3. *The procedure* CoverGridtree$(H, \mathcal{T})$ *returns a set of κ spanning forests on H, for some $\kappa = O(\varepsilon^{-2})$.*

Proof: From the procedure CoverGridtree, F_j^η is the union of spanning trees in the set $\{T^\eta_{j+k\cdot(3\gamma+2)\varepsilon^{-1}} :$ integer $k\}$. We first show that, for every column η in $\mathcal{T}$, the graph F_j^η is a spanning forest of H. Let T_{i_1} and T_{i_2} be two trees from the set, with $T_{i_1} \neq T_{i_2}$. We want to prove T_{i_1} and T_{i_2} are disjoint. Let c_{i_1} and c_{i_2} denote the cluster centers that are roots of T_{i_1} and T_{i_2}, respectively. If T_{i_1} and T_{i_2} were not disjoint, then there would be some vertex v in $T_{i_1} \cap T_{i_2}$ such that $\delta_{H_\eta}(c_{i_1}, v) \leq (\gamma+1)\Delta$ and $\delta_{H_\eta}(c_{i_2}, v) \leq (\gamma+1)\Delta$. But the choice of cluster centers guarantees that c_{i_1} and c_{i_2} are at distance at least $|i_1 - i_2| \cdot \varepsilon\Delta > (3\gamma+2)\Delta$ with respect to H_η. Thus, there are no vertices in both T_{i_1} and T_{i_2}.

Forest F_j^ℓ is the union of each spanning forest F_j^η over the nodes η whose level $\pmod{(\gamma+2)\varepsilon^{-1}}$ in $\mathcal{T}$ is equal to ℓ, defined in the procedure CoverGridtree. We now show that, for each level ℓ in the gridtree, the graph F_j^ℓ is a spanning forest. Let η_1 and η_2 be two such nodes. We want to prove that $F_j^{\eta_1}$ and $F_j^{\eta_2}$ are disjoint. If η_1 and η_2 are not in an ancestor-descendent relationship, then $V(H_{\eta_1})$ and $V(H_{\eta_2})$ are disjoint and so the claim holds. Otherwise, suppose without loss of generality that η_1

```
COVERGRIDTREE(H, 𝒯):
  ⟨⟨Create forests for each column⟩⟩
  for each column η in 𝒯:
      c_1, ..., c_m ← cluster centers for η from the shortcut partition 𝒞
      for each c_i ← c_1, ..., c_m:
          B_i ← ball of radius (γ+1)Δ with respect to H_η, centered at c_i
          T_i^η ← SSSP tree (w.r.t. H_η) of B_i rooted at c_i
      for j ← 0, ..., (3γ+2)ε^{-1} − 1:
          F_j^η ← forest containing trees T^η_{j+k·(3γ+2)ε^{-1}} for every integer k
          ⟨⟨Note: If there are fewer than j portals on π_η, then F_j^η = ∅⟩⟩

  ⟨⟨Combine disjoint forests from columns that are far apart⟩⟩
  for each ℓ ← 0, ..., (γ+2)ε^{-1} − 1:
      for each j ← 0, ...(3γ+2)ε^{-1} − 1:
          F_j^ℓ ← ⋃_η F_j^η over column η whose level equals to ℓ (mod (γ+2)ε^{-1}) in 𝒯

  return {F_j^ℓ} where ℓ ∈ [(γ+2)ε^{-1}] and j ∈ [(3γ+2)ε^{-1}]
```

Figure 9. Algorithm CoverGridtree$(H, \mathcal{T})$ constructs a forest cover for vertices in columns of $\mathcal{T}$.

is an ancestor of η_2. There are at least $(\gamma+2)\varepsilon^{-1}-1$ columns separating η_1 and η_2 in the gridtree $\mathcal{T}$. By Claim 5.2, every vertex in η_1 is at least $((\gamma+2)\varepsilon^{-1}-1)\cdot\varepsilon\Delta > (\gamma+1)\Delta$ distance away from the closest vertex in H_{η_2}. As every vertex in $F_j^{\eta_1}$ is within distance $(\gamma+1)\Delta$ of some vertex in η_1, the forests $F_j^{\eta_1}$ and $F_j^{\eta_2}$ are disjoint. Thus, each of the $(3\gamma+2)(\gamma+2)\varepsilon^{-2} = O(\varepsilon^{-2})$ graphs F_j^ℓ returned by CoverGridtree$(\mathcal{T})$ is a forest. □

As a brief aside, we can in fact prove something slightly stronger. We say that a two (spanning) trees are *cluster-disjoint* with respect to $\mathcal{C}$ if no cluster in $\mathcal{C}$ intersects nontrivially with the two trees. The following lemma will not be used in the proof of $O_\varepsilon(1)$-size tree cover, but it will be helpful in Section 7, when we embed any planar graph into a bounded-treewidth graph.

Lemma 5.4. *Every forest returned by* CoverGridtree$(H, \mathcal{T})$ *contains trees that are pairwise cluster-disjoint with respect to* $\mathcal{C}$.

Proof: The proof follows that of Lemma 5.3 almost verbatim. We use one additional fact: By the construction of clusters based on the ClusterGridtree algorithm from Section 3.2.2, every vertex in a subgraph H_η belongs to a cluster that is fully contained within H_η.

We first prove that two trees T_{i_1} and T_{i_2} in a forest F_j^η are cluster-disjoint. This follows from the fact that each cluster has strong diameter $\gamma\varepsilon\Delta < \gamma\Delta$: if there was some cluster $C \subseteq H_\eta$ containing both a vertex v_1 in T_{i_1} and a vertex v_2 in T_{i_2}, then triangle inequality would imply that $\delta_{H_\eta}(c_{i_1}, c_{i_2}) \leq \delta_{H_\eta}(c_{i_1}, v_1) + \delta_C(v_1, v_2) + \delta_{H_\eta}(v_2, c_{i_2}) \leq (2(\gamma+1)+\gamma\varepsilon)\Delta < (3\gamma+2)\Delta$, a contradiction.

We then prove that two forests $F_j^{\eta_1}$ and $F_j^{\eta_2}$ in a forest F_j^i are cluster-disjoint. If there was some cluster containing both a vertex in $F_j^{\eta_1}$ and $F_j^{\eta_2}$, then that cluster must be entirely in H_{η_2} — but we proved above that $F_j^{\eta_1}$ contains no vertices in H_{η_2}. □

We now construct a tree cover for G by repeatedly apply CoverGridtree to the gridtrees in each layer of the gridtree hierarchy $\mathcal{H}$. The procedure CoverHierarchy$(G, \mathcal{H})$ takes as input a depth-d gridtree hierarchy $\mathcal{H}$ of the graph G and returns a forest cover of size $O(d\cdot\varepsilon^{-2})$.

Lemma 5.5. *Let G be a graph with a gridtree hierarchy $\mathcal{H}$. Then the algorithm* CoverHierarchy$(G, \mathcal{H})$ *returns a set of $O(d\cdot\varepsilon^{-2})$ spanning forests on G. Further, the trees in any forest are pairwise cluster-disjoint with respect to $\mathcal{C}$.*

```
CoverHierarchy(G, ℋ)
  d ← depth of gridtree hierarchy ℋ
  for each layer λ ← 1, ..., d of ℋ:
      for each node (H^μ, 𝒯^μ) in layer λ of ℋ:
          ⟨⟨κ is the constant from Lemma 5.3⟩⟩
          F_1^μ, ... F_κ^μ ← CoverGridtree(H^μ, 𝒯^μ)
      for each index j ← 1, ..., κ:
          F_j^λ ← ⋃_{μ at layer λ} F_j^μ
  return the set {F_j^λ} for λ ∈ [d] and j ∈ [κ]
```

Proof: For every layer λ and index j, the graph F_j^λ is the union of forests F_j^μ over all nodes $(H^\mu, \mathcal{T}^\mu)$ in layer λ of the gridtree hierarchy $\mathcal{H}$. By Lemma 5.3, each F_j^μ is a spanning forest of H^μ. For every H^μ in a single layer λ, the sets $V(H^\mu)$ are disjoint, so F_j^λ is the disjoint union of forests. Further, every vertex in H^μ belongs to a cluster entirely within H^μ, so the trees in F_j^λ are pairwise cluster-disjoint. There are $d \cdot \kappa = O(d \cdot \varepsilon^{-2})$ such forests F_j^λ. □

Lemma 5.6. *Let G be a graph with diameter Δ and a gridtree hierarchy $\mathcal{H}$. Then for every pair of vertices (u, v) in G, there is some forest F returned by the* CoverHierarchy$(G, \mathcal{H})$ *such that $\delta_F(u, v) \le \delta_G(u, v) + O(\varepsilon\Delta)$.*

Proof: Let P be a shortest path between u and v in G. Let $(H, \mathcal{T})$ be the lowest node of $\mathcal{H}$ such that H contains all vertices of P. Note that this implies $\delta_H(u, v) = \delta_G(u, v)$.

Let η be the highest column in $\mathcal{T}$ that contains a vertex in P. Let p be a vertex in $P \cap \eta$. There is some cluster center c_i in η with $\delta_{H_\eta}(c_i, p) \le \gamma\varepsilon\Delta$. We now claim that H_η contains all vertices of P: Indeed, the column adjacency property of $\mathcal{T}$ implies that vertices of H_η are only incident other vertices in H_η and to vertices in $\mathrm{Pa}(\eta)$, and the choice of η implies that P contains no vertices in $\mathrm{Pa}(\eta)$. Thus, we have $\delta_{H_\eta}(u, c_i) \le \delta_G(u, p) + \gamma\varepsilon\Delta < (\gamma + 1)\Delta$ and $\delta_{H_\eta}(c_i, v) \le \delta_G(p, v) + \gamma\varepsilon\Delta < (\gamma + 1)\Delta$. The algorithm CoverGridtree$(\mathcal{T})$ returns a forest F containing an SSSP tree connecting root c_i to every vertex in H_η within distance $(\gamma + 1)\Delta$ of c_i. This forest satisfies $\delta_F(u, v) \le \delta_G(u, v) + 2\gamma\varepsilon\Delta$. □

Observation 5.7. *Every tree in a forest returned by* CoverGridtree$(H, \mathcal{T})$ *is an SSSP tree with radius $(\gamma + 1)\Delta$. Thus, every tree has diameter $O(\Delta)$.*

Together with the fact that $\varepsilon\Delta$-gridtree hierarchy $\mathcal{H}$ of G has depth $O(\varepsilon^{-1})$ from Lemma 5.1, we conclude:

Theorem 5.8. *Every planar graph G with diameter Δ has an $O(\Delta)$-bounded spanning forest cover of size $O(\varepsilon^{-3})$, with additive distortion $+O(\varepsilon\Delta)$.*

Combining with the multiplicative-to-additive distortion reduction (Lemma 1.7), this proves Theorem 1.2.

