Approximation Schemes for Minimum 2-Connected Spanning Subgraphs in Weighted Planar Graphs*

André Berger[1], Artur Czumaj[2], Michelangelo Grigni[1], and Hairong Zhao[2]

[1] Department of Mathematics and Computer Science, Emory University, Atlanta GA 30322, USA. Email: aberge2@emory.edu, mic@mathcs.emory.edu.
[2] Department of Computer Science, New Jersey Institute of Technology, Newark NJ 07102, USA. Email: {czumaj,hairong}@cis.njit.edu.

Abstract. We present new approximation schemes for various classical problems of finding the minimum-weight spanning subgraph in edge-weighted undirected *planar graphs* that are resistant to edge or vertex removal. We first give a PTAS for the problem of finding minimum-weight 2-edge-connected spanning subgraphs where duplicate edges are allowed. Then we present a new greedy spanner construction for edge-weighted planar graphs, which augments any connected subgraph A of a weighted planar graph G to a $(1+\varepsilon)$-spanner of G with total weight bounded by weight$(A)/\varepsilon$. From this we derive quasi-polynomial time approximation schemes for the problems of finding the minimum-weight 2-edge-connected or biconnected spanning subgraph in planar graphs. We also design approximation schemes for the minimum-weight 1-2-connectivity problem, which is the variant of the survivable network design problem where vertices have non-uniform (1 or 2) connectivity constraints. Prior to our work, for all these problems no polynomial or quasi-polynomial time algorithms were known to achieve an approximation ratio better than 2.

1 Introduction

The *survivable network design problem* is to design graphs that resist edge and/or vertex removal. This is a fundamental problem in algorithmic graph theory with numerous applications in computer science and operations research (see, e.g., [13,16,23]). The classical k*-connectivity* problems are perhaps the most extensively studied problems in network design. We are given a graph G with n vertices and a nonnegative weight $w(e)$ on each edge e, and we want to find a k-edge or k-vertex connected *spanning* subgraph S, such that its total edge weight $w(S)$ equals OPT, the minimum possible. It is well-known that all non-trivial variants of the survivable network design problem are $\mathcal{NP}$-hard and therefore the main research interest lies in the design of efficient approximation algorithms (see, e.g., [16] for a survey).

In this paper we consider approximation algorithms for the most basic case of the survivable network design problem in which the resulting subgraphs should be resistant to the removal of a *single* edge or vertex. The two classical problems here are to find a minimum-weight 2-edge-connected (2-EC) spanning subgraph (a 2-ECSS) of G, or a 2-vertex-connected (2-VC or biconnected) spanning subgraph (a 2-VCSS) of G. We also

* Research supported in part by NSF grants CCR-0208929 and ITR-CCR-0313219.

consider a standard relaxation of the 2-ECSS problem of finding a minimum weight 2-EC spanning sub-*multigraph* (2-ECSSM) H of G, meaning that an edge of G can be used multiple times in H (consequently its weight is also counted multiple times in H). Another classical extension is the *1-2-connectivity problem*: each vertex v is assigned a connectivity type $r_v \in \{1, 2\}$. The problem is to find a minimum weight spanning subgraph such that for any pair of vertices $v, u \in V$, there are at least $r_{uv} = \min\{r_u, r_v\}$ edge-disjoint or vertex-disjoint paths between v and u. We denote the 1-2-edge-connectivity by {1,2}-EC, and 1-2-vertex-connectivity by {1,2}-VC. We also consider the relaxed 1-2-edge-connectivity problem where each edge may be used more than once.

All the problems mentioned above have been extensively studied in the literature. Since all these problems are $\mathcal{NP}$-hard, the main research has been devoted to design efficient approximation algorithms, see the survey [16] and more recent advances [5,11,12,15,18]. In general, we would prefer to design a *polynomial-time approximation scheme (PTAS)*, which is a c-approximation algorithm taking both G and c as inputs, and running in polynomial time for each fixed $c > 1$. However, all the problems we consider in this paper are max-SNP-hard [7], even for unweighted graphs or when duplicate edges are allowed; therefore they do not have a PTAS unless $\mathcal{P} = \mathcal{NP}$. But this does not preclude a PTAS for restricted classes of graphs: indeed, there exist PTAS's for all these problems in *geometric graphs* in low dimensions [7,9], and also for the 2-ECSS and 2-VCSS problems in *unweighted planar graphs* [6]. In fact, the approximation schemes of [6] allow *weighted* planar graphs, but then the algorithms will either run in time $n^{O(\frac{w(G)}{\varepsilon \cdot \mathrm{OPT}})}$ to ensure an $(1+\varepsilon)$-approximate solution or they run in polynomial time with an approximation guarantee of 1+$O(\frac{w(G)}{\varepsilon \cdot \mathrm{OPT}})$. Since the ratio $w(G)/\mathrm{OPT}$ could be arbitrarily large, these algorithms are in general not PTAS's for weighted planar graphs.

