A PTAS for Three-Edge-Connected Survivable Network Design in Planar Graphs*†

Glencora Borradaile[1] and Baigong Zheng[2]

1 **Oregon State University, Corvallis, OR, USA**
`glencora@eecs.oregonstate.edu`
2 **Oregon State University, Corvallis, OR, USA**
`zhengb@oregonstate.edu`

Abstract

We consider the problem of finding the minimum-weight subgraph that satisfies given connectivity requirements. Specifically, given a requirement $r \in \{0, 1, 2, 3\}$ for every vertex, we seek the minimum-weight subgraph that contains, for every pair of vertices u and v, at least $\min\{r(v), r(u)\}$ edge-disjoint u-to-v paths. We give a polynomial-time approximation scheme (PTAS) for this problem when the input graph is planar and the subgraph may use multiple copies of any given edge (paying for each edge separately). This generalizes an earlier result for $r \in \{0, 1, 2\}$. In order to achieve this PTAS, we prove some properties of triconnected planar graphs that may be of independent interest.

1998 ACM Subject Classification G.2.2 [Graph Theory] Graph Algorithms

Keywords and phrases Three-Edge Connectivity, Polynomial-Time Approximation Schemes, Planar Graphs

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.3

1 Introduction

The survivable network design problem aims to find a low-weight subgraph that connects a subset of vertices and will remain connected despite edge failures, an important requirement in the field of telecommunications network design. This problem can be formalized as the I-edge connectivity problem for an integer set I as follows: for an edge-weighted graph G with a requirement function on its vertices $r : V(G) \to I$, we say a subgraph H is a feasible solution if for any pair of vertices $u, v \in V(G)$, H contains $\min\{r(u), r(v)\}$ edge-disjoint u-to-v paths; the goal is to find the cheapest such subgraph. In the *relaxed* version of the problem, H may contain multiple (up to $\max I$) copies of G's edges (H is a *multi*-subgraph) in order to achieve the desired connectivity, paying for the copies according to their multiplicity; otherwise we refer to the problem as the *strict* version. Thus $I = \{1\}$ corresponds to the minimum spanning tree problem and $I = \{0, 1\}$ corresponds to the minimum Steiner tree problem. Here our focus is when $\max I \geq 2$.

This problem and variants have a long history. The I-edge connectivity problem, except when $I = \{1\}$ and $I = \{0\}$, is MAX-SNP-hard [13]. There are constant-factor approximation algorithms for the strict $\{k\}$-edge-connectivity problem: for $k = 2$, Frederickson and Jájá [16] gave a 3-approximation for this problem, and Sebő and Vygen [24] gave a

* The full version of this work can be found in [11], `http://arxiv.org/abs/1611.03889`.
† This material is based upon work supported by the National Science Foundation under Grant No. CCF-1252833.

4/3-approximation for this problem in unweighted graphs; for any k, Khuller and Vishkin [19] gave a 2-approximation for this problem. Klein and Ravi [23] gave a 2-approximation for the strict $\{0, 1, 2\}$-edge-connectivity problem. For general requirements, Jain [18] gave a 2-approximation for both the strict and relaxed versions of the problem.

We study this problem in planar graphs. In planar graphs, the I-edge connectivity problem, except when $I = \{1\}$ and $I = \{0\}$, is NP-hard (by reduction from Hamiltonian cycle). Berger, Czumaj, Grigni, and Zhao [4] gave a polynomial-time approximation scheme[1] (PTAS) for the relaxed $\{1, 2\}$-edge-connectivity problem, and Berger and Grigni [5] gave a PTAS for the strict $\{2\}$-edge-connectivity problem. Zheng [26] gave a linear PTAS for the strict $\{3\}$-edge-connectivity problem in unweighted planar graphs. Borradaile and Klein [8] gave an *efficient*[2] PTAS (EPTAS) for the relaxed $\{0, 1, 2\}$-edge-connectivity problem[3]. The only planar-specific algorithm for non-spanning, *strict* edge-connectivity is a PTAS for the following problem: given a subset R of edges, find a minimum weight subset S of edges, such that for every edge in R, its endpoints are two-edge-connected in $R \cup S$ [22]; otherwise, the best known results for the strict versions of the edge-connectivity problem when I contains 0 and 2 are the constant-factor approximations known for general graphs.

In this paper, we give an EPTAS for the relaxed $\{0, 1, 2, 3\}$-edge-connectivity problem in planar graphs. This is the first PTAS for connectivity beyond 2-connectivity in planar graphs:

▶ **Theorem 1.** *For any $\epsilon > 0$ and any planar graph instance of the relaxed $\{0, 1, 2, 3\}$-edge connectivity problem, there is an $O(n \log n)$-time algorithm that finds a solution whose weight is at most $1 + \epsilon$ times the weight of an optimal solution.*

In order to give this EPTAS, we must prove some properties of triconnected (three-vertex connected) planar graphs that may be of independent interest. One simple-to-state corollary of the sequel is:

▶ **Theorem 2.** *In a planar graph that minimally pairwise triconnects a set of terminal vertices, every cycle contains at least two terminals.*

In the remainder of this introduction we overview the PTAS framework for network design problems in planar graphs [9] that we use for the relaxed $\{0, 1, 2, 3\}$-edge connectivity problem. In this overview we highlight the technical challenges that arise from handling 3-edge connectivity. We then overview why we use properties of vertex connectivity to address an edge connectivity problem and state our specific observations about triconnected planar graphs that we require for the PTAS framework to apply. In the remainder, 2-EC refers to "relaxed $\{0, 1, 2\}$-edge-connectivity" and 3-EC refers to "relaxed $\{0, 1, 2, 3\}$-edge-connectivity".