6 A shortcut partition for bounded treewidth graphs

In this section we present an algorithm for creating an $(\varepsilon, O_\varepsilon(1))$-shortcut partition for graphs with $O(1)$-treewidth. Our algorithm is inspired by the technique developed by Friedrich et al. [FIK+23] for solving the mulicut problem in bounded treewidth graphs.

6.1 Algorithm description

Let $\varepsilon \in (0,1)$ be a fixed constant. Our algorithm takes as input a graph G with diameter Δ, and its width-k tree decomposition T. It returns a set of clusters $\mathcal{C}'$ that form an $(\varepsilon, O_\varepsilon(1))$-shortcut partition.

In the preprocessing phase, we modify G and T so that the resulting graph has the same treewidth up to a small constant, but the resulting tree decomposition becomes more suitable for creating clusters; we abuse the notation and refer to the resulting graph and tree decomposition as G and T. First, for each bag in T we connect all its vertices by a clique in G. Note that each vertex v in G may appear in multiple bags of T; in every such bag, we create a fresh copy of v. In addition, for every pair of bags X and Y when both contain copies of v and X is a parent of Y, we add a 0-weight edge between the corresponding copies of v in G. This concludes the preprocessing phase.

Next, we build a set of "preliminary clusters" $\mathcal{C}$, which might not be vertex-disjoint. This is done in $k+1$ rounds, numbered from $k+1$ to 1 for technical convenience. In the ith round, we invoke a recursive procedure ClusteringRound that creates clusters. A recursive call of this procedure operates on the root bag R of a subtree T' (of the entire tree T) and grows balls of radius $\varepsilon\Delta$ centered at each of the unclustered vertices in R. Importantly, the ball is restricted to the vertices in the subtree T' (rather than the whole tree). Let B be the set of vertices clustered in this recursive call. The algorithm proceeds recursively with the set of subtrees obtained by removing from T' all the bags containing vertices in B.

As mentioned, the preliminary clusters in $\mathcal{C}$ created in the $k+1$ rounds of the aforementioned recursive procedure might not be vertex-disjoint. To achieve disjoint clusters, the algorithm creates a dummy graph G' by adding a vertex s and connecting it via weight-0 edge to all the centers in $\mathcal{C}'$. The clusters are then obtained by computing an SSSP tree τ_s of G' rooted at s. Specifically, for every child u of s, we let B'_u be the set of vertices in the subtree of τ_s rooted at u and add it to $\mathcal{C}'$. The output is $\mathcal{C}'$. Our algorithm is described in more detail in Figure 10.

6.2 Algorithm analysis

Observation 6.1. *Consider any call of* ClusteringRound$(T, G, i, \mathcal{C})$. *Let $S(B)$ be the set of bags in T containing a vertex from B. Then the subgraph of T induced by $S(B)$ is connected.*

Proof: We shall assume that the root bag R contains at least one unclustered vertex, otherwise the subgraph of T induced by $S(B)$ is a single node R. Recall that B is the union of balls B_u, taken over all centers u from the root bag R.

Consider any vertex u in R and the corresponding ball B_u. Since B_u is connected in G, we note that the bags containing vertices in B_u form a connected subtree of T. (This follows from the facts that (1) the bags containing any vertex v form a connected subtree of T, and (2) the bags containing vertices along a path between any pair (u, v) form a connected subtree of T.)

Taking union over all centers in R gives us a union of connected subtrees in T that all contain bag R in common. It follows that the union is connected. $\square$

Lemma 6.2. *The set $\mathcal{C}'$ of clusters returned by* Clustering(T, G) *form a partition of V. Moreover, every cluster is connected and has a strong diameter at most $O(\varepsilon\Delta)$.*

Proof: In every round i, the algorithm visits every bag X of T and either clusters all the unclustered vertices in it (when X is a round-i root bag) or clusters at least one vertex in it. It follows that after $k+1$ rounds every vertex becomes clustered. The set of resulting clusters in $\mathcal{C}$ (each of which is a ball B_u) is a cover of V rather than a partition. After the last round in Clustering, the algorithm transforms $\mathcal{C}$ to a collection $\mathcal{C}'$ of *disjoint* clusters, where every vertex that is clustered under $\mathcal{C}$ remains clustered in $\mathcal{C}'$.

```
CLUSTERING(T, G):
  ⟨⟨Preprocess G and T⟩⟩
  for each bag in T, connect its vertices by a clique
  for each v ∈ V(G):
      replace every occurrence of v with a fresh copy
      for every pair parent-child in T_v add an edge between the copies of v
  𝒞 ← ∅
  for i ← k+1 down to 1: ⟨⟨Proceed in k+1 rounds, where k is the treewidth of G⟩⟩
      CLUSTERINGROUND(T, G, i, 𝒞)

  ⟨⟨𝒞 may contain overlapping preliminary clusters⟩⟩
  𝒞' ← ∅
  G' ← add to G a dummy vertex s and connect s to every cluster center in 𝒞'
  ⟨⟨all new edges have weight 0⟩⟩
  τ_s ← an SSSP tree of G' rooted at s
  for each child u of s in τ_s:
      B'_u ← the set of vertices in the subtree of τ_s rooted at u
      𝒞' ← 𝒞' ∪ B'_u
  return 𝒞'
```

```
CLUSTERINGROUND(T, G, i, 𝒞):
  R ← root bag of T
  ⟨⟨If R contains unclustered vertices, R is called a round-i root bag⟩⟩
  ⟨⟨Otherwise, we continue recursively with each one of its children⟩⟩
  B ← ∅
  for each unclustered u in R:
      B_u ← {v ∈ T | δ_G(u, v) ≤ εΔ} ⟨⟨Cluster downwards in the tree⟩⟩
      𝒞 ← 𝒞 ∪ B_u ⟨⟨Add B_u as a preliminary-cluster⟩⟩
      B ← B ∪ B_u
  S(B) ← the set of bags in T containing a vertex from B
  if R does not contain unclustered vertices:
      S(B) ← {R}
  ⟨⟨The subgraph of T induced by S(B) is connected⟩⟩
  T_1, ..., T_g ← subtrees of T \ S(B)
  for j ← 1, ..., g:
      CLUSTERINGROUND(T_j, G, i, 𝒞)
```

Figure 10. Algorithms ClusteringRound$(T, G, i, \mathcal{C})$ and Clustering$(T, G, \mathcal{C})$

Every cluster is formed by taking all vertices in a subtree of the SSSP tree rooted at the dummy vertex s, and is thus connected. By definition, every cluster of $\mathcal{C}$, which is a ball of radius $\varepsilon\Delta$, has a strong diameter of at most $O(\varepsilon\Delta)$. After running the SSSP algorithm, every cluster may reduce its size, but all the vertices remaining in that cluster are at the same distance from the respective center as before. □

Definition 6.3. *Let $\pi = \pi_G(u, v) = \langle u = x_0, x_1, \ldots, x_\ell = v\rangle$ be a path between u and v in G and let T be a tree decomposition of G. We say that a walk w in T corresponds to π if it is the shortest walk in T that visits bags containing (x_i, x_{i+1}) for $0 \le i \le \ell - 1$ in that order.*

Observation 6.4. *For every pair of points u and v in G, there is a shortest path $\pi(u, v)$ in G such that the corresponding walk π_T in T is a simple path. This follows from the fact that every bag of T induces a clique in G.*

Observation 6.5. *Let X and Y be two round-i root bags such that X is an ancestor of Y. Then, the distance between every center of X clustered at the ith round and every center of Y clustered at the ith round is greater than $\varepsilon\Delta$.*

Proof: Suppose towards contradiction that there were two centers of round-i clusters $x \in X$ and $y \in Y$ at distance at most $\varepsilon\Delta$. Consider the recursive call of ClusteringRound when the ball centered at x is created. Then, y is in B_x by definition, as it is distance at most $\varepsilon\Delta$ from x, $y \in Y$ and Y is a descendant of X. Hence, the bag Y is in B, and by the algorithm description Y will not be visited in any recursive call at the ith round. Hence, Y cannot be a round-i bag, yielding a contradiction. □

Observation 6.6. *Consider an arbitrary round $i \in \{k+1, \ldots, 1\}$ during a call to* Clustering. *Let π be a path in G that corresponds to a path π_T in T. Suppose that π_T contains no round-i' root bag, for all of the rounds $i' \geq i$. Then, the vertices of π are contained in at most $k+1$ clusters created at the ith round.*

Proof: Since no bag in the path π_T is a round-i bag, the algorithm implies that every bag on π_T intersects a cluster created at some ancestor root bag of the ith round. (This is true because the algorithm creates clusters only *down* the tree, and every occurrence of a vertex in the tree is replaced by a fresh copy.) By Observation 6.1, there is a single such ancestor root bag, which gives rise to at most $k+1$ clusters. Thus, vertices in the bags of π_T are contained in at most $k+1$ clusters in the i-th round. The observation follows from the fact that every vertex in π lies in some bag in π_T. □

Lemma 6.7. *Let $i \in \{k+1, \ldots, 1\}$ be an arbitrary round in a call to* Clustering. *Let u and v be any two vertices in G, let π be a shortest path between u and v in G, and let π_T be a simple path corresponding to π. Suppose that there is no round-i' root bag on w, for any earlier round $i' > i$. Then, the union of the vertices in bags in π_T is contained in at most $J_i = O((i/\varepsilon)^i)$ clusters created at any round $i, i-1, \ldots, 1$.*

Proof: The proof proceeds by induction on the round number i. For the basis $i = 1$, there at most $k+1$ clusters on π by Observation 6.5.

