Two-connected Augmentation Problems in Planar Graphs

J. Scott Provan [1] Roger C. Burk [2]

UNC/OR TR94-12

October 1996

Abstract

Given an weighted undirected graph G and a subgraph S of G, we consider the problem of adding a minimum-weight set of edges of G to S so that the resulting subgraph satisfies specified (edge or vertex) connectivity requirements between pairs of nodes of S. This has important applications in upgrading telecommunications networks to be invulnerable to link or node failures. We give a polynomial algorithm for this problem when S is connected, nodes are required to be at most 2-connected, and G is planar. Applications to network design and multicommodity cut problems are also discussed.

Key words: planar graphs; two-connectedness; multicommodity cuts; Steiner trees

Subject classifications: 05C85; 68Q25; 68R10; 90B12

[1]Department of Operations Research, University of North Carolina, Chapel Hill, NC 27599-3180. The work of this author was supported in part by NSF grant No. CCR-9200572

[2]Science Applications International Corporation, 4501 Daly Dr., Suite 400, Chantilly, VA 22021-3707

1 Introduction

The *Connectivity Augmentation Problem* has as input tuple $(G, \boldsymbol{w}, S, \boldsymbol{r})$, where $G = (V, E)$ is an undirected graph, $\boldsymbol{w} = (w_e : e \in E)$ is a set of edge weights, $S = (V_S, E_S)$ is a subgraph of G, and $\boldsymbol{r} = (r_{ij} : i, j \in V_S)$ is a set of nonnegative integers representing *connectivity requirements* on the vertices of S. The *edge (vertex)-connectivity* between i and j is defined to be the minimum number of edges (vertices distinct from i and j) whose removal disconnects i and j. The objective is to find the minimum weight subgraph $F \subseteq E \setminus E_S$ for which the connectivity in $F \cup E_S$ between every pair i, j of vertices in V_S is at least r_{ij}. We will refer to the edge- and vertex-connectivity versions of these problems as ECAP and VCAP, respectively. [1]

Connectivity augmentation problems were first introduced in this general context in [10], and include as a special cases the construction of networks with given connectivities, such as minimum spanning trees, Steiner trees, minimum weight 2-connected networks, and minimum weight networks with low connectivities (see [18] and [17] for extensive surveys of these problems). General connectivity augmentation problems have also been studied extensively: see [9] for an excellent survey. One of the earliest versions was studied in [7] has S a connected subgraph and $r_{ij} = 2$ for all vertices i, j in S. We will refer to these problems as 2EC and 2VC, respectively, and drop the parameter $\boldsymbol{r}$ when referring to instances of these problems.

Both ECAP and VCAP are known to be NP-hard, even when

- G is planar, all weights are 1, S is the entire set of isolated vertices of G, and all connectivity requirements are 2 (essentially equivalent to finding a Hamiltonian circuit in G [14]);
- G is planar, all weights are 1, S is a subset of isolated vertices, and all connectivity requirements are 1 between the vertices of S (the planar Steiner tree problem [13]);
- G is the complete graph, S is a tree, weights are arbitrary, and all connectivity requirements are 2 on the vertices of S [7, 12].

Polynomial algorithms exist for

[1] A frequently used alternative definition for the connectivity between points is the maximum cardinality of a set of edge- or vertex-disjoint paths between the given vertices. For the ECAP version the problem the two definitions are identical, and for the VCAP version the problem can be easily modified to use this definition by bisecting each edge and modifying the edge weights and vertex connectivities appropriately.

- VCAP and ECAP in the case where G is arbitrary, S is the entire set of isolated vertices, weights are arbitrary, and all connectivity requirements on S are 1 (the min spanning tree problem);
- 2EC and 2VC in the case where G is a complete graph and all weights are 1 [7];
- 2VC in the case where G is series-parallel, S is the entire set of isolated vertices, and weights are arbitrary [23];
- VCAP and ECAP in the case where G is planar, S is a set of isolated vertices lying on one face of G, all connectivity requirements on S are 1, and weights are arbitrary [6], and 2EC and 2VC under the same restrictions [22];
- ECAP in the case where G is a complete graph with arbitrarily high edge multiplicity (i.e., any edge is allowed to be used arbitrarily many times), all weights are 1, and connectivity requirements are arbitrary [8].