1.1 Overview of the planar PTAS framework

The planar PTAS framework grew out of a PTAS for travelling salesperson problem [21] and has been used to give PTASes for Steiner tree [7, 10], Steiner forest [3] and 2-EC [9] problems. For simplicity of presentation, we follow the PTAS whose running time is doubly

[1] A polynomial-time approximation scheme for an minimization problem is an algorithm that, given a fixed constant $\epsilon > 0$, runs in polynomial time and returns a solution within $1 + \epsilon$ of optimal. The algorithm's running time need not be polynomial in ϵ.

[2] A PTAS is efficient if the running time is bounded by a polynomial whose degree is independent of ϵ.

[3] Note that Borradaile and Klein [8] claimed their PTAS would generalize to relaxed $\{0, 1, \ldots, k\}$-edge-connectivity, but this did not come to fruition.

exponential in $1/\epsilon$ [7]; this can be improved to singly exponential as for Steiner tree [10]. Note that for all these problems (except Steiner forest, which requires a preprocessing step to the framework), the optimal value OPT of the solution is within a constant factor of the optimal value of a Steiner tree on the same terminal set where we refer to vertices with non-zero requirement as *terminals*. In the following, O_ϵ-notation hides factors depending on ϵ.

The PTAS framework

The PTAS framework for a planar connectivity problem in graph G consists of the following steps. We describe the steps in terms of the relaxed I-edge connectivity problem, which, at this high level, are easy to generalize from the application of this framework to Steiner tree [7] and 2-EC [9]:

Step 1: Find the *spanner* subgraph H (described below) having the properties:
- **(S1)** $w(H) = O_\epsilon(\mathrm{OPT})$, and
- **(S2)** H contains a feasible solution to the connectivity problem of value at most $(1+\epsilon)\mathrm{OPT}$.

To find a $(1+O(\epsilon))$-approximate solution in G, it is sufficient to find a $(1+\epsilon)$-approximate nearly-optimal solution in H by (S2).

Step 2: Decompose the spanner into a set of subgraphs, called *slices*, such that:
- **(A1)** each slice has *branchwidth* $O_\epsilon(1)$,
- **(A2)** the boundary of a slice is a set of cycles and every cycle bounds exactly two slices,
- **(A3)** the weight of all boundary edges is at most $\epsilon\mathrm{OPT}$.

The slice boundaries correspond to every k^{th} breadth-first level in the dual graph; this gives property (A2). By choosing $k = O_\epsilon(1)$, we get property (A1). Property (A3) follows from (S1) for k sufficiently large.

Step 3: Add artificial terminals to slice boundaries and assign connectivity requirements so that:
- **(B1)** for each slice, there is a feasible solution over the original and artificial terminals whose weight is bounded by the weight of the slice boundary plus the weight of the optimal solution in the slice.
- **(B2)** the union of these slice solutions is a feasible solution for the original original.

This can be done by adding a terminal to a boundary cycle if the cycle separates any two original terminals and assigning this terminal a connectivity requirement equal to the maximum connectivity requirement the cycle separates (e.g. 2 if the cycle separates two terminals each having a connectivity requirement of 2); this process and the fact that edge connectivity is transitive guarantees property (B2). Property (B1) is guaranteed by property (A3) as seen by adding $2\max I$ copies of the slices to a solution in H.

Step 4: Solve the problem with respect to original and artificial terminals in each slice.

By property (A1), we can do this by dynamic programming over the branch decomposition.

Step 5: Return the union of the slice solutions.

We apply this PTAS framework to the 3-EC problem. Algorithmically, the modifications needed for 3-EC (as compared to 2-EC or Steiner tree) are limited to Step 4; we can obtain an $O_\epsilon(n)$-time dynamic program for the I-edge connectivity problem on graphs with branchwidth $O_\epsilon(1)$, which is similar to that for the k-vertex-connectivity spanning subgraph problem in Euclidean space given by Czumaj and Lingas in [12, 13]. We will argue that the spanner construction (with larger constants) is the same as used for Steiner tree and 2-EC; this

argument is the bulk of the technical challenge of this work. Borradaile, Klein and Mathieu show that Step 1 can be done in $O_\epsilon(n \log n)$ time [10, 9] and Steps 2 and 3 can be done in $O(n)$ time. Therefore, we will achieve an $O_\epsilon(n \log n)$ running time for 3-EC.

Spanners for connectivity problems

The spanner construction for Steiner tree and 2-EC [10] (and, as we will argue, for 3-EC) starts with finding the *mortar graph* MG of the input graph G. The mortar graph is a grid-like subgraph of G that spans all the terminals and has total weight bounded by $O_\epsilon(1)$ times the minimum weight of a Steiner tree spanning all the terminals (i.e. weight $O_\epsilon(\mathrm{OPT})$). To construct the mortar graph, we first find an approximate Steiner tree connnecting all terminals and recursively add some short paths. Each face of MG is bounded by four $(1+\epsilon)$ approximations to short paths; the subgraph of G that is enclosed by a face of MG is called a *brick*.