For the induction step, we assume the statement holds for all values smaller than i (i.e., for all later rounds) and proceed to prove it for i, with $i > 1$. We first consider clusters created at the ith round. Every time the algorithm creates new clusters, it creates up to $k+1$ clusters from the vertices of some root bag. Suppose first that π_T does not pass through any root bag. By Observation 6.6, vertices in bags of π_T are contained in at most $k+1$ clusters, created by some root bag higher than the bags visited by π_T. Otherwise, let $X_1, X_2, \ldots, X_\ell$ be the root bags which π_T passes through. The number of different round-i clusters visited is at most $(k+1)(\ell+1)$, since the path touches at most $(k+1)\ell$ clusters created by the root bags it visits and at most $k+1$ additional clusters created by some root bag higher than the highest bag of the path. We now find an upper bound for ℓ. Consider a pair X_i, X_{i+1} of root bags along π_T such that X_i is an ancestor of X_{i+1}. It follows from Observation 6.5 that the distance between the centers of X_i and X_{i+1} is at least $\varepsilon\Delta$. The same conclusion can be reached when X_i is a descendant of X_{i+1}. Among the root bags in $X_1, X_2, \ldots, X_\ell$, there is at most one pair of consecutive root bags which are not in ancestor/descendant relation. The path between them is contained in at most $k+1$ clusters by Observation 6.6. We conclude that if length of π is Δ, then the corresponding walk π_T passes through at most $1/\varepsilon + 2$ different root bags, i.e., $\ell \leq 1/\varepsilon + 2$. Hence, the number of clusters visited by π is at most $(k+1)(1/\varepsilon + 3)$.

The above argument bounds the number of intersections with the clusters built in the i-th round by $(k+1)(1/\varepsilon+3)$. This splits the path into at most $(k+1)(1/\varepsilon+3)+1$ connected parts that were unclustered by clusters from the ith round. Inductively, each such part intersect J_{i-1} clusters. Hence, J_i satisfies recurrence

$$J_i = ((k+1)(1/\varepsilon+3)+1) \cdot J_{i-1} + (k+1)(1/\varepsilon+3)$$

with $J_1 = 1$. The solution is given by $J_i = O((i/\varepsilon)^i)$. The lemma follows since the number of rounds is $k+1$ and $J_{k+1} = O((k/\varepsilon)^k)$. $\square$

By combining this lemma with Observation 6.4, we conclude:

Corollary 6.8. *For every pair of vertices u and v within distance at most Δ in G, there is an $O((k/\varepsilon)^{k+1})$-hop path in the cluster graph induced by the subset of clusters that has nontrivial intersections with some (approximate) shortest path $\pi(u,v)$ between u and v in G.*

From Corollary 6.8 and Lemma 6.2, we conclude:

Theorem 6.9. *For any $\varepsilon \in (0,1)$, any graph G with treewidth k embeds exactly into a graph G' with treewidth $O(k)$, such that G' has an $(\varepsilon, O(k/\varepsilon)^{k+1})$-shortcut partition.*

Combining this theorem with our general framework for constructing tree cover (Lemma 1.7 and Theorem 1.8) proves Theorem 1.4.

7 Embedding planar graphs into low-treewidth graphs

We show that every planar graph embeds, with distortion $+O(\varepsilon\Delta)$, into a graph with $O_\varepsilon(1)$ treewidth. The proof is simple and uses two facts of planar graphs: it can be covered by $O_\varepsilon(1)$ forests, and it has (a weak version of) a shortcut partition.

Claim 7.1. *Let G be a planar graph with diameter Δ, and let $\varepsilon > 0$. Then there is a set $\mathcal{F}$ of $O(\varepsilon^{-3})$ forests of rooted trees and a partition $\mathfrak{P}$ of G into components with diameter $\varepsilon\Delta$, such that:*

- [Low-hop.] *For every pair of vertices u and v in G, there is a path in G between u and v that intersects at most $O(\varepsilon^{-1})$ clusters.*
- [Root preservation.] *For every $u, v \in V(G)$, there is some tree T in a forest of $\mathcal{F}$ such that (i) $\delta_G(u,v) \le \delta_T(u,v) \le \delta_G(u,v) + O(\varepsilon\Delta)$, and (ii) the shortest path between u and v in T passes through the root of T.*
- [Cluster disjointness.] *Every forest in $\mathcal{F}$ is a spanning forest (i.e., it contains no Steiner vertices or edges). Further, no two trees in any forest in $\mathcal{F}$ intersect the same cluster of $\mathfrak{P}$.*

This claim is satisfied by choosing $\mathfrak{P}$ to be an $(\varepsilon, O(\varepsilon^{-1}))$-shortcut partition with $O(1)$ distortion (which exists by Theorem 3.10), and choosing $\mathcal{F}$ to be the associated spanning forest cover of size $O(\varepsilon^{-3})$ described in Section 5 (cf. Theorem 5.8). In particular, the low-hop property in the claim follows from the low-hop property of shortcut partitions; the root preservation property follows from the proof of Lemma 5.6; and the cluster disjointness property follows from Lemma 5.5.

7.1 Adding edges to a low-treewidth graph

Before describing the construction of the low-treewidth embedding, we prove a general lemma to bound the treewidth of a graph produced by adding certain edges to a low-treewidth graph. We begin by stating a simple property of tree decompositions. (For definition and basic properties of treewidth and tree decompositions, we refer the readers to the textbook by Diestel [Die17] for an introduction.)

Claim 7.2. *Let G be a graph, and let $\mathcal{T}$ be a tree decomposition for G. If H is a connected subgraph of G, then the set of all bags intersecting nontrivially with H forms a connected subtree in $\mathcal{T}$.*

Proof: Let v be an arbitrary vertex in H. Let V_i denote the set of all vertices that are within hop-distance i of v in H, and let B_i be the set of bags in $\mathcal{T}$ that intersect nontrivially with V_i. Because H is connected, there is some i where all bags in B_i intersect nontrivially with H. We prove, by induction on i, that B_i is a connected subtree in $\mathcal{T}$. In the base case where $i = 0$, $V_0 = \{v\}$ and so the claim follows from the definition of tree decomposition. For the inductive step, assume that the claim holds for every B_j with $j < i$. For every $u \in V_i \setminus V_{i-1}$, let B_u^+ denote the set of bags in $\mathcal{T}$ that contain u. Notice that every $u \in V_i \setminus V_{i-1}$ is adjacent to some vertex in V_{i-1}, so there is some bag (containing both u and a vertex in V_{i-1} adjacent to u) that is in both B_u^+ and B_{i-1}. As B_u^+ and B_{i-1} are both connected subsets of $\mathcal{T}$, we conclude that $B_i = \bigcup_{u \in V_i \setminus V_{i-1}} B_u^+ \cup B_{i-1}$ is a connected subset of $\mathcal{T}$. □

Lemma 7.3. *Let G be a graph, and let $\mathcal{T}$ be a tree decomposition for G with treewidth w. Let $\mathcal{F} = \{F_1, \ldots, F_k\}$ be a set of spanning forests (that is, each forest is a subgraph of G containing rooted trees). Then there is a tree decomposition $\mathcal{T}'$ for G such that (i) $\mathcal{T}'$ has width $O(wk)$, and (ii) for every tree T in some forest of $\mathcal{F}$ and for every vertex v in T, there is some bag in $\mathcal{T}'$ that contains both v and* $\mathrm{root}(T)$.

Proof: Let $\mathcal{T}' \leftarrow \mathcal{T}$. For every $v \in V(G)$, let $R_v := \{\mathrm{root}(T) \mid T \text{ is a tree in } \mathcal{F} \text{ where } v \in V(T)\}$, and add the vertices R_v to each bag in $\mathcal{T}'$ that contains v.

Width of $\mathcal{T}'$. Notice that for every v, at most one tree per forest of $\mathcal{F}$ contains v, so $|R_v| \le k$. Each bag in $\mathcal{T}$ has at most $w+1$ vertices. The corresponding bag in $\mathcal{T}'$ contains at most $(k+1)(w+1)$ vertices: it contains each of the $w+1$ original vertices v, plus the $w+1$ sets R_v that are each of size at most k.

Connectivity in $\mathcal{T}'$. We need to show that for every vertex v in G, the bags in $\mathcal{T}'$ containing v form a connected subtree. For an arbitrary v in G, let H_v denote the union of all trees in (some forest of) $\mathcal{F}$ that contain v. Notice that H_v is a connected subgraph of G, as it is the union of connected subgraphs that share a common vertex. Claim 7.2 implies that the set of bags B_v of $\mathcal{T}$ that intersects nontrivially with $V(H_v)$ form a connected subset in $\mathcal{T}$. However, the set of bags in $\mathcal{T}'$ containing v is precisely B_v. □

7.2 Embedding a planar graph

Let G be a planar graph with diameter Δ, and let $\varepsilon > 0$. Let $\mathcal{F}$ be the forest cover and let $\mathfrak{P}$ be the partition guaranteed by Claim 7.1. We construct a graph G' by first contracting each cluster in $\mathfrak{P}$. For each cluster, choose an arbitrary vertex in that cluster to be the *center vertex* of the cluster. Replace each supernode in G' with a star: the center vertex of the cluster is the center of the star, and the non-center vertices in the cluster are the other vertices. Edges between supernodes are replaced with edges between the corresponding center vertices. Assign each edge $(u, v) \in E(G')$ weight equal to $\delta_G(u, v)$.

Lemma 7.4. *The graph G' has treewidth $O(\varepsilon^{-1})$.*

Proof: The low-hop property of Claim 7.1 guarantees that there is a path of hop-length $O(\varepsilon^{-1})$ between every pair of vertices in $V(G')$. Further, G' is planar: Planar graphs are minor-closed, and replacing supernodes with stars does not affect planarity. Planar graphs have treewidth asymptotically upper-bounded by hop diameter, which proves the claim. □

We now augment G' with extra edges into $\hat{G}$ to reduce the distortion. Let $\hat{G} \leftarrow G'$. For every tree T in $\mathcal{F}$, and for every vertex v in T, add an edge to $\hat{G}$ between $\mathrm{root}(T)$ and v. Assign the edge weight to be $\delta_G(\mathrm{root}(T), v)$.