For both the 2-ECSS and 2-VCSS problems in weighted planar graphs, the best known polynomial-time or even quasi-polynomial-time approximation guarantee is still 2 [17,21], which is achieved by polynomial-time algorithms working for general weighted graphs. On the other hand, besides the PTAS for unweighted planar graphs [6], there are better constant approximation guarantees known for general unweighted graphs. For example, there exists a $\frac{5}{4}$-approximation algorithm for the unweighted 2-ECSS problem [15], and a $\frac{4}{3}$-approximation algorithm for the unweighted 2-VCSS problem [24].

A similar phenomenon can be seen for the 1-2-connectivity problems. For both, the unweighted {1,2}-ECSS and the unweighted {1,2}-VCSS problem, Krysta [19] gives $\frac{3}{2}$-approximation algorithms. If the graph is weighted, the best known result for {1,2}-ECSS is a 2-approximation algorithm, due to Jain [14], which in fact solves the more general problem where $r_v \leq k$ for any k. For the weighted {1,2}-VCSS problem, Fleischer [10] gives a 2-approximation algorithm, which actually solves the {0,1,2}-VCSS problem. A PTAS for the geometric version of these problems is presented in [9].

The discrepancy between approximation guarantees of the unweighted and weighted versions of these problems is a phenomenon that is known also for general graphs. A striking discrepancy is known for the k-VCSS problem ($k \gg 2$), which admits a $(1 + 1/k)$-approximation for the unweighted case [4], while for the weighted version of the problem the best known approximation guarantee is $O(\ln k \cdot \min\{\sqrt{k}, \frac{n}{n-k} \ln k\})$ [18]

for any k, and is $O(\ln k)$ when $n \geq 6\,k^2$ [5]. In general, a PTAS for unweighted graphs in some family does not seem to imply a PTAS for weighted graphs in the same family. In particular, the existence of polynomial-time or even quasi-polynomial-time approximation schemes for these problems in weighted planar graphs has remained as a major open question in the area.

1.1 New Contributions and Techniques

We present efficient approximation schemes for all the above mentioned problems in weighted planar graphs. Our approximation algorithms depend in a crucial way on our new construction of *light spanners* for planar graphs.

Let G be a weighted graph. We use $d_G(u, v)$ to denote the weighted shortest path distance between the vertices u and v in G. An *s-spanner* of G is a spanning subgraph H of G such that $d_H(u, v) \leq s \cdot d_G(u, v)$ for all u, v. A spanner provides an approximate representation of the shortest path metric (1-connectivity) in G, but it may be much lighter than G.

Althöfer et al. [1] designed a simple greedy algorithm that for an arbitrary graph G computes an s-spanner H of G for any $s > 1$. In the case of *planar* graphs, it is shown in [1] that this spanner has weight $w(H) \leq (1 + 2/(s-1))\mathrm{MST}(G)$, where $\mathrm{MST}(G)$ is the weight of a minimum spanning tree in G. Since $\mathrm{MST}(G) \leq$ OPT for all the problems we consider, this bounds the ratio $w(H)/$OPT in terms of just s. If all weighted graphs in a graph family have spanners with such a bound on $w(H)/$OPT (depending only on s), then we say the family has *light spanners* for this problem. Light spanners are known to be very useful for solving various optimization problems on graphs. For example, planar graphs have light spanners for metric-TSP: the first step in the metric-TSP PTAS for weighted planar graphs [2] is to replace the input graph with an accurate enough s-spanner (using [1]), thus effectively bounding $w(G)/$OPT for the remainder of the algorithm. Spanners are also used in complete geometric graphs to design efficient PTAS's for geometric TSP and related problems [22], and to design PTAS's for the 2-edge and 2-vertex-connectivity problems [8,9].

By combining the spanner constructed in [1] with the planar separator decomposition approach tuned to analyze 2-connected graphs [6], we show that one can design a PTAS for the 2-ECSSM problem and a PTAS for the $\{1,2\}$-ECSSM problem. However, this approach of replacing the input graph with an s-spanner fails for the 2-ECSS and 2-VCSS problems. The reason is that a spanner does not have to be 2-connected, thus may not contain the optimal or a near optimal solution in most cases. Naturally, one may think to use light *fault-tolerant* spanners (see, e.g., in [20]), which are subgraphs that persist as s-spanners even after deleting a constant number of vertices or edges. Unfortunately, this concept is not useful in our setting, since simple examples show that light fault-tolerant spanners do not exist in weighted planar graphs, not even for a single edge deletion.

To solve the problem mentioned above, we present our main contribution: a new greedy spanner construction which produces a light planar spanner with certain desirable properties. Specifically, given a weighted planar graph G, a connected spanning subgraph A of G and $s > 1$, it computes an s-spanner H of G. H contains A as a subgraph and has total weight $w(H) = O(1/(s-1) \cdot w(A))$. Thus if we feed the algorithm

with α-approximate solutions H to the various connectivity problems in a weighted planar graph G, then we obtain an $O(\alpha/(s-1))$-approximation H^* for that problem, which at the same time is an s-spanner for G. Furthermore, we can show that while H^* need not contain an $(1+\varepsilon)$-approximate solution S, we do put a bound on the number of edges of S "crossing" each face of H^* (Lemma 3).