We now state the main result of this paper.

Main Theorem *The ECAP and VCAP problems can be solved in $O(n^2k^2)$ time for any instance $(G, \boldsymbol{w}, S, \boldsymbol{r})$ with G an n-vertex planar graph, S a k-vertex connected subgraph, and $r_{ij} \leq 2$ for all $i, j \in V_S$.*

The first five sections of the paper are devoted to proving this theorem. Sections 2 and 3 give $O(n^2k^2)$ algorithms for the planar 2EC and 2VC problems, respectively, in the case where S is a tree. Section 4 extends the algorithms for 2EC and 2VC to cover the general planar ECAP and VCAP problems when S is a tree and $r_{ij} \leq 2$; Section 5 completes the proof of the Main Theorem by extending the algorithms to apply when S is a general connected set and $r_{ij} \leq 2$. In Section 6 we discuss the relationship of these problems to network design heuristics, and also to a special case of the *multicommodity cut problem* of finding a minimum weight set of edges whose removal disconnects a given set of source-sink vertex pairs in a graph.

2 2EC on trees

We will assume throughout the presentation that edge weights are *positive*. This assumption is made only for convenience, since any instance having nonpositive edge weights can be solved by replacing all nonpositive weights by sufficiently small positive weights. The ECAP or VCAP problem is solved for this case, and then all of the nonpositive edges are added to the solution form an optimal solution for the original ECAP or VCAP problem.

In this section we solve 2EC on planar graphs G when the given subgraph S is a tree. For any subset B of edges, a *bridge* with respect to B is any edge whose removal disconnects some pair of vertices spanned by B.

Lemma 1 *Let $(G, \boldsymbol{w}, S)$ be an instance of 2EC and let $F \subseteq E \setminus E_S$. Then F is an optimal solution to 2EC if and only if F is a minimum weight forest such that no edge of S is a bridge with respect to $F \cup E_S$.*

Proof First, let F be a minimum weight forest such no element of E_S is a bridge with respect to $F \cup E_S$. Then no edge in F can be a bridge with respect to $F \cup E_S$ either, since its removal must leave one component with no edges of S. This can then be removed from the solution, contradicting the edge-minimality of F. It follows that $F \cup E_S$ is 2-connected, and hence F is feasible to 2EC.

Conversely, let F be an optimal solution for 2EC. Since $F \cup E_S$ is 2-edge-connected then it contains no bridges, and hence contains no bridges in S. To show that F is a forest, let Ω be some cycle of F. Removing any edge from Ω will not create any bridges among the edges of S, and hence by the above argument there is a strict subset of $F \cup E_S$ such that no edge of S is a bridge with respect to $F \cup E_S$. Again this contradicts the minimality of F.

Since the sets of minimal solutions to both problems are identical and the solutions to each have the same weight, then the optimal solutions to both problems are also identical. ∎

We now show how to solve 2EC for instance $(G, \boldsymbol{w}, S)$ with G planar and S a tree. To do this we first define the concept of an *interval*. Because G is planar, we can traverse S clockwise by a closed walk $W = \sigma_0, \sigma_1, \ldots, \sigma_m = \sigma_0$ of vertices σ_i, so that no point of S lies immediately to the left of W in the traversal. This means that in the traversal of W each edge of S will appear exactly twice and each vertex of S will appear exactly as many times as its degree, so that $m = 2k - 2$ (see Figure 1a). We will consider the σ_i's as distinct elements, using the notation $\langle \sigma_i \rangle$ to denote the vertex corresponding to σ_i. For any edge e of $G \setminus E_S$ that has an endpoint on S, we will refer to that endpoint as σ_j if σ_j