Lemma 7.5. *For every pair of vertices u and v in $\hat{G}$, we have $\delta_G(u, v) \le \delta_{\hat{G}}(u, v) \le \delta_G(u, v) + O(\varepsilon\Delta)$.*

Proof: Let u and v be vertices in $\hat{G}$. By the root preservation property in Claim 7.1, there is some tree T in $\mathcal{F}$ such that $\delta_T(u, \mathrm{root}(T)) + \delta_T(\mathrm{root}(T), v) \le \delta_G(u,v) + O(\varepsilon\Delta)$. The construction of $\hat{G}$ guarantees that there is an edge in $\hat{G}$ from u to $\mathrm{root}(T)$ and an edge from $\mathrm{root}(T)$ to v, weighted according to their distances in G. This proves the upper-bound for $\delta_{\hat{G}}(u,v)$. The lower bound follow from the fact that every edge (u,v) in $\hat{G}$ has weight $\delta_G(u,v)$. □

To prove that $\hat{G}$ has low treewidth, we want to use Lemma 7.3 applied to G' and the forest cover for $\mathcal{F}$. To do this, we need first to state a lemma that lets us translate between spanning forests in G and spanning forests in G'.

Lemma 7.6. *For every F in $\mathcal{F}$, there is a corresponding spanning forest F' of rooted trees (each a subgraph of G'), such that the following holds: For every tree T in F, there is a tree T' in F' such that $V(T) \subseteq V(T')$ and $\mathrm{root}(T) = \mathrm{root}(T')$.*

Proof: For each tree T in F, Let $V_T \subseteq V(G)$ be the union of cluster vertices in $\mathfrak{P}$ that intersect nontrivially with T. Let V'_T is the corresponding subset of $V(G')$ by identifying vertices in V_T that came from the same cluster in $\mathfrak{P}$ into a single vertex. The sets $\{V'_T\}_{T\in F}$ are pairwise disjoint, because of the cluster disjointness property of $\mathfrak{P}$ in Claim 7.1. Further, because each cluster in $\mathfrak{P}$ is connected and because T is a connected subgraph of G, each set V'_T induces a connected subgraph of G'. Let T' be a spanning tree in G' rooted at $\mathrm{root}(T)$ with vertex set V'_T. The collection of all trees T' is pairwise vertex-disjoint, and thus forms a forest on G'. □

Lemma 7.7. *The graph $\hat{G}$ has treewidth $O(\varepsilon^{-4})$.*

Proof: For every forest F in $\mathcal{F}$, apply Lemma 7.6 to find a corresponding forest F' in G'. The resulting set $\mathcal{F}'$ contains $O(\varepsilon^{-3})$ forests. By construction of $\hat{G}$, every edge $(u,v) \in E(\hat{G}) \setminus E(G')$ is induced by some tree T where (without loss of generality) $u = \mathrm{root}(T)$ and $v \in V(T)$. By Lemma 7.6, there is a corresponding tree T' in $\mathcal{F}'$ such that $u = \mathrm{root}(T')$ and $v \in V(T')$. Thus, applying Lemma 7.3 to G' (which, by Lemma 7.4, has treewidth $O(\varepsilon^{-1})$) and $\mathcal{F}'$ yields a valid tree decomposition for $\hat{G}$ with width $O(|\mathcal{F}'| \cdot \mathrm{tw}(G')) = O(\varepsilon^{-4})$. □

We now observe that Theorem 1.3 follows from Lemma 7.5 and Lemma 7.7.

8 Reduction from additive to multiplicative tree cover

In this section, we prove Lemma 1.7, which we restate below. Recall that a tree cover $\mathcal{T}$ is *Δ-bounded* if every tree of $\mathcal{T}$ has diameter at most Δ.

Lemma 1.7 (Reduction to additive tree covers). *Let (X, δ_X) be a K_r-minor-free metric (for any constant r) with n points. For any parameter $\varepsilon \in (0,1)$, suppose that any K_r-minor-free submetric induced by a subset $Y \subseteq X$ with diameter Δ has an $O(\Delta)$-bounded tree cover $\mathcal{F}$ of size $\tau(\varepsilon)$ of additive distortion $+\varepsilon\Delta$. Then (X, δ_X) has a tree cover of size $O(\tau(O(\varepsilon)) \cdot \log(1/\varepsilon))$ with multiplicative distortion $1+\varepsilon$.*

We introduce the notion of a *family of pairwise hierarchical partitions* and base our reduction on this family. To formally define this notion, we first define a hierarchical partition.

Definition 8.1 (Hierarchical Partition). *Let $\mu > 1$ be a parameter. Let (X, δ_X) be a metric space with minimum distance 1 and maximum distance at most Φ. A μ-hierarchical partition for (X, δ_X), denoted by $\mathbb{P} = \{\mathcal{P}_0, \mathcal{P}_1, \dots, \mathcal{P}_{i_{\max}}\}$ where $i_{\max} = O(\log_\mu(\Phi))$, is a set of partitions such that:*

1. *$\mathcal{P}_0$ contains singletons only and $\mathcal{P}_{i_{\max}}$ contains a single set X.*
2. *Each $\mathcal{P}_i$ is a partition of X into clusters of diameter at most μ^i for every integer $i \in [0, i_{\max}]$. We call $\mathcal{P}_i$ a partition at scale i of $\mathbb{P}$.*
3. *Each set S in $\mathcal{P}_i$ for $i \geq 1$ is the union of some sets in $\mathcal{P}_{i-1}$.*

Item 3 implies that the partitions are nested and hence form a hierarchy.

Definition 8.2 (Hierarchical Pairwise Partition Family (HPPF)). *Let $\sigma, \mu, \rho \geq 1$ be parameters. Let (X, δ_X) be a metric space with minimum distance 1 and maximum distance Φ. A (σ, μ, ρ)-hierarchical pairwise partition family, which we abbreviate as (σ, μ, ρ)-HPPF, is a family of μ-hierarchical partitions of size σ, denoted by $\mathfrak{P} = \{\mathbb{P}_1, \ldots, \mathbb{P}_\sigma\}$, of (X, δ_X) such that:*

1. *Each $\mathbb{P}_j$, $j \in [\sigma]$, is a μ-hierarchical partition of (X, δ_X).*
2. *For every two different points x and y in X, there exists a hierarchy $\mathbb{P}_j$ in $\mathfrak{P}$ and a partition $\mathcal{P}_i$ at scale i of $\mathbb{P}_j$ such that both x and y belong to the same cluster and $\delta_X(x, y) \geq \mu^i/\rho$.*

Item 2 in Definition 8.2 is called the *pairwise property*. Recall that each cluster in $\mathcal{P}_i$ has diameter at most μ^i. The pairwise property posits that for every pair of points, there is a cluster in some partition at scale i containing the pair, whose diameter is roughly the same as the distance of the pair up to a factor of ρ. The pairwise property is crucial in our reduction.

The HPPF is a variant of the *hierarchical partition family* (HPF) introduced earlier by [BFN19, KLMN05] without the pairwise property. Instead, the HPF in these works has a *padded property*: every ball $B_X(x, \mu^i/\rho)$ is wholly contained in some cluster of some partition at scale i.

Lemma 8.3. *Let $\varepsilon \in (0, 1)$ be a parameter. Any K_r-minor-free metric (X, δ_X) admits (σ, μ, ρ)-HPPF for $\sigma = O(3^r \log(1/\varepsilon)/\log r), \mu \geq 1/\varepsilon, \rho = O(r^2)$.*

We defer the proof of Lemma 8.3 to Section 8.2. Equipped with a HPPF as in Lemma 8.3, we now show the reduction as claimed in Lemma 1.7.

8.1 The reduction: proof of Lemma 1.7

In this section, we prove Lemma 1.7, assuming that Lemma 8.3 holds. Let $\mathfrak{P}$ be a (σ, μ, ρ)-HPPF for $\sigma = O(3^r \log(1/\varepsilon)/\log r), \mu = 1/\varepsilon, \rho = O(r^2)$ as in Lemma 8.3. For each hierarchy of partitions $\mathbb{P} \in \mathfrak{P}$, we will construct a tree cover $\mathcal{T}_{\mathbb{P}}$. The final tree cover will be $\mathcal{T} := \cup_{\mathbb{P} \in \mathfrak{P}} \mathcal{T}_{\mathbb{P}}$.

For each scale i, we construct a *net N_i* as a subset of X inductively as follows. $N_{i_{\max}}$ contains a single point chosen arbitrarily from X. Suppose that we are given N_{i+1}. For every set S in $\mathcal{P}_i$ (the partition at scale i), if $S \cap N_{i+1} = \varnothing$ then we chose a point in S arbitrarily and add it to N_i. (Initially, N_i is empty.) Otherwise, we add $S \cap N_{i+1}$ to N_i. The sets N_i's satisfy the following claim.

Claim 8.4. *One has $N_{i_{\max}} \subseteq \ldots \subseteq N_0 = X$. Furthermore for every scale i and every set $S \in \mathcal{P}_i$, $|S \cap N_i| = 1$.*

Proof: The first claim follows directly from the fact that $\mathcal{P}_i$ is a partition of X and whenever $S \cap N_{i+1} \neq \varnothing$ we add all points of $S \cap N_{i+1}$ to N_i. We prove the second claim by induction: the base case is $N_{i_{\max}}$ that has a single point and hence the claim holds. For the inductive case, if $S \cap N_{i+1} \neq \varnothing$, then $|S \cap N_{i+1}| = 1$. This is because $\mathcal{P}$ is a hierarchy and hence there exists a superset of S in $\mathcal{P}_{i+1}$, say S', such that $S \subseteq S'$ and thus has $|S' \cap N_{i+1}| = 1$ by induction. It follows that $|S \cap N_i| = 1$ by construction. Otherwise, $S \cap N_{i+1} = \varnothing$, we choose a single point of S to add to N_i by construction and hence $|S \cap N_i| = 1$. □

Let $r_i := \mu^i$. Claim 8.4 shows that N_i is a r_i-*cover* of N_{i-1} as each point $x \in N_{i-1}$ has a point $y \in N_i$ such that $\delta_X(x,y) \le r_i$ since the diameter of sets in $\mathcal{P}_i$ is at most r_i. For every point $x \in S \in \mathbb{P}_i$, we call the point $x_i \in N_i \cap S$ the *ancestor at scale i* of x. Observe that:

Observation 8.5. *For any point $x \in X$, $\delta_X(x,x_i) \le r_i$ where x_i is the ancestor at scale i of x.*

Proof: This follows from the fact that $x, x_i \in S$ for some $S \in \mathbb{P}_i$, and S has diameter at most r_i. □

In the following construction, we regard points in different sets N_i as *different* and write (x,i) as a copy of x if $x \in N_i$, and the tree cover we construct is for all points in $N_0 \cup N_1 \cup \ldots \cup N_{i_{\max}}$. For each tree in the final tree cover $\mathcal{T}_{\mathbb{P}}$, we keep only one copy per point, which is the copy in N_0 of the point.

