A PTAS for the Minimum Cost 2-edge-Connected Spanning Subgraph in Planar Graphs

Michelangelo Grigni[*] Papa Sissokho[†]

July 12, 2002

Abstract

Suppose we are given a planar graph G with edge costs and we want to find a 2-edge-connected spanning subgraph of minimum cost. We present a polynomial time approximation scheme (PTAS) for this problem when the costs are uniform, or when the total cost of G is within a constant factor of the optimal.

1 Introduction

For an integer k, a graph G is *k-edge-connected* (k-EC) if the deletion of any $k-1$ of its edges leaves it connected. Similarly, G is *k-vertex-connected* (k-VC) if it has at least $k+1$ vertices and the deletion of any $k-1$ of its vertices leaves it connected. A *spanning* subgraph $H \subseteq G$ is a subgraph such that $V(H) = V(G)$. Suppose each edge e of G has a non-negative cost c_e; in case these costs are all equal, we say the costs are uniform. For a subgraph $H \subseteq G$, its cost $c(H)$ is the sum of the costs of its edges. A k-EC spanning subgraph of G is called a *k-ECSS*, and the problem of finding a minimum cost k-ECSS is the *k-ECSS problem*. We define *k-VCSS* similarly.

Both the k-ECSS and k-VCSS problems are NP-hard for $k \geq 2$ [8]. There are several constant factor approximation algorithms for these problems [3, 4, 12], where the constant depends on k. We would prefer a *polynomial time approximation scheme* (PTAS): an algorithm taking an instance G and a positive ε, returning a solution with cost at most $(1+\varepsilon)$ times optimal, and running in time that is polynomial for each fixed ε. However, even for $k = 2$ these problems are MaxSNP-hard [5, 7], so there is no PTAS unless P=NP.

[*]This work was supported by NSF Grant number CCR-9820931. Department of Mathematics and Computer Science, Emory University, Atlanta GA 30322, USA. Email: mic@mathcs.emory.edu.

[†]Same support and address. Email: psissok@emory.edu.

However, we may still find a PTAS for special classes of graphs. For example, there is a PTAS for the k-VCSS problem in complete Euclidean graphs in $\mathbf{R}^d$ [6] (fixed d). In this paper we present a PTAS for the 2-ECSS problem in planar graphs with uniform edge costs, or more generally when $c(G)/\mathrm{OPT}(G)$ (the ratio of the total cost of G to the optimal solution cost) is bounded.

Our general approach resembles the approximation schemes for metric-TSP in planar graphs [2, 9]. We use a separator theorem, hierarchical decomposition, and dynamic programming. Our separator finds some low-cost cycles in a planar graph so that after contracting those cycles (and committing their edges to our approximate solution), the remaining graph has a logarithmic size vertex separator. Using this, we recursively divide the input graph G into pieces, forming a decomposition tree $\mathcal{T}$ of logarithmic depth. Each piece has a logarithmic number of "portal" vertices connecting it to the rest of G. For each piece, we enumerate all the different ways that some subgraph of G may influence the connectivity constraints within this piece. We call these the *external types* of the piece, and we show that the number of such types is a simple exponential in the number of portals. For each piece in $\mathcal{T}$ and for each external type, we must find a near minimum cost subgraph H of the piece, so that H together with the external type can meet the 2-EC constraints. We solve these problems by dynamic programming, working up $\mathcal{T}$ from the leaves to the root G.

Our main technical contributions are the modified separator theorem, the use of types in our dynamic program, and the enumeration of types. We will prove:

Theorem 1 *Let $\varepsilon > 0$, let G be a 2-EC planar graph with non-negative edge costs, and let $\mathrm{OPT}(G)$ be the minimum cost of a 2-ECSS in G. There is an algorithm taking inputs G and ε, running in time $n^{O(c(G)/(\mathrm{OPT}(G)\cdot\varepsilon))}$, and producing a 2-ECSS H in G such that $c(H) \leq (1+\varepsilon)\cdot\mathrm{OPT}(G)$.*

The claimed algorithm is a PTAS for the 2-ECSS problem whenever $c(G)/\mathrm{OPT}(G)$ is bounded, and in particular when the edge costs are uniform. After proving our main result, we also remark on some related problems in Section 5.

