An $O(n \log n)$ Approximation Scheme for Steiner Tree in Planar Graphs

GLENCORA BORRADAILE, PHILIP KLEIN and CLAIRE MATHIEU
Brown University

We give a polynomial-time approximation scheme (PTAS) for the Steiner tree problem in planar graphs. The running time is $O(n \log n)$.

Categories and Subject Descriptors: F. Theory of Computation [**F.2 ANALYSIS OF ALGORITHMS AND PROBLEM COMPLEXITY**]: F.2.2 Nonnumerical Algorithms and Problems

General Terms: Algorithms

Additional Key Words and Phrases: Steiner tree, planar graphs, approximation scheme

1. INTRODUCTION

Given a graph with edge lengths, and given a subset Q of vertices called *terminals*, the *Steiner tree problem in networks* aims to find a minimum-length connected subgraph that spans all vertices in Q. The minimum spanning tree problem is the special case where every vertex in the graph is a terminal.

The Steiner tree problem in networks is one of the most well-studied problems in combinatorial optimization. It was one of Karp's original NP-complete problems [Karp 1975], and is now known to be max SNP-complete [Bern and Plassmann 1989], so there is no polynomial-time approximation scheme for the problem unless P=NP. 2-approximation algorithms have been presented in [Takahashi and Matsuyama 1980; Kou et al. 1981] (among others), with improvedments to running time in [Wu et al. 1986; Widmayer 1986; Mehlhorn 1988]. The approximation ratio has been improved by [Zelikovsky 1993; Berman and Ramaiyer 1994; Zelikovsky 1994; Prömel and Steger 1997; Karpinski and Zelikovsky 1997; Hougardy and Prömel 1999], leading to a 1.55-approximation [Robins and Zelikovsky 2005].

A natural restriction on the Steiner tree problem in networks is to require that the input vertices are points in the Euclidean plane (or in low-dimensional Euclidean space) and the lengths are Euclidean distances. For the Euclidean Steiner tree problem, Arora [Arora 1998] and Mitchell [Mitchell 1999] gave polynomial-time approximation schemes (the running time of [Arora 1998] is near-linear in n with a

Glencora Borradaile was supported by NSERC PGS-D, Rowland-Lloyd CST, Kanellakis, and Brown University Dissertation Fellowships.
Philip Klein was supported by NSF Grant CCF-0635089.

polylog factor whose degree depends on ϵ). Rao and Smith [Rao and Smith 1998] gave an $O(n \log n)$-time approximation scheme. In this Euclidean case, n denotes the number of terminals.

Another natural restriction on the Steiner tree problem in networks is to require that the input graph be planar. Even in this restricted case, the problem is NP-hard [Garey and Johnson 1977]. Baker, in a very influential paper [Baker 1994], gave a general approach for deriving approximation schemes for planar graphs. The approach is useful for problems without global connectivity requirements. Demaine and Hajiaghayi [Demaine and Hajiaghayi 2005] showed how to derive other approximation schemes, for planar graphs and generalizations. However, until now, the best approximation ratio known for the Steiner tree problem in planar graphs was no better than for general graphs.

Here, we first give a different kind of result: a *spanner* type result for Steiner trees. For a graph G and terminal set Q, let $\mathrm{OPT}(G,Q)$ denote the minimum length of a Steiner tree in G that spans all vertices in Q. For a fixed number $0 < \epsilon < 1$, a subgraph H of G is a *Steiner-tree spanner* with respect to Q if it has the following two properties:

Spanning Property:. There is a connected subgraph of H that spans Q and has length at most $(1+\epsilon)\mathrm{OPT}(G,Q)$ (Lemma 4.2.)

Shortness Property:. The total length of H is at most some function $f(\epsilon)$ times $\mathrm{OPT}(G,Q)$ (Lemma 4.1.)

THEOREM 1.1. *For any $\epsilon > 0$, there is an algorithm that, given a planar graph G with edge-lengths and a set Q of vertices of G, finds a Steiner-tree spanner. The running time is $O(n \log n)$, where n is the number of vertices of G.*

We prove the above theorem for $f(\epsilon) = 2^{\mathrm{poly}(1/\epsilon)}$ in Section 4. The complete running time to build the spanner, including the dependence on ϵ, is $O(2^{\mathrm{poly}(1/\epsilon)}n + n \log n)$.

Such a spanner-type result can be used to obtain an $O(n \log n)$ approximation scheme, following the approach of Klein [Klein 2005a;]. (The resulting running time is $O(2^{2^{\mathrm{poly}(1/\epsilon)}}n + n \log n)$.)

The idea of using such a spanner-type result for an approximation scheme was also used by Arora, Grigni, Karger, Klein, and Woloszyn [Arora et al. 1998], and by Rao and Smith [Rao and Smith 1998]. In fact, Rao and Smith gave a version of Theorem 1.1 for fixed-dimensional Euclidean space. (Their construction was in fact more powerful, in that the spanner included a nearly optimal Steiner tree spanning *any* subset of Q.)

In our second and main result, we exploit a main structural theorem (Theorem 3.2), used in proving Theorem 1.1, to give an approximation scheme. The algorithm is much faster than what can be achieved using Theorem 1.1 as a black box. Namely, our algorithm is singly exponential, instead of doubly exponential, in $\mathrm{poly}(1/\epsilon)$.

THEOREM 1.2. *For any $\epsilon > 0$, there is an algorithm that, given a planar graph G with edge lengths and a set Q of vertices of G, finds a Steiner tree that spans Q and whose length is at most $1+\epsilon$ times the length of the optimal Steiner tree* *spanning Q. The running time is $O(n \log n)$, where n is the number of vertices of G.*

More specifically, the running time, including the dependence on ϵ, is $O(2^{\text{poly}(1/\epsilon)}n + n \log n)$. This theorem is proved in Section 5.4.

Both the approximation scheme and the spanner construction algorithm use a dynamic-programming algorithm of Erickson, Monma, and Veinott [Erickson et al. 1987] to find the optimal Steiner tree in a planar embedded graph for a set of terminals on the boundary of a single face. We summarize their result in the following theorem:

Theorem 1.3. **[Erickson et al. 1987]** *Let G be a planar embedded graph and Q be a set of k terminals that all lie on the boundary of a single face. Then there is an algorithm*[1] *to find an optimal Steiner tree of Q in G in time $O(nk^3 + (n \log n)k^2)$.*

The algorithm of [Erickson et al. 1987] uses as a subroutine an algorithm for computing single-source shortest paths. Using instead the linear-time planarity-exploiting algorithm of [Henzinger et al. 1997] as a subroutine, one can improve the running time to $O(nk^3)$.

This article is based on two conference papers. In the first [Borradaile et al. 2007], we proved Theorem 1.1 for a function $f(\epsilon)$ that is doubly exponential in $1/\epsilon$. We showed that this result, combined with the framework of [Klein 2005a], yielded an $O(n \log n)$ approximation scheme. However, the resulting dependence of the approximation scheme's running time on ϵ^{-1} was triply exponential.

In the second conference paper [Borradaile et al. 2007], we showed how to improve the spanner result so that $f(\epsilon)$ was only singly exponential. We also developed a different approach to obtaining an approximation scheme, one that more directly uses the structure theorem underlying the spanner construction. The different approach yielded an approximation scheme for which the running time was singly exponential in $\text{poly}(1/\epsilon)$. The approach turns out to be more generally applicable to other connectivity problems, which we mention at the end of the paper.

Outline of this paper

For notational convenience, we give a spanner construction H with a slightly weakened version of the spanning property:

> There is a connected subgraph of H that spans Q and has length at most $(1 + c\epsilon)\text{OPT}(G, Q)$, where c is an absolute constant.

The difference between this property and the spanning property is the constant c. To prove Theorem 1.1, for any given $\bar{\epsilon} > 0$, we assign $\epsilon = \bar{\epsilon}/c$ and carry out the construction; the result is a subgraph H that satisfies the original spanning property and shortness property with respect to $\bar{\epsilon}$.

[1]This algorithm has been generalized by Bern [Bern 1990] and by Bern and Bienstock [Bern and Bienstock 1991] to handle some additional special cases, e.g. where the terminals lie on a constant number of faces. Provan [Provan 1988b; 1988a] used the same approach to give exact and approximate algorithms for some geometric special cases.

Similarly, we show that, for any $\epsilon > 0$, there is an $O(n \log n)$ algorithm to find a Steiner tree whose length is at most $(1 + 2c\epsilon)\text{OPT}(G, Q)$, where c is a constant. To prove Theorem 1.2, for any given $\bar{\epsilon} > 0$, we set $\epsilon = \bar{\epsilon}/2c$ and invoke the algorithm.

Sections 3 through 5 give a broad outline of the proof, omitting technical elements that are deferred to later sections. In Section 3, we define a particular graph decomposition, define portal vertices, and state our main Structure Theorem (Theorem 3.2). This is used to prove both Theorem 1.1 (Section 4) and Theorem 1.2 (Section 5). The full details of the construction are given in Sections 6 through 9. In Section 10, we prove the Structure Theorem. In Section 4, we complete the proof of Theorem 1.1.

2. PRELIMINARIES

The *boundary of a face* of a planar embedded graph is the set of edges adjacent to the face; it does not always form a simple cycle (Figure 2(a)). The *boundary* ∂H of a planar embedded graph H is the set of edges bounding the infinite face. An edge is *strictly enclosed* by the boundary of H if the edge belongs to H but not to ∂H.

Graphs are identified with sets of edges, thus a subgraph H of a graph G is also considered a subset of the edges of G. The set of vertices that are endpoints of edges in H is denoted $V(H)$. For a tree T and vertices $x, y \in V(T)$, we denote the unique simple x-to-y path in T by $T[x, y]$. In particular, if T is a path then $T[x, y]$ is the x-to-y subpath, while $T(x, y]$ denotes $T[x, y]$ with x removed. $P \circ Q$ denotes the concatenation of paths P and Q. We denote the length of a shortest x-to-y path in G as $dist_G(x, y)$.

The dual of a planar graph G is denoted G^*; it is the graph with vertices corresponding to faces of G and edges between adjacent faces.

We assume, without loss of generality, that G_{in} (the input graph) is planar embedded. Such an embedding can be found in $O(n)$ time [Hopcroft and Tarjan 1974]. Further, we assume that G_{in} has degree at most three. This is used in the proof of Lemma 9.2, Lemma 8.5 and Corollary 8.9. Degree three can be achieved by triangulating the dual with edges of large length. The input graph has positive edge-lengths $\ell(\cdot)$ and an identified set of terminal vertices Q. For a set A of edges, we use $\ell(A)$ to denote $\sum_{e \in A} \ell(e)$.

3. GRAPH DECOMPOSITION

In this section, we present a decomposition of the input graph into regions called *bricks* interconnected with a *mortar graph*. This decomposition is used for both Theorem 1.1 and Theorem 1.2. The first few steps of constructing this decomposition (Steps 1(a) through 1(d)) are borrowed from [Klein 2006], for a spanner construction for distances between a subset of vertices of a planar graph.

3.1 Mortar Graph

We first find a connected grid-like subgraph of the input graph G_{in}, based on the set Q of terminals and the given precision ϵ. The subgraph is called the *mortar graph* and is denoted MG (see Figure 1(b)).

Step 1: Construct the mortar graph, MG.

An $O(n \log n)$ algorithm for finding it is given in Section 6. We show in that section that MG satisfies the following two properties:

Terminal Property:. Every terminal in Q is a vertex of MG (Lemma 6.8).

Mortar-Graph Length Property:. The length of MG is at most $9\epsilon^{-1}\mathrm{OPT}(G_{in}, Q)$ (Lemma 6.9).

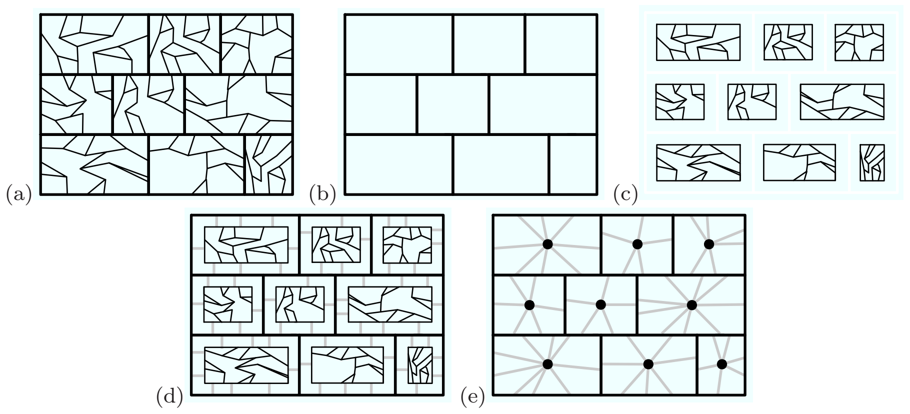

Fig. 1. (a) An input graph G_{in} with bold edges forming the mortar graph MG in (b). (c) The set of bricks corresponding to MG (d) A portal-connected graph, $\mathcal{B}^+(MG)$. The portal edges are grey. (e) $\mathcal{B}^+(MG)$ with the bricks contracted, resulting in $\mathcal{B}^{\div}(\mathcal{B}^+(MG))$. The dark vertices are brick vertices.

3.2 Bricks

Each face f of the mortar graph that strictly encloses at least one edge of G_{in} defines a graph called a *brick*. The brick consists of the edges of G_{in} that are enclosed by the boundary ∂f of f. This boundary is a cycle of edges, possibly with repetition if some edges occur twice in the boundary (an example of such a situation is shown on Figure 2). We duplicate the repeated edges as follows:

> Cut the original graph G_{in} along ∂f, duplicating the edges you cut along (and replicating the vertices), and define the brick to be the subgraph of G_{in} embedded inside that cycle, including the boundary edges according to their multiplicity in ∂f. That is, if an edge occurs twice in the boundary of the face, then there are two copies of that edge in the corresponding brick.