For each scale $i \in [i_{\max}]$, we construct a set of κ forests $\mathcal{F}^i = \{F_1^i, \ldots, F_\kappa^i\}$ rooted at points in N_i as follows. (The value of κ will be set later.) For each set S in $\mathcal{P}_i$, we construct a tree cover, denoted by $\mathcal{T}_S^i$, with additive distortion $+\varepsilon r_i/\rho$ for $S \cap N_{i-1}$. Note that r_i is the diameter of S. By Lemma 1.7, $|\mathcal{T}_S^i| \le \tau(|S \cap N_{i-1}|, \varepsilon/\rho) \le \tau(n, O(\varepsilon/r^2))$. (Recall that $\rho = O(r^2)$ in Lemma 8.3.) We choose $\kappa = \tau(n, O(\varepsilon/r^2))$ so that $|\mathcal{T}_S^i| \le \kappa$ for every S in $\mathcal{P}_i$. By duplication, we assume that $\mathcal{T}_S^i$ contains exactly κ trees, denoted by $T_{1,S}, \ldots, T_{\kappa,S}$. We root each tree $T_{t,S}$ in $\mathcal{T}_S^i$ at the copy (x,i) of the (single) point x in $S \cap N_i$; by Claim 8.4, there is only one such point. Next, for each $t \in [\kappa]$, let $F_t^i := \{T_{t,S} : S \in \mathcal{P}_i\}$ to be the t-th forest of $\mathcal{F}^i$. When $i = 0$, we define F_t^i to be the forest of singletons.

Finally, we construct the t-th tree for $t \in [\kappa]$, denoted by T_t, in the cover $\mathcal{T}_{\mathbb{P}}$ by connecting the forests F_t^i at different scales i. Specifically, for every scale i in $[i_{\max}]$, for each point x in N_{i-1}, add an edge of length 0 between $(x, i-1)$ and (x,i). This operation effectively connects the connected components of F_t^i and the components of F_t^{i-1} for every $i \in [i_{\max}]$.

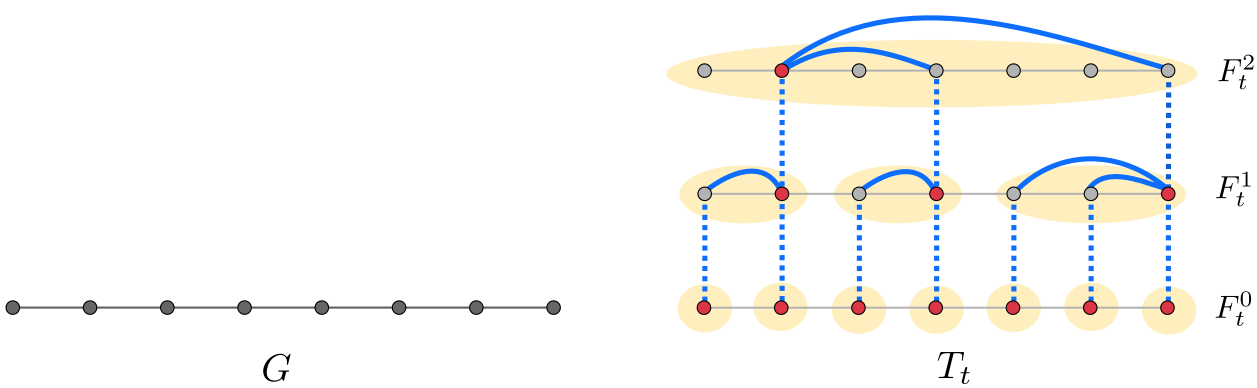

Figure 11. A graph G and a tree T_t. The tree T_t consists of forests F_t^2, F_t^1, and F_t^0, connected (by blue dashed lines) to form T_t. At each scale i: there is a partition of $\mathcal{P}_i$ (highlighted in yellow), a set of net points N_i (drawn in red), and a forest F_t^i (drawn with blue lines).

Claim 8.6. *T_t is a tree.*

Proof: Let $\hat{F}_t^0 := F_t^0$ which is a forest of singletons. Let $\hat{F}_t^i$ be the forest obtained by connecting F_t^i and F_t^{i-1} in the construction algorithm via edges of length 0 between (x,i) and $(x,i-1)$ for every $(x,i) \in N_i$. Then $\hat{F}_t^i$ is a forest containing trees rooted at points of N_i. As $N_{i_{\max}}$ consists of a single point, the forest $\hat{F}_t^{i_{\max}}$ (which is also T_t) is a tree. □

The following claim is crucial in bounding the distortion of the tree cover.

Claim 8.7. *Let x_0 be any point in N_0 and $x_i \in N_i$ be the ancestor at scale i of x_0 for any $i \ge 1$. Then $\delta_G(x_i, x_0) \le d_{T_t}(x_i, x_0) = O(r_i)$ when $\varepsilon < 1$.*

Proof: The lower bound follows from the fact that each tree in the forest F_t^i is a dominating tree for $N_{i-1} \cap S$. We now focus on proving the upper bound. Let c_0 be the constant such that every tree cover of additive distortion $+\varepsilon\Delta$ is $(c_0 \cdot D)$-bounded following the assumption of Lemma 1.7. Since every $S \in \mathcal{P}_i$ has diameter at most r_i, it follows that $d_{F_t^i}(x_{i-1}, x_i) \le c_0 r_i$. Thus, $d_{T_t}(x_i, x_0) \le c_0 \sum_{j=0}^{i} r_i \le c_0 r_i(1 + \varepsilon + \varepsilon^2 + \ldots) \le O(r_i)$ when $\varepsilon < 1$. □

The following claim concludes the proof of Lemma 1.7.

Claim 8.8. *Let $\mathcal{T} = \cup_{\mathbb{P}\in\mathfrak{P}} \mathcal{T}_{\mathbb{P}}$. Then $\mathcal{T}$ is a tree cover with multiplicative distortion $(1 + O(r^2\varepsilon))$ of size $O(\tau(n, O(\varepsilon/r^2)) \cdot 3^r \log(1/\varepsilon)/\log r)$.*

Proof: Since $|\mathfrak{P}| = \sigma = O(3^r \log(1/\varepsilon)/\log r)$ and $|\mathcal{T}_{\mathbb{P}}| \le \kappa = \tau(n, O(\varepsilon/r^2))$, $|\mathcal{T}| \le O(\tau(n, O(\varepsilon/r^2)) \cdot 3^r \log(1/\varepsilon)/\log r)$ as claimed. It remains to bound the distortion. We observe by the construction that every tree in $\mathcal{T}$ is a dominating tree since every tree in $\mathcal{T}_{\mathbb{P}}$ is dominating.

Let x, y be any two points in X. Let $\mathbb{P} \in \mathfrak{P}$ be the hierarchy and $\mathcal{P}_i$ be a partition of X at scale i of $\mathbb{P}$ such that $\{x, y\} \subseteq S$ for some set $S \in \mathcal{P}_i$ and $\delta_X(x, y) \ge r_i/\rho$. $\mathbb{P}$ and i exist by the definition of HPPF. Let $\hat{x}, \hat{y}$ be the ancestors of x and y, respectively, at scale $i-1$. As $\delta_X(x, y) \ge r_i/\rho$, we have:

$$r_{i-1} \le \varepsilon r_i \le \varepsilon\rho\delta_X(x, y). \tag{3}$$

Since x, y are both in S, $\hat{x}$ and $\hat{y}$ are both in S by the definition of N_i's. Since the tree cover $\mathcal{T}_S^i$ for $N_{i-1} \cap S$ in the construction of $\mathcal{F}^i$ has additive distortion $\varepsilon r_i/\rho$, there is a forest $F_t^i \in \mathcal{F}^i$ for some $t \in [\kappa]$ such that $d_{F_t^i}(\hat{x}, \hat{y}) \le \delta_X(\hat{x}, \hat{y}) + \varepsilon r_i/\rho$. It follows that:

$$\begin{aligned} d_{T_t}(\hat{x}, \hat{y}) &\le \delta_X(\hat{x}, \hat{y}) + \varepsilon r_i/\rho \\ &\le \delta_X(x, y) + O(r_{i-1}) + \varepsilon r_i/\rho \qquad \text{(by triangle inequality and Observation 8.5)} \\ &= \delta_X(x, y) + O(\varepsilon\rho\delta_X(x, y)) + \varepsilon\delta_X(x, y) \qquad \text{(by Equation (3))} \\ &= \delta_X(x, y) + O(\rho)\varepsilon\delta_X(x, y) \end{aligned} \tag{4}$$

Furthermore, by Claim 8.7 and the triangle inequality, we have:

$$\begin{aligned} \delta_{T_t}(x, y) &\le d_{T_t}(\hat{x}, \hat{y}) + O(r_{i-1}) \\ &\le \delta_X(x, y) + O(\rho)\varepsilon\delta_X(x, y) + O(r_{i-1}) \qquad \text{(by Equation (4))} \\ &\le \delta_X(x, y) + O(\rho)\varepsilon\delta_X(x, y) + O(\rho)\varepsilon\delta_X(x, y) \qquad \text{(by Equation (3))} \\ &= (1 + O(r^2\varepsilon))\delta_X(x, y) \end{aligned} \tag{5}$$

since $\rho = O(r^2)$. That is, there exists a tree in $\mathcal{T}_{\mathbb{P}}$ (which is T_t), and hence a tree in $\mathcal{T}$, such that the distance between x and y is preserved in the tree up to a factor of $1 + O(r^2\varepsilon)$. The claim now follows. □

8.2 HPPF construction: proof of Lemma 8.3

We base our construction of a HPPF on the *hierarchical partition family* (HPF) formally define below. The HPF was formally introduced in [BFN19] though its ideas appeared earlier [KLMN05].