2 Cuts and Types

In this paper graphs are undirected, without self-loops but possibly with parallel edges and nonnegative edge costs; a subgraph or minor inherits its edge costs from its parent graph. Suppose $G = (V, E)$ is a graph and S_1, S_2 are disjoint subsets of its vertex set $V = V(G)$. S_1 and S_2 are *separated* if there is no path in G from a vertex of S_1 to a vertex of S_2. An edge set $F \subseteq E$ *separates* S_1 and S_2 if they are separated in $G - F$; we also say that F is an *edge cut* for S_1 and S_2. Similarly, a *vertex cut* for S_1 and S_2 is some $U \subseteq V - (S_1 \cup S_2)$ such that S_1 and S_2 are separated in $G - U$. We let $\mathrm{Cut}^e_G(S_1, S_2)$ denote a minimum size edge cut, and $\mathrm{C}^e_G(S_1, S_2) = |\mathrm{Cut}^e_G(S_1, S_2)|$ is the *edge capacity* between S_1 and S_2. We define $\mathrm{Cut}^v_G(S_1, S_2)$ and $\mathrm{C}^v_G(S_1, S_2)$ similarly. We may efficiently compute such min cuts using max-flow. For $P \subseteq V$, we say that a (vertex or edge) cut *crosses* P if it separates some S_1 and S_2 such that $S_1 \cup S_2 = P$.

A *cycle* has no repeated vertex, but it may consist of two vertices joined by two parallel edges. For $e \in E$, G/e denotes the graph obtained by contracting e (that is, identifying its endpoints). If C is a cycle of G, G/C is the graph obtained by contracting the edges of C. After contraction we discard self-loops, but we retain parallel edges. More precisely, suppose we have several parallel edges between two vertices: when solving k-ECSS we may discard all but the k least cost edges, and when solving k-VCSS we may discard all but the one least cost edge. A *minor* of graph G is a graph obtained from G through a series of such edge contractions and edge/vertex deletions.

Definition 2 *A* bipartition *of a set P is a pair of nonempty subsets $\{S_1, S_2\}$ such that $S_1 \cup S_2 = P$ and $S_1 \cap S_2 = \emptyset$.*

Suppose G is a graph, $P \subseteq V(G)$, and k is a positive integer. The $(\mathrm{k}{-}EC, P)$-type *of G is a table t indexed by bipartitions $\{S_1, S_2\}$ of P, and holding the values $t(S_1, S_2) = \min(k, \mathrm{Cut}^e_G(S_1, S_2))$.*

Suppose t_1 and t_2 are $(k{-}EC, P)$-types of two graphs sharing the vertex subset P; they are compatible *iff $t_1(S_1, S_2) + t_2(S_1, S_2) \geq k$ for all $\{S_1, S_2\}$.*

Intuitively, the $(\mathrm{k}{-}EC, P)$-type describes how P is crossed by edge cuts using less than k edges. We are usually only interested in the type when G is *$(k{-}EC, P)$-safe*, meaning that all edge cuts of G not crossing P use at least k edges. The relevance of such types to the k-ECSS problem follows from this simple claim:

Claim 3 *Suppose H_1 and H_2 are graphs with disjoint edge sets, and $V(H_1) \cap V(H_2) = P$. Then $H_1 \cup H_2$ is k-EC iff:*

1. *H_1 is $(k{-}EC, P)$-safe,*
2. *H_2 is $(k{-}EC, P)$-safe, and*
3. *the $(k{-}EC, P)$-types of H_1 and H_2 are compatible.*

We may abbreviate the third condition above by saying that H_1 and H_2 (or H_1 and the type of H_2, or vice versa) are *compatible*.