Using our new spanner construction technique and the planar separator decomposition, we design approximation schemes for the 2-ECSS and 2-VCSS, {1,2}-ECSS and {1,2}-VCSS problems, which find solutions with weight at most $(1+\varepsilon)\cdot$ OPT in $n^{O(\log n \log(1/\varepsilon)/\varepsilon)}$ time; these are *quasi-polynomial time approximation schemes* (QPTAS's).

Organization. We first present a PTAS for the 2-ECSSM problem in Section 2. This section contains also a description of the main algorithmic approach used in our approximation schemes, which is a combination of the use of spanners, a recursive approach driven by a variant of the planar separator theorem, and dynamic programming. Next, in Sections 3 and 4, we describe our new construction of spanners and discuss the special properties of the spanners. In Section 5, we present quasi-polynomial approximation schemes for the 2-ECSS and the 2-VCSS problems. Finally in Section 6, we consider {1,2}-ECSS and {1,2}-VCSS problems: we show a PTAS for the {1,2}-ECSSM problem, and a QPTAS for each of the {1,2}-ECSS and {1,2}-VCSS problems.

2 PTAS for the 2-ECSSM Problem

Let G be a connected weighted graph. A 2-ECSSM H of G is a spanning sub-multigraph of G in which edges can have some multiplicity and in which every pair of vertices is connected by at least two edge-disjoint paths. Note that G may not have any multiple edges at all. If an edge is used multiple times in H, its weight also contributes multiple times to the weight of H. Since it never helps to use an edge more than twice, we may cap all edge multiplicities at two. We now present a PTAS for this problem, running in $n^{O(1/\varepsilon^2)}$ time.

Given G and $\varepsilon > 0$, we choose s so that $s^2 \le 1+\varepsilon$. We first compute an s-spanner H in G by the greedy spanner algorithm [1], with weight $w(H) = O((1/\varepsilon)\cdot \text{OPT})$. Now we show that there is a $(1+\varepsilon)$-approximate 2-ECSSM that uses only edges from H. Suppose S^* is an optimal 2-ECSSM in G with $w(S^*) = \text{OPT}$. Now we modify S^* such that it uses only edges from H. For each edge e of S^* not in H, we remove e and add a shortest path from H of total weight at most $s\cdot w(e)$. When we add the path, we add the edges with multiplicity, but capped at two. The result of all these modifications is another 2-ECSSM S, using only edges from H, each edge used at most twice, with $w(S) \le s\cdot \text{OPT}$.

Next we apply the 2-ECSS s-approximation algorithm from [6] to the graph H', which is H with each edge duplicated. Since this algorithm forms also a core of our algorithm for other problems discussed later in Sections 5 and 6, we briefly describe it here. The algorithms in [6] use a recursive approach driven by the following planar separator theorem from [3] (see also [6]):

Lemma 1. *Let G be a connected planar graph on $n \geq 3$ vertices embedded in the plane. Suppose G has non-negative weights on its vertices, edges and faces, and non-negative costs on its edges. Let W be the total weight of the graph and let M be its total cost and assume that no edge has weight more than $(3/4) \cdot W$. Then for any positive integer k, we can find a subgraph F of G and a closed Jordan curve J in $O(n)$ time such that:*

1. *F is the union of at most two vertex-disjoint simple cycles (maybe none). The total cost of the edges on each cycle is at most M/k. If F contains two cycles A and C, then $interior(C) \subset interior(A)$. The interior of C and the exterior of A (if they exist) both have weight at most $W/2$.*
2. *Denote by G' the embedded graph that results after deleting the interior of C and the exterior of A (if they exist) and contracting each cycle in F to a vertex of weight 0. Then J is a Jordan curve which intersects edges of G' only at their endpoints and passes through $O(k)$ vertices ("portals") including the new contracted vertices. The interior and exterior of J both have weight at most $(3/4) \cdot W$.*

First, we decompose H' according to Lemma 1 (with $k = \Theta((\log n/\varepsilon) \cdot \frac{w(H')}{\mathrm{OPT}}) = \Theta(\log n/\varepsilon^2)$) into at most four pieces: the interior of the cycle C, the exterior of the cycle A, and the interior and the exterior of the Jordan curve J. By assigning weight to the new portals and faces properly, we can make sure that each piece has weight at most a constant fraction of the H'. We continue to decompose the small pieces recursively. It is easy to see that the depth of the recursion is logarithmic, and the number of pieces is $O(n \log n)$. By the weighting scheme of the new portals ([2,6]), one can show that each piece has a portal set P with size $O(k)$.