A *structure theorem* shows that there is a nearly optimal solution for Steiner tree and 2-EC whose intersection with each brick is a set of non-crossing trees with $O_\epsilon(1)$ leaves that are *portals* (a subset of $O_\epsilon(1)$ designated vertices of the boundary of the brick) [9]. Each such tree can be computed efficiently since each is a Steiner tree with vertices on the boundary of a planar graph (a brick) [14].

We compute the *spanner* subgraph H by starting with the mortar graph, assigning $O_\epsilon(1)$ vertices of each brick boundary to be portals and adding to the spanner all Steiner trees for each subset of portals in each brick. Since there are $O_\epsilon(1)$ Steiner trees per brick and each has weight at most the boundary of the brick, the spanner has weight $O_\epsilon(\mathrm{OPT})$. By the structure theorem, it is sufficient to solve the given problem in the spanner.

Extension to the 3-EC problem

To prove that the PTAS framework extends to 3-edge connectivity, we need to show this construction results in a spanner for 3-EC, that is, that H contains a $(1+\epsilon)$-approximate solution to 3-EC. This is the main technical challenge of this work. We will prove:

▶ **Theorem 3** (Structure Theorem for 3-EC). *For any $\epsilon > 0$ and any planar graph instance (G, w, r) of the 3-EC problem, there exists a feasible solution S in the spanner H such that*

- *the weight of S is at most $(1 + c\epsilon)\mathrm{OPT}$ where c is an absolute constant, and*
- *the intersection of S with the interior of any brick is a set of $O_\epsilon(1)$ trees whose leaves are on the boundary of the brick and each tree has $O_\epsilon(1)$ leaves.*

The *interior* of a brick is the set of brick edges that are not on the boundary of the brick (that is, not in MG). We denote the interior of a brick B by $\mathrm{int}(B)$. Consider a brick B of G whose boundary is a face of MG and consider the intersection of OPT with the interior of this brick, $\mathrm{OPT} \cap \mathrm{int}(B)$. To prove the Structure Theorem, we will show that:

(P1) $\mathrm{OPT} \cap \mathrm{int}(B)$ can be partitioned into a set of trees $\mathcal{T}$ whose leaves are on the boundary of B.

(P2) If we replace any tree in $\mathcal{T}$ with another tree spanning the same leaves, the result is a feasible solution.

(P3) There is another set of $O(1)$ trees $\mathcal{T}'$ and a set of brick boundary edges B' that costs at most a $1+\epsilon$ factor more than $\mathcal{T}$, such that each tree of $\mathcal{T}'$ has $O(1)$ leaves and $(\mathrm{OPT} \setminus \mathcal{T}) \cup \mathcal{T}' \cup B'$ is a feasible solution.

Property P1 implies that we can decompose an optimal solution into a set of trees inside of bricks plus some edges of MG. Property P2 shows that we can treat those trees independently

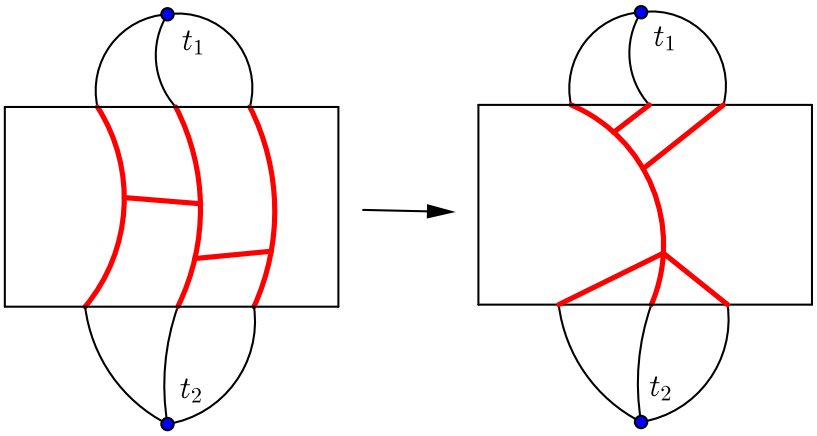

Figure 1 If the bold red tree (left) is $\mathrm{OPT} \cap \mathrm{int}(B)$ (where B is denoted by the rectangle), replacing the tree with another tree spanning the same leaves (right) could destroy 3-connectivity between t_1 and t_2. We will show that such a tree cannot exist in a minimally connected graph.

with regard to connectivity, and this gives us hope that we can replace $\mathrm{OPT} \cap \mathrm{int}(B)$ with some Steiner trees with terminals on the boundary which we can efficiently compute in planar graphs [14]. Property P3 shows that we can compute an approximation to $\mathrm{OPT} \cap \mathrm{int}(B)$ by guessing $O(1)$ leaves.

For the Steiner tree problem, P1 and P2 are nearly trivial to argue; the bulk of the work is in showing P3 [7].