Suppose G_1 and G_2 are edge disjoint graphs with $V(G_1) \cap V(G_2) = P$. To solve the k-ECSS problem in $G_1 \cup G_2$, it suffices to do the following:

1. For each possible type t_1 of a subgraph of G_1, find a min-cost $(\mathrm{k}{-}EC, P)$-safe spanning subgraph H_2 of G_2 compatible with t_1.
2. For each possible type t_2 of a subgraph of G_2, find a min-cost $(\mathrm{k}{-}EC, P)$-safe spanning subgraph H_1 of G_1 compatible with t_2.

3. Consider all pairs of an H_1 from Step 1 and an H_2 from Step 2. Return the min-cost compatible pair.

We will use a similar approach in our algorithm; in particular we need a polynomial bound on the number of distinct subgraph types considered.

We may succinctly represent the (k$-EC, P$)-type of G by a smaller graph $t(G)$ which contains P and has the same type. In particular a minor of G contains P as long as we have neither deleted a vertex of P, nor contracted two vertices of P together.

In the special case of $k = 2$, we construct such a $t(G)$ from G by repeatedly applying the following rules, until none apply:

1. If a cycle C has a chord (an edge $e \notin E(C)$ connecting two vertices of C), delete the chord.
2. If a cycle C has at most one vertex in P, contract C to a point.
3. If a vertex $v \notin P$ has degree 2, contract it with a neighbor.

The correctness of the above follows from observing that all 0-edge cuts and 1-edge cuts of P are invariant under the above rules. In our application we need to consider the situation where G is embedded in a disk with the vertices of P on the boundary. In the next two lemmas we bound the size of $t(G)$, and we also bound the total number of possible $(2-EC, P)$-types induced by subgraphs of G.

Lemma 4 *Suppose G is a $(2-EC, P)$-safe planar graph embedded in the disk, with the vertices of P on the disk boundary. Then $t(G)$ is a planar graph embedded in the same way, with $O(|P|)$ vertices.*

Proof: By considering the three rules used to form $t(G)$, we see that it is also a planar $(2-EC, P)$-safe graph embedded in the disk with P on the boundary. Every internal face f of $t(G)$ has at least two portals. If f has exactly two portals, we draw an arc e_f inside f between those two portals. If f has $d \geq 3$ portals, we draw a cycle of d arcs within f connecting the portals. These arcs form an outerplanar graph A on the portals.

We claim A has no parallel edges. Suppose instead that two portals $p, q \in P$ are connected by two parallel arcs a_1 and a_2, from faces f_1 and f_2. Since all faces must involve at least two portals, we can choose a_1 and a_2 consecutive at p, so that f_1 and f_2 share at least one edge. Now consider the part of $t(G)$ drawn between a_1 and a_2: it has no cycles (by rule 2) and is connected, so it is a tree. Because $t(G)$ is $(2-EC, P)$-safe, it is a path from p to q. By rule 3 it must be an edge directly between p and q. But then it is a chord between f_1 and f_2, so it should have been deleted by rule 1.

Therefore A is a simple outerplanar graph on vertex set P, so it has less than $2|P|$ arcs. Further if we add arcs from the outer faces of $t(G)$ (those faces bounded by a segment of the boundary), we still have at most three parallel arcs per pair of portals, therefore at most $6|P|$ arcs. For each $p \in P$, its degree in $t(G)$ is at most the number of adjacent arcs; therefore the sum of the degree of p in $t(G)$, over all $p \in P$, is at most $\ell = 12|P|$.