We would like to find a low cost spanning subgraph in each piece and then combine them together to form an almost minimum wight 2-ECSS of H'. Of course we do not know the remaining subgraph outside this piece. Thus, for each piece, we enumerate all the different ways that some subgraph of H' (outside this piece) may influence the connectivity constraints within this piece. We call these the *external types* of the piece, and one can show that the number of such types is $2^{O(|P|)} = n^{O(1/\varepsilon^2)}$ [6, Lemma 2.4], where P is the set of portals of this piece.

For each piece and each external type, we must find a near minimum cost subgraph of the piece, so that this subgraph together with the external type can meet the global connectivity constraints. We use dynamic programming to solve the subproblems in each of the pieces.

During the course of the algorithm, we always commit the cycle edges found in the separator to the solution, which is the only source of the error in our algorithm. Since the cycle edges has total cost at most $w(H')/k$ at each level of the recursion and there are at most $O(\log n)$ levels, by setting appropriately the leading constant in k, we can show that the total error introduced in the algorithm is $\varepsilon \cdot \mathrm{OPT}$. For each piece, the number of types is bounded by $n^{O(w(H')/(\varepsilon \cdot \mathrm{OPT}))} = n^{O(1/\varepsilon^2)}$. There are $O(n \log n)$ pieces. Therefore the algorithm solves $n^{O(1/\varepsilon^2)}$ subproblems, each in time $n^{O(1/\varepsilon^2)}$.

In summary, we have the following theorem.

Theorem 1. *Let $\varepsilon > 0$ and let G be a connected weighted planar graph with n vertices. There is an algorithm running in time $n^{O(1/\varepsilon^2)}$ that outputs a 2-ECSSM of G whose weight is at most $(1+\varepsilon)$ times the minimum.*

Why this technique fails for other problems: The above technique does not work for other problems considered in this paper, because there we are not allowed to duplicate edges from G in the output graph. Instead, our approximation schemes must consider the possibility that the near-optimal S needs some "extra" edges from outside the spanner. In Sections 3 and 4 we develop a new type of light planar spanners and we limit the number and arrangement of those extra edges outside the spanner.

3 Augmented Planar Spanners

In this section we present a new greedy algorithm constructing s-spanners in weighted planar graphs, resembling the standard greedy algorithm [1] for general graphs. Just as in the standard algorithm, we take a connected weighted graph G and a parameter $s \geq 1$, and produce an s-spanner H. Unlike the general algorithm, our G must be planar, and for each edge e of G not in H we guarantee that $s \cdot w(e)$ is at least the length of some path in the face of H containing e. We also provide our algorithm with a third argument: a "seed" spanning subgraph A, containing edges that must appear in H. In Section 4 we will use A to enforce some 2-connectivity properties in the spanner.

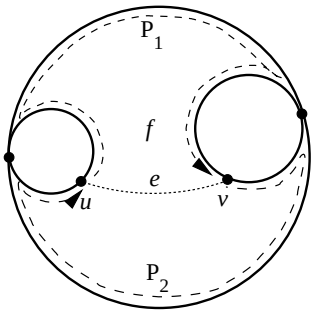

Fig. 1. A non-simple face f in H, a chord e, and walks P_1 and P_2.

Suppose G is a weighted *plane graph* (that is, an embedded planar graph) and H is a subgraph. A *chord* e of H is an edge of G not in H. Note that H and e inherit embeddings from G. For each chord e we define $w_H(e)$ as the length of the shortest walk connecting the endpoints of e, along the boundary of the face of H containing e.

More precisely, if the endpoints of e are disconnected in H, then we define $w_H(e) = +\infty$. Otherwise e connects two vertices in a component of H, and e is embedded in some face f of this component. The boundary of f is a cyclic walk of (oriented) edges, with total weight $w(f)$; note that a cut-edge may appear twice in the boundary (once per orientation), and its weight would then count twice in $w(f)$. Similarly a cut-vertex

may appear multiple times. The edge e splits the boundary sequence into two walks P_1 and P_2, both connecting the endpoints of e, with $w(P_1) + w(P_2) = w(f)$. Now we define $w_H(e) = \min(w(P_1), w(P_2))$ (see Figure 1).

Given G, s, and A as above, we compute $H = \text{Augment}(G, s, A)$ as follows:

```
Augment(G, s, A):
    H ← A
    for all edges e of G in non-decreasing w(e) order do
        if e is not in H and s · w(e) < w_H(e) then
            add e to H
    return H
```

Note $A \subseteq H \subseteq G$. If A is empty (has all vertices of G but no edges), then this is like the general greedy spanner algorithm [1], except that we have w_H in place of d_H.

Theorem 2. *Let G be a weighted plane graph, $s > 1$, and A a spanning subgraph of G. Then $H = \text{Augment}(G, s, A)$ is an s-spanner of G. If A is connected, then $w(H) \leq (1 + 2/(s-1)) \cdot w(A)$.*

Proof. To show that H is an s-spanner it suffices to show that each edge of G is s-approximated in H. For e not in H, at the moment it was rejected we had $w_H(e) \leq s \cdot w(e)$. Note that $w_H(e)$ may only decrease after that, so $d_H(e) \leq w_H(e) \leq s \cdot w(e)$ at the end of the algorithm.