Step 2: Compute the set of bricks, $\mathcal{B}$.

It is easy to see that Step 2 takes $O(n)$.

The boundary ∂B of a brick B is the simple cycle of boundary edges. The corresponding face of MG is called the *mortar boundary* of B. Each edge of the mortar graph occurs at most twice in the disjoint union of the boundaries of the bricks. Since we defined bricks corresponding only to non-empty faces, every brick contains at least one edge not belonging to MG. Figure 1(c) is an example of the set of bricks corresponding to the mortar graph of Figure 1(b). The construction of a brick is illustrated in Figures 2(a) and (b).

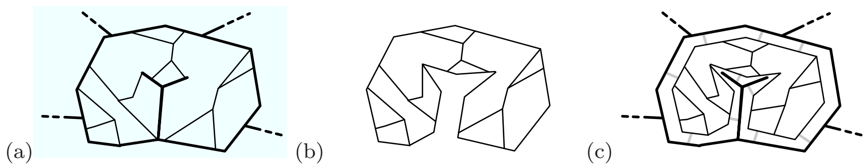

Fig. 2. Construction of a brick: (a) The boundary of a face f of MG is a cycle of edges (thick edges), possibly with repetition (i.e. an edge can occur twice in the boundary). The light edges are those in the interior of f in G_{in}. (b) We obtain the corresponding *brick* via Step 2. The resulting brick B has boundary ∂B. (c) A brick, copied.

3.3 Portals

The next step uses a positive integer parameter θ which depends polynomially on $1/\epsilon$ (Equation (5)).

Step 3: For each brick B, designate some vertices of ∂B as *portals*.

A linear-time greedy algorithm for portal selection is given in Section 7. We show that the portals selected for brick B satisfy the following properties:

Coverage Property:. For any vertex x on ∂B, there is a portal y such that the x-to-y subpath of ∂B has length at most $\ell(\partial B)/\theta$ (Lemma 7.1).

Cardinality Property:. There are at most θ portals on ∂B (Lemma 7.2).

3.4 Portal-connected graph and the operation $\mathcal{B}^+$

In preparation for stating the Structure Theorem, we define an operation called *brick insertion*. For any subgraph G of MG, we derive a planar embedded graph $\mathcal{B}^+(G)$ as follows. For each face f of G corresponding to a brick B, embed a copy of B inside the face f, and, for each portal vertex v of B, connect v in the brick to the corresponding vertex in f, using a zero-length artificial edge (Figure 2(c)). We refer to the artificial edges as *portal edges*. This step is illustrated in Figure 1(d).

We refer to $\mathcal{B}^+(MG)$ as the *portal-connected graph*. Intuitively, this graph is almost the same as the input graph G_{in}, except that artificial zero-cost separations have been added so that paths that connect vertices stricly enclosed by faces of the mortar graph to outside vertices are forced to go through the portals.

Even if a vertex of MG is a terminal, we do not consider its copy on the brick to be a terminal vertex. Thus a brick has no terminals; this is used in Section 9.3.

The following simple lemma follows directly from the fact that each portal edge in $\mathcal{B}^+(MG)$ connects a vertex of a brick to the corresponding vertex of MG.

LEMMA 3.1. *If A is a connected subgraph of $\mathcal{B}^+(MG)$, then $A - \{$portal edges$\}$ is a connected subgraph of G_{in} that spans the same vertices of $\mathcal{B}^+(MG)$.*

The following theorem, proved in Section 10, is central to the proof of correctness of the spanner construction and the approximation scheme. Indeed, taken together, Lemma 3.1 and Theorem 3.2 provide a reduction from the Steiner tree problem in G_{in} to the Steiner tree problem in $\mathcal{B}^+(MG)$.

THEOREM 3.2 STRUCTURE THEOREM. *There exists a constant $\theta = \theta(\epsilon)$ depending polynomially on $1/\epsilon$ such that, for any choice of portals satisfying the Coverage Property, the corresponding portal-connected graph $\mathcal{B}^+(MG)$ satisfies*

$$OPT(\mathcal{B}^+(MG), Q) \leq (1 + c\epsilon)OPT(G_{in}, Q)$$

where c is an absolute constant.

4. SPANNER RESULT: PROOF OF THEOREM 1.1

To construct the spanner we exhaustively compute optimal Steiner trees within each brick.

4.1 Spanner Construction

Step Spanner 4: For each brick B and for each subset X of the portals of B, find the optimal Steiner tree for B and X.

Since the number of terminals of each Steiner-tree problem solved here is at most θ by the Cardinality Property, the remark after Theorem 1.3 implies that the running time of this step is $O(2^\theta \theta^3 n)$.

Step Spanner 5: Return the union of all the edge sets found in Step 4, together with the edges of MG.

This completes the spanner construction.

Combining the running time of Step Spanner 4 with the running times of Steps 1 through 3, gives a total running time for constructing the spanner of $O(2^\theta \theta^3 n + n \log n)$ which is $O(2^{\text{poly}(1/\epsilon)} n + n \log n)$ since θ depends polynomially on $1/\epsilon$.

4.2 Correctness

To complete the analysis, we prove that the output set of edges satisfy the shortness and the spanning properties defined in the introduction.

LEMMA 4.1. *(Shortness property) The total length of the output edges is at most*

$$(1 + 2^{1+\theta})9\epsilon^{-1} OPT(G_{in}, Q).$$

PROOF. For each brick B, the length of any Steiner tree spanning terminals on ∂B is bounded by $\ell(\partial B)$. By construction of the bricks, each edge occurs at most twice as the edge of a brick boundary, so the sum of brick boundary lengths is at most $2\ell(MG)$. For each brick B, the Cardinality Property for B implies that Step Spanner 4 finds at most 2^θ Steiner trees, each of length at most $\ell(\partial B)$, so the total length of all Steiner trees found, summing over bricks B, is at most $2^{1+\theta} \cdot \ell(MG)$. The output also includes each edge of the mortar graph, so the total length of the output is at most $(1 + 2^{1+\theta}) \cdot \ell(MG)$. Appealing to the Mortar-Graph Length Property completes the proof. □

LEMMA 4.2. *(modified Spanning property) The output contains a tree that spans all vertices of Q and has length at most $(1 + c\epsilon)OPT(G_{in}, Q)$ where c is a constant.*

Proof. By the Structure Theorem (Theorem 3.2), there exists a tree T in $\mathcal{B}^+(MG)$ that spans all vertices of Q and has length at most $(1+c\epsilon)\mathrm{OPT}(G_{in}, Q)$. For each brick B, for each connected component K of the intersection of T with B, let Q' be the set of portals of B belonging to K, and replace K with the optimal Steiner tree for B and Q' found in Step Spanner 4. Let T' be the subgraph resulting from all these replacements: we have $\ell(T') \leq \ell(T) \leq (1+c\epsilon)\mathrm{OPT}(G_{in}, Q)$. By Lemma 3.1, the edges of T', not including the portal edges, form a solution to the Steiner tree problem for G_{in} and Q. □

We have now completed the proof of Theorem 1.1.

5. APPROXIMATION SCHEME: PROOF OF THEOREM 1.2

In this section we will give the remaining steps of the approximation scheme.

The approximation scheme makes heavy use of planar duality. For a connected planar embedded graph G, there is another connected planar embedded graph denoted G^*. The faces of G and the vertices of G^* are identified; the edges of the two graphs are identified. We refer to G as the *primal* graph and to G^* as the *dual*.

5.1 Parcels

First we further decompose MG into a set $\mathcal{H}$ of subgraphs called *parcels*. The construction uses a positive integer parameter η is used at the end of Section 5.4. We define

$$\eta = \eta(\epsilon) = 18/c\epsilon^2 \tag{1}$$

where c is the constant appearing in Theorem 3.2 (and will be defined in the proof of Theorem 10.7).

Step 4: Decompose the mortar graph MG into a set $\mathcal{H}$ of parcels.

A linear-time algorithm for Step 4 is given in Section 8. The basic idea, which comes from [Klein 2005a], is to use breadth-first search in the dual of MG, together with the shifting technique of [Baker 1994]. Each parcel is a planar embedded subgraph of MG. Each edge is in at most two parcels. A *boundary edge* is one that belongs to two parcels. We denote the set of boundary edges by $\partial\mathcal{H}$. We say two parcels are *adjacent* if they share a vertex. The *parcel graph* is defined to be the simple graph whose vertex set is the set of parcels, and whose edges are defined by parcel adjacency.

The parcel decomposition has the following properties:

Parcel-Boundary Length. $\ell(\partial\mathcal{H}) \leq \ell(MG)/\eta$ (Corollary 8.2).

Parcel Graph. The parcel graph is a tree (Corollary 8.9).

Parcel Boundary. For two adjacent parcels, the subgraph of shared vertices and edges in MG forms a simple nonempty cycle (Corollary 8.9).

Radius. The planar dual of each parcel has a spanning tree of depth at most $\eta+1$ (Lemma 8.10).

As we show in Section 8, for each parcel H, the Radius Property makes it possible to compute an optimal Steiner tree within $\mathcal{B}^+(H)$ in polynomial time. Our plan is to compute such an optimal Steiner tree for each parcel and take their union to

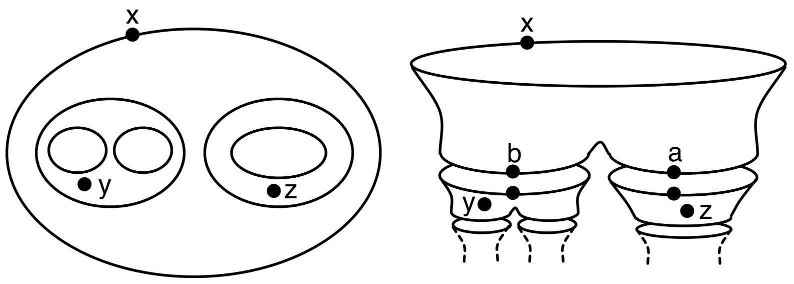

Fig. 3. On the left, the nested cycles that form the boundaries of the parcels are pictured. On the right, we see that parcels drawn according to their level in the breadth-first search. Given original terminals x, y and z (left and right), new terminals a and b must be introduced to connect these through the parcel boundaries (right). Note that when the parcels are separated (as shown on the right), we get two copies of each terminal that is on the boundary of a parcel.

obtain a solution for the original graph. To ensure that the union of the the parcel solutions is a connected subgraph of the original graph, in Step 5 we introduce new terminals on the boundaries of the parcels. Our algorithm for selecting new terminals uses the Parcel Boundary and Parcel Graph Properties. In Section 5.4, we use the Parcel-Boundary Length Property to bound the increase in length due to new terminals and we use the Structure Theorem (Theorem 3.2) to prove that the length of the union is nearly optimal.

5.2 New terminals

The next step is to introduce some *new* terminals. The Parcel-Boundary Length Property, together with Lemma 5.1 below, ensures that connecting to these new terminals does not increase the length of the optimal parcel solution by much. As we show in Lemma 5.2, these new terminals will ensure that the Steiner trees we find within parcels will combine to form a connected subgraph.

A pair of adjacent parcels gives rise to an edge e in the parcel graph. By the Parcel Boundary Property, the common vertices and edges form a simple cycle, and we denote it by C_e.

Step 5: Root the parcel graph at some parcel containing a terminal. For each edge e of the parcel graph that lies on the path from the root to another parcel containing a terminal, designate as a new terminal some vertex of $V(C_e)$.

(Note that the new terminals are vertices of the mortar graph, not of the bricks.)

Recall that $\partial\mathcal{H}$ denotes the set of parcel boundary edges.

Lemma 5.1 Spannability. *Let T be a tree in $\mathcal{B}^+(MG)$ that spans the original terminals and let H be a parcel. Then $T \cup \partial\mathcal{H}$ contains a tree in $\mathcal{B}^+(H)$ that spans the original and new terminals in H.*

Proof. We show that the following set of edges is connected and spans all terminals:

$$E(T) \bigcup \{E(C_e) \;:\; e \text{ lies on the path from the root to a parcel containing a terminal}\} \tag{2}$$

Every original terminal belongs to T. Every new terminal belongs to some cycle C_e appearing in (2). This shows that (2) spans all terminals.

To show that (2) is connected, we show that, for each cycle C_e appearing in (2), some vertex of C_e belongs to T.

Let R denote the root of the parcel graph as chosen in Step 5. Suppose e lies on the path in the parcel graph from the root R to a parcel H containing a terminal. Because R and H both contain terminals, T must contain a path P between a vertex in R and a vertex in H. The path P induces a path $\hat{P}$ in the parcel graph between R and H. By choice of H, this path $\hat{P}$ must contain e. Therefore P contains a vertex common to the two parcels whose adjacency is represented by e. Each such vertex belongs to C_e, so P contains a vertex of C_e. □

LEMMA 5.2 CONNECTING PROPERTY. *For each parcel H, let T_H be a tree in $\mathcal{B}^+(H)$ that spans the original and new terminals in H. (If H has no terminals, then T_H is empty.) Then $\bigcup_H T_H$ is a connected subgraph of $\mathcal{B}^+(MG)$.*

PROOF. Let R be the root chosen in Step 5. We prove that, for each nonnegative integer i,

$$\bigcup \{T_H \;:\; H \text{ at distance} \leq i \text{ from } R \text{ in the parcel graph}\} \tag{3}$$

is connected. The case $i = 0$ is trivial. We prove the induction step as follows.