Now if erase each portal and an infinitesimal neighborhood around it, the graph $t(G)$ is transformed into a forest (by rule 2) with ℓ leaves, and all internal vertices of degree at least 3 (by rule 3). Then $t(G)$ has less than ℓ vertices not in P, or in other words $t(G)$ has less than $13|P|$ vertices overall. □

Lemma 5 *With G and P embedded as in the previous lemma, the number of distinct $(2{-}EC, P)$-types defined by subgraphs of G is $2^{O(|P|)}$.*

Proof: Let H be a subgraph of G containing P. By trimming H, we can make it $(2{-}EC, P)$-safe without changing its $(2{-}EC, P)$-type. In the previous proof we saw that $t(H)$ can be described by a planar forest T with at most $12|P|$ leaves, internal vertices of degree at least three, and each leaf labeled by some $p \in P$, where the labels for a given p are on consecutive leaves. By standard tree counting techniques, there are $2^{O(|P|)}$ such graphs $t(H)$, and therefore at most that many distinct $(2{-}EC, P)$-types. □

3 Planar 2-EC Separators

We want to approximately solve the 2-ECSS problem in a planar graph G, embedded on a sphere. If we can find a low-cost simple cycle C in G, then we may divide our problem into subproblems by contracting C. This follows from two observations:

Fact 6 *For any subgraph H of G containing C, H is a 2-ECSS in G iff H/C is a 2-ECSS in G/C. (This does not use planarity.)*

Fact 7 *When we contract C, the sphere pinches into two spheres kissing at the new contracted vertex (a cut-point). Therefore the 2-ECSS problem in G/C is equivalent to two disjoint 2-ECSS problems, one on each sphere.*

Therefore to approximately solve the 2-ECSS problem in G, we may first contract C and approximately solve the two independent 2-ECSS subproblems. Then we lift the edges of those two solutions back to G and add the edges of C, to get a 2-ECSS in G. The additive error of this solution (the difference between its cost and the optimal cost) is at most the sum of the errors on the two subproblems plus $c(C)$.

However, we will not always be lucky enough to find a light cycle which does a good enough job of separating G, therefore we consider a more general kind of separator combining cycles with a *Jordan cut*: a Jordan cut of G is a closed Jordan curve in the embedding of G that does not cross (intersect the interior of) any edge. Given a Jordan cut J, every edge is either in the interior or the exterior of J, but those vertices and faces intersected by J are not counted in either the interior or the exterior of J.

The following theorem is a modification of Miller's planar separator theorem, as already used for the planar TSP [2, 9, 11]. Our main technical change is to use up to three "caps" of Miller's cycle tree (the root cap at at most two leaf caps) as contractible cycles:

Theorem 8 *Let G be a planar graph with n vertices, with non-negative weights on its vertices and faces, and non-negative costs on its edges. Let W be the total weight and let M be the total cost. Given parameter $k \geq 1$, in $O(n \log n)$ time we may find a subgraph F of G and a Jordan cut J (as described below) so that:*

1. *F is the union of at most three vertex-disjoint simple cycles. Their total edge cost $c(F)$ is $O(M/k)$. The cycles are non-nesting, meaning we may choose an embedding of G so that their interiors are disjoint.*
2. *The interior of each C_i has weight at most $(2/3)W$.*
3. *Let G' be the embedded graph that results after we "pinch off" the interiors of the cycles (G' is G/F minus the interior parts from each cycle). Each new contracted vertex has weight 0.*
4. *J is a Jordan cut of G' passing through the new contracted vertices from F, and the interior and exterior of J each have weight at most $(2/3)W$.*
5. *Q, the set of vertices of G' on J, has size at most k.*

When we apply the above theorem, we say Q is the set of new[1] *portal* vertices introduced by the separator. The original graph G has been divided into at most five parts of weight at most $(2/3)W$: the (up to three) pinched cycle interiors, the interior of J, and the exterior of J. We let G_1 denote Q together with the subgraph of G' interior to J, and we let G_2 denote Q together with the subgraph of G' exterior to J. In this way $G_1 \cup G_2 = G'$, $E(G_1) \cap E(G_2) = \emptyset$, and $V(G_1) \cap V(G_2) = Q$. In the inherited embedding of G_1 (or G_2) all vertices of Q appear on a single new face which we call a *portal face*; the old faces that intersected J are gone. When solving 2-ECSS in G, we will see that the "pinched off" cycle interiors become independent 2-ECSS problems, but the subproblems in G_1 and G_2 are dependent because they share Q.