For the second part we need to show that the weight of all edges in H but not A is at most $(2/(s-1)) \cdot w(A)$. Suppose e is such an edge; then e is not a cut edge in H since A is a connected spanning subgraph. Therefore e is bounded by two distinct faces. Let f be either face bounding e. We first claim that $w(f) > (1 + s) \cdot w(e)$. To see this, consider the *last* edge e' added to f whose boundary consists of a path P plus e'. Since e' is added to H, we must have that $s \cdot w(e') < w_H(e')$ and $w_H(e') \leq w(P)$. Adding $w(e')$ to both sides of $s \cdot w(e') < w(P)$, and noting $w(e) \leq w(e')$, we get the claim.

For each face f of A, let E_f be the set of edges in H crossing the interior of f. Since the sum of $w(f)$ over all faces of A is $2 \cdot w(A)$, it suffices to show that $w(E_f) \leq (1/(s-1)) \cdot w(f)$. Note that the edges dual to E_f define a tree on the faces of H inside f. Orient this dual tree away from some arbitrarily chosen root: now for each $e \in E_f$, we have chosen an adjacent face f_e of H (only the root was not picked). For each $e \in E_f$ we know $w(f_e) - 2 \cdot w(e) > (s-1) \cdot w(e)$, from the previous paragraph. Summing these inequalities over all $e \in E_f$, we get at most $w(f)$ on the left hand side, and exactly $(s-1) \cdot w(E_f)$ on the right.

4 Spanners and 2-EC Subgraphs

Suppose we are given a weighted plane 2-EC graph G, where we want to find an $(1+\varepsilon)$-approximate 2-ECSS. We first construct an auxiliary subgraph H^*, as follows:

1. Compute a 2-approximate 2-ECSS A, in polynomial time.
2. Compute $H^* = \text{Augment}(G, \sqrt{2}, A)$.

The constant $\sqrt{2}$ here is not critical, just convenient. By Theorem 2, H^* is a 12-approximate 2-ECSS. Below we show that for every $\varepsilon > 0$, this H^* has nice intersection properties with some $(1+\varepsilon)$-approximate 2-ECSS in G.

Given a face f in H^*, the chords of f are the edges of G embedded inside this face, according to G's embedding. A *face-edge* e of f is an abstract edge connecting two vertices of f; unlike a chord, a face-edge is not necessarily an edge of G. (If vertices appear more than once on f, we must specify which appearances we want as the endpoints of e.) We say the face edge e *crosses* a chord c if: c is a chord of the same face f, their endpoints are distinct vertex appearances on f, and they appear in cyclic "$ecec$" order around the boundary of f. Note that we may embed e inside f so e intersects only the crossed chords.

Suppose S is a 2-ECSS in G, and an edge c of S is not in H^*. Then c is a chord of some face f of H^*. Let P_c be the path in f connecting the endpoints of c, such that $w(P_c) \leq \sqrt{2} \cdot w(c)$. Then the *chord move at* c is the following modification of S: add to S all the edges of P_c that were not already in S, and remove from S any chords inside the cycle $c \cup P_c$ (see Figure 2(a)). Since H^* is 2-EC, the cycle has no repeated edges, and therefore S is still a 2-ECSS after the chord move. The chord move is *improving* if $w(S)$ decreases; this happens whenever $w(P_c)$ (or $\sqrt{2} \cdot w(c)$) is less than the weight of the discarded chords. Any non-trivial chord move brings S closer to H^* (in Hamming distance), thus at most $O(n)$ improving chord moves apply to any given 2-ECSS S.

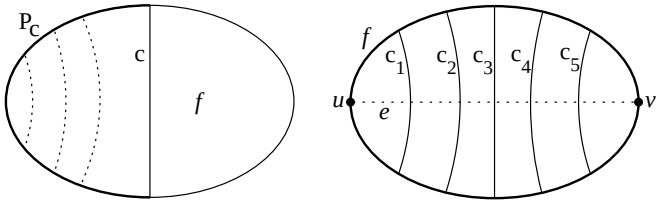

Fig. 2. (a) Face f of H^* (oval) with chord c, path P_c (bold), and chords removed from S by the chord move at c (dotted). (b) Face f with a face-edge e (dashed) crossed by five chords from S.

Lemma 2. *Let e be a face edge of a face f in H^* and S be a 2-ECSS of G. Suppose C is a set of edges of S all of which are chords crossing e, $\sqrt{2} \cdot \min_{c \in C} w(c) > \max_{c \in C} w(c)$, and no chord in C gives an improving chord move. Then $|C| \leq 4$.*

Proof. If not, S has five chords crossing e as in Figure 2(b). But then we have an improving chord move at c_3, since the discarded chords ($\{c_1, c_2\}$ or $\{c_4, c_5\}$) weigh more than $\sqrt{2} \cdot w(c_3)$.