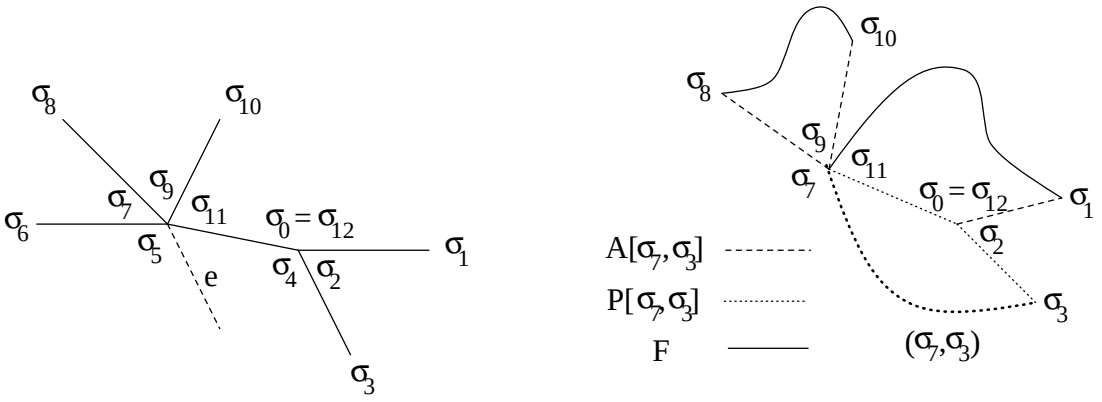

Figure 1: Interval definitions

is the vertex of W passed when e comes into S from the left. In Figure 1a, the edge e has endpoint σ_5. For two vertices σ_i, σ_j on W, the *interval* $[\sigma_i, \sigma_j]$ is defined to be that portion $\sigma_i, \sigma_{i+1}, \ldots, \sigma_j$ of W appearing clockwise from σ_i to σ_j (the indices for σ will always be taken modulo m). For technical convenience we take the interval $[\sigma_i, \sigma_{i+m}]$ to represent the entire walk W; $[\sigma_i, \sigma_i]$ will represent the one point path containing σ_i. Also, for $i \neq j$ we set $(\sigma_i, \sigma_j] = [\sigma_i, \sigma_j] \setminus \{\sigma_i\}$ and $[\sigma_i, \sigma_j) = [\sigma_i, \sigma_j] \setminus \{\sigma_j\}$. The edges traversed by $[\sigma_i, \sigma_j]$ can be partitioned into two sets: one, $P[\sigma_i, \sigma_j]$, being the set of edges in the unique path in S from σ_i to σ_j, and the other, $A[\sigma_i, \sigma_j]$, being the set of distinct edges of $[\sigma_i, \sigma_j]$ that are not in $P[\sigma_i, \sigma_j]$. In other words, $A[\sigma_i, \sigma_j]$ is that portion of S attached to $P[\sigma_i, \sigma_j]$ from the left when traversing $P[\sigma_i, \sigma_j]$ from σ_i to σ_j (see Figure 1b).

Now let $[\sigma_i, \sigma_j]$ be an interval and let F be a set of edges of G. We call F *one-sided* with respect to $[\sigma_i, \sigma_j]$ if it is incident to S only at elements of $[\sigma_i, \sigma_j]$. Denote by F^* the set $F \cup P[\sigma_i, \sigma_j] \cup A[\sigma_i, \sigma_j] \cup \{(\sigma_i, \sigma_j)\}$, and note that this is a planar set of edges whenever F is one-sided. We will assume that the edge (σ_i, σ_j) is positioned so that $A[\sigma_i, \sigma_j]$ lies on the outside of $P[\sigma_i, \sigma_j] \cup \{(\sigma_i, \sigma_j)\}$, and define ∂F^* to be the exterior boundary of F^*. Finally, F is called a $[\sigma_i, \sigma_j]$-*feasible* forest if F is one-sided and $A[\sigma_i, \sigma_j]$ contains no bridges with respect to F^*. The F in Figure 1b, for example, is a $[\sigma_7, \sigma_3]$-feasible forest. The optimal solution to 2EC will then be a minimum weight $[\sigma_0, \sigma_m]$-feasible forest.