Definition 8.9 (Hierarchical Partition Family (HPF)). *Let $\sigma, \mu, \rho \ge 1$ be parameters. Let (X, δ_X) be a metric space with minimum distance 1 and maximum distance Φ. A (σ, μ, ρ)-hierarchical partition family ((σ, μ, ρ)-HPF for short) is a family of μ-hierarchical partitions of size σ, denoted by $\mathfrak{P} = \{\mathbb{P}_1, \ldots, \mathbb{P}_\sigma\}$ of (X, δ_X), such that:*

1. *Each $\mathbb{P}_j$, $j \in [\sigma]$, is a μ-hierarchical partitions of (X, δ_X).*
2. *For every point $x \in X$, there exists a hierarchy $\mathbb{P}_j \in \mathfrak{P}$ and a partition $\mathcal{P}_i$ at scale i of $\mathbb{P}_j$ such $B_X(x, \mu^i/\rho) \subseteq S$ for some cluster $S \in \mathcal{P}_i$.*

The following lemma was shown in [KLMN05] as noted by [BFN19].

Lemma 8.10 ([KLMN05]). *Any K_r-minor-free metric admits a (σ, μ, ρ)-HPF with $\sigma = O(3^r)$, $\mu = O(r^2)$ and $\rho = O(r^2)$.*

Next, we show how to construct a HPPF from a HPF. The following lemma and Lemma 8.10 implies Lemma 8.3 as all parameters $\hat{\sigma}, \hat{\mu}, \hat{\rho}$ are $O(1)$ for planar graphs.

Lemma 8.11. *Let (X, δ_X) be a metric space admitting a $(\hat{\sigma}, \hat{\mu}, \hat{\rho})$-HPF. Then for any $\varepsilon \in (0,1)$, (X, δ_X) admits a (σ, μ, ρ)-HPPF with $\sigma = O(\hat{\sigma} \log(1/\varepsilon)/\log(\hat{\mu}))$, $\mu \geq 1/\varepsilon$, and $\rho = \hat{\rho}$.*

Proof: Let $\hat{\mathfrak{P}} = \{\hat{\mathbb{P}}_1, \ldots, \hat{\mathbb{P}}_{\hat{\sigma}}\}$ be a $(\hat{\sigma}, \hat{\mu}, \hat{\rho})$-HPF by the assumption of the lemma. For every hierarchy $\hat{\mathbb{P}}_a \in \hat{\mathfrak{P}}$ for $a \in [\hat{\sigma}]$, we "partition" it into $\kappa := \lceil \log_{\hat{\mu}}(1/\varepsilon) \rceil$ hierarchies $\{\mathbb{P}_{a,0}, \mathbb{P}_{a,1}, \ldots, \mathbb{P}_{a,\kappa-1}\}$ as follows: each hierarchy $\mathbb{P}_{a,t}$ for $t \in [\kappa]$ includes all partitions $\mathcal{P}_i$ in all scales i's of $\hat{\mathbb{P}}_a$ such that $i \equiv t \pmod{\kappa}$. We then define:

$$\mathfrak{P} := \{\mathbb{P}_{a,t} : a \in [\hat{\sigma}], t \in [\kappa]\}. \tag{6}$$

Clearly, $\sigma = |\mathfrak{P}| = \hat{\sigma} \cdot \kappa = O(\hat{\sigma} \log(1/\varepsilon)/\log(\hat{\mu}))$ as claimed. Furthermore, for every hierarchy $\mathbb{P} \in \mathfrak{P}$, the ratio of the diameter of scale $i+1$ to the diameter of scale i is $\hat{\mu}^{\kappa} = \hat{\mu}^{\lceil \log_{\hat{\mu}}(1/\varepsilon) \rceil} \geq 1/\varepsilon$; thus, $\mu \geq 1/\varepsilon$.

Finally, let (x, y) be any two different points in X. Let $i \geq 0$ be such that $\hat{\mu}^{i-1}/\rho \leq \delta_X(x,y) < \hat{\mu}^i/\rho$; such i exists since $\rho \leq 1$, $\delta_X(x,y) \geq 1, \mu \geq 1$. By Item 2 in Definition 8.9, there exist a hierarchy $\hat{\mathbb{P}}_a \in \hat{\mathfrak{P}}$, a partition $\mathcal{P}_i \in \hat{\mathbb{P}}_a$, and a cluster $S \in \mathcal{P}_i$ such that $B_X(x, \mu^i/\rho) \subseteq S$. Since hierarchies $\{\mathbb{P}_{a,0}, \mathbb{P}_{a,1}, \ldots, \mathbb{P}_{a,\kappa-1}\}$ partition $\hat{\mathbb{P}}_a$, there exists a hierarchy $\mathbb{P}_{a,t}$ for some $t \in [\kappa]$ such that S belongs to the partition at some scale of $\mathbb{P}_{a,t}$. Furthermore, both x and y are in S since $B_X(x, \mu^i/\rho) \subseteq S$, as desired. $\square$

9 Algorithmic applications

In this section, we discuss the algorithmic applications of our tree cover theorem (Theorem 1.2).

Distance oracles. Let $\mathcal{T}$ be a tree cover of size $O(\varepsilon^{-3} \log(1/\varepsilon))$. In the preprocessing stage, we construct a distance oracle for each tree T in $\mathcal{T}$ using the LCA data structure, following a standard construction [LWN21, FGNW17], as follows. Root T at a vertex r. For each vertex v in T, we store the distance $d_T(v, r)$ at v. We then construct the LCA data structure for T, denoted by $\mathcal{LCA}_T$. To query the distance between two vertices u and v in T, first we query $\mathcal{LCA}_T$ to get the LCA of u and v, denoted by w. We then return $d_T(u,r) + d_T(v,r) - 2d_T(w,r)$ as the distance between u and v in G.

The distance oracle for G will contain the distance oracle for each tree T in the tree cover. The total space over all trees is then $\sum_{T \in \mathcal{T}}(S_{LCA}(|V(T)|) + O(|V(T)|)) = O(S_{LCA}(n) \cdot \varepsilon^{-3} \log(1/\varepsilon))$ as $|V(T)| = O(n)$ for every T in $\mathcal{T}$. To query the distance between u and v, we simply go over each tree T in $\mathcal{T}$, query the distance $d_T(u,v)$ and finally return $\min_{T \in \mathcal{T}} d_T(u,v)$ as the distance. The query time is $O(Q_{LCA}(n) \cdot \varepsilon^{-3} \log(1/\varepsilon))$. The returned distance is a $(1+\varepsilon)$-approximation of $\delta_G(u,v)$ since the distortion of $\mathcal{T}$ is $(1+\varepsilon)$. Our Theorem 1.6 now follows.

Emulator of linear size. Given a planar graph G and a set S of k terminals in G. We say that a graph H is a $(1+\varepsilon)$-*emulator* if $S \subseteq V(H)$ and for every two terminals $t_1, t_2 \in S$,

$$\delta_G(t_1,t_2) \le d_H(t_1,t_2) \le (1+\varepsilon)\cdot\delta_G(t_1,t_2).$$

The size of the emulator is the number of edges. Cheung, Goranci, and Henzinger constructed a *planar* $(1+\varepsilon)$-emulator of almost quadratic size $\tilde{O}(k^2/\varepsilon^2)$. Recently, Chang, Krauthgamer, and Tan [CKT22a, CKT22b] obtained a planar $(1+\varepsilon)$-emulator of *almost linear* size $k\,\mathrm{poly}(\log k, 1/\varepsilon)$, which breaks below the $\Omega(k^2)$ lower bound when no distortion is allowed.

Using our tree cover in Theorem 1.2, we can construct a $(1+\varepsilon)$-emulator of *linear size* as follows. For each tree T in $\mathcal{T}$, we prune T so that it only has $O(k)$ vertices by (i) repeatedly removing non-terminal leaves until every leaf is a terminal in S and (ii) contracting non-terminal vertices of degree 2 and reweighting the new edge appropriately. This pruning does not change the distances in T between terminals. Finally, we simply union all the tree T in the cover $\mathcal{T}$. The same copies of a terminal in different trees will become a single terminal in the emulator. The size of our emulator is $O(k\cdot\varepsilon^{-3}\log(1/\varepsilon))$ since each tree has $O(k)$ vertices. We note that, unlike prior work, our emulator may not be planar.

Low-hop emulators of planar metrics. We have just shown that any point set in a planar metric has a linear size $(1+\varepsilon)$-emulator. In some applications, we would want our emulators to have the low-hop property: every vertex u can reach a vertex v by a shortest path having few edges in the emulator. Our tree cover theorem provides a simple way to construct such an emulator, following the line of [KLMS22]. Specifically, for each tree in the tree cover, construct a low-hop spanner for the tree using the result by Solomon [Sol13], and take the union of all the spanners. The end result is an emulator with k hops for any integer $k \ge 1$ and size $O(n\alpha_k(n)\cdot\varepsilon^{-3}\log(1/\varepsilon))$ where $\alpha_k(n)$ is the function $\alpha_k(n)$ is the inverse of an Ackermann-like function at the $\lfloor k/2\rfloor$-th level of the primitive recursive hierarchy, where $\alpha_0(n) = \lceil n/2\rceil$, $\alpha_1(n) = \lceil\sqrt{n}\rceil$, $\alpha_2(n) = \lceil\log n\rceil$, $\alpha_3(n) = \lceil\log\log n\rceil$, $\alpha_4(n) = \log^* n$, $\alpha_5(n) = \lfloor\frac{1}{2}\log^* n\rfloor$, etc.