Now we argue that by accepting a small additive error in our 2-ECSS, we may assume it has only a small number of chords crossing a given face-edge:

Lemma 3. *Suppose G and H^* are as above, $\varepsilon > 0$, S is a 2-ECSS, f is a face in H^*, and e is a face-edge in f. Then there exists a 2-ECSS S' such that $w(S') \leq w(S) + \varepsilon \cdot w(C'_f)$, where C'_f is the set of edges of S' that are chords crossing e, $C'_f \subseteq S$, and $|C'_f| = O(\log(1/\varepsilon))$.*

Proof. First we may suppose that S has no improving chord move at a chord crossing e, since such a move could only remove some chords crossing e. Let C_f be the set of chords in S crossing e. Arrange C_f in "left to right" order, according to how they intersect e. Let $c_0 \in C_f$ be the chord with maximum weight. Say that a chord $c \in C_f$ is *short* if $w(c) \leq \varepsilon \cdot w(c_0)/(2\sqrt{2})$. Now if there are short chords to the left of c_0, perform a chord move at the rightmost one, c_l. Similarly if there are short chords to the right of c_0, perform a chord move at the leftmost one, c_r. S' is the result of these (at most) two chord moves; note that C'_f contains no short chords except possibly c_l and c_r.

Map each non-short chord $c \in C'_f$ to the real number $\log(w(c_0)/w(c))$, a point in the real interval $I = [0, \log(1/\varepsilon) + 3/2]$. Note that two edges can be mapped to the same semi-open subinterval of I of length $1/2$ only if the heavier edge has weight less than $\sqrt{2}$ times that of the lighter edge. By Lemma 2, at most four edges can be mapped into the same subinterval of I of length $1/2$. This implies $|C'_f| = O(\log(1/\varepsilon))$.

The chord moves in f increased $w(S')$ by at most $\sqrt{2}(w(c_l) + w(c_r)) \leq \varepsilon \cdot w(c_0)$, which is at most $\varepsilon \cdot w(C'_f)$.

Remarks: In the 2-VCSS case, the initial A should be a 2-approximate 2-VCSS, so that H^* is a 12-approximate 2-VCSS. Then in the chord move the cycle has no repeated vertices, therefore S remains a 2-VCSS after the move. In Lemmas 2 and 3, the only properties of H^* that we needed were that it was 2-EC (or 2-VC), and that $w_{H^*}(e) \leq \sqrt{2} \cdot w(e)$ for each chord e.

5 Approximation Schemes for the 2-ECSS and 2-VCSS Problems

In this section we will show how to use our new spanner construction to find quasi-polynomial time approximation schemes for the 2-ECSS and the 2-VCSS problems in weighted planar graphs. We start with the QPTAS for 2-ECSS problem.

We use a similar framework as that in the PTAS for 2-ECSSM problem in Section 2. But instead of using the spanner constructed as in [1], now we use the augmented spanner H^* as constructed in Section 4.

We first apply Lemma 1 to H^* with $k = \Theta(\log n/\varepsilon)$ to decompose H^*. However, different from the PTAS for the 2-ECSSM problem, H^* may not contain a near-optimal solution of the 2-ECSS problem. Thus we cannot work on the pieces of H^* directly. Fortunately, Lemma 3 guarantees that there exists a near-optimal solution with at most $O(k \log(1/\varepsilon))$ edges crossing the Jordan curve J. We guess these crossing edges by trying all $n^{O(k \log(1/\varepsilon))}$ possibilities. We add the guessed edges to the corresponding pieces, and the vertices of H^* along J together with the endpoints of the guessed edges determine the set of portals P for the new pieces. For each possible new piece with the guessed edges, we assign weights to the new portals such that each new piece has cost at most constant fraction of H^* and $O(k)$ portals. Then we recursively decompose the new pieces.

As in the PTAS for the 2-ECSSM problem, for each piece we define edge-connectivity types which describe how these portals may be connected outside one of the pieces in a $(1+\varepsilon)$-approximate solution. The number of types for each piece is $2^{O(k \log(1/\varepsilon))}$, exponential in the number of portals. Then, we use dynamic programming to solve the subproblems as before and we commit the cycle edges to the solution.

The approximation scheme for the 2-VCSS problem is similar, and we only mention the differences: first, we redefine H^* as remarked at the end of Section 4, and then we need to define vertex-connectivity types using the same techniques as in [6].

The error of our final solution comes from two sources. First, we committed the edges of the cycles that arose from the application of the separator theorem to the solution. Since each piece in the decomposition has weight at most constant fraction of its parent weight, the depth of the recursive calls is $O(\log n)$. As before, the total error per recursive level is $O((w(H^*)/k) \log n)$, where $k = \Theta(\log n/\varepsilon)$ and $w(H^*) = O(\text{OPT}/\varepsilon)$. By an appropriate choice of the leading constant defining k, this is at most $(\varepsilon/2) \cdot \text{OPT}$.