The remainder of the section will be devoted to giving functions that will produce minimum weight $[\sigma_i, \sigma_j]$-feasible forests. For $[\sigma_i, \sigma_j]$-feasible forest F define an S-*path* to be a path in F that has none of its interior vertices in S, and an S-*component* to be a maximal subtree of F having no interior (nonpendant) vertices in common with S. We can

now define the following four auxiliary functions, which find various types of $[\sigma_i, \sigma_j]$-feasible forests. Let $[\sigma_i, \sigma_j]$ be an interval, and v a vertex of G. Then define

$\mathbf{S}([\sigma_i, \sigma_j]) =$ minimum weight of a $[\sigma_i, \sigma_j]$-feasible forest;

$\mathbf{B}([\sigma_i, \sigma_j], v) =$ minimum weight of a $[\sigma_i, \sigma_j]$-feasible forest having σ_i, σ_j, and v in the same S-component and such that v is of degree at least 2 or equal to σ_i or σ_j;

$\mathbf{C}([\sigma_i, \sigma_j], v) =$ minimum weight of a $[\sigma_i, \sigma_j]$-feasible forest having σ_i, σ_j, and v in the same S-component;

$\mathbf{D}([\sigma_i, \sigma_j], v) =$ minimum weight of a $[\sigma_i, \sigma_j]$-feasible forest having σ_j and v in the same S-component;

and for vertices v and u define

$\mathbf{d}_S(v, u) =$ length of a shortest S-path from v to u.

(These values are assumed to be ∞ if no such objects exist.) We also use the notation

$[\sigma_i, \sigma_j]_+ = \{[\sigma_r, \sigma_s]$: $\sigma_r, \sigma_s \in [\sigma_i, \sigma_j]$, $[\sigma_r, \sigma_s] \neq [\sigma_i, \sigma_j]$, $\sigma_r \neq \sigma_s$, and $P[\sigma_r, \sigma_s]$ and $P[\sigma_i, \sigma_j]$ have at least one vertex in common$\}$

Proposition 1 *For any interval $[\sigma_i, \sigma_j]$, $\mathbf{S}([\sigma_i, \sigma_j]))$ can be computed using the following set of equations:*

$$\mathbf{S}([\sigma_i, \sigma_j]) = \min_{[\sigma_r, \sigma_s] \in [\sigma_i, \sigma_j]_+} \{\mathbf{S}([\sigma_i, \sigma_r]) + \mathbf{B}([\sigma_r, \sigma_s], \sigma_r) + \mathbf{S}([\sigma_s, \sigma_j])\} \quad (1)$$

$$\mathbf{B}([\sigma_r, \sigma_s], v) = \min_{\sigma_p \in [\sigma_r, \sigma_s)} \{\mathbf{C}([\sigma_r, \sigma_p], v) + \mathbf{D}([\sigma_p, \sigma_s], v)\} \quad (2)$$

$$\mathbf{C}([\sigma_r, \sigma_p], v) = \min_{u \in V} \{\mathbf{d}_S(v, u) + \mathbf{B}([\sigma_r, \sigma_p], u)\} \quad (3)$$

$$\mathbf{D}([\sigma_p, \sigma_s], v) = \min_{\sigma_q \in (\sigma_p, \sigma_s]} \{\mathbf{S}([\sigma_p, \sigma_q]) + \mathbf{C}([\sigma_q, \sigma_s], v)\} \quad (4)$$

with these values being 0 whenever $A[\sigma_\cdot, \sigma_\cdot] = \emptyset$.

Proof The reader should refer to Figure 2 for the proofs of each of the recursive formulae above, as marked.