For the 2-EC problem, P1 depends on first converting G and OPT into G' and OPT' such that OPT' biconnects (two-vertex connects) the terminals requiring two-edge connectivity and using the relatively easy-to-argue fact that every cycle of OPT' contains at least one terminal. By this fact, a cycle in OPT' must contain a vertex of the brick's boundary (since MG spans the terminals), allowing the partition of $\mathrm{OPT}' \cap \mathrm{int}(B)$ into trees. P2 and P3 then require that two-connectivity across the brick is maintained.

For the 3-EC problem, P1 is quite involved to show, but further to that, showing Property P2 is also involved; the issues[4] are illustrated in Figure 1 and are the focus of Sections 2 and 3. As with 2-EC, we convert OPT into a vertex connected graph to simplify the arguments. Given Properties P1 and P2, we illustrate Property P3 by following a similar argument as for 2-EC; since this requires reviewing more details of the PTAS framework, we cover this in Section 4.

Non-planar graphs

We point out that, while previously-studied problems that admit PTASes in planar graphs (e.g. independent set and vertex cover [2], TSP [21, 20, 1], Steiner tree [10] and forest [3], 2-EC [9]) generalize to surfaces of bounded genus [6], the method presented in this paper for 3-EC is hard to be generalized to higher genus surfaces. In the generalization to bounded genus surfaces, the graph is preprocessed (by removing some provably unnecessary edges) so that one can compute a mortar graph whose faces bound disks. This guarantees that even though the input graph is not planar, the bricks are; this is sufficient for proving

[4] The issues also appear in 2-ECP, but we explain why it is easy to handle in 2-ECP in the next subsection.

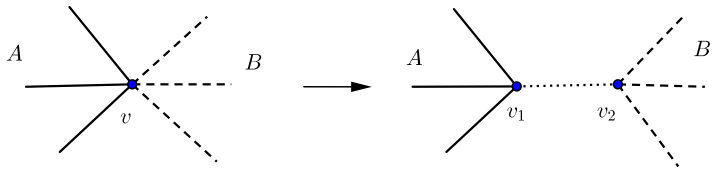

Figure 2 Vertex v is cleaved into vertices v_1 and v_2. The edges incident to v are partitioned into two sets A and B to become incident to distinct copies.

above-numbered properties in the case of TSP, Steiner tree and forest and 2-EC. However, for 3-ECP, in order to prove P2, we require *global* planarity, not just planarity of the brick. To the authors' knowledge, this is the only problem that we know to admit a PTAS in planar graphs that does not naturally generalize to toroidal graphs.

1.2 Reduction to vertex connectivity

Now we overview how we use vertex connectivity to argue about the structural properties of edge-connectivity required for the spanner properties.

We require a few definitions. Vertices x and y are k-vertex-connected in a graph G if G contains k pairwise vertex disjoint x-to-y paths. If $k = 3$ ($k = 2$), then x and y are also called triconnected (biconnected). For a subset Q of vertices in G and a requirement function $r : Q \to \{2, 3\}$, subgraph H is said to be (Q, r)-vertex-connected if every pair of vertices x, y in Q is k-vertex-connected where $k = \min\{r(x), r(y)\}$. We call the vertices of Q *terminals*. If $r(x) = 3$ ($r(x) = 2$) for all $x \in Q$, we say H is Q-triconnected (Q-biconnected). We say a (Q, r)-vertex-connected graph is *minimal*, if no edge or vertex can be deleted without violating the connectivity requirements.

We *cleave* vertices to transform edge-connectivity into vertex-connectivity. Informally, cleaving a vertex is splitting the vertex into two copies and adding a zero-weight edge between the copies; incident edges choose between the copies in a planarity-preserving way (Figure 2). We can cleave the vertices of OPT, creating OPT$'$, so that if two terminals are k-edge-connected in OPT, there are corresponding terminals in OPT$'$ that are k-vertex-connected. We will prove that OPT$'$ satisfies Properties P1 and P2 and since OPT$'$ is obtained from OPT by cleavings, these two properties also hold for OPT.

To prove that OPT$'$ satisfies Property P1, we show that every cycle in OPT$'$ contains at least one terminal (Section 2). To prove that OPT$'$ satisfies Property P2, we define the notion of a *terminal-bounded component*: a connected subgraph is a terminal-bounded component if it is an edge between two terminals or obtained from a maximal terminal-free subgraph S (a subgraph containing no terminals), by adding edges from S to its neighbors (which are all terminals by maximality of S). In Section 3, we show that in a minimal Q-triconnected graph any terminal-bounded component is a tree whose leaves are terminals as well as:

▶ **Theorem 4** (Connectivity Separation Theorem)**.** *Given a minimal (Q, r)-vertex-connected planar graph, for any pair of terminals x and y that require triconnectivity (biconnectivity), there are three (two) vertex disjoint paths from x to y in G such that any two of them do not contain edges of the same terminal-bounded tree.*

▶ **Corollary 5.** *Given a minimal (Q, r)-vertex-connected planar graph, for any pair of terminals x and y that require triconnectivity (biconnectivity), there exist three (two) vertex*

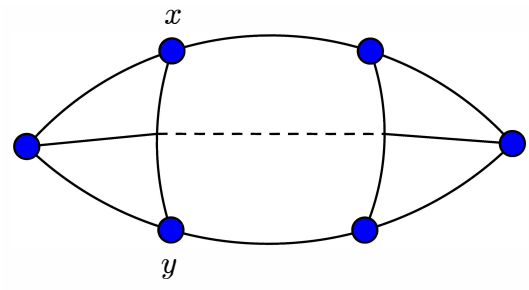

Figure 3 A minimal Q-triconnected graph. The bold vertices are terminals. The dashed path connects two x-to-y paths but it does not contain any terminal.

disjoint x-to-y paths such that any path that connects any two of those x-to-y paths contains a terminal.