Assume that (3) is connected. We show that adding T_H for each parcel H at distance $i + 1$ from R preserves connectivity. If T_H is empty, this is trivial. Otherwise, H contains a terminal. Let e be the first edge on the path in the parcel graph from H to R. In Step 5, some vertex v of C_e is designated a new terminal. Therefore v belongs to the subgraph (3) and also to T_H. This shows that adding T_H to the subgraph (3) preserves connectivity. □

5.3 Construction

Step 6: For each parcel H, find an optimal Steiner tree in $\mathcal{B}^+(H)$ spanning the original and new terminals in H.

This step is solved by a $O(k^{\theta\eta} m)$-time dynamic programming algorithm where m is the number of edges in H and k is a constant. The details are in Section 9.

Step 7: Return the union of the edge-sets of all the Steiner trees found in Step 6 (not including portal edges).

This completes the high-level description of the approximation scheme.

The running time of the approximation scheme is given by the sum of the running times of Steps 1 through 6, which is $O(2^{\text{poly}(1/\epsilon)} n + n \log n)$ and will be given by the algorithms details in Sections 6 through 9.

5.4 Correctness

We first need to check that the output is a valid solution to the Steiner tree problem for G_{in} and Q. By the Connecting Property of new terminals (Lemma 5.2), the union of the Steiner trees spans the original terminals in $\mathcal{B}^+(MG)$. By Lemma 3.1, therefore, the output is a connected subgraph of G_{in} that spans the original terminals.

We need to compare the length of the output to the length of an optimal solution. Let $\widehat{Q}$ denote the set of new terminals.

Let T be the optimal Steiner tree in $\mathcal{B}^+(MG)$ spanning the set Q of original terminals. By the Spannability Property of the new terminals (Lemma 5.1), for each parcel H, there is a (possibly empty) tree T_H in $\mathcal{B}^+(H)$ consisting of edges of $T \cup \partial\mathcal{H}$ spanning all the new and original terminals in H. We have:

$$\mathrm{OPT}(\mathcal{B}^+(H), Q \cup \widehat{Q}) \leq \ell\,(T_H) = \ell\,(T_H - \partial\mathcal{H}) + \ell\,(T_H \cap \partial\mathcal{H}).$$

Every edge of T not in $\partial\mathcal{H}$ appears in T_H for exactly one parcel H, and so $\sum_H \ell\,(T_H - \partial\mathcal{H}) \leq \ell\,(T) = \mathrm{OPT}(\mathcal{B}^+(MG), Q)$. Every edge of $\partial\mathcal{H}$ appears in two parcels, and so $\sum_H \ell\,(T_H \cap \partial\mathcal{H}) \leq 2 \cdot \ell\,(\partial\mathcal{H})$. Thus the length of the output is at most

$$\mathrm{OPT}(\mathcal{B}^+(MG), Q) + 2\ell\,(\partial\mathcal{H}).$$

By Theorem 3.2, the first term is bounded by $(1+\epsilon)\ \mathrm{OPT}(G_{in}, Q)$. For the second term, combining the Parcel-Boundary Length Property (Section 5.1), the definition of η (Equation (1)), and the Mortar-Graph Length Property (Section 3), we obtain $\ell\,(\partial\mathcal{H}) \leq \frac{1}{2}c\epsilon\mathrm{OPT}(G_{in}, Q)$, where c is the constant in the statement of Theorem 3.2. Adding the two terms completes the proof of Theorem 1.2.

6. STEP 1: DEFINING THE MORTAR GRAPH

In this section, we give the details for Step 1, construction of the mortar graph.

The construction of the mortar graph MG of G_{in} and Q is illustrated in Figures 1(a) and (b). Steps 1(a) through (d) of the algorithm for constructing MG are identical to the first steps of [Klein 2006] for a subset spanner.

Step 1(a): Find a 2-approximate Steiner tree T spanning Q in G_{in}.

Viewed as a planar embedded graph, the graph induced by the edges of T has a single face whose boundary is an Euler tour that traverses each edge once in each direction.

Step 1(b): Let G_1 be the planar embedded graph obtained by duplicating each edge of T and introducing multiple copies of vertices, thereby transforming the Euler tour corresponding to T into a simple cycle H that encloses no vertices and bounds a single new face.

This process is illustrated in Figure 4. Change the embedding to take this new face to be the infinite face of G_1.

Lemma 6.1. [Klein 2006] $\ell\,(H) \leq 4\,\mathrm{OPT}(G_{in}, Q)$.

Step 1 can be done in $O(n \log n)$ time [Mehlhorn 1988; Widmayer 1986; Wu et al. 1986].

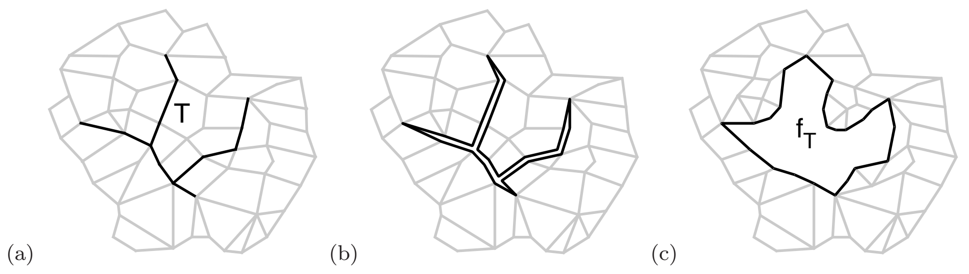

Fig. 4. The process of cutting open a graph along a tree T (a): duplicate edges, replicate vertices (b) and create a new face, f_T (c).

We next decompose G_1 into *strips*. Let $H[x, y]$ denote the x-to-y subpath of ∂G_1 (which is precisely H) in the counterclockwise direction in the embedding. If $x = y$ then by convention $H[x, y] = H$. We use a recursive algorithm to find ϵ-short paths in G_1.

Definition 6.2 *ϵ-short.* A path P in a graph G is *ϵ-short in G* if for every pair of vertices x and y on P, the distance from x to y along P is at most $(1 + \epsilon)$ times the distance from x to y in G: $dist_P(x, y) \leq (1 + \epsilon) dist_G(x, y)$.

Step 1(c): Find vertices x, y on H such that $H[x, y]$ is a minimal subpath of H that is not ϵ-short in H. (Such a pair of vertices always exists since $H[x, y]$ with $x = y$ is not ϵ-short.) Let N be a shortest path from x to y in G_1: the subgraph enclosed by $H[x, y] \cup N$ is called a strip. Recursively decompose the subgraph of G_1 enclosed by $N \cup (H - H[x, y])$ into strips (if this subgraph is nontrivial).

Step 3 is illustrated in Figure 5(a) and (b). The path N (a shortest path) is called the *north* boundary of the strip. The path $H[x, y]$ (whose interior is an ϵ-short path) is called the *south* boundary of the strip, and is denoted S.

Lemma 6.3. (Inequality (10), Klein [Klein 2006]) *The total length of all the boundary edges of all the strips is at most $(\epsilon^{-1} + 1)$ times the length of H.*

Klein [Klein 2006] shows that the strip decomposition of an n-vertex planar graph can be found in $O(n \log n)$ time using the shortest-path algorithm of [Klein 2005b].

In the next step, for each strip we find short paths crossing the strip, called *columns*. Consider a strip, with north and south boundaries N and S. We select vertices $s_0, s_1, \ldots$ on S inductively as follows. Embedding the strip with the north boundary above the south boundary, we direct S and N from left to right. Let s_0 be the left endpoint common to S and N. By convention column C_0 is defined to be the (empty) shortest path from s_0 to N.

Step 1(d): For $i = 1, 2, \ldots$, find the first vertex s_i on S (in left-to-right order) such that the distance from s_{i-1} to s_i along S is greater than ϵ times the distance from s_i to N in the strip: $dist_S(s_{i-1}, s_i) > \epsilon\, dist_{\text{strip}}(s_i, N)$ Column C_i is defined to be the shortest path in the strip from s_i to N.

We also include as a column the trivial path starting and ending at the rightmost vertex common to S and N. Columns are illustrated in Figure 5(c).

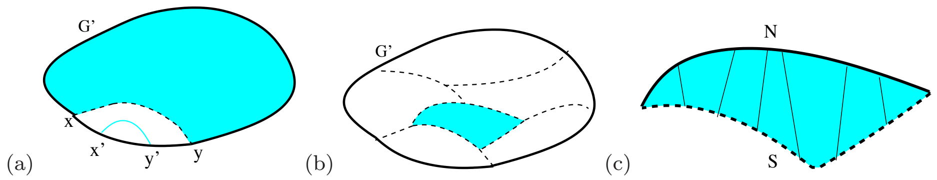

Fig. 5. (a) The first strip is created by a path (dashed) connecting x to y. The distance between every pair of vertices, x' and y', between x and y on the boundary is well approximated by the boundary distance. We recurse on the shaded face. (b) A graph is divided into strips (by the dashed lines). One strip is shaded and enlarged in (c). Columns (vertical lines) are taken from the set of shortest paths from the lower, south boundary S (dashed) to the upper, north boundary N (solid).

LEMMA 6.4 LEMMA 5.2, KLEIN [KLEIN 2006]. *The sum of the lengths of the columns in a strip is at most $\epsilon^{-1}\ell(S)$.*

Klein [Klein 2006] describes how to reduce the problem of finding the columns in a strip to a single shortest-path computation in the strip. It is therefore easy to find all the columns in $O(n)$ time.

Let

$$\kappa = \kappa(\epsilon) = 4\epsilon^{-2}(1+\epsilon^{-1}). \tag{4}$$

In the next step, for each strip we select a subset of the columns $C_0, C_1, \ldots, C_s$ of that strip as follows:

Step 1(e): Let $\mathcal{C}_i = C_i \cup C_{i+\kappa} + \cup C_{i+2\kappa} \cup \ldots$ for $i \in \{0, 1, \ldots, \kappa - 1\}$. Let i^* be the index that minimizes $\ell(\mathcal{C}_i)$. We designate the columns in $\mathcal{C}_{i^*}$ as *supercolumns.*

LEMMA 6.5. *The sum of lengths of the supercolumns in a strip is at most $1/\kappa$ times the sum of the lengths of the columns in the strip.*

LEMMA 6.6. *The sum over the strips of the lengths of all the supercolumns is at most $\epsilon\,\mathrm{OPT}(G_{in}, Q)$.*

PROOF. Combine Lemmas 6.1, 6.3, 6.4, 6.5 and Equation (4). □

Definition 6.7 *Mortar graph.* We define the *mortar graph MG* of G_{in} to be the planar embedded subgraph consisting of the edges of the 2-approximate Steiner tree T of Step 1, the edges of the shortest paths used in Step 2, and the edges of the supercolumns in Step 5.

The mortar graph is illustrated in Figure 1(b).

LEMMA 6.8. *(Terminal Property) Every terminal in Q is a vertex of MG.*

In fact every terminal is a vertex of a strip boundary. This establishes the Terminal Property given in the overview of the algorithm. Next we establish the Length Property. We bound the total length of MG as follows:

Lemma 6.9. *(Mortar-Graph Length Property) The length of MG is at most* $9\epsilon^{-1} OPT(G_{in}, Q)$.

Proof. From Lemma 6.1 and Lemma 6.3, we have that the lengths of the strip boundaries is at most $4(\epsilon^{-1}+1)\mathrm{OPT}(G_{in}, Q)$. From Lemma 6.6, the lengths of the supercolumns is at most $\epsilon\,\mathrm{OPT}(G_{in}, Q)$. Adding those quantities yields

$$\ell\,(MG) \leq 4(1+\epsilon+\epsilon^2/4)\epsilon^{-1}\mathrm{OPT}(G_{in}, Q),$$

hence the Lemma for $\epsilon < 1$. $\square$

The construction of the mortar graph takes $O(n \log n)$ time (independent of ϵ) in total.

The boundary of a brick B is the union of subpaths N_B and S_B of the north and south boundaries of a strip and two supercolumns E_B and W_B (east and west). The construction implies the following lemma, which summarizes the properties of a brick.

Lemma 6.10. *The boundary of a brick B, in counterclockwise order, is the concatenation of four paths $W_B \cup S_B \cup E_B \cup N_B$ such that:*

(1) *The set of edges $B - \partial B$ is nonempty.*
(2) *Every terminal of Q that is in B is on N_B or on S_B.*
(3) *N_B is 0-short in B, and every proper subpath of S_B is ϵ-short in B.*
(4) *There exists a number $k \leq \kappa$ and vertices $s_0, s_1, s_2, \ldots, s_k$ ordered from west to east along S_B such that, for any vertex x of $S_B[s_i, s_{i+1})$, the distance from x to s_i along S_B is less than ϵ times the distance from x to N_B in B: $dist_{S_B}(x, s_i) < \epsilon\, dist_B(x, N_B)$.*

7. STEP 3: DESIGNATING PORTALS

In this section, we give the details for Step 3, portal selection.

For each brick B, we designate a subset of the vertices of the boundary ∂B as portals of the brick. Let

$$\theta = \theta(\epsilon) = 2\alpha(\epsilon) 9\epsilon^{-2}, \tag{5}$$

where $\alpha(\epsilon)$ will be defined in Theorem 10.7. We use the following greedy algorithm:

```
Step 3(a):

Let $v_0 \in V(\partial B)$ be the endpoint of an edge strictly enclosed by $\partial B$.
Designate $v_0$ as a portal vertex.
Set $i = 0$.
Repeat:
    Let $v_i$ be the first vertex of $\partial B$ such that $\ell\,(\partial B[v_{i-1}, v_i]) > \ell\,(\partial B)/\theta$.
    If $v_0 \in V(\partial B(v_{i-1}, v_i])$, stop.
    Otherwise, designate $v_i$ as a portal vertex.
    Set $i = i + 1$.
```

The following lemma follows trivially:

LEMMA 7.1. *(Coverage Property) For any vertex x on ∂B, there is a portal y such that the x-to-y subpath of ∂B has length at most $\ell(\partial B)/\theta$.*

LEMMA 7.2. *(Cardinality Property) There are at most θ portals on ∂B.*

PROOF. Suppose there are p iterations. Each iteration selects a subpath of length more than $\ell(\partial B)/\theta$, so we have $\ell(\partial B) \geq \sum_{i=1}^{p} \ell(\partial B[v_{i-1}, v_i]) > p\,\ell(\partial B)/\theta$, and so it follows that $p < \theta$. □

The running time of this step is linear.