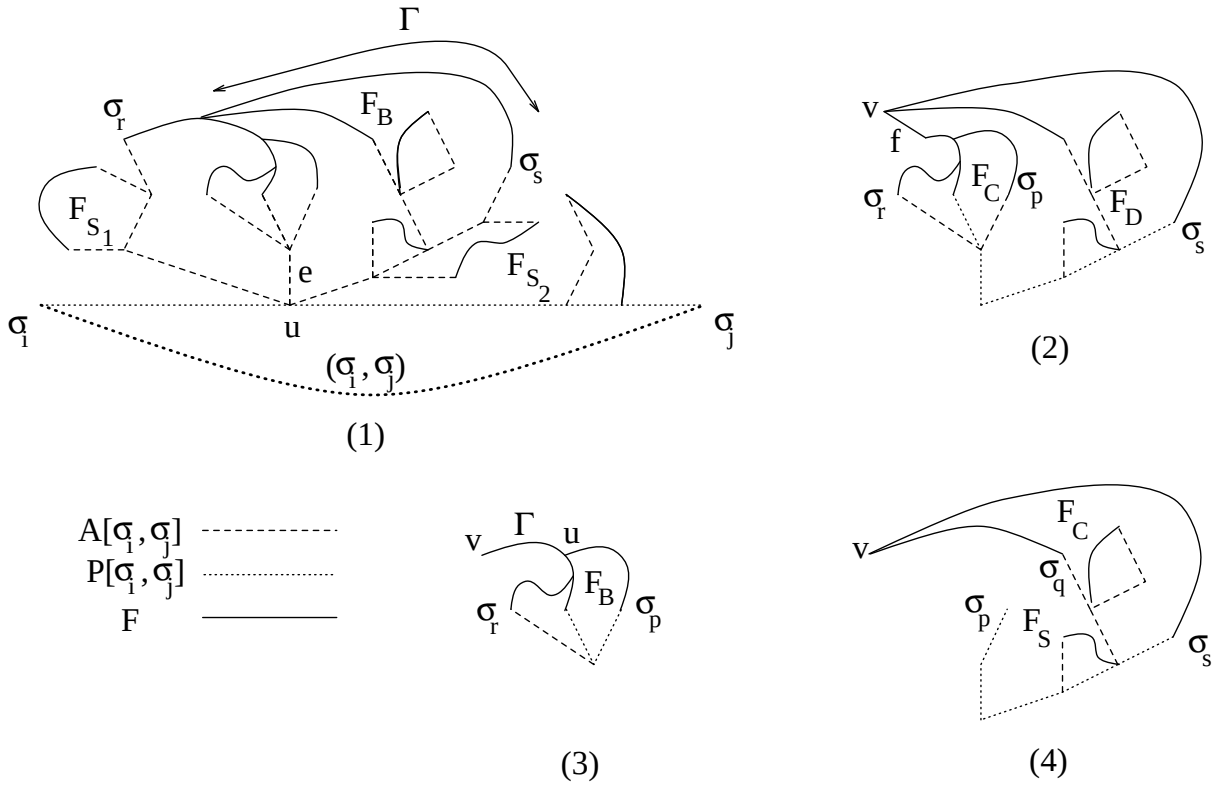

Figure 2: Example of recursive forest decomposition

(1): Let F be a minimum weight $[\sigma_i, \sigma_j]$-feasible forest, and let e be an edge of $A[\sigma_i, \sigma_j]$ touching $P[\sigma_i, \sigma_j]$ at vertex u. Since e is not a bridge, then ∂F^* is made up of tree edges and S-path segments that together either contain or enclose e. This means that one of these S-paths Γ creates a cycle with $[\sigma_i, \sigma_j]$ that also contains or encloses e. Let $\sigma_r \neq \sigma_s$ be the endpoints of Γ, in clockwise order. Then $P[\sigma_r, \sigma_s]$ must also contain the vertex u, and so $P[\sigma_r, \sigma_s]$ and $P[\sigma_i, \sigma_j]$ have at least one vertex in common. Further, $[\sigma_r, \sigma_s]$ cannot be equal to $[\sigma_i, \sigma_j]$, since any cycle in $F \cup E_S$ containing both σ_i and σ_j would contain no edges of $A[\sigma_i, \sigma_j]$, and hence the edge at either end of this cycle could be removed without creating a bridge in S with respect to F^*. This contradicts the fact that F was minimal, and it follows that $[\sigma_r, \sigma_s]$ is in $[\sigma_r, \sigma_s]_+$. Also, by the choice of Γ and the planarity of F^*, there can be no pair of points in different sets of of $[\sigma_i, \sigma_r]$, $[\sigma_r, \sigma_s]$, or $[\sigma_s, \sigma_j]$ that are connected by an S-path of F. We can therefore unambiguously partition the edges of F into subforests F_{S_1}, F_B and F_{S_2}, corresponding to those S-components of F incident to vertices in $[\sigma_i, \sigma_r]$, $[\sigma_r, \sigma_s]$, and $[\sigma_s, \sigma_j]$, respectively. These must in fact be $[\sigma_i, \sigma_r]$-feasible, $[\sigma_r, \sigma_s]$-feasible, and $[\sigma_s, \sigma_j]$-feasible forests, since they are clearly one-sided and induce no bridges in S since F does not. Finally, F_B has σ_r, σ_s, and $v = \sigma_r$ in the same S-component. Thus F has weight at least $\mathbf{S}([\sigma_i, \sigma_r]) + \mathbf{B}([\sigma_r, \sigma_s], \sigma_r) + \mathbf{S}[\sigma_s, \sigma_j])$.