This corollary can be viewed as a generalization of the following by Borradaile and Klein for 2-ECP [9]:

▶ **Theorem 6.** (Theorem 2.8 [9]). *Given a graph that minimally biconnects a set of terminals, for any pair of terminals x and y and for any two vertex disjoint x-to-y paths, any path that connects these paths must contain a terminal.*

Note that Theorem 6 holds for general graphs while we only know Corollary 5 to hold for planar graphs, underscoring why our PTAS does not generalize to higher-genus graphs. Further *"for any"* is sufficient for biconnectivity (Theorem 6) whereas *"there exists"* is necessary for triconnectivity (Corollary 5) as illustrated by the example in Figure 3. Higher connectivity comes at a price.

For OPT′, Corollary 5 implies Property P2. Consider the set of disjoint paths guaranteed by Corollary 5. If any tree replacement in a brick merges any two disjoint paths, say P_1 and P_2, in the set (the replacement in Figure 1 merges three paths), then the replaced tree must contain at least one vertex of P_1 and one vertex of P_2. This implies the replaced tree contains a P_1-to-P_2 path P such that each vertex in P has degree at least two in the replaced tree. Further, P contains a terminal by Corollary 5. However, all the terminals are in the mortar graph, which forms the boundaries of the bricks. So P must have a common vertex with the boundary of the brick. By Property P1, the replaced tree, which is in the intersection of OPT′ with the interior of the brick, can only contain leaves on the boundary of the brick. Therefore, the replaced tree can not contain such a P_1-to-P_2 path, otherwise there is a vertex in P that has degree one in the tree.

2 Vertex-connectivity basics

In this section, we consider minimal (Q, r)-vertex-connected graphs for a subset Q of vertices and a requirement function $r : Q \to \{2, 3\}$.

Borradaile and Klein prove that in a minimal Q-biconnected graph, every cycle contains a terminal (Theorem 2.5 [9]). We show a similar property for a minimal (Q, r)-vertex connected graph here. This property implies property P1, that is the intersection of an optimal solution with the interior of any brick can be partitioned into a set of trees whose leaves are on the boundary of the brick. Note that our proof for this property does not depend on planarity.

For a Q-triconnected graph H, we can obtain another graph H' by contracting all the edges incident to the vertices of degree two in H. We say H' is *contracted version of* H and,

alternatively, is *contracted* Q-triconnected. We can prove that H' is triconnected. Further, if H is a minimal Q-triconnected graph, then the contracted version of H is also a minimal Q-triconnected graph. And if $|Q| > 3$, then we can prove H' is simple by the result of Eswaran and Tarjan [15].

Holton, Jackson, Saito and Wormald study the *removability* of edges in triconnected graphs [17]. For an edge $e = uv$ of a simple, triconnected graph G, removing e consists of (i) deleting uv from G, (ii) if u or v now have degree 2, contracting incident edges, and (iii) deleting parallel edges. If the resulting graph after removing e is triconnected, then e is said to be *removable*.

By applying several results of Holton et al. [17] about removable edges, we can prove that every cycle in a minimum contracted Q-triconnected graph contains a terminal. For a graph G that is (Q, r)-vertex connected, let G' be a minimum Q-triconnected graph that is a supergraph of G. Let G'' is the contracted version of G'. Then every cycle in G'' contains a terminal. Since G' is a subdivision of G'', we know every cycle in G' contains a terminal. Since G is a subgraph of G', we have the following theorem.

▶ **Theorem 7.** *For a requirement function $r : Q \to \{2, 3\}$, let G be a minimal (Q, r)-vertex-connected graph. Then every cycle in G contains a vertex of Q.*

3 Connectivity Separation

In this section we continue to focus on vertex connectivity and prove the Connectivity Separation Theorem. The Connectivity Separation Theorem for biconnectivity follows easily from Theorem 6. To see why, consider two paths P_1 and P_2 that witness the biconnectivity of two terminals x and y. For an edge of P_1 to be in the same terminal-bounded component as an edge of P_2, there would need to be a P_1-to-P_2 path that is terminal-free. However, such a path must contain a terminal by Theorem 6. Herein we mainly focus on triconnectivity.

For a requirement function $r : Q \to \{2, 3\}$, let G be a minimal (Q, r)-vertex-connected planar graph. We say a subgraph is *terminal-free* if it is connected and does not contain any terminals. It follows from Theorem 7 that any terminal-free subgraph of G is a tree. We partition the edges of G into *terminal-bounded* components as follows: a terminal-bounded component is either an edge connecting two terminals or is obtained from a *maximal* terminal-free tree T by adding the edges from T to its neighbors, all of which are terminals. Theorem 8 will show that any terminal-bounded subgraph is also a tree.