8. STEP 4: DECOMPOSITION INTO PARCELS

In this section, we give details for Step 4, decomposition into parcels.

Let t_0 be a terminal, and let r be a face of MG whose boundary includes t_0.

Step 4(a): Do breadth-first search in the planar dual MG^* starting from r.

Define the *level* of a vertex of MG^* (face of MG) as its breadth-first-search distance from r. Let E_i denote the set of edges whose two endpoints are at levels i and $i+1$. Recall the parameter η defined in Equation (1).

Step 4(b): For $k = 0, 1, \ldots, \eta - 1$, let $\mathcal{E}_k = E_k \cup E_{k+\eta} \cup E_{k+2\eta} \cup \ldots$. Let k^* be the index that minimizes $\ell(\mathcal{E}_k)$.

Let $\mathcal{Y}$ denote the set of connected components of $MG^* - \mathcal{E}_{k^*}$. For each $Y \in \mathcal{Y}$, let H_Y denote the subgraph of MG consisting of the boundaries of faces in $V(Y_K)$ The set of *parcels* of MG is $\mathcal{H} = \{H_Y \ : \ Y \in \mathcal{Y}\}$

Step 4(c): Find the set $\mathcal{H}$ of parcels of MG.

This step takes $O(n)$ time.

Next we prove the Parcel-Boundary Length, Parcel Boundary, Parcel Graph, and Radius Properties given in Section 5.1. Except for Corollary 8.2 given below, the rest of this section gives formal proofs of properties which are intuitively obvious considering Figure 3. (The cursory reader may wish to simply stare at that figure until the Parcel Boundary, Parcel Graph, and Radius Properties are believable.)

LEMMA 8.1. *Every boundary edge belongs to $\mathcal{E}_{k^*}$.*

PROOF. Suppose e is a boundary edge, belonging to two parcels H_{Y_1} and H_{Y_2}. In MG, e is on the boundary of a face $f_1 \in V(Y_1)$ and a face $f_2 \in V(Y_2)$. In MG^*, e is incident to both vertex f_1 and vertex f_2. Since Y_1 and Y_2 are distinct connected components of $MG^* - \mathcal{E}_{k^*}$, we conclude that $e \in \mathcal{E}_{k^*}$. □

COROLLARY 8.2. *(Boundary Length Property) $\ell(\partial\mathcal{H}) \leq \ell(MG)/\eta$.*

PROOF. By definition of k^*, $\ell(\mathcal{E}_{k^*}) \leq \ell(MG)/\eta$. □

For a graph G and a subset W of vertices, let $\delta(W, G)$ denote the set of edges e of G such that e has exactly one endpoint in W. The following is a classical result in planarity [Whitney 1933]:

LEMMA 8.3. *Let G be a planar embedded graph. If the subgraph of G induced by W is connected and the subgraph of G induced by $V(G) - W$ is also connected, then the edges of $\delta(W, G)$ form a simple cycle in the planar dual G^*.*

For an interval $[i, j]$, we denote by $MG^*[i, j]$ the subgraph of MG^* induced by vertices whose breadth-first-search distance from r is in $[i, j]$.

LEMMA 8.4. *For a positive integer i, and for any connected component K of $MG^*[i, \infty)$, the edges of $\delta(V(K), MG^*)$ form a simple cycle in MG.*

PROOF. Consider any vertex v of MG that is not in K, and let P be the v-to-r path through the breadth-first search tree. Then $P = P_1 \circ P_2$ where P_1 (which might be empty) consists of vertices of level at least i, and P_2 consists of vertices of level less than i. Because v is not in K, P_1 contains no vertices of K. Certainly P_2 contains no vertices of K because every vertex of K has level at least i. Hence P contains no vertices of K.

We have shown that the subgraph of MG^* consisting of vertices not in K is connected. By Lemma 8.3, the edges of $\delta(V(K), MG^*)$ form a simple cycle in MG. $\square$

We denote by C_K the simple cycle of Lemma 8.4. When we consider the face r of MG as the infinite face, the faces enclosed by C_K are exactly the vertices of K.

LEMMA 8.5. *Two vertices u and v of MG^* whose levels are in $[i, j]$ are B connected in $MG^*[i, j]$ iff they are connected in $MG^*[i, \infty)$.*

PROOF. Let P be a u-to-v path in $MG^*[i, \infty)$, chosen to minimize the number of vertices of P that are at levels greater than j. Suppose for a contradiction that this number is positive, and let P' be a maximal subpath of P consisting of vertices at levels greater than j. Let K be the connected component of $MG^*[j + 1, \infty)$ containing P'. The edges of C_K form a simple cycle $e_1\ e_2\ \cdots\ e_g$ in MG. For $j = 1, \ldots, g$, let f_j be the face whose corresponding vertex in MG^* is at level i and whose boundary includes e_j. Note that the vertices of P just before and just after P' are among $f_1, \ldots, f_g$.

For $j = 1, \ldots, g - 1$, since the common endpoint of e_j and e_{j+1} in MG has degree at most three, either f_j and f_{j+1} are the same or they share an edge on their boundaries. This shows that $f_1, \ldots, f_j$ are connected, which contradicts the choice of P. $\square$

For $Y \in \mathcal{Y}$, define the *level* of Y to be the minimum breadth-first-search level of a vertex belonging to Y. For $Y, \hat{Y} \in \mathcal{Y}$, if the level of $\hat{Y}$ is less than that of Y and $\hat{Y}$ is adjacent to Y, we say $\hat{Y}$ is a *parent* of Y.

LEMMA 8.6. *For $Y \in \mathcal{Y}$, if the level of Y is greater than zero then Y has exactly one parent.*

PROOF. Let $i + 1$ be the level of Y, and let u be a vertex of Y whose level is $i + 1$. Since $i + 1 > 0$, u has a parent v in the breadth-first-search tree. Thus at least one component of $MG^*[i - \eta, i]$ is adjacent to Y. Let w be any other level-i vertex that is adjacent to Y. By Lemma 8.5, v and w are connected in $MG^*[i - \eta, i]$. This shows that there is only one component of $MG^*[i - \eta, i]$ adjacent to Y. By the properties of breadth-first-search, no node adjacent to Y has level less than i. $\square$

LEMMA 8.7. *For some $i > 0$, let K be a connected component of $MG^*[i, \infty)$. Suppose that in MG there is a vertex v that is on the boundary of some face in $V(K)$ and also on the boundary of some face not in $V(K)$. Then v is on C_K.*

PROOF. The vertex v is on the boundary of a face enclosed by C_K and a face not enclosed by C_k, so by planarity it must be on C_K. $\square$

LEMMA 8.8. *Suppose $Y \in \mathcal{Y}$ has level greater than 0 and $\hat{Y}$ is the parent of Y. The vertices and edges in MG common to H_Y and $H_{\hat{Y}}$ form a simple cycle in MG.*

PROOF. Let $i + 1$ be the level of Y. Let K be the connected component of $MG^*[i + 1, i + \eta]$ that contains Y. An edge belongs to both H_Y and $H_{\hat{Y}}$ if in MG^* the edge has level i and is incident to Y. Such edges are exactly the edges of $\delta(V(K), MG^*)$, so they form the simple cycle C_K.

Suppose a vertex v is common to H_Y and $H_{\hat{Y}}$. Then it belongs to the boundary of a face in $V(K)$ and a face not in $V(K)$, so by Lemma 8.7 it is on C_K. Conversely, any vertex on C_K is on the boundary of a face of H_Y and a face of $H_{\hat{Y}}$. $\square$

COROLLARY 8.9. *(Parcel Graph Property and Parcel Boundary Property) The parcel graph is a tree, and, for two adjacent parcels, the subgraph of shared vertices and edges in MG forms a simple nonempty cycle.*

PROOF. Let H_{Y_1} and H_{Y_2} be any pair of adjacent parcels. We show that Y_1 is the parent of Y_2 or vice versa. By Lemma 8.6, this shows that the parcel graph is a tree, and Lemma 8.8 shows that the vertices and edges shared by H_{Y_1} and H_{Y_2} form a simple nonempty cycle.

By definition of adjacent parcels, there must exist faces f_1, f_2 of MG such that $f_1 \in V(Y_1)$, $f_2 \in V(Y_2)$, and v is on the boundary of both f_1 and f_2. Since the input graph has degree at most three, v has degree at most three, so it follows that f_1 and f_2 have an edge e in common. By Lemma 8.1, $e \in \mathcal{E}_{k^*}$. Therefore the level of e must be $k^* + i \cdot \eta$ for some nonnegative integer i. That is, the levels of f_1 and f_2 must be $k^* + i \cdot \eta$ and $k^* + i \cdot \eta + 1$, in some order. Therefore the level of one of Y_1, Y_2 must be less than that of the other, so one is the parent of the other. $\square$

LEMMA 8.10. *(Radius Property) The planar dual of each parcel has a spanning tree of depth at most $\eta + 1$.*

PROOF. Consider a component $Y \in \mathcal{Y}$, and let i be the level of Y. . First suppose $i > 0$.

Let K be the connected component of $MG^*[i, \infty)$ that contains Y. The edges of $\delta(V(K), MG^*)$ appear in the parcel H_Y because they are on the boundaries of faces belonging to $V(Y)$. Moreover, by Lemma 8.4 they form a simple cycle C_K in MG. The faces in $V(K)$ (including faces in $V(Y)$ are all embedded on one of the two sides of C_K in MG. Therefore the other side has no edges of H_Y, so C_K is the boundary of a face of H_Y. Let us denote this face by r_Y. The edges comprising its boundary all have level $i - 1$.

Now suppose $i = 0$, which means Y contains the breadth-first-search root r. In this case, the edges of level zero are the edges incident to r in MG^*. These edges form the boundary of a face that is in MG^* and in Y. Let r_Y denote this face.

Let j be the maximum breadth-first-search level of a node in Y. By Step 2, $j-i+1 \leq \eta$. Hence each such node is at most η hops from the face r_Y in the graph H_Y^*.

Finally, let f be a face of H_Y^* that is not r_Y and is not a face of MG. By definition of H_Y, some edge e of the boundary of f also belongs to the boundary of some face $f' \in V(Y)$. Hence f is at most $\eta+1$ hops from r_Y in H_Y^*. $\square$

9. STEP 6: FINDING A STEINER TREE IN THE PARCELS

In this section, we give the details of Step 6. We use dynamic programming.

9.1 Defining the recursion tree

Let H be a parcel in the set of parcels $\mathcal{H}$ found in Step 4. In this section, we will show that for any set of terminals that are in the vertex set of H, there is a dynamic programming algorithm with running time $O(c^{\theta\eta}m)$ to find an optimal Steiner tree in $\mathcal{B}^+(H)$, where m is the number of edges of $\mathcal{B}^+(H)$. Recall that $\mathcal{B}^+(H)$ is obtained by embedding bricks in those faces of H for which bricks are defined and connecting each brick to the corresponding face of H via at most θ portal edges.

By Lemma 8.10, the dual graph H^* has a breadth-first search tree T^* of depth at most $\eta+1$. The following lemma is a classical result in planarity [Sommerville 1929]:

Lemma 9.1. *For any connected planar graph G, the set of edges not belonging to a spanning tree of G^* forms a spanning tree of G.*

Step 6(a): Let T be the set of edges of H not in T^*.

Next we describe another operation, applicable to a graph $\mathcal{B}^+(H)$ obtained from a graph H by applying the brick-insertion operation from Section 3.3. The new operation is called *brick contraction*, and is denoted $\mathcal{B}^{\div}(\mathcal{B}^+(H))$. Starting with $\mathcal{B}^+(H)$, contract each brick to a single vertex, called a *brick vertex*, as illustrated by Figure 1(e). The mortar edges of $\mathcal{B}^+(H)$ are unaffected by this operation. Note that the degree of each brick vertex is the number of portals for the corresponding brick, which is at most θ by Lemma 7.1. The graph $\mathcal{B}^{\div}(\mathcal{B}^+(H))$ differs from H in that it has a single vertex connected to H via portal edges rather than a brick.

Step 6(b): For each face of H that is the mortar boundary of a brick B, let e_B be the portal edge corresponding to the first portal vertex selected for B (see Step 1). Let $\widehat{T} := T \bigcup_B \{e_B\}$.

Observe that $\widehat{T}$ is a spanning tree of $\mathcal{B}^{\div}(\mathcal{B}^+(H))$.

Lemma 9.2. *The degree of $\widehat{T}$ in $\mathcal{B}^{\div}(\mathcal{B}^+(H))$ is at most 3.*

Proof. Note that each vertex of $\widehat{T}$ is either a vertex of H or the result of a brick contraction.

If a vertex v of $\widehat{T}$ is obtained by contracting a brick B, then e_B is the only edge in $\widehat{T}$ adjacent to v: v has degree 1.