Distance labeling schemes. A distance labeling scheme is an assignment of labels (binary strings) to all vertices of a graph, so that the distance between any two nodes can be computed solely from their labels and the size of labels is as small as possible. A major open problem is to determine the complexity of distance labeling in an n-vertex unweighted and undirected planar graphs. An upper bound of $O(\sqrt{n}\log n)$ bits, complemented with a lower bound of $\Omega(n^{1/3})$, were achieved in the pioneering work of Gavoille et al. [GPPR04]. The upper bound was later improved to $O(\sqrt{n})$ [GU16]. For unweighted planar graphs of diameter bounded by Δ, we obtain a distance labeling scheme of $O(2^\Delta\log^2 n)$ bits, as a direct corollary of our tree cover result of Theorem 1.5 in conjunction with known distance labeling schemes (e.g., [Pel00]) for trees. This result generalizes for unweighted minor-free graphs of bounded diameter.

For $(1+\varepsilon)$-approximate distance labeling in unweighted planar graphs, the state-of-the-art is a labeling scheme of $O(\log n\cdot\log\log n\cdot\varepsilon^{-1})$ bits [Tho04]. We obtain a $(1+\varepsilon)$-approximate labeling scheme of $O(\log n\cdot\varepsilon^{-3}\cdot\log^2(1/\varepsilon))$ bits, as a direct corollary of our tree cover theorem of Theorem 1.2 in conjunction with the approximate distance labeling scheme for trees from [FGNW17]. This bound is optimal up to the ε-dependence, even for trees [FGNW17].

Routing in planar metrics. In compact routing schemes, the basic goal is to achieve efficient tradeoffs between the space usage — the maximal number of bits per node — and the stretch, which is the maximum ratio between distances of the route used and the shortest path over all source-destination pairs. There are two basic variants. In the *labeled* model, the designer is allowed to assign nodes with short labels that can be used for routing. In the *name-independent* model, the node labels are determined by an adversary.

There is a large body of work on routing in metric spaces, including in restricted families of metrics, most notably in Euclidean and doubling metrics [HP00, AM04, Tal04, Sli05, AGGM06, GR08, CGMZ16]. Many known routing protocols are rather complex, both conceptually and in implementation details.

Our tree cover theorems (Theorems 1.2, 1.4 and 1.5) essentially provide a reduction from the problem of routing in planar metrics to that of routing in tree metrics. Our approach has several advantages over previous work. (1) *Simple*: Routing along trees is as basic as it gets. (2) *General*: Our approach applies beyond planar metrics, to bounded treewidth metrics, to unweighted minor-free graphs of bounded diameter, and in principle to any graph family for which an efficient tree cover theorem exists. (3) *Low-hop*: By "shortcutting" the trees in our tree cover using the 2-hop sparse 1-spanners by Solomon [Sol13], one can carry out the routing protocol on these spanners; they are not as basic as tree metrics, but they achieve hop-diameter 2 with size $O(n \log n)$. As discussed and demonstrated in [KLMS22], a routing protocol in which the hop-distances are guaranteed to be bounded by 2 is advantageous.

We summarize our result on 2-hop routing in planar metrics.

Theorem 9.1. *For any n-point planar metric, one can construct a $(1+\varepsilon)$-stretch 2-hop routing scheme in the labeled, fixed-port model with headers of $\lceil \log n \rceil$ bits, labels of $O_\varepsilon(\log^2 n)$ bits, local routing tables of $O_\varepsilon(\log^2 n)$ bits, and local decision time $O(1)$.*

Acknowledgement. Hung Le and Cuong Than are supported by the NSF CAREER Award No. CCF-2237288 and an NSF Grant No. CCF-2121952. Shay Solomon is funded by the European Union (ERC, DynOpt, 101043159). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them. Shay Solomon is also supported by the Israel Science Foundation (ISF) grant No.1991/1. Shay Solomon and Lazar Milenković are supported by a grant from the United States-Israel Binational Science Foundation (BSF), Jerusalem, Israel, and the United States National Science Foundation (NSF).

References

[ADM+95] Sunil Arya, Gautam Das, David M. Mount, Jeffrey S. Salowe, and Michiel Smid. Euclidean spanners. In *Proceedings of the twenty-seventh annual ACM symposium on Theory of computing,STOC' 95*, 1995. doi:10.1145/225058.225191.

[AFGN22] Ittai Abraham, Arnold Filtser, Anupam Gupta, and Ofer Neiman. Metric embedding via shortest path decompositions. *SIAM Journal on Computing*, 51(2):290–314, 2022. doi:10.1137/19m1296021.

[AGGM06] Ittai Abraham, Cyril Gavoille, Andrew V. Goldberg, and Dahlia Malkhi. Routing in networks with low doubling dimension. In *Proc. of 26th ICDCS*, page 75, 2006.

[AKP94] Baruch Awerbuch, Shay Kutten, and David Peleg. On buffer-economical store-and-forward deadlock prevention. *IEEE transactions on communications*, 42(11):2934–2937, 1994.

[AKPW95] Noga Alon, Richard M. Karp, David Peleg, and Douglas West. A graph-theoretic game and its application to the k-server problem. *SIAM Journal on Computing*, 24(1):78–100, 1995. doi:10.1137/s0097539792224474.

[AM04] Ittai Abraham and Dahlia Malkhi. Compact routing on euclidian metrics. In *PODC*, pages 141–149. ACM, 2004.

[AN12] Ittai Abraham and Ofer Neiman. Using petal-decompositions to build a low stretch spanning tree. In *Proceedings of the 44th annual ACM symposium on Theory of Computing*, 2012. doi:10.1145/2213977.2214015.

[AP92] Baruch Awerbuch and David Peleg. Routing with polynomial communication-space trade-off. *SIAM J. Discret. Math.*, 5(2):151–162, may 1992. doi:10.1137/0405013.

[Bar96] Yair Bartal. Probabilistic approximations of metric spaces and its algorithmic applications. In *37th Annual Symposium on Foundations of Computer Science, FOCS '96, Burlington, Vermont, USA, 14-16 October, 1996*, pages 184–193, 1996. URL: https://doi.org/10.1109/SFCS.1996.548477.

[BC05] Bo Brinkman and Moses Charikar. On the impossibility of dimension reduction in ℓ_1. *Journal of the ACM*, 52(5):766–788, 2005. doi:10.1145/1089023.1089026.

[BDHM07] MohammadHossein Bateni, Erik D. Demaine, MohammadTaghi Hajiaghayi, and Mohammad Moharrami. Plane embeddings of planar graph metrics. *Discrete & Computational Geometry*, 38(3):615–637, 2007. doi:10.1007/s00454-007-1353-4.

[BFC00] M. A. Bender and M. Farach-Colton. The LCA problem revisited. In *Latin American Symposium on Theoretical Informatics (LATIN '00)*, pages 88–94, 2000. doi:10.1007/10719839_9.

[BFN19] Yair Bartal, Nova Fandina, and Ofer Neiman. Covering metric spaces by few trees. In *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, pages 20:1–20:16, 2019. doi:10.4230/LIPIcs.ICALP.2019.20.

[BLMN05] Yair Bartal, Nathan Linial, Manor Mendel, and Assaf Naor. On metric Ramsey-type phenomena. *Annals of Mathematics*, 162(2):643–709, 2005. doi:10.4007/annals.2005.162.643.

[BLT14] Costas Busch, Ryan LaFortune, and Srikanta Tirthapura. Sparse covers for planar graphs and graphs that exclude a fixed minor. *Algorithmica*, 69(3):658–684, July 2014. doi:10.1007/s00453-013-9757-4.

[Cab18] S. Cabello. Subquadratic algorithms for the diameter and the sum of pairwise distances in planar graphs. *ACM Transactions on Algorithms*, 15(2), 2018. Announced at SODA'17. doi:10.1145/3218821.

[CFKL20] Vincent Cohen-Addad, Arnold Filtser, Philip N. Klein, and Hung Le. On light spanners, low-treewidth embeddings and efficient traversing in minor-free graphs. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science, FOCS' 22*, 2020. doi:10.1109/focs46700.2020.00061.

[CGH16] Yun Kuen Cheung, Gramoz Goranci, and Monika Henzinger. Graph minors for preserving terminal distances approximately – lower and upper bounds. In *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*, Leibniz International Proceedings in Informatics (LIPIcs), pages 131:1–131:14, 2016. URL: http://drops.dagstuhl.de/opus/volltexte/2016/6267, doi:10.4230/LIPIcs.ICALP.2016.131.

[CGKL21] Vincent Cohen-Addad, Anupam Gupta, Philip N. Klein, and Jason Li. A quasipolynomial $(2 + \varepsilon)$-approximation for planar sparsest cut. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, 2021. doi:10.1145/3406325.3451103.

[CGMZ16] T.-H. Hubert Chan, Anupam Gupta, Bruce M. Maggs, and Shuheng Zhou. On hierarchical routing in doubling metrics. *ACM Trans. Algorithms*, 12(4):55:1–55:22, 2016. Preliminary version appeared in SODA 2005.

[CK15] Julia Chuzhoy and David H. K. Kim. On approximating node-disjoint paths in grids. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2015)*, Leibniz International Proceedings in Informatics (LIPIcs), pages 187–211, 2015. URL: http://drops.dagstuhl.de/opus/volltexte/2015/5303, doi:10.4230/LIPIcs.APPROX-RANDOM.2015.187.

[CKT22a] Hsien-Chih Chang, Robert Krauthgamer, and Zihan Tan. Near-linear ϵ-emulators for planar graphs. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, 2022. doi:10.1145/3519935.3519998.

[CKT22b] Hsien-Chih Chang, Robert Krauthgamer, and Zihan Tan. Near-linear ε-emulators for planar graphs. arXiv:2206.10681, 2022.

[Die17] Reinhard Diestel. *Graph Theory*. Number 173 in Graduate Texts in Mathematics. Springer Berlin Heidelberg : Imprint: Springer, Berlin, Heidelberg, 5th ed. edition, 2017. URL: https://diestel-graph-theory.com/, doi:10.1007/978-3-662-53622-3.