For a connected subgraph χ of G and an embedding of G with outer face containing no edge of χ, let $C(\chi)$ be the simple cycle that strictly encloses the fewest faces and all edges of χ, if such a cycle exists. (Note that $C(\chi)$ does not exist if there is no aforementioned choice for an outer face.) In order to prove the Connectivity Separation Theorem for bi- and triconnectivity, we start with the following theorem:

▶ **Theorem 8** (Tree Cycle Theorem)**.** *Let T be a terminal-bounded component in a minimal Q-triconnected planar graph H. Then T is a tree and $C(T)$ exists with the following properties*

- **(a)** *The internal vertices of T are strictly inside of $C(T)$.*
- **(b)** *All vertices strictly inside of $C(T)$ are on T.*
- **(c)** *All leaves of T are in $C(T)$.*
- **(d)** *Any pair of distinct maximal terminal-free subpaths of $C(T)$ does not contain vertices of the same terminal-bounded tree.*

Theorem 2 follows from this Tree Cycle Theorem.

Proof of Theorem 2. For a contradiction, assume there is a cycle in H that only containing one terminal, then there is a terminal-bounded component containing that cycle, which can not be a tree, contradicting the Tree Cycle Theorem. ◀

We give an overview of the proof the Tree Cycle Theorem in Subsection 3.2. First, let us see how the Tree Cycle Theorem implies the Connectivity Separation Theorem.

3.1 The Tree Cycle Theorem implies the Connectivity Separation Theorem

For a requirement function $r : Q \to \{2, 3\}$, let G be a minimal (Q, r)-vertex-connected planar graph. Let Q_3 be the set of terminals requiring triconnectivity, and let H be a minimal Q_3-triconnected subgraph of G. Let $Q_2 = Q \setminus Q_3$. Consider two terminals x and y. We sketch the proof here.

Suppose x and y only require biconnectivity. For this case, we know the graph is biconnected and by applying a result of Whitney [25] for the ear decomposition of a biconnected graph, we can find a simple cycle C containing x and y such that every C-to-C path contains a terminal as an internal vertex. As argued at the start of Section 3, this proves the Connectivity Separation Theorem for x and y.

Suppose x and y require triconnectivity, that is $x, y \in Q_3$. Since graph H is Q_3-triconnected, there are three internally vertex-disjoint paths from x to y in H. We modify these three paths such that they do not contain edges of the same terminal-bounded tree. Suppose all three paths contain some edges of a common terminal-bounded tree T. By the Tree Cycle Theorem, there is a cycle $C(T)$ that contains all leaves of T and all other vertices of T are enclosed by $C(T)$. So all the tree paths must intersect cycle $C(T)$. Note that since both of x and y are terminals, the edges incident to x and y are not in the same terminal-bounded tree. So, for each x-to-y path, we can identify non-trivial subpaths: one to-$C(T)$ prefix and one from-$C(T)$ suffix. We can find two subpaths of $C(T)$ and one path in T such that they are vertex-disjoint and the union of these three subpaths together with the to-$C(T)$ prefices and the from-$C(T)$ suffices defines another three internally vertex-disjoint x-to-y paths in H. Only one of the three new paths will contain edges of T. By property (d) of the Tree Cycle Theorem, the two subpaths of $C(T)$ will not introduce any *shared* terminal-bounded tree. We can apply a similar modification when there are only two x-to-y paths containing edges of the same terminal-bounded tree. The argument for extending the property from H to G requires minimal extra work.

3.2 Proof of Tree Cycle Theorem

Let G be a minimal Q_3-triconnected planar graph. We prove the Tree Cycle Theorem for the contracted Q_3-triconnected graph H obtained from G. If the theorem is true for H, then it is true for G since subdivision will maintain the properties of the theorem. We give a high-level overview of the proof.

We focus on a maximal terminal-free tree T^*, rooted arbitrarily, of H and the corresponding terminal-bounded component T (that is, $T^* \subset T$). We show that there is a face of H that does not touch any internal vertex of T^*, which guarantees that there is a drawing of H such that T^* is enclosed by some cycle. We take this face of H as the infinite face. We view T^* as a set $\mathcal{P}$ of root-to-leaf paths. For each path in $\mathcal{P}$, we can find a cycle that strictly encloses only vertices on the paths. The outer cycle of the cycles for all the paths in $\mathcal{P}$ defines $C(T)$. See Figure 4. Property (a) directly follows from the construction. Property (b) is proved

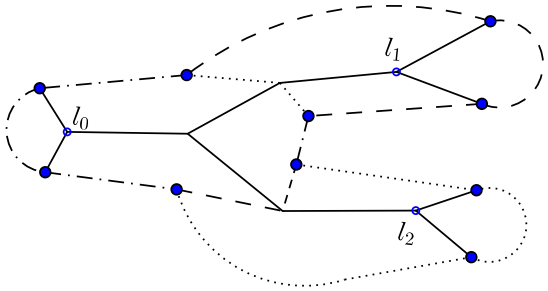

Figure 4 Illustration of $C(T)$. The dashed cycle is C_P for P from l_0 to l_1 and the dotted cycle is $C_{P'}$ for P' from l_0 to l_2. The outer boundary forms $C(T)$.

by induction on the number of root-to-leaf paths of T: when we add a new cycle for a path from $\mathcal{P}$, the new outer cycle will only strictly enclose vertices of the root-to-leaf paths so far considered. After that, we show any two terminals are triconnected when T is a tree: by modifying the three paths between terminals in a similar way to the proof for Connectivity Separation, only one path will require edges in T. Since T is connected, this proves T is a tree by minimality of H. Combining the above properties and triconnectivity of H, we can obtain property (c). Property (d) is proved by contradiction: if there is another terminal-bounded tree T' that shares two terminal-free paths of $C(T)$, then there is a terminal-free path in T'. We can show there is a removable edge in this path of T', contradicting the minimality of H.