If v is a vertex of H, then certainly v has degree at most 3 in G_{in} (by assumption as mentioned in Section 2). So v has degree at most 3 in both H and T, which are subgraphs of G_{in}. Since the first portal for a brick B has degree 3, with one edge enclosed by ∂B and two edges in ∂B contributing to the degree, a vertex can only be the first portal for one brick (see Step 1). If v is the first portal chosen for brick B, then v has degree at most 2 in H, and the addition of e_B gives degree at most 3. □

Root $\widehat{T}$ at a non-brick-vertex of degree at most two. For each vertex v of $\widehat{T}$, let $\widehat{T}(v)$ denote the set of descendents of v in $\widehat{T}$.

Lemma 9.3. *In the graph $\mathcal{B}^{\div}(\mathcal{B}^{+}(H)))$, there are at most $2\theta\eta + 1$ edges between $\widehat{T}(v)$ and the rest of the graph.*

Proof. Let $\widehat{T}^*$ be the set of edges of $\mathcal{B}^{\div}(\mathcal{B}^{+}(H))$ not in $\widehat{T}$. By Lemma 9.1, $\widehat{T}^*$ forms a spanning tree of the dual graph $(\mathcal{B}^{\div}(\mathcal{B}^{+}(H)))^*$. Comparing $\widehat{T}^*$ to the spanning tree T^* of H^*, we see that each vertex of T^* obtained by contracting a brick B corresponds to a path in $\widehat{T}^*$ consisting of all but one of the portal edges associated with B. Such a path has length at most $\theta - 1$. Since the depth of T^* is at most η (Lemma 8.10), the depth of $\widehat{T}^*$ is at most $\theta\eta$.

Now we use an argument from [Klein 2005a]. Let v be any vertex of $\widehat{T}$ other than the root, and let e_v be the edge connecting v to its parent. Then e_v is not in the tree $\widehat{T}^*$. The path in $\widehat{T}^*$ between the endpoints of e_v has at most $2\theta\eta$ edges. Combining this path with e_v yields a simple cycle in the dual graph $(\mathcal{B}^{\div}(\mathcal{B}^{+}(H)))^*$ having at most $2\theta\eta + 1$ edges. The edges of this cycle are exactly the edges in the cut $\delta(V(\widehat{T}(v)), \mathcal{B}^{\div}(\mathcal{B}^{+}(H)))$ in the primal graph (Lemma 8.3). □

9.2 The dynamic programming table

Step 6(c): Solve the Steiner problem on $\mathcal{B}^{+}(H)$ by doing dynamic programming guided by the tree $\widehat{T}$.

For each vertex v of $\widehat{T}$, define

$$f(v) = \begin{cases} B & \text{if } v \text{ is the result of contracting a brick } B \\ v & \text{otherwise} \end{cases},$$

and define $W(v)$ to be the subgraph of $\mathcal{B}^{+}(H)$ induced by $\bigcup\{f(w) \;:\; w \in \widehat{T}(v)\}$. It follows from Lemma 9.3 that $\delta(V(W(v)), \mathcal{B}^{+}(H)) = 2\theta\eta + 1$ which equals $\delta(V(\widehat{T}(v)), \mathcal{B}^{\div}(\mathcal{B}^{+}(H)))$.

Since $\widehat{T} - \widehat{T}(v)$ is connected, the complement of $V(W(v))$ is connected and by Lemma 8.3, $\delta(V(W(v)), \mathcal{B}^{+}(H))$ is a cycle $e_1 e_2 \ldots e_d$ in the dual graph $(\mathcal{B}^{+}(H))^*$. This defines a cyclic ordering of the edges of $\delta(V(W(v)), \mathcal{B}^{+}(H))$. A *noncrossing partition* of $\delta(V(W(v)), \mathcal{B}^{+}(H))$ is a partition in which no two blocks "cross" each other, i.e., if e_i and e_j belong to one block and e_k and e_ℓ to another, in the cyclic ordering they are not arranged in the order $e_i e_k e_j e_\ell$. It follows from Lemma 9.3 and the following well-known Catalan bound that the number of noncrossing partitions is at most $4^{2\theta\eta+1}$:

Lemma 9.4. [Catalan 1838] *The number of non-crossing partitions of $\{1, 2, \ldots, n\}$ is C_n, the nth Catalan number, which is less than $4^n/(n+1)$.*

For a partition $\mathcal{C}$ of $\delta(V(W(v)),\mathcal{B}^+(H))$, we say a subgraph L of $W(v)$ is *consistent* with $\mathcal{C}$ if it satisfies the following *consistency properties*:

(1) For each set $S \in \mathcal{C}$, there is a connected component K_S of L that contains an endpoint of each edge in S.

(2) Every connected component of L that does not contain all terminals of $\mathcal{B}^+(H)$ includes an endpoint of some edge appearing in $\mathcal{C}$.

Define the $\mathcal{C}$-cost of L to be $\sum\{\ell(K_S) + \frac{1}{2}\ell(S) \ : \ S \in \mathcal{C}\}$.

Suppose A is a connected subgraph of $\mathcal{B}^+(H)$ that spans the terminal of H. Let L be the intersection of A with the subgraph $W(v)$. Let the connected components of L be $K_1,\ldots,K_p$. Define S_i to be the set of edges of $A\cap\delta(V(W(v)),\mathcal{B}^+(H))$ incident to vertices of K_i. Then note that L is consistent with the partition $\mathcal{C} = \{S_1,\ldots,S_p\}$, that $\mathcal{C}$ is noncrossing, and that the $\mathcal{C}$-cost of L is $\ell(L)+\frac{1}{2}\ell(A\cap\delta(V(W(v)),\mathcal{B}^+(H)))$.

A subgraph L of $W(v)$ is *terminal-covering* if every terminal in $W(v)$ is in L. Note that if A is a connected subgraph of $\mathcal{B}^+(H)$ that spans its terminals, then the intersection of A with $W(v)$ is terminal-covering.

We define the *subproblem corresponding to* $\widehat{T}(v)$ to be the construction of a table $\textsc{Tab}_v[\cdot]$ indexed by noncrossing partitions $\mathcal{C}$ of $\delta(V(W(v)),\mathcal{B}^+(H))$ such that $\textsc{Tab}_v[\mathcal{C}] = \min_{L\in\mathcal{L}}(\mathcal{C}\text{-cost of } L)$ where the minimum is over all terminal-covering subgraphs L of $W(v)$ that are consistent with C.

9.3 Filling in the entries of the dynamic programming table

Solving the subproblems uses dynamic programming.

First suppose v is a leaf of $\widehat{T}$. If v is a vertex of H then the corresponding subproblem is trivial. If v is the result of having contracted a brick B, then $W(v) = B$ and $\delta(V(W(v)),\mathcal{B}^+(H))$ is the set of portal edges corresponding to B. By Lemma 7.1, there are at most θ such edges. Since H contains all the terminals, B contains no terminals, and every subgraph of B is terminal-covering. For each partition $\mathcal{C} = \{S_1,\ldots,S_p\}$ of the portal edges of B, we compute the value of $\textsc{Tab}_v[\mathcal{C}]$ as follows: for each set S_i, find the optimal Steiner tree spanning the endpoints of S_i in B in $O(\theta^3 m)$ time, where m is the number of edges of B (Theorem 1.3). Let $\textsc{Tab}_v[\mathcal{C}]$ be assigned the sum of the lengths of these trees.[2] The number of noncrossing partitions is at most 4^θ, so the time to populate $\textsc{Tab}_v[\cdot]$ is $O(4^\theta m)$.

Given a collection $\{X_1,\ldots,X_p\}$ of sets of edges, say two edges are related if they appear in the same set X_i. Let $Y_1,\ldots,Y_q$ be the equivalence classes of the transitive closure of this relation. In the following pseudo-code, we refer to $\{Y_1,\ldots,Y_q\}$ as the *transitive closure* of $\{X_1,\ldots,X_p\}$.

Suppose v is not a leaf. Then v is a vertex of H, (i.e. $f(v) = v$), and v has either one or two children in our recursion tree $\widehat{T}$. Let $v_1,\ldots,v_s$ be its children, $1 \le s \le 2$. We use the following algorithm to fill in $\textsc{Tab}_v[\cdot]$:

```
1   Initialize each entry of $\textsc{Tab}_v$ to $\infty$.
2   For each tuple $(\mathcal{C}_0,\mathcal{C}_1,\ldots,\mathcal{C}_s)$ of noncrossing partitions of
    $\delta(\{v\},\mathcal{B}^+(H)),\delta(V(W(v_1)),\mathcal{B}^+(H)),\ldots,\delta(V(W(v_s)),\mathcal{B}^+(H))$,
3       if each edge not in $\delta(V(W(v)),\mathcal{B}^+(H))$ occurs zero or two times in
```

[2]The edges of $\delta(B)$ are portal edges, so have zero length.

```
      $\mathcal{C}_0 \cup \cdots \cup \mathcal{C}_s$, and $\mathcal{C}_0$ is nonempty if $v$ is a terminal,
4       let $\mathcal{C}'$ be the transitive closure of $\mathcal{C}_0 \cup \cdots \cup \mathcal{C}_s$.
5       If every set in $\mathcal{C}$ includes some edge of $\delta(V(W(v)), \mathcal{B}^+(H))$,
6           let $\mathcal{C}$ be the restriction of $\mathcal{C}'$ to $\delta(V(W(v)), \mathcal{B}^+(H))$.
7           Assign $\text{TAB}_v[\mathcal{C}] := \max\{\text{TAB}_v[\mathcal{C}], \sum_{i=1}^{s} \text{TAB}_{v_i}[\mathcal{C}_i] + \frac{1}{2}\ell\left(\bigcup \mathcal{C}_0\right)\}$.
```

To prove correctness, consider a tuple in Line 2. By induction, for each $\mathcal{C}_i$ ($i = 1, \ldots, s$), there is a corresponding subgraph L_i. By unioning these subgraphs with the edges in $\bigcup_{i=0}^{s} \mathcal{C}_i - \delta(V(W(v)), \mathcal{B}^+(H))$, we obtain a subgraph L. The inductive hypothesis and the condition in Line 3 ensure that L is terminal-covering. The inductive hypothesis and the condition in Step 3 imply the first consistency property. The inductive hypothesis and the condition in Line 5 imply the second consistency property.

We now analyze the running time of the dynamic program. Since v has degree at most 3, for a non-leaf vertex v, the number of partitions $\mathcal{C}_0$ is at most 2^3. Since $s \leq 2$, the number of tuples of partitions is at most $4^{2\theta\eta}$. The time required for the dynamic program, not including the leaves, is therefore $O(2^3 \cdot 4^{2\theta\eta} n_1)$, where $n_1 \leq n$ is the number of non-leaf vertices in the tree $\widehat{T}$. The time for all the leaves of $\widehat{T}$ is bounded by $O(4^{\theta} m)$ where $m = O(n)$ is the total number of edges in all bricks B in $\mathcal{B}^+(H)$. The total time is therefore $O(2^{4\theta\eta} n + 2^{2\theta} m) = O(2^{4\theta\eta} n)$ where n is the number of vertices in $\mathcal{B}^+(H)$.

10. PROOF OF THE STRUCTURE THEOREM (THEOREM 3.2)

10.1 Simplifying trees

In this self-contained section, we establish a few combinatorial lemmas that simplify trees whose leaves lie on an ϵ-short path (Definition 6.2). These lemmas will be used in Section 10.2.

LEMMA 10.1. *Let G be a planar embedded graph and let T be a tree in G whose leaves all lie on an ϵ-short path P. There is a subpath of P spanning the vertices of $T \cap P$ whose total length is at most $(1+\epsilon)\ell(T)$.*

PROOF. Let P' be the shortest subpath of P that spans all the vertices of $T \cap P$. There is a path Q in T between the endpoints of T. Since P is ϵ-short, $\ell(P') < (1+\epsilon)\ell(Q) \leq (1+\epsilon)\ell(T)$. □

In the rest of this subsection, G is a planar embedded graph and T is a tree in G rooted at a vertex r with leaves on an ϵ-short path P that is a subpath of the boundary ∂G of G.

LEMMA 10.2. *There is a tree T' that (a) is also rooted at r, (b) spans all the vertices of $T \cap P$, (c) is binary, and (d) has length at most $(1+\epsilon)\ell(T)$,*

PROOF. If T is binary then $T' = T$. Else, let $\mathcal{U}$ be the set of rootmost vertices of T with at least three children, and for each $u \in \mathcal{U}$, let T_u be the subtree of T rooted at u. The trees T_u are disjoint from one another. For each $u \in \mathcal{U}$, replace T_u with T'_u consisting of (1) the subpath P' of P that spans all the leaves of T_u and (2) the shortest u-to-P path. The resulting tree T' satisfies properties (a), (b) and (c) by construction.

To analyze the length of T', note that T_u has length equal to $dist_G(u,P')+\ell(P')$. Since u has at least three children in T, it must be that T_u contains three disjoint paths from u to P', including paths Q_1 and Q_2 to the endpoints of P' and a third path Q_3 to some other vertex of P'. Note that $\ell(Q_1)+\ell(Q_2)$ is at least the distance in G between the endpoints of P' which in turn is at least $\ell(P')/(1+\epsilon)$ because P is ϵ-short. Also, $\ell(Q_3) \geq dist(u,P')$. Combining, we get the following bound on the length of T'_u:

$$\begin{aligned}\ell(T'_u) &= \ell(P') + dist_G(u,P')\\ &< (1+\epsilon)(\ell(Q_1)+\ell(Q_2)) + \ell(Q_3)\\ &< (1+\epsilon)\ell(T_u).\end{aligned}$$

Summing over all $u \in \mathcal{U}$, we infer that $\ell(T') \leq (1+\epsilon)\ell(T)$. □

Definition 10.3 *Joining vertex*. Let H be a subgraph of G such that P is a path in H. A *joining vertex* of H with P is a vertex of P that is the endpoint of an edge of $H-P$.