[DJM+20] Vida Dujmović, Gwenaël Joret, Piotr Micek, Pat Morin, Torsten Ueckerdt, and David R. Wood. Planar graphs have bounded queue-number. *Journal of the ACM*, 67(4):1–38, 2020. doi:10.1145/3385731.

[DPBF09] Frederic Dorn, Eelko Penninkx, Hans L. Bodlaender, and Fedor V. Fomin. Efficient exact algorithms on planar graphs: Exploiting sphere cut decompositions. *Algorithmica*, 58(3):790–810, 2009. doi:10.1007/s00453-009-9296-1.

[EKM13] David Eisenstat, Philip N. Klein, and Claire Mathieu. Approximating k-center in planar graphs. In *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms*, 2013. doi:10.1137/1.9781611973402.47.

[Fed87] Greg N. Federickson. Fast algorithms for shortest paths in planar graphs, with applications. *SIAM Journal on Computing*, 16(6):1004–1022, 1987. doi:10.1137/0216064.

[FGNW17] Ofer Freedman, Paweł Gawrychowski, Patrick K. Nicholson, and Oren Weimann. Optimal distance labeling schemes for trees. In *Proceedings of the ACM Symposium on Principles of Distributed Computing*, 2017. doi:10.1145/3087801.3087804.

[FIK+23] Tobias Friedrich, Davis Issac, Nikhil Kumar, Nadym Mallek, and Ziena Zeif. Approximate max-flow min-multicut theorem for graphs of bounded treewidth. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, STOC '23, 2023. doi:10.1145/3564246.3585150.

[Fil20a] Arnold Filtser. A face cover perspective to ℓ_1 embeddings of planar graphs. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1945–1954. 2020. doi:10.1137/1.9781611975994.120.

[Fil20b] Arnold Filtser. Scattering and sparse partitions, and their applications. In *47th International Colloquium on Automata, Languages, and Programming (ICALP 2020)*, volume 168 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 47:1–47:20, 2020. URL: https://drops.dagstuhl.de/opus/volltexte/2020/12454, doi:10.4230/LIPIcs.ICALP.2020.47.

[FKS19] Eli Fox-Epstein, Philip N. Klein, and Aaron Schild. Embedding planar graphs into low-treewidth graphs with applications to efficient approximation schemes for metric problems. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1069–1088. 2019. doi:10.1137/1.9781611975482.66.

[FL22a] Arnold Filtser and Hung Le. Locality-sensitive orderings and applications to reliable spanners. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2022, page 1066–1079, 2022. doi:10.1145/3519935.3520042.

[FL22b] Arnold Filtser and Hung Le. Low treewidth embeddings of planar and minor-free metrics. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science, FOCS' 22*, 2022. doi:10.1109/focs54457.2022.00105.

[FRT04] Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. A tight bound on approximating arbitrary metrics by tree metrics. *J. Comput. Syst. Sci.*, 69(3):485–497, 2004. doi:10.1016/j.jcss.2004.04.011.

[GH15] Mohsen Ghaffari and Bernhard Haeupler. Distributed algorithms for planar networks II: Low-congestion shortcuts, MST, and min-cut. In *Proceedings of the 27th Annual ACM-SIAM Symposium on Discrete Algorithms*, 2015. doi:10.1137/1.9781611974331.ch16.

[GH16] Mohsen Ghaffari and Bernhard Haeupler. Distributed algorithms for planar networks I. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing*, 2016. doi:10.1145/2933057.2933109.

[GKR01] A. Gupta, A. Kumar, and R. Rastogi. Traveling with a Pez dispenser (or, routing issues in MPLS). In *Proceedings 42nd IEEE Symposium on Foundations of Computer Science,FOCS' 01*, 2001. doi:10.1109/sfcs.2001.959889.

[GNRS04] Anupam Gupta, Ilan Newman, Yuri Rabinovich, and Alistair Sinclair. Cuts, trees and ℓ_1-embeddings of graphs. *Combinatorica*, 24(2):233–269, 2004. doi:10.1007/s00493-004-0015-x.

[GPPR04] Cyril Gavoille, David Peleg, Stéphane Pérennes, and Ran Raz. Distance labeling in graphs. *Journal of Algorithms*, 53(1):85–112, 2004. doi:10.1016/j.jalgor.2004.05.002.

[GR08] L. Gottlieb and L. Roditty. Improved algorithms for fully dynamic geometric spanners and geometric routing. In *Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'08, pages 591–600, 2008. doi:10.5555/1347082.1347148.

[GU16] Paweł Gawrychowski and Przemysław Uznański. A note on distance labeling in planar graphs. *arXiv preprint arXiv:1611.06529*, 2016. doi:10.48550/arXiv.1611.06529.

[HIZ16] Bernhard Haeupler, Taisuke Izumi, and Goran Zuzic. Low-congestion shortcuts without embedding. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing*, 2016. doi:10.1145/2933057.2933112.

[HP00] Yehuda Hassin and David Peleg. Sparse communication networks and efficient routing in the plane (extended abstract). In *PODC*, pages 41–50. ACM, 2000.

[HPM06] S. Har-Peled and M. Mendel. Fast construction of nets in low-dimensional metrics and their applications. *SIAM Journal on Computing*, 35(5):1148–1184, 2006.

[HT84] D. Harel and R. E. Tarjan. Fast algorithms for finding nearest common ancestors. *SIAM Journal on Computing*, 13(2):338–355, 1984. doi:10.1137/0213024.

[Kle02] Philip Klein. Preprocessing an undirected planar network to enable fast approximate distance queries. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '02, page 820–827, 2002.

[KLMN05] Robert Krauthgamer, James R. Lee, Manor Mendel, and Assaf Naor. Measured descent: a new embedding method for finite metrics. *Geometric and Functional Analysis*, 15(4):839–858, 2005. URL: http://dx.doi.org/10.1007/s00039-005-0527-6, doi:10.1007/s00039-005-0527-6.

[KLMS22] O. Kahalon, H. Le, L. Milenkovic, and S. Solomon. Can't see the forest for the trees: Navigating metric spaces by bounded hop-diameter spanners. In *Proceedings of the 41st Symposium on Principles of Distributed Computing (to appear)*, PODC '22, 2022. doi:10.48550/ARXIV.2107.14221.

[Kos82] A. V. Kostochka. A lower bound for the Hadwiger number of a graph as a function of the average degree of its vertices. *Diskret. Analiz. Novosibirsk*, 38:37–58, 1982.

[KPR93] P. N. Klein, S. A. Plotkin, and S. Rao. Excluded minors, network decomposition, and multicommodity flow. In *Proceedings of the 25th Annual ACM Symposium on Theory of Computing*, STOC '93, page 682–690, 1993. doi:10.1145/167088.167261.

[LLR95] Nathan Linial, Eran London, and Yuri Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15(2):215–245, 1995. doi:10.1007/bf01200757.

[LP19] Jason Li and Merav Parter. Planar diameter via metric compression. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC' 19.*, 2019. doi:10.1145/3313276.3316358.

[LT79] R. Lipton and R. Tarjan. A separator theorem for planar graphs. *SIAM Journal on Applied Mathematics*, 36(2):177–189, 1979.

[LWN21] H. Le and C. Wulff-Nilsen. Optimal approximate distance oracle for planar graphs. *To appear in the Proceedings the 62nd Annual Symposium on Foundations of Computer Science*, 2021. https://arxiv.org/abs/2111.03560.

[Mil86] G. L. Miller. Finding small simple cycle separators for 2-connected planar graphs. *Journal of Computer and System Sciences*, 32(3):265 – 279, 1986. doi:https://doi.org/10.1016/0022-0000(86)90030-9.

[MN06] Manor Mendel and Assaf Naor. Ramsey partitions and proximity data structures. In *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS' 06)*, 2006. doi:10.1109/focs.2006.65.

[NR02] Ilan Newman and Yuri Rabinovich. A lower bound on the distortion of embedding planar metrics into Euclidean space. *Discrete and Computational Geometry*, 29(1):77–81, 2002. doi:10.1007/s00454-002-2813-5.

[NW64] C. St.J. A. Nash-Williams. Decomposition of finite graphs into forests. *Journal of the London Mathematical Society*, 1(1):12–12, 1964.

[Pel00] David Peleg. Proximity-preserving labeling schemes. *J. Graph Theory*, 33(3):167–176, mar 2000. URL: https://dl.acm.org/doi/10.5555/1379811.1379818, doi:10.5555/1379811.1379818.

[Rao99] Satish Rao. Small distortion and volume preserving embeddings for planar and Euclidean metrics. In *Proceedings of the 15th Annual Symposium on Computational Geometry*, 1999. doi:10.1145/304893.304983.

[RS86] Neil Robertson and P. D. Seymour. Graph minors. V. Excluding a planar graph. *Journal of Combinatorial Theory, Series B*, 41(1):92–114, 1986. doi:10.1016/0095-8956(86)90030-4.

[Sli05] Aleksandrs Slivkins. Distance estimation and object location via rings of neighbors. In *PODC*, pages 41–50. ACM, 2005.

[Sol13] Shay Solomon. Sparse euclidean spanners with tiny diameter. *ACM Transactions on Algorithms*, 9(3):1–33, 2013. doi:10.1145/2483699.2483708.

[Tal04] Kunal Talwar. Bypassing the embedding: algorithms for low dimensional metrics. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing, Chicago, IL, USA, June 13-16, 2004*, pages 281–290, 2004. doi:10.1145/1007352.1007399.

[Tho04] Mikkel Thorup. Compact oracles for reachability and approximate distances in planar digraphs. *Journal of the ACM*, 51(6):993–1024, 2004. doi:10.1145/1039488.1039493.

[TZ01] M. Thorup and U. Zwick. Approximate distance oracles. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing*, STOC'01, pages 183–192, 2001.

[Van76] J. VanLeeuwen. Finding lowest common ancestors in less than logarithmic time, 1976.

[WN16] C. Wulff-Nilsen. Approximate distance oracles for planar graphs with improved query time-space tradeoff. In *Proceedings of the 27th Annual ACM-SIAM Symposium on Discrete Algorithm*, SODA '16, page 351–362, 2016. doi:10.1137/1.9781611974331.ch26.