4 Proof of the Structure Theorem

In this section, we give a brief overview of the proof of the Structure Theorem (Theorem 3); full details are in the full version of the paper. First we introduce some properties of the mortar graph and bricks. For a brick B, let ∂B be its boundary and $\mathrm{int}(B) = E(B) \setminus E(\partial B)$ be its interior. A path is *ϵ-short* if the distance between every pair of vertices on that path is at most $(1+\epsilon)$ times the distance between them in G. Bricks have the following properties.

▶ **Lemma 9** (Lemma 6.10 [10] rewritten). *The boundary of a brick B, in counterclockwise order, is the concatenation of four paths W_B, S_B, E_B and N_B (west, south, east and north) such that:*

- *Every vertex of $Q \cap B$ is in $N_B \cup S_B$.*
- *N_B is 0-short and every proper subpath of S_B is ϵ-short.*

The paths that form eastern and western boundaries of bricks are called *supercolumns*, and the weight of all edges in supercolumns is at most $\epsilon\mathrm{OPT}$ (Lemma 6.6 [10]). We designate a set of vertices, called *portals*, evenly spaced on the boundary of each brick. Each brick has only constant number (depending on ϵ) of portals on its boundary.

To prove the Structure Theorem, we transform OPT for the instance (G, Q, r) so that it satisfies the following properties (repeated from the introduction):

(P1) $\mathrm{OPT} \cap \mathrm{int}(B)$ can be partitioned into a set of trees $\mathcal{T}$ whose leaves are on the boundary of B.

(P2) If we replace any tree in $\mathcal{T}$ with another tree spanning the same leaves, the result is a feasible solution.

(P3) There is another set of $O(1)$ trees $\mathcal{T}'$ and a set of brick boundary edges B' that costs at most a $1+\epsilon$ factor more than $\mathcal{T}$, such that each tree of $\mathcal{T}'$ has $O(1)$ leaves and $(\mathrm{OPT} \setminus \mathcal{T}) \cup \mathcal{T}' \cup B'$ is a feasible solution.

The transformation consists of the following steps:

Augment. We add four copies of each supercolumn; we take two copies each to be interior to the two adjacent bricks. After this, connectivity between the east and west boundaries of a brick will be transformed to that between the north and south boundaries. Since the weight of all supercolumns is at most $\epsilon\mathrm{OPT}$, this only increases the weight by an small fraction of OPT.

Cleave. By cleaving a vertex, we split it into multiple copies while keeping the connectivity as required by adding artificial edges of weight zero between two copies and maintaining a planar embedding. We call the resulting solution OPT_C. In this step, we turn k-edge-connectivity into k-vertex-connectivity for $k = 1, 2, 3$. By Theorem 7, we can obtain Property P1: $\mathrm{OPT}_C \cap \mathrm{int}(B)$ can be partitioned into a set $\mathcal{T}$ of trees whose leaves are in ∂B. By Corollary 5, we can obtain Property P2: we can obtain another feasible solution by replacing any tree in $\mathcal{T}$ with another tree spanning the same leaves.

Flatten. For each brick B, we consider the connected components of $\mathrm{OPT}_C \cap \mathrm{int}(B)$. If the component only spans vertices in the north or south boundary, we replace it with the minimum subpath of the boundary that spans the same vertices. This will not increase the weight much by the ϵ-shortness of the north and south boundaries. Note that vertex-connectivity may break as a result, but edge-connectivity is maintained. In the remainder, we only maintain edge-connectivity. We call the resulting solution OPT_F.

Restructure. For each brick B, we consider the connected components of $\mathrm{OPT}_F \cap \mathrm{int}(B)$. We replace each component with a subgraph through a mapping ϕ. The new subgraph may be a tree or a subgraph $\widehat{C}$ that is the union of a cycle and two subpaths of ∂B. The mapping ϕ has the following properties:

- For any component χ of $\mathrm{OPT}_F \cap \mathrm{int}(B)$, $\phi(\chi)$ is connected and spans $\chi \cap \partial B$.
- For two components χ_1 and χ_2 of $\mathrm{OPT}_F \cap \mathrm{int}(B)$, if $\phi(\chi_i) \neq \widehat{C}$ for at least one of $i = 1, 2$, then $\phi(\chi_1)$ and $\phi(\chi_2)$ are edge-disjoint, taking into account edge multiplicities.
- The new subgraph $\phi(\mathrm{OPT}_F \cap \mathrm{int}(B))$ has only constant number (depending on ϵ) of vertices in the boundary ∂B.

We can prove that the total weight is increased by at most $\epsilon\mathrm{OPT}_F$, giving Property P3. We call the resulting solution OPT_R.