Lemma 10.4. *There is a tree $\widehat{T}$ that (a) is also rooted at r, (b) spans all the vertices of $T \cap P$, (c) has length at most $(1+4\cdot\epsilon)\ell(T)$, and such that (d) $\widehat{T}$ has at most $11\cdot\epsilon^{-1.45}$ joining vertices with P.*

Proof. Let T' be the tree derived from T in Lemma 10.2. Starting from T', we construct a tree T'' from T' that satisfies the properties of the lemma. We partition the edges of T' into *super-edges*, defined by maximal paths in T' whose internal vertices all have degree 2 in T'. The *level* of a superedge is equal to the number of super-edges traversed when going from the root of T' to the beginning of the super-edge. The endpoints of a super-edge are called *super-vertices* and the level of a super-vertex is equal to the number of super-edges traversed when going from the root of T' to the super-vertex.

Select a level $k>0$ (to be determined shortly). Let $\mathcal{U}$ be the set of all super-vertices at level k. For each $u \in \mathcal{U}$, replace the subtree T'_u of T' rooted at u with another tree T''_u rooted at u that is the union of the shortest subpath P' of P spanning the vertices of $T'_u \cap P$ and the shortest u-to-P' path (Figure 6(a)). After all such replacements, we get a new tree, T'', which satisfies Properties (a) and (b) by construction.

Say that a super-edge (u,v) is a *zig-zag* edge if the two-step path from the parent $p(u)$ of u to u to v, either goes from $p(u)$ to a left child and from u to a right child, or goes from $p(u)$ to a right child and from u to a left child. For each level i, let L_i denote the total length of the zig-zag super-edges at level i.

Note that for each vertex in level $i>0$, exactly one child super-edge is in L_i. Let u be a vertex in level k. Let e be the super-edge in L_k whose parent is u (as illustrated in Figure 6 (a)). Let Q_1 be the path in T' between the endpoints of P' and let Q_2 be the path from u to P that traverses e, makes a zig-zag turn, and subsequently never makes a turn again until reaching P'. We have:

$$\ell(T''_u) = \ell(P') + dist_G(u,P') \leq (1+\epsilon)[\ell(Q_1)+\ell(Q_2)].$$

Note that T''_u contains no turning superedge of level greater than k, and that only

e appears in both Q_1 and Q_2, and so

$$\ell(T'') \le (1+\epsilon)[\ell(T') + L_k - (L_{k+2} + L_{k+3} + L_{k+4} + \cdots)].$$

Claim 10.5. *Let $k_0 = \lceil \log_\Phi(\sqrt{5}/\epsilon + 1)\rceil$ (where Φ is the golden ratio). Then there exists $k \le k_0$ such that $L_k \le \epsilon\ell(T') + L_{k+2} + L_{k+3} + \cdots + L_{k_0}$.*

Proof. Otherwise, for every $k = 1, 2, \ldots, k_0$, we have $L_k > \epsilon\ell(T') + L_{k+2} + L_{k+3} + \cdots + L_{k_0}$, hence $L_1 > \epsilon\ell(T')\mathrm{Fib}_{k_0}$, where Fib_{k_0} is the ${k_0}^{th}$ Fibonacci number, which is greater than $1/\epsilon$. Thus $L_1 > \ell(T')$, a contradiction. □

Choosing k according to the above Lemma yields $\ell(T'') \le (1+\epsilon)^2\ell(T')$, hence Property (c).

Since each vertex of T'' has at most two children, the number of super-vertices at level k is at most

$$2^k \le 2^{1+\log_\Phi(\sqrt{5}(1/\epsilon+1))} < 11 \cdot \epsilon^{-1.45}$$

using our assumption that $\epsilon < 1$, hence Property (d). □

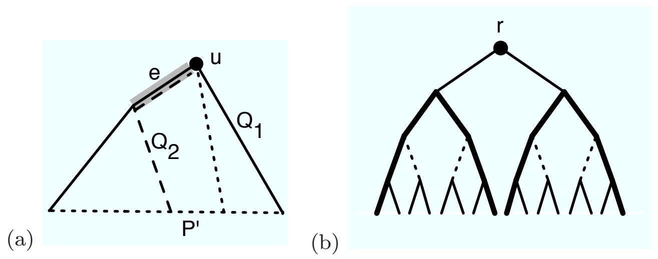

Fig. 6. (a) The dotted tree replaces the subtree rooted at u. The solid path between the endpoints of P' through u is Q_1. The dashed path from u to P' whose first edge is e is Q_2. (b) The bold edges are in E_1. The dotted edges are in L_2.

Lemma 10.6. *Let p and q be two vertices of T. There is another tree $\widehat{T}$ that spans p and q and the vertices of $T \cap P$ such that $\ell(\widehat{T}) \le (1 + c_1\epsilon)\ell(T)$ and $\widehat{T}$ has at most $c_2 \cdot \epsilon^{-2.5}$ joining vertices with P, where c_1 and c_2 are constants.*

Proof. Let Q be the unique p-to-q path in T. Removing the edges of Q from T breaks T into a forest with k trees rooted at vertices of Q and leaves on P: $\mathcal{T} = \{T_1, T_2, \ldots, T_k\}$, numbered according to the order of their leaves along P (which is well defined, since P is on the boundary of the graph). We include as (trivial) trees in this sequence all joining vertices of P with Q. Without loss of generality, assume that p is the root of T_1 and q is the root of T_k. The root of T_i is r_i.

First we define a transformation:

> Let T_a and T_b be such that $a \le b$. Let x be the first vertex of T_a along P and let y be the last vertex of T_b along P. Obtain a tree T' from T by removing $T_a, T_{a+1}, T_{a+2}, \ldots, T_b$ and the r_a-to-r_b subpath of Q and adding the paths from r_a to x in T_a, from x to y along P, and from y to r_b in T_b (as illustrated in Figure 7).

The p-to-q path in T', which we denote Q', is composed of the q-to-r_a subpath of Q, the r_a-to-x path in T_a, the x-to-y path of P, y-to-r_b path in T_b, and the r_b-to-p subpath of Q. By construction, p and q are vertices of T'. Further, this transformation guarantees that T' spans the vertices of $T \cap P$: a tree T_i that is removed from T has leaves on the x-to-y subpath of P, which is included in T'. So, T' spans the vertices required by the Lemma.

The increase in length due to this transformation is given by

$$\Phi = \ell\left(T_a[r_a, x]\right) + \ell\left(T_b[y, r_b]\right) + \ell\left(P[x, y]\right) - \ell\left(Q[r_a, r_b]\right) - \sum_{i=a}^{b} \ell\left(T_i\right).$$

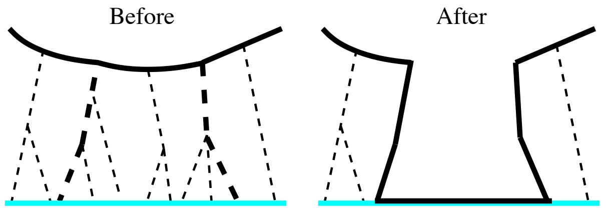

Fig. 7. The path Q is replaced by Q', allowing us to remove the trees rooted between a and b.

Now we find values of a and b for the transformation that reduces the number of joining vertices of the tree with P while not increasing the length of the tree by much. Let P' be the shortest subpath of P that spans the vertices of $T \cap P$. Say a subtree T_i is *light* if $\ell\left(T_i\right) < \epsilon \ell\left(P'\right)$ and *heavy* otherwise. Let $I = \{i \ : \ T_i \text{ is light}\}$. Let

$$w = \begin{cases} (\min I) - 1 + k - (\max I) & \text{if } I \neq \emptyset \\ k & \text{otherwise} \end{cases}$$

Case I, $w \geq \epsilon^{-1} + 2$. In this case, there are at least $\epsilon^{-1} + 2$ heavy trees in $\mathcal{T}$. We apply the transformation described above with $a = 1$ and $b = k$. The increase in length is given by:

$$\begin{aligned}
\Phi &= \ell\left(T_1[p, x]\right) + \ell\left(T_k[y, q]\right) + \ell\left(P[x, y]\right) - \ell\left(Q\right) - \sum_{i=1}^{k} \ell\left(T_i\right) \\
&\leq \ell\left(P'\right) - \sum_{i=2}^{k-1} \ell\left(T_i\right) \\
&< \ell\left(P'\right) - \epsilon^{-1} \epsilon \ell\left(P'\right), \text{ since there are at least } \epsilon^{-1} + 2 \text{ heavy trees} \\
&= 0.
\end{aligned}$$

The resulting tree T' is a single path from p to q containing exactly two joining vertices with P: x and y (since $\mathcal{T}' = \{x, y\}$). Since $\ell\left(T'\right) < \ell\left(T\right)$, T' achieves the properties of the Lemma.

Case II, $w < \epsilon^{-1} + 2$. In this case, we apply the transformation described with $a = \min I$ and $b = \max I$. The increase in length is given by:

$$\begin{aligned}
\Phi &= \ell\left(T_a[r_a, x]\right) + \ell\left(T_b[y, r_b]\right) + \ell\left(P[x, y]\right) - \ell\left(Q[r_a, r_b]\right) - \sum_{i=a}^{b} \ell\left(T_i\right) \\
&\leq 2\epsilon\ell\left(P'\right) + \ell\left(P[x, y]\right) - \underbrace{\left(\ell\left(T_a[r_a, x] \cup Q[r_a, r_b] \cup T_b[y, r_b]\right)\right)}_{\geq \ell\left(P[x,y]\right)/(1+\epsilon),\text{ since } P \text{ is } \epsilon\text{-short}} \\
&= 2\epsilon\ell\left(P'\right) + \ell\left(P[x, y]\right) + \epsilon\ell\left(T_a[r_a, x] \cup Q[r_a, r_b] \cup T_b[y, r_b]\right) - \ell\left(P[x, y]\right) \\
&\leq 2\epsilon(1 + \epsilon)\ell\left(T\right) + \epsilon\ell\left(T\right),
\end{aligned}$$

For the last inequality, observe that there is a path in T between the endpoints of P'; since P' is ϵ-short, $(1 + \epsilon)\ell\left(P'\right)$ is at most the length of this path in T (which in turn is at most the length of T itself).

The new tree T' consists of the p-to-q path Q' and, attached to Q', the trees $T_1, T_2, \ldots, T_{a-1}, T_{b+1}, \ldots, T_k$. The joining vertices of T' with P include p, q, and the joining vertices of all the trees T_i. Though there are fewer than $\epsilon^{-1} + 2$ such trees remaining, each of the trees might itself have many joining vertices with P.

Let T_i' be the tree obtained from T_i by applying Lemma 10.4 with the ϵ-short path P. Obtain a new tree T'' from T' by replacing each tree T_i with T_i'. By Lemma 10.4, there are at most $11\epsilon^{-1.45}$ joining vertices with P per tree T_i'. Since $w < \epsilon^{-1} + 2$, the new tree T'' has at most $11\epsilon^{-1.45}(\epsilon^{-1} + 2) + 2$ joining vertices with P (the extra 2 counts the joining vertices x and y), achieving the last property of the lemma.

By Lemma 10.4, $\ell\left(T_i'\right) < (1 + \epsilon)\ell\left(T_i\right)$, and so $\ell\left(T''\right) < (1 + \epsilon)\ell\left(T'\right)$ which, using the bound on the length of T', is at most $(1 + \epsilon)(1 + c\epsilon)\ell\left(T\right)$, satisfying T', is at most $(1 + \epsilon)(1 + 2\epsilon(2 + \epsilon))\ell\left(T\right)$, satisfying the length bound for the lemma. □

10.2 Simplifying forests inside bricks

Why is our decomposition into bricks useful? Because there exists a near-optimal Steiner tree that crosses the boundary of each face of MG a small number of times. In this subsection, we formally state and prove this property (Theorem 10.7). The proof relies on the Lemmas of Section 10.1 and on Lemma 6.10, and the result will be a key ingredient to the proof of Theorem 3.2.

THEOREM 10.7 STRUCTURAL PROPERTY OF BRICKS. *Let B be a plane graph with boundary $N \cup E \cup S \cup W$, satisfying the brick properties of Lemma 6.10. Let F be a set of edges of B. There is a forest $\tilde{F}$ of B with the following properties:*

—(F1) If two vertices of $N \cup S$ are connected in F then they are connected in $\tilde{F}$.

—(F2) The number of joining vertices of F with both N and S is at most $\alpha(\epsilon)$.

—(F3) $\ell(\tilde{F}) \leq \ell\left(F\right)(1 + c\epsilon)$.

In the above, $\alpha(\epsilon) = o(\epsilon^{-5.5})$ and c is a fixed constant.

The rest of this section is devoted to the proof of Theorem 10.7. Let F_0 be a minimal subgraph of F such that if two vertices of $N \cup S$ are connected in F, then they are connected in F_0: F_0 is a forest whose leaves are all on N and S. We partition F_0 into two forests F_1 and F_2. Let $\mathcal{T}$ be the set of trees obtained from F_0

by cutting F_0 at every joining vertex of F_0 with $N \cup S$. The forest F_1 is the union of trees in $\mathcal{T}$ that have vertices of either N or S (but not both) and $F_2 = F_0 - F_1$.

From forest F_i (for $i = 1, 2$), we will build a forest $\widehat{F_i}$ satisfying the three properties of Theorem 10.7 for $\alpha_i(\epsilon)$ and c_i.

Before specifying $\widehat{F_1}$ and $\widehat{F_2}$, let us see how this implies Theorem 10.7. Define $\widehat{F}$ as the union of $\widehat{F_1}$ and $\widehat{F_2}$.