Conversely, let F_{S_1}, F_B and F_{S_2} be minimum-weight $[\sigma_i, \sigma_r]$-feasible, $[\sigma_r, \sigma_s]$-feasible, and $[\sigma_s, \sigma_j]$-feasible forests, respectively, with $P[\sigma_i, \sigma_j]$ and $P[\sigma_r, \sigma_s]$ sharing a common vertex, and let $F = F_{S_1} \cup F_B \cup F_{S_2}$. Since S contains no bridges with respect to $F^*_{S_1}$, F^*_B and $F^*_{S_2}$ and each of these in turn contains at least one vertex in common with $P[\sigma_r, \sigma_s]$, then S likewise contains no bridges with respect to F^*. Thus F is a $[\sigma_i, \sigma_j]$-feasible set of weight $\mathbf{S}([\sigma_i, \sigma_r]) + \mathbf{B}([\sigma_r, \sigma_s], \sigma_r) + \mathbf{S}[\sigma_s, \sigma_j])$, and so Equation (1) follows.

(2): Let F_B be a minimum-weight $[\sigma_r, \sigma_s]$-feasible forest with v, σ_r, and σ_s in the same S-component, $r \neq s$, and v of degree at least 2 or equal to σ_r or σ_s. Remove the edge f adjacent to v that disconnects the counterclockwise-most set of vertices in $[\sigma_r, \sigma_s]$, and let σ_p be the *clockwise*-most vertex in the disconnected S-component. Note that by the conditions on F_B we have $p \neq s$. Then the edges of F_B can be unambiguously partitioned into subforests F_C and F_D corresponding to those S-components incident to vertices in $[\sigma_r, \sigma_p]$ and $[\sigma_p, \sigma_s]$, respectively. Since F_B is a planar forest with v, σ_r, and σ_p in the same S-component, then F_C and F_D can have only the vertex v in common. It follows that F_C must be a $[\sigma_r, \sigma_p]$-feasible forest with v, σ_r and σ_p in the same S-component, and F_D must be a $[\sigma_p, \sigma_s]$-feasible forest with v and σ_s in the same S-component, again since a bridge in S with respect to F^*_C or F^*_D would also be a bridge with respect to F^*. Thus F_B has weight at least $\mathbf{C}([\sigma_r, \sigma_p], v) + \mathbf{D}([\sigma_p, \sigma_s], v)$.