Moreover, each time a face of H^* (or its pieces) is cut by a Jordan curve, we guess $O(\log(1/\varepsilon))$ crossing edges. If we guess these edges optimally (they were edges in some original optimal S^*), then by Lemma 3 we may pay an additive error of at most $\varepsilon/2$ times the weight of these guessed edges. Summing over the entire assembly of a possible solution, the total of these errors is at most $(\varepsilon/2) \cdot \text{OPT}$.

The dominating factor in the running time comes from trying all $n^{O(k \log(1/\varepsilon))}$ possibilities for the guessed edges. The weights of the subproblems are only a constant times the weight of their respective parents and therefore a pure recursive approach (without dynamic programming) leads to a time bound of $T(n) \leq n^{O(k \log(1/\varepsilon))} T(c \cdot n)$ $(0 < c < 1)$, with solution $n^{O((1/\varepsilon) \cdot \log(1/\varepsilon) \cdot \log^2 n)}$. We may improve this bound by a logarithmic factor in the exponent by using dynamic programming and by a more careful count of subproblems. The following lemma proved in [3] bounds the number of ways how a graph can be decomposed regardless of its weight scheme.

Lemma 4. *Let G be a planar graph on n vertices with non-negative edge costs, embedded in the plane, and a parameter $k \geq 1$. Then we can find a list of $O(n^2)$ separations of G, such that for any valid weight scheme of the vertices, edges and faces of G, some separation in this list satisfies the properties of Lemma 1.*

Lemma 4 shows that a piece is partitioned in only $O(n^2)$ different ways, no matter how many arrangements of the vertices along the Jordan curve and incident with the "guessed" edges we try. This implies:

Lemma 5. *The total number of distinct pieces (contracted subgraphs) of the original H^* that occur during our recursive decomposition is $n^{O(\log n)}$. Therefore the number of distinct subproblems (a piece, $|P| = O(k \log(1/\varepsilon))$ portals selected in the piece, and an external connectivity type on those portals) is $n^{O(\log n)} n^{O(|P|)} 2^{O(|P|)} = n^{O(k \log(1/\varepsilon))}$.*

Theorem 3. *Let $\varepsilon > 0$ and let G be a 2-EC (2-VC) weighted planar graph with n vertices. There is an algorithm running in time $n^{O(\log n \cdot \log(1/\varepsilon)/\varepsilon)}$ that outputs a 2-ECSS (2-VCSS) H of G such that $w(H) \leq (1+\varepsilon) \cdot \text{OPT}$.*

6 Extensions to the $\{1, 2\}$-Connectivity Problem

In this section, we extend our results to the $\{1, 2\}$-connectivity problems in weighted planar graphs. We focus on the algorithm for the $\{1, 2\}$-ECSS problem only. The algorithm for the $\{1, 2\}$-VCSS problem can be obtained similarly. The algorithms presented are modifications of the respective algorithms in Sections 2 and 5.

First, consider the $\{1, 2\}$-ECSSM problem, which is a relaxed version of the $\{1, 2\}$-ECSS problem where duplicate edges are allowed. As in Section 2, we can show that there is a $(1+\varepsilon)$-approximate $\{1, 2\}$-ECSSM that uses only edges from a light $(1+\varepsilon)$-spanner H. So instead of G, we can work on H with duplicated edges.

The main difference from Section 2 is the dynamic programming part. We need to redefine the connectivity types to reflect the non-uniform connectivity requirement. For this, we can use the connectivity type construction in [9]. Informally, the main difference is that each time we contract a 2-connected component or path, we assign the highest connectivity requirement among all contracted vertices to the new vertex. This increases the number of types from $2^{O(|P|)}$ to $2^{O(2|P|)}$, where P is the set of portals in the given graph. We again obtain a PTAS with running time $n^{O(1/\varepsilon^2)}$.

Now consider the $\{1, 2\}$-ECSS problem. We first find a 2-approximate solution A using algorithms from [14] (or [10] for $\{1, 2\}$-VCSS). Then we augment A into a light spanner H^* as in Section 4. Using similar arguments as in the proof of Lemma 3, we can show that there is a $(1 + \varepsilon)$-approximate $\{1, 2\}$-ECSS S so that for each picked face-edge e, only $O(\log(1/\varepsilon))$ edges of S cross e. Now redefine the connectivity types as above. Finally, we use dynamic programming to solve the problem. The running time is still dominated by the number of subproblems $n^{O(\log n \cdot \log(1/\varepsilon)/\varepsilon)}$. Hence, we get a QPTAS in this case.

Our results in this section are summarized as follows.