Suppose two vertices z_0 and z_k of $N \cup S$ are connected in F. By construction, they are connected in F_0. By definition of F_1 and F_2, there are vertices $z_1, \ldots, z_{k-1}$ of $S \cup N$ such that for $i = 1, \ldots k$, vertices z_{i-1} and z_i are connected in either F_1 or F_2. By (F1), z_{i-1} and z_i are connected in either $\widehat{F_1}$ or $\widehat{F_2}$, and so they are connected in $\widehat{F}$. It follows that z_1 and z_k are connected in $\widehat{F}$.

We have that $\widehat{F}$ has at most $\alpha(\epsilon) = \alpha_1(\epsilon) + \alpha_2(\epsilon) = o(\epsilon^{-5.5})$ joining vertices with $N \cup S$.

Moreover,

$$\begin{aligned}\ell(\widehat{F}) &\leq \ell(\widehat{F_1}) + \ell(\widehat{F_2}) \leq (1 + c_1\epsilon)\ell(F_1) + (1 + c_2\epsilon)\ell(F_2) \\ &\leq (1 + c\epsilon)(\ell(F_1) + \ell(F_2)) = (1 + c\epsilon)\ell(F_0) \leq (1 + c\epsilon)\ell(F),\end{aligned}$$

where $c = \max(c_1, c_2)$, and so $\widehat{F}$ satisfies all of the requirements of Theorem 10.7.

We now give the construction of $\widehat{F_1}$. Let T be a connected component of F_1. Without loss of generality, assume that T does not span any vertices of S. Apply Lemma 10.1 to T with ϵ-short path N (using Lemma 6.10) to get tree $\widehat{T}$. Repeating for every connected component of F_1 produces a forest $\widehat{F_1}$ with the desired properties (here $\alpha_1(\epsilon) = 0$ and $c_1 = 1$).

In the rest of the subsection, we give the construction of $\widehat{F_2}$.

Let $s_0, \ldots, s_t$ be the vertices of S guaranteed by Lemma 6.10 (where s_0 is the vertex common to S and W and s_t is the vertex common to S and E). We greedily define a collection of disjoint S-to-N paths $P_0, P_1, \ldots, P_{t'}$ (and associated indices $k_0, k_1, \ldots k_{t'}$) as follows. (See Figure 8.) Let P_0 be the easternmost path in F_2 from S to N that does not go through any vertices of S or N. For each $i \geq 0$, k_i is the integer such that the start vertex of P_i belongs to $S[s_{k_i}, s_{k_i+1})$. For $i \geq 1$, let P_i be the easternmost path in F_2 from $S[s_0, s_{k_i-1})$ to N that does not go through any vertex of S or N or P_{i-1}. Let $t' \leq t$ be the last index for which $P_{t'}$ is defined.

For $i \geq 0$, let x_i and y_i be the start vertex and end vertex, respectively, of P_i.

Now we define a partition (H_i) of F_2. For $i < t'$, let H_i be the subgraph of F_2 consisting of edges enclosed by the simple cycle in $P_i \cup N \cup P_{i+1} \cup S$, including edges of P_i but not edges of P_{i+1}. For $i = t'$, let $H_{t'}$ be the subgraph of F_2 enclosed by $P_{t'}$, N, W, and S. For $i \geq 1$, if there is a vertex in P_i that belongs to H_{i-1}, define q_i to be such a vertex. For each i, we will define a tree $\widehat{H_i}$ and set $\widehat{F_2}$ to be the union of $\widehat{H_i}$ over all i.

Claim 10.8. *Let Q be a minimal P_{i+1}-to-$P_i \cup S[s_{k_i}, x_i]$ path in H_i (if there is no path, then Q is empty). Then $H_i \cup S[s_{k_i}, x_i] - Q$ is connected.*

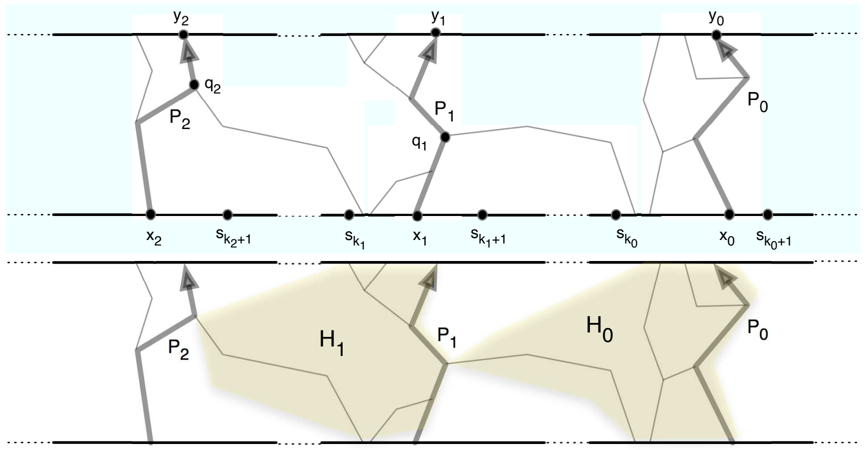

Fig. 8. The north and south boundaries are indicated by horizontal lines. The paths P_0, P_1, and P_2 are indicated by thick gray lines. In the lower figures, the subgraphs H_0 and H_1 are indicated.

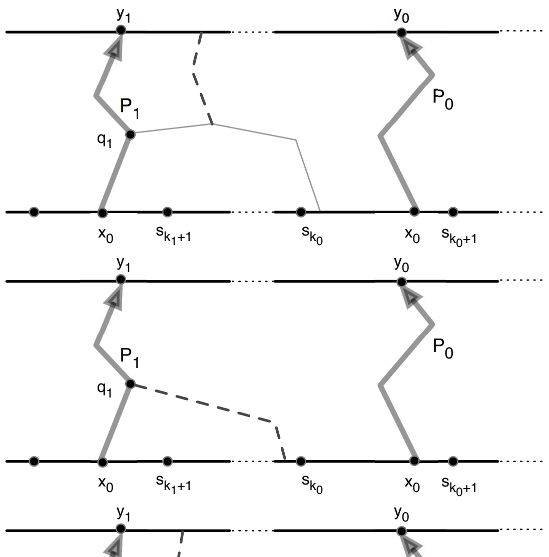

Fig. 9. Illustration for the proof of Claim 10.8. The dashed lines are forbidden positions for P'.

PROOF. Let e be any edge of H_i that is not in $P_i \cup S[s_{k_i}, x_i] \cup Q$. We claim that e belongs to the same connected component of $H_i \cup S[s_{k_i}, x_i] - Q$ as $S[s_{k_i}, x_i] \cup P_i$.

Because e is in F_2, F_2 contains some S-to-N path P through e. Let P' be the maximal subpath of P that contains e and such that no internal vertex of P' is belongs to $P_i \cup P_{i+1} \cup N \cup S \cup Q$. Then all edges of P' belong to H_i, and the endpoints of P' lie on $P_i \cup P_{i+1} \cup N \cup S \cup Q$. Assume for a contradiction that neither endpoint belongs to $S[s_{k_i}, x_i] \cup P_i$. Since F_2 has no cycles, at most one endpoint of P' belongs to $P_{i+1} \cup Q$. Since P' is a subpath of a minimal S-to-N path, at most one endpoint belongs to S and at most one endpoint belongs to N. Then P' either connects $S[x_{i+1}, s_{k_i})$ to $P_{i+1} \cup Q \cup N$ or connects $P_{i+1} \cup Q$ to N. Each possibility contradicts the choice of P_{i+1} as the easternmost path in H_2 from $S[s_0, s_{k_i})$ to N. $\square$

Suppose that q_{i+1} is not defined, i.e. no vertex of P_{i+1} is in H_i. By Lemma 10.8, $H_i \cup S[s_{k_i}, x_i]$ is connected. Let T_i be a tree of edges of $H_i \cup S[s_{k_i-1}, x_i]$ that contains all of the edges of $P_i \cup S[s_{k_i-1}, x_i]$ and spans all vertices of $H_i \cap N$ and of $H_i \cap S$. Applying Lemma 10.4 to T_i with $r_i = s_{k_i}$ and ϵ-short path S defines a tree $\widehat{H}_i$.

Now suppose q_{i+1} is defined to be some vertex of P_{i+1} that is in H_i. Let Q_i be a minimal path from q_{i+1} to $S[s_{k_i}, x_i] \cup P_i$ and let r_i be the vertex common to Q_i and $S[s_{k_i}, x_i] \cup P_i$. By Lemma 10.8, $H_i \cup S[s_{k_i}, x_i] - Q_i$ is connected. By choice of P_{i+1}, Q_i contains no vertex of N. Let T_i be a tree of edges of $H_i \cup S[s_{k_i}, x_i] - Q_i$ that contains all the edges of $S[s_{k_i}, x_i] \cup P_i$ and spans all vertices of $H_i \cap N$, $H_i \cap S$.

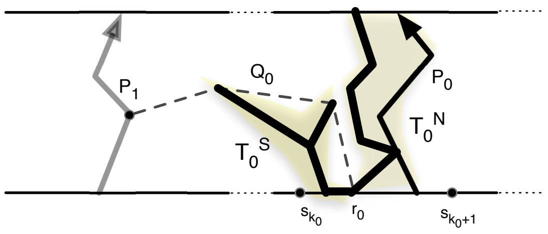

Fig. 10. The path Q_0 is indicated by the dashed line. The trees T_0^N and T_0^S are indicated in solid black and highlighted. They meet at r_0.

As illustrated in Figure 10, let T_i^S be the subgraph of T_i that is enclosed by the cycle formed by Q_i, S, P_i and P_{i+1}. Since Q_i contains no vertex of N, neither does T_i^S. Let $T_i^N = T_i - T_i^S$. Since T_i does not use any edges of Q_i, both T_i^N and T_i^S are trees. The leaves of T_i^N (except perhaps r_i) are all on N. Without loss of generality, assume that T_i^S does not contain q_i.

Let $\widehat{T}_i^S$ be the tree guaranteed by Lemma 10.4 as applied to tree T_i^S, vertex r_i, and ϵ-short path S: $\widehat{T}_i^S$ spans $V(T_i^S \cap S) \cup \{r_i\}$. Let $\widehat{T}_i^N$ be the tree guaranteed by Lemma 10.6 as applied to tree T_i^N, vertices r_i and q_i (if q_i is defined, else use Lemma 10.4), and ϵ-short path N. Let $\widehat{T}_i = \widehat{T}_i^S \cup \widehat{T}_i^N \cup S[s_{k_i}, x_i]$. If two vertices of $N \cup S \cup \{q_i, r_i\}$ are connected in T_i, then they are connected in $\widehat{T}_i$. Let $\widehat{H}_i$ be a spanning tree of $\widehat{T}_i \cup Q_i$ that contains $\widehat{T}_i$. If two vertices of $N \cup S \cup \{q_{i-1}, q_i, r_i\}$ are connected in H_i, then they are connected in $\widehat{H}_i$.

Finally, we define $\widehat{F}_2$ to be the union of $\widehat{H}_i$ over all i.

It remains to show that $\widehat{F}_2$ satisfies properties (F1), (F2), and (F3).

(F1) Suppose vertices z and z' are vertices of $N \cup S$ that are connected in F_2. Let R be the path from z to z' in F_2. Let $H_a, H_{a+1}, \ldots, H_b$ be the subgraphs of F_2 that contain edges of R. Since R is a path in F_2, H_i is connected to H_{i+1} for $a \le i < b$, and so q_i and q_{i+1} are connected in H_i. Likewise since z is a vertex of $N \cup S$, z is connected to q_j for some $a \le j \le b$ (and likewise for z'). It follows that z and z' are connected in $\widehat{F}_2$.

(F2) The number of joining vertices of $\widehat{H}_i$ with $N \cup S$ is at most $c_2\epsilon^{-1.45} + c_3\epsilon^{-2.5}$ where the first term comes from $\widehat{T}_i^S$ and the second term comes from $\widehat{T}_i^N$ using Lemmas 10.4 and 10.6. Since $i \le t \le 4\epsilon^{-2}(1+\epsilon^{-1})$ by Lemma 6.10, the total number of joining vertices of $\widehat{F}_2$ is at most $c_4\epsilon^{-5.5}$.

(F3) To bound the length of $\widehat{F}_2$, we bound the length of $\widehat{H}_i$ in terms of the length of H_i. Using the fact that $\ell(S[s_{k_i}, x_i]) < \epsilon\ell(P_i)$ (Lemma 6.10), we have that

$$\begin{aligned}
\ell(\widehat{H}_i) &= \ell(\widehat{T}_i^S) + \ell(\widehat{T}_i^N) + \ell(Q_i) + \ell(S[s_{k_i}, x_i]) \\
&\le (1+c\epsilon)\ell(T_i^S) + (1+c'\epsilon)\ell(T_i^N) + \ell(Q_i) + \ell(S[s_{k_i}, x_i]) \\
&\qquad \text{for constants } c, c', \text{ by Lemmas 10.4, 10.6 and 6.10} \\
&\le (1+c'\epsilon)\ell(T_i \cup Q_i) + \ell(S[s_{k_i}, x_i]) \\
&\le (1+c'\epsilon)(\ell(H_i) + \ell(S[s_{k_i}, x_i]) + \ell(S[s_{k_i}, x_i]) \\
&\le (1+c'\epsilon)(\ell(H_i) + \epsilon\ell(P_i)) + \epsilon\ell(P_i) \text{ by Lemma 6.10} \\
&\le (1+(c'+2)\epsilon)\ell(H_i).
\end{aligned}$$

This completes the proof of Theorem 10.7.