4 The Algorithm

We are given as input an embedded planar 2-EC graph G_0 with n vertices, non-negative edge costs, and a parameter $\varepsilon > 0$. We assign weight 1 to each vertex and weight 0 to each face. By existing approximation algorithms we estimate $\mathrm{OPT}(G_0)$, the minimum cost of a 2-ECSS, within a constant factor. We fix an integer $k = \Theta((\gamma/\varepsilon) \log n)$, where $\gamma = c(G_0)/\mathrm{OPT}(G_0))$.

[1]We are reserving "P" to denote all portals in a graph, new or old.

We build a rooted decomposition tree $\mathcal{T}$ from G_0 as follows. Each node of $\mathcal{T}$ stores an embedded planar graph G, which has edge costs, vertex/face weights, and some distinguished subset P of "portal" vertices. The root of T stores G_0 itself, with no portals. Each node of $\mathcal{T}$ has at most five children, defined inductively as follows.

Let G be the graph stored at a node of $\mathcal{T}$, and let W be its total vertex/face weight. If W is $O(k^2)$, then this node is a leaf of $\mathcal{T}$. Otherwise, apply Theorem 8 to partition G into at most five pieces (up to three pinched cycle interiors and G_1, G_2), each of total weight at most $(2/3)W$. Uncontracted portal vertices from G remain as portals in each piece where they appear; the graphs G_1 and G_2 each get at most k new portals, the set Q. In G_1 and G_2, we assign a weight of $W/(12k)$ to each new portal, and weight $W/12$ to the new portal face. With these new weights, G_1 and G_2 still have total weight at most $(5/6)W$. By the separator properties, we see that each G in $\mathcal{T}$ is $(2{-}EC, P)$-safe, and that the tree $\mathcal{T}$ has depth $O(\log n)$ and size $O(n \log n)$.

By our construction P is the set of portals in G, which have been introduced by some Jordan cut but not yet cut off by a cycle contraction or another Jordan cut. It follows from our portal/hole weighting scheme [2, 9] that $|P|$ is $O(k)$, and G has $O(1)$ portal faces. Each portal face contains a hole made by a Jordan cut at some ancestor of G in $\mathcal{T}$. Note that a Jordan cut might cut (simply) across an existing portal face, in which case some old portals may appear on the new portal face, but this still counts as a single portal face. Or in terms of an embedding on a sphere with holes, all old holes crossed by the Jordan cut disappear, with segments of their boundaries incorporated into the one new hole boundary.

G is connected via P to the rest of G_0 (really a pinched and contracted version of G_0) which can be embedded as disjoint pieces, one in each portal face of G. Therefore the $(2{-}EC, P)$-type imposed on P by the rest of G_0 decomposes into independent types, one in each portal face of G. By applying Lemma 5 to each portal face, we may bound and enumerate the $2^{O(|P|)} = n^{O(\gamma/\varepsilon)}$ different $(2{-}EC, P)$-types that may may be imposed on P by the rest of G_0. Call this list the list of *external types* for G. It is more efficient, although not essential for our puposes, if we represent each external type t of G as a planar graph of size $O(|P|)$ (see Lemma 4) embedded in the portal faces of G. In particular at the root of $\mathcal{T}$ the input graph G_0 has no portals, and therefore it has the "empty" external type t_0.