Theorem 4. *Let $\varepsilon > 0$ and let G be a weighted planar graph with n vertices. There is an algorithm running in time $n^{O(1/\varepsilon^2)}$ that outputs a $\{1,2\}$-ECSSM of G whose weight is at most $(1 + \varepsilon) \cdot$ OPT.*

Theorem 5. *Let $\varepsilon > 0$ and let G be a weighted planar graph with n vertices. There is an algorithm running in time $n^{O(\log n \cdot \log(1/\varepsilon)/\varepsilon)}$ that outputs a $\{1,2\}$-ECSS H of G such that $w(H) \leq (1 + \varepsilon) \cdot$ OPT.*

Theorem 6. *Let $\varepsilon > 0$, and let G be a weighted planar graph with n vertices. There is an algorithm running in time $n^{O(\log n \cdot \log(1/\varepsilon)/\varepsilon)}$ that outputs a $\{1,2\}$-VCSS H of G such that $w(H) \leq (1 + \varepsilon) \cdot$ OPT.*

References

1. I. Althöfer, G. Das, D. Dobkin, D. Joseph, and J. Soares. On sparse spanners of weighted graphs. *Discrete Comput. Geom.*, 9: 81–100, 1993.
2. S. Arora, M. Grigni, D. Karger, P. Klein, and A. Woloszyn. A polynomial time approximation scheme for weighted planar graph TSP. *SODA* 1998, pp. 33–41.
3. A. Berger, M. Grigni and H. Zhao. A well-connected separator for planar graphs. *Manuscript*, 2004.

4. J. Cheriyan and R. Thurimella. Approximating minimum-cost k-connected spanning subgraphs via matching. *SIAM J. Comput.*, 30(2):528-560, 2000.
5. J. Cheriyan, S. Vempala, and A. Vetta. An approximation algorithm for the minimum-size k-vertex connected subgraph. *SIAM J. Comput.*, 32(4):1050–1055, 2003.
6. A. Czumaj, M. Grigni, P. Sissokho, and H. Zhao. Approximation schemes for minimum 2-edge-connected and biconnected subgraphs in planar graphs. *SODA* 2004, pp. 489–498.
7. A. Czumaj and A. Lingas. On approximability of the minimum-cost k-connected spanning subgraph problem. *SODA* 1999, pp. 281–290.
8. A. Czumaj and A. Lingas. Fast approximation schemes for Euclidean multi-connectivity problems. *ICALP* 2000, pp. 856–868.
9. A. Czumaj, A. Lingas, and H. Zhao Polynomial-time approximation schemes for the Euclidean survivable network design problem. *ICALP* 2002, pp. 973–984.
10. L. Fleischer. A 2-approximation for minimum cost $\{0, 1, 2\}$ vertex connectivity. *IPCO* 2001, pp. 115–129.
11. H. N. Gabow. An ear decomposition approach to approximating the smallest 3-edge connected spanning subgraph of a multigraph. *SODA* 2002, pp. 84–93.
12. H. N. Gabow. Better performance bounds for finding the smallest k-edge connected spanning subgraph of a multigraph. *SODA* 2003, pp. 460–469.
13. M. Grötschel, C. L. Monma, and M. Stoer. Design of survivable networks. In M. O. Ball et al., eds., *Handbooks in Operations Research and Management Science*, vol 7: *Network Models*, chapter 10, pp. 617–672. North-Holland, Amsterdam, 1995.
14. K. Jain. A factor 2 approximation algorithm for the generalized Steiner network problem. *Combinatorica*, 21(1):39–60, 2001.
15. R. Jothi, B. Raghavachari, and S. Varadarajan. A $5/4$-approximation algorithm for minimum 2-edge-connectivity. *SODA* 2003, pp. 725–734.
16. S. Khuller. Approximation algorithms for finding highly connected subgraphs. In D. S. Hochbaum, ed., *Approximation Algorithms for $\mathcal{NP}$-Hard Problems*, pp. 236–265, 1996.
17. S. Khuller and U. Vishkin. Biconnectivity approximations and graph carvings. *Journal of the ACM*, 41(2):214–235, March 1994.
18. G. Kortsarz and Z. Nutov. Approximation algorithm for k-node connected subgraphs via critical graphs. *STOC* 2004, pp. 138–145.
19. P. Krysta. Approximating minimum size 1,2-connected networks, *Discrete Appl. Math.*, 125:267–288, 2003.
20. C. Levcopoulos, G. Narasimhan, and M. H. M. Smid. Efficient algorithms for constructing fault-tolerant geometric spanners. *STOC* 1998, pp. 186–195.
21. M. Penn and H. Shasha-Krupnik. Improved approximation algorithms for weighted 2- and 3-vertex connectivity augmentation problems. *J. Algorithms*, 22:187–196, 1997.
22. S. B. Rao and W. D. Smith. Approximating geometrical graphs via "spanners" and "banyans". *STOC* 1998, pp. 540–550.
23. M. Stoer. *Design of Survivable Networks*. *Lect. Notes in Math.* 1531, Springer-Verlag, 1992.
24. S. Vempala and A. Vetta Factor $4/3$ approximations for minimum 2-connected subgraphs. *APPROX* 2000, pp. 262–273.