10.3 Completion of the proof of Theorem 3.2

The Structure Theorem (Theorem 3.2) states that

$$\mathrm{OPT}(\mathcal{B}^+(MG), Q) \le (1+\epsilon)\mathrm{OPT}(G_{in}, Q).$$

We now give the proof using the Lemmas of Sections 6 and 7 as well as Theorem 10.7.

Proof. We start from an optimal solution T to the Steiner tree problem in G_{in} and gradually transform it into a solution $\widehat{T}$ to the Steiner tree problem in $\mathcal{B}^+(MG)$, while almost preserving its length: $\ell(\widehat{T}) < (1+\epsilon)\ell(T)$.

First, we add the east and west boundaries of each brick. Let T_1 be the union of T with the east and west boundaries (E_B and W_B) for every brick B in G. Using Lemma 6.6, we have

$$\ell(T_1) \le \mathrm{OPT} + \epsilon\mathrm{OPT}(G_{in}, Q). \tag{6}$$

We next reduce the number of joining vertices on the north and south boundaries of each brick. Let T_1' be the subgraph of T_1 that is strictly embedded in a brick of G_{in}. Replace T_1' with the forest T_2' that is guaranteed by Theorem 10.7: from the third part of the Theorem, we deduce that $\ell(T_2') \le (1+\epsilon)\ell(T_1')$. Repeating this process for every brick of G_{in} produces the subgraph T_2'. Since the bricks are disjoint, we have

$$\ell(T_2) \le (1+\epsilon)\ell(T_1). \tag{7}$$

Then, we map the edges to a subgraph of $\mathcal{B}^+(MG)$. Every edge of G_{in} has at least one corresponding edge in $\mathcal{B}^+(MG)$. For every edge e of T_2, we select one corresponding edge in $\mathcal{B}^+(MG)$ as follows: if e is an edge of MG select the corresponding mortar edge of $\mathcal{B}^+(MG)$, otherwise select the unique edge corresponding to e in $\mathcal{B}^+(MG)$. This process constructs a subgraph T_3 of $\mathcal{B}^+(MG)$ such that

$$\ell\,(T_3) = \ell\,(T_2). \tag{8}$$

Since T_3 is not connected, we connect it via portal and mortar edges. Let V_B be the set of joining vertices of T_3 with $N_B \cup S_B$ for a brick B of $\mathcal{B}^+(MG)$. For any vertex v on the interior boundary ∂B of a brick, let p_v be the portal on ∂B that is closest to v, let P_v be the shortest v-to-p_v path along ∂B and let P'_v be the corresponding path of mortar edges. Let e be the portal edge corresponding to p_v. Add P_v, P'_v, and e to T_3. Repeat this process for every $v \in V_B$ and for every brick B, building a graph $\widehat{T}$. This completes the definition of $\widehat{T}$.

We now need to analyze the length of $\widehat{T}$:

$$\ell\,(\widehat{T}) \le \ell\,(T_3) + \sum_{B\in\mathcal{B}} \sum_{v\in V_B} (\ell\,(P_v) + \ell\,(e) + \ell\,(P'_v)), \tag{9}$$

and we have:

$$\begin{aligned}
\sum_{B\in\mathcal{B}} \sum_{v\in V_B} \ell\,(P_v) + \ell\,(e) + \ell\,(P'_v) \;&=\; 2\sum_{B\in\mathcal{B}} \sum_{v\in V_B} \ell\,(P_v), \text{ since } \ell(\text{portal edges}) = 0\\
&\le\; 2\sum_{B\in\mathcal{B}} \sum_{v\in V_B} \ell\,(\partial B)/\theta(\epsilon), \text{ by Lemma 7.1}\\
&\le\; 2\sum_{B\in\mathcal{B}} \frac{\alpha(\epsilon)}{\theta(\epsilon)}\ell\,(\partial B), \text{ by Theorem 10.7, Part 2}\\
&\le\; 2\frac{\alpha(\epsilon)}{\theta(\epsilon)}\nu\epsilon^{-1}\mathrm{OPT}(G_{in}, Q), \text{ using Lemma 6.9}\\
&\le\; \epsilon\,\mathrm{OPT}(G_{in}, Q), \text{ using Equation (5).}
\end{aligned}$$

Combining this with inequalities (9), (8), (7) and (6), we obtain

$$\ell\,(\widehat{T}) \le \epsilon\mathrm{OPT}(G_{in}, Q) + (1+\epsilon)^2\mathrm{OPT}(G_{in}, Q) < (1+4\epsilon)\mathrm{OPT}(G_{in}, Q)$$

for the fixed constant c given by Theorem 10.7. The construction can be modified to obtain $\ell\,(\widehat{T}) < (1+\epsilon)\mathrm{OPT}(G_{in}, Q)$ by inputting $\epsilon' = \epsilon/4$ to the algorithm.

It remains to show that $\widehat{T}$ is a solution to the Steiner tree problem in $\mathcal{B}^+(MG)$. First we show that T_2 is a solution to the Steiner tree problem in G_{in}.

Clearly, since T is a subgraph of T_1 and T is a solution in G_{in}, T_1 is a solution in G_{in}. Now we argue that T_2 is a solution to the Steiner tree problem in G_{in}. Let P be a path of T that connects two terminals s and t. We partition P into a sequence of subpaths as follows: P_i is a subpath of the partition if it is a maximal subpath strictly enclosed by a brick, or if it is a maximal subpath on the boundary of a single brick. Each subpath P_i is an x_i-to-y_i path. For a vertex x in MG, if x is a vertex internal to an east boundary E_B of a brick B, then define $\widehat{x}$ to be the vertex common to E_B and N_B (likewise for a vertex internal to a west boundary). If x is a vertex of a north or south boundary, then define $\widehat{x} = x$. Note that for each path P_i, the first step of the construction guarantees that there is a corresponding $\widehat{x}_i$-to-$\widehat{y}_i$ path in T_1. Since $\widehat{x}_i$ and $\widehat{y}_i$ are vertices on $S_B \cup N_B$ for a brick B, Theorem 10.7 guarantees that there is a $\widehat{x}_i$-to-$\widehat{y}_i$ path in T_2. It follows that there is a $\widehat{s}$-to-$\widehat{t}$ path in T_2. By Lemma 6.10 since s and t are terminals of Q, they are on north or south brick boundaries, and so $\widehat{s} = s$ and $\widehat{t} = t$: there is an s-to-t path $\widehat{P}$ in T_2.

The definition of T_3 breaks $\widehat{P}$ into disjoint paths. Consider one such path, $\widehat{P}_i$, that is not a subpath of MG. By construction, the endpoints of $\widehat{P}_i$ are joining vertices. To go from T_3 to $\widehat{T}$, these endpoints are connected to the corresponding vertices on MG via portal edges. It follows that there is an s-to-t path $\widehat{P}$ in $\widehat{T}$. □

11. CLOSING REMARKS

The method used here to obtain a PTAS for the Steiner tree problem has been extended to higher edge-connectivity problems in planar graphs. In particular, we can give an $O(n \log n)$-time approximation scheme for the following problem: given integer requirements $r_v \in \{0, 1, 2\}$ for each vertex v, find a subgraph H that contains at least $\min\{r_u, r_v\}$ edge-disjoint u-to-v paths for every pair u, v of vertices, where H is allowed to use multiple copies of edges [Borradaile and Klein 2008]. This includes the well studied 2-edge connectivity problem. In forthcoming work, we will show that we can satisfy node requirements $r_v \in \{0, 1, \ldots, k\}$ for fixed k.

The algorithm presented here has been implemented in [Tazari and Müller-Hannemann 2008] with good results: by using small constants in the implementation (ie. number of portals per brick), good approximations are found.

REFERENCES

Arora, S. 1998. Polynomial-time approximation schemes for Euclidean TSP and other geometric problems. *Journal of the ACM 45,* 5, 753–782.

Arora, S., Grigni, M., Karger, D., Klein, P., and Woloszyn, A. 1998. A polynomial-time approximation scheme for weighted planar graph TSP. In *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*. 33–41.

Baker, B. 1994. Approximation algorithms for NP-complete problems on planar graphs. *Journal of the ACM 41,* 1, 153–180.

Berman, P. and Ramaiyer, V. 1994. Improved approximations for the Steiner tree problem. *Journal of Algorithms 17*, 381–408.

Bern, M. 1990. Faster exact algorithms for Steiner trees in planar networks. *Networks 20*, 109–120.

Bern, M. and Bienstock, D. 1991. Polynomially solvable special cases of the Steiner problem in planar networks. *Mathematics of Operations Research 33*, 405–418.

Bern, M. and Plassmann, P. 1989. The Steiner problem with edge lengths 1 and 2. *Information Processing Letters 32*, 171–176.

Borradaile, G., Kenyon-Mathieu, C., and Klein, P. 2007. A polynomial-time approximation scheme for Steiner tree in planar graphs. In *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms*. 1285–1294.

Borradaile, G. and Klein, P. 2008. The two-edge connectivity survivable network problem in planar graphs. In *Proceedings of the 35th International Colloquium on Automata, Languages and Programming*. To appear.

Borradaile, G., Klein, P., and Mathieu, C. 2007. Steiner tree in planar graphs: An $O(n \log n)$ approximation scheme with singly exponential dependence on epsilon. In *Proceedings of the 10th International Workshop on Algorithms and Data Structures*. Lecture Notes in Computer Science, vol. 4619. 275–286.

Catalan, E. 1838. Note sur un problème de combinaisons. *Journal de Mathématiques Pures et Appliquées 3*, 111 – 112.

Demaine, E. D. and Hajiaghayi, M. 2005. Bidimensionality: New connections between FPT algorithms and PTASs. In *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms*. 367–376.

Erickson, R., Monma, C., and Veinott, A. 1987. Send-and-split method for minimum-concave-cost network flows. *Mathematics of Operations Research 12*, 634–664.

Garey, M. and Johnson, D. S. 1977. The rectilinear Steiner tree problem is NP-complete. *SIAM Journal on Applied Mathematics 32,* 4, 826–834.

Henzinger, M. R., Klein, P. N., Rao, S., and Subramanian, S. 1997. Faster shortest-path algorithms for planar graphs. *Journal of Computer and System Sciences 55,* 1, 3–23.

Hopcroft, J. and Tarjan, R. 1974. Efficient planarity testing. *Journal of the ACM 21,* 4, 549–568.

Hougardy, S. and Prömel, H. J. 1999. A 1.598 approximation algorithm for the Steiner problem in graphs. In *Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms*. 448–453.

Karp, R. 1975. On the computational complexity of combinatorial problems. *Networks 5*, 45–68.

Karpinski, M. and Zelikovsky, A. 1997. New approximation algorithms for the Steiner tree problem. *Journal of Combinatorial Optimization 1*, 47–65.

Klein, P. A linear-time approximation scheme for planar TSP. *SIAM Journal on Computing*. To appear.

Klein, P. 2006. A subset spanner for planar graphs, with application to subset TSP. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*. 749–756.

Klein, P. N. 2005a. A linear-time approximation scheme for planar weighted TSP. In *Proceedings of the 46th Annual Symposium on Foundations of Computer Science*. 647–656.

Klein, P. N. 2005b. Multiple-source shortest paths in planar graphs. In *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms*. 146–155.

Kou, L., Markowsky, G., and Berman, L. 1981. A fast algorithm for Steiner trees. *Acta Informatica 15*, 141–145.

Mehlhorn, K. 1988. A faster approximation algorithm for the Steiner problem in graphs. *Information Processing Letters 27,* 3, 125–128.

Mitchell, J. 1999. Guillotine subdivisions approximate polygonal subdivisions: A simple polynomial-time approximation scheme for geometric TSP, k-MST, and related problems. *SIAM Journal on Computing 28,* 4, 1298–1309.

Prömel, H. and Steger, A. 1997. RNC approximation algorithms for the Steiner problem. In *Proceedings of the 14th International Symposium on Theoretical Aspects of Computer Science*. 559–570.

Provan, J. 1988a. An approximation scheme for finding Steiner trees with obstacles. *SIAM Journal on Computing 17,* 920-934, 920–934.

Provan, J. 1988b. Convexity and the Steiner tree problem. *Networks 18*, 55–72.

Rao, S. and Smith, W. 1998. Approximating geometrical graphs via "spanners" and "banyans". In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*. 540–550.

Robins, G. and Zelikovsky, A. 2005. Tighter bounds for graph Steiner tree approximation. *SIAM Journal on Discrete Mathematics 19,* 1, 122–134.

Sommerville, D. 1929. *An introduction to the geometry of n dimensions*. London.

Takahashi, H. and Matsuyama, A. 1980. An approximate solution for the Steiner problem in graphs. *Mathematica Japonicae 24*, 571–577.

Tazari, S. and Müller-Hannemann, M. 2008. To fear or not to fear large hidden constants: Implementing a planar Steiner tree ptas. Tech. Rep. TUD-CD-2008-2, Technische Universität Darmstadt, Department of Computer Science, Darmstadt, Germany.

Whitney, H. 1933. Planar graphs. *Fundamenta mathematicae 21*, 73–84.

WIDMAYER, P. 1986. A fast approximation algorithm for Steiner's problem in graphs. In *Graph-Theoretic Concepts in Computer Science*. Lecture Notes in Computer Science, vol. 246. Springer Verlag, 17–28.

WU, Y., WIDMAYER, P., AND WONG, C. 1986. A faster approximation algorithm for the Steiner problem in graphs. *Acta informatica 23,* 2, 223–229.

ZELIKOVSKY, A. 1993. An 11/6-approximation algorithm for the network Steiner problem. *Algorithmica 9*, 463–470.

ZELIKOVSKY, A. 1994. Better approximation bounds for the network and Euclidean Steiner tree problems. Tech. Rep. CS-96-06, University of Virginia.