Having computed $\mathcal{T}$ and these external type lists, we may now define a set of subproblems that we want to approximately solve:

Definition 9 *For G in $\mathcal{T}$ (with portal set P) and an external type t for G, the* subproblem (G, t) *is this: find a min-cost $(2{-}EC, P)$-safe spanning subgraph H of G which is compatible with t, or else declare that (G, t) is infeasible (no such H).*

Checking feasibility is simple: just check whether t is compatible with G itself.

The total number of subproblems (over all choices of G and t) is $n^{O(\gamma/\varepsilon)}$, and the subproblem (G_0, t_0) is our original 2-ECSS problem. We will approximately solve each subproblem starting at the leaves of $\mathcal{T}$ and finishing at the root. We use dynamic programming, storing our solutions to avoid recomputation.

In the base case, G is a leaf of $\mathcal{T}$ and has size $N = O(k^2)$. Then we apply an enumerative method based on Lipton-Tarjan separtors [10] to exactly solve such subproblems in $2^{O(\sqrt{N})} = n^{O(\gamma/\varepsilon)}$ time (this may be regarded as a continuation of our method, using Jordan cuts without cycle contractions). We omit details.

Otherwise G is not a leaf, and has up to five children in $\mathcal{T}$ as found by Theorem 8. We have a specific external type t, which decomposes into an independent external type in each portal face of G.

For each child G_C of G which was pinched off in the interior of some cycle C, let t_C be the subtype of t induced by the portal faces inside C, and lookup our solution H_C to the subproblem (G_C, t_C). We lift the edges of H_C and the edges of C to be part of our approximate solution H for the (G, t) subproblem.

Now we must consider the two remaining children G_1 and G_2. As in Theorem 8, let G' be what is left of G after we contract the (up to three) cycles and pinch off their interiors; so $G_1 \cup G_2 = G'$. Let t' denote the external type induced by t in the portal faces of G'. Not knowing the optimal choice of external types t_1 and t_2 for G_1 and G_2, we try them all. That is, for every pair (t_1, t_2) where subproblems (G_1, t_1) and (G_2, t_2) were found feasible, we lookup their solutions H_1 and H_2 and check whether $H' = H_1 \cup H_2$ is compatible with t'. We take the cheapest compatible H' found, and lift its edges back to G. These edges of H', together with the C and H_C edges mentioned earlier, comprise our approximate solution H for the (G, t)-subproblem.

Although G' is not actually associated with a node of $\mathcal{T}$, note that we can still speak sensibly of the (G', t') subproblem as defined above. In fact it would be a simple matter to reformulate $\mathcal{T}$ as a binary tree including G': at each internal node of $\mathcal{T}$ we would either pinch one cycle, or apply a Jordan cut.

4.1 Analysis

Our algorithm solves $n^{O(\gamma/\varepsilon)}$ subproblems, each in $n^{O(\gamma/\varepsilon)}$ time, so the total running time is $n^{O(\gamma/\varepsilon)}$.

Consider a feasible subproblem (G, t). By planarity, t decomposes into independent types in each portal face, and these faces cannot cross a cycle; therefore each cycle-pinched subproblem (G_C, t_C) and the remaining subproblem (G', t') are all feasible. Taking the external type on G_1 induced by $G_2 \cup t'$ as t_1, we see that (G_1, t_1) is feasible. Supposing (by induction) that our algorithm found some solution H_1 for (G_1, t_1), then $H_1 \cup t$ induces an external type t_2 on G_2 such that (G_2, t_2) is also feasible. Therefore by induction up $\mathcal{T}$, our algorithm finds some solution for each feasible (G, t)-subproblem.

Now suppose H is the solution our algorithm finds for a feasible subproblem (G, t). Define the *error* on (G, t) as the difference between the found cost

$c(H)$ and the minimum possible cost. For each pinched cycle C (up to three), by Facts 6 and 7 subproblem (G, t) will inherit the error of (G_C, t_C) plus an additional additive errror of at most $c(C)$.