Redirect. We connect each vertex j of $\mathrm{OPT}_R \cap \mathrm{int}(B)$ in the boundary ∂B to the nearest portal p on ∂B by adding multiple copies of the short j-to-p subpath of ∂B. Similar to 2-ECP, we can prove this only increases the weight by an ϵ fraction of OPT and the resulting solution satisfies the Structure Theorem.

Acknowledgements. We thank Hung Le, Amir Nayyeri and David Pritchard for helpful discussions.

References

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

2 B. Baker. Approximation algorithms for NP-complete problems on planar graphs. *Journal of the ACM*, 41(1):153–180, 1994. doi:10.1145/174644.174650.

3 M. Bateni, M. Hajiaghayi, and D. Marx. Approximation schemes for Steiner forest on planar graphs and graphs of bounded treewidth. *J. ACM*, 58(5):21, 2011. doi:10.1145/2027216.2027219.

4 A. Berger, A. Czumaj, M. Grigni, and H. Zhao. Approximation schemes for minimum 2-connected spanning subgraphs in weighted planar graphs. In *Proceedings of the 13th European Symposium on Algorithms*, volume 3669 of *Lecture Notes in Computer Science*, pages 472–483, 2005.

5 A. Berger and M. Grigni. Minimum weight 2-edge-connected spanning subgraphs in planar graphs. In *Proceedings of the 34th International Colloquium on Automata, Languages and Programming*, volume 4596 of *Lecture Notes in Computer Science*, pages 90–101, 2007. doi:10.1007/978-3-540-73420-8_10.

6 G. Borradaile, E. Demaine, and S. Tazari. Polynomial-time approximation schemes for subset-connectivity problems in bounded-genus graphs. *Algorithmica*, 2012. Online. doi:10.1016/j.jda.2012.04.011.

7 G. Borradaile, C. Kenyon-Mathieu, and P. Klein. A polynomial-time approximation scheme for Steiner tree in planar graphs. In *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms*, volume 7, pages 1285–1294, 2007.

8 G. Borradaile and P. Klein. The two-edge connectivity survivable network problem in planar graphs. In *Proceedings of the 35th International Colloquium on Automata, Languages and Programming*, pages 485–501, 2008.

9 G. Borradaile and P. Klein. The two-edge connectivity survivable-network design problem in planar graphs. *ACM Transactions on Algorithms*, 12(3):30:1–30:29, 2016.

10 G. Borradaile, P. Klein, and C. Mathieu. An $O(n \log n)$ approximation scheme for Steiner tree in planar graphs. *ACM Transactions on Algorithms*, 5(3):1–31, 2009.

11 G. Borradaile and B. Zheng. A PTAS for three-edge-connected survivable network design in planar graphs. *CoRR*, abs/1611.03889, 2016. URL: http://arxiv.org/abs/1611.03889.

12 A. Czumaj and A. Lingas. A polynomial time approximation scheme for euclidean minimum cost k-connectivity. In *Automata, Languages and Programming*, pages 682–694. Springer, 1998.

13 A. Czumaj and A. Lingas. On approximability of the minimum cost k-connected spanning subgraph problem. In *Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 281–290, 1999.

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

15 K. Eswaran and R. Tarjan. Augmentation problems. *SIAM Journal on Computing*, 5(4):653–665, 1976.

16 G. Frederickson and J. Jájá. Approximation algorithms for several graph augmentation problems. *SIAM Journal on Computing*, 10(2):270–283, 1981.

17 D. A. Holton, B. Jackson, A. Saito, and N. C. Wormald. Removable edges in 3-connected graphs. *J. Graph Theory*, 14:465–475, 1990.

18 K. Jain. A factor 2 approximation algorithm for the generalized Steiner network problem. *Combinatorica*, 2001(1):39–60, 21.

19 S. Khuller and U. Vishkin. Biconnectivity approximations and graph carvings. *Journal of the ACM*, 41(2):214–235, 1994.

20 P. Klein. A subset spanner for planar graphs, with application to subset TSP. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, pages 749–756, 2006. doi:10.1145/1132516.1132620.

21 P. Klein. A linear-time approximation scheme for TSP in undirected planar graphs with edge-weights. *SIAM Journal on Computing*, 37(6):1926–1952, 2008.

22 P. Klein, C. Mathieu, and H. Zhou. Correlation clustering and two-edge-connected augmentation for planar graphs. In Ernst W. Mayr and Nicolas Ollinger, editors, *32nd International*

Symposium on Theoretical Aspects of Computer Science (STACS 2015), volume 30 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 554–567. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015.

23 P. Klein and R. Ravi. When cycles collapse: A general approximation technique for constraind two-connectivity problems. In *Proceedings of the 3rd International Conference on Integer Programming and Combinatorial Optimization*, pages 39–55, 1993.

24 A. Sebő and J. Vygen. Shorter tours by nicer ears: 7/5-approximation for the graph-tsp, 3/2 for the path version, and 4/3 for two-edge-connected subgraphs. *Combinatorica*, pages 1–34, 2014.

25 H. Whitney. Non-separable and planar graphs. *Trans. Amer. Math. Soc.*, 34:339–362, 1932.

26 B. Zheng. Linear-time approximation schemes for planar minimum three-edge connected and three-vertex connected spanning subgraphs. *CoRR*, abs/1701.08315, 2017. URL: `http://arxiv.org/abs/1701.08315`.