Conversely, for $\sigma_p \in [\sigma_p, \sigma_s)$ let F_C be a $[\sigma_r, \sigma_p]$-feasible forest with v, σ_r and σ_p in the same S-component, and let F_D be a $[\sigma_p, \sigma_s]$-feasible forest with v and σ_s in the same S-component. Now since neither $A[\sigma_r, \sigma_p]$ nor $P[\sigma_r, \sigma_p]$ have bridges with respect to F_C^*, and $A[\sigma_p, \sigma_s]$ has no bridges with respect to F_D^*, then $A[\sigma_r, \sigma_s]$ will have no bridges with respect to $F_C \cup F_D$ (since $A[\sigma_r, \sigma_s] \cap P[\sigma_p, \sigma_s] \subseteq A[\sigma_r, \sigma_s] \cap P[\sigma_r, \sigma_p]$). Thus $F_C \cup F_D$ is a $[\sigma_r, \sigma_s]$-feasible set with v, σ_r, and σ_s in the same S-component, and so has weight at most $\mathbf{C}([\sigma_r, \sigma_p], v) + \mathbf{D}([\sigma_p, \sigma_s], v)$. Equation (2) follows.

(3): Let F_C be the set found in (2), with v, σ_r, and σ_p in the same S-component. Follow the path Γ in F_C from v toward σ_p until the first point at which a vertex u is encountered that is either of degree at least 3 or is in $[\sigma_r, \sigma_p]$, and set $F_B = F_C \setminus \Gamma$. Then Γ is an S-path. Further, $A[\sigma_r, \sigma_p]$ contains no bridges with respect to F_B^* it had none with respect to F_C^*, and so F_B is a $[\sigma_r, \sigma_p]$-feasible set. Finally, since F_C is part of the planar forest F, then u cannot be of degree 1 in F_B unless it is equal to σ_r or σ_p, since otherwise F would cross itself. Thus F_B as weight at most $B([\sigma_r, \sigma_p], u)$, and so F_C will have weight at least $\mathbf{d}_S(v, u) + \mathbf{B}([\sigma_r, \sigma_p], u)$.

Conversely, for $u \in V$ let Γ be a shortest S-path from v to u, and let F_B be a minimum weight $[\sigma_r, \sigma_p]$-forest with u, σ_r, and σ_p in the same S-component and u of degree at least 2 or equal to σ_r or σ_p. Clearly $F_C = \Gamma \cup F_B$ is a $[\sigma_r, \sigma_p]$-feasible set with v, σ_r, and σ_p in the same S-component, and has weight $\mathbf{d}_S(v, u) + \mathbf{B}([\sigma_r, \sigma_p], u)$. Equation (3) follows.

(4): Let F_D be the minimum-weight $[\sigma_p, \sigma_s]$-feasible forest found in (2), with v and σ_s lying in the same S-component. Let σ_q be the counterclockwise-most element of $[\sigma_p, \sigma_s]$ lying in the same S-component as v and σ_s. From the construction of F_D in (2), it follows that $q \neq p$. Then F_D is partitioned into subforests F_S of those S-components incident to vertices in $[\sigma_p, \sigma_q]$, and subforest F_C comprised of the remaining edges of F_D. Again, the planarity of F implies that these forests are disjoint, and hence F_S must be a $[\sigma_p, \sigma_q]$-feasible forest, and F_C must be $[\sigma_q, \sigma_s]$-feasible forest with v, σ_q, and σ_s in the same S-component. Thus F_D has weight at least $\mathbf{S}([\sigma_p, \sigma_q], v) + \mathbf{D}([\sigma_q, \sigma_s], v)$.

Conversely, for $q \neq p$ let F_S and F_C be minimum weight forests corresponding to $\mathbf{S}([\sigma_p, \sigma_q])$ and $\mathbf{C}([\sigma_q, \sigma_s], v)$, and set $F_D = F_S \cup F_C$. Then again we have that F_D is a $[\sigma_p, \sigma_s]$-feasible set, and clearly v and σ_s are in the same S-component. Further, F_D has weight $\mathbf{S}([\sigma_p, \sigma_{q-1}]) + \mathbf{C}([\sigma_q, \sigma_s], v)$, and Equation (4) follows.

∎