After pinching cycles, the remaining error of (G, t) is that from (G', t'). Recall $G' = G_1 \cup G_2$; let H^* be the unknown optimal solution for (G', t'). Let t_1^* denote the external type of G_1 induced by $(H^* \cap G_2) \cup t'$, and similarly let t_2^* denote the external type of G_2 induced by $(H^* \cap G_1) \cup t'$. Then (G_i, t_i^*) has the optimal solution $H^* \cap G_i$ (for $i = 1, 2$), and (t_1^*, t_2^*) is a compatible type pair considered by our algorithm; if we solved these two subproblems optimally, our solution cost would be $c(H^*)$. Therefore our error on (G', t') is at most the sum of our errors on (G_1, t_1^*) and (G_2, t_2^*), even though we might not actually find our best solution H' using this pair. Therefore error terms simply add at a Jordan cut.

Therefore the total error of our root problem (G_0, t_0) is at most the sum of $c(C)$ over all cycles contracted in $\mathcal{T}$. We see that for any level of $\mathcal{T}$, the total edge cost of that level is at most $c(G_0)$, therefore the total edge cost of all cycles contracted on that level is $O(c(G_0)/k)$. Summing over all $O(\log n)$ levels of $\mathcal{T}$, the total error from all levels of $\mathcal{T}$ is $O((c(G_0)/k) \log n)$. By an appropriate choice of the leading constant defining k, this is at most $\varepsilon \cdot \mathrm{OPT}(G_0)$. Therefore our final solution has cost at most $(1 + \varepsilon)\mathrm{OPT}(G_0)$, proving Theorem 1.

5 Concluding remarks

A deficiency of our PTAS is its dependence on $\gamma = c(G)/\mathrm{OPT}(G)$, where $\mathrm{OPT}(G)$ is the minimum cost of a 2-ECSS in G. We know of no hardness result justifying this dependency, and indeed a very similar dependency in the context of the TSP was eliminated using a known “spanner” construction [1, 2], which safely prunes some heavy edges out of G. We conjecture that a similar approach works for the 2-ECSS problem.

Definition 10 *Given $s \geq 1$ and a 2-EC graph G with non-negative edge costs, a* $(2{-}EC, s)$-spanner *of G is a 2-ECSS G' in G such that $\mathrm{OPT}(G') \leq s \cdot \mathrm{OPT}(G)$. The* cost ratio *of such a spanner is $c(G')/\mathrm{OPT}(G')$.*

Of course such spanners exist even with $s = 1$ and cost ratio one: just let G' be the optimal 2-ECSS in G. In polynomial time, known constant factor approximation algorithms find a spanner with bounded s and cost ratio. But what we need is an efficient algorithm achieving $s = 1 + \varepsilon$ and bounded cost ratio (depending on ε), whenever G is planar:

Conjecture 11 *There is a polynomial time algorithm taking inputs a planar G and $\varepsilon > 0$, and returning a $(2{-}EC, 1 + \varepsilon)$-spanner of cost ratio at most $f(\varepsilon)$. Here $f(\varepsilon)$ is some finite function depending only on ε.*

If the conjectured algorithm exists (or slightly weaker: an algorithm which is polynomial time for each fixed $\varepsilon > 0$), then we could use it as a preprocessor for our algorithm, reducing our effective γ to (say) $f(\varepsilon/2)$. The resulting PTAS for the 2-ECSS problem would then have running time $n^{O(f(\varepsilon/2)/\varepsilon)}$, for all planar graphs G.

Another obvious direction of research is to extend our method to similar problems, such as the 2-VCSS or 3-ECSS problems in planar graphs, or the 2-ECSS problem is "slightly non-planar" graphs (such as graphs with a fixed forbidden minor). We remark on some of these problems below; in all cases, progress seems to depend on the development of appropriate separator theorems.

Planar 2-VCSS: The notion of a *$(k{-}EC, P)$-type* easily generalizes to *$(k{-}VC, P)$-type*, but one must also take care to generalize Claim 3. We can generalize Lemma 5, and low-cost cycles are again useful. But it is no longer safe to contract cycles; instead we may simply commit those edges to the solution and then reset their costs to zero in the two subproblems (interior and exterior). But this requires duplicating the cycle vertices, which implies some reconsideration of the separator theorem.

Planar 3-ECSS: Here we need to generalize Lemma 5; let us remark that counting planar Gomory-Hu trees is not sufficient. Also cycle contraction no longer works here, because the cycle vertices are not 3-edge-connected. However we can modify our separator theorem to replace cycles with *bicycles*. A bicycle is two nested cycles joined by edges. The endpoints of the connecting edges are 3-edge-connected, and we may safely contract them to a cut-point. The bicycle edges surviving this contraction are committed to the solution by resetting their costs to zero. Again this requires some reconsideration of the separator theorem, which we avoid here.

Forbidden minor 2-ECSS: Here we have two difficulties: first, the usual separator theorems do not produce cycles. Also the appropriate generalization of Lemma 5 may require a careful application of the Robertson-Seymour theory, which appears difficult. For the special case of bounded-genus graphs, both of these issues look much more tractable.

Acknowledgment: We thank Piotr Krysta for suggesting this problem.

References

[1] I. Althöfer, G. Das, D. P. Dobkin, D. Joseph, and J. Soares. On sparse spanners of weighted graphs. *Discrete Comput. Geom.*, 9(1):81–100, 1993. An early version appeared in SWAT'90, LNCS V. 447.

[2] S. Arora, M. Grigni, D. Karger, P. Klein, and A. Woloszyn. A polynomial time approximation scheme for weighted planar graph TSP. *Proc. ACM-SIAM 9th Annual Symp. on Discrete Algorithms*, pages 33–41, 1998.

[3] J. Cheriyan, A. Sebö, and Z. Szigeti. An improved Approximation Algorithm for Minimum Size 2-Edge Connected Spanning Subgraphs. *Proc. 6th IPCO,LNCS*, 1412:126–136, 1998.

[4] J. Cheriyan, S. Vampala, and A. Vetta. Approximation Algorithms for Minimum-Cost k-vertex Connected Subgraphs. *34th ACM Symposium on Theory of Computing*, To appear, 2002.

[5] B. Csaba, M. Karpinski, and P. Krysta. Approximability of Dense Sparse Instances of Minimum 2-Connectivity, TSP and Path Problems. *Proc. ACM-SIAM 13th Annual Symp. on Discrete Algorithms*, pages 74–83, 2002.

[6] A. Czumaj and A. Lingas. On Approximability of the Minimum Cost Spanning Subgraph Problem. *Proc. ACM-SIAM 10th Annual Symp. on Discrete Algorithms*, pages 281–290, 1999.

[7] C. G. Fernandes. A Better Approximation Ratio for the Minimum Size k-edge-Connected Spanning Subgraph Problem. *Journal of Algorithms*, 28:105–124, 1988.

[8] M. R. Gary and D. S. Johnson. Computer and Intractability: A guide to the Theory of NP-completeness. *Freeman*, 1979.

[9] M. Grigni, E. Koutsoupias, and C. Papadimitriou. An approximation scheme for planar graph TSP. *Proc. IEEE Symposium on Foundations of computer Science*, pages 640–645, 1995.

[10] R. Lipton and R. Tarjan. Applications of a planar separator theorem. *SIAM Journal on Computing*, 9(3):615–627, 1980.

[11] G. L. Miller. Finding small simple cycle separators for 2-connected planar graphs. *Journal of Computer and System Sciences*, 32:265–279, 1986.

[12] S. Vampala and A. Vetta. Factor 4/3-Approximations for Minimum 2-Connected Subgraphs. *Proc. 3rd Workshop APPROX, LNCS*, 1913:262–273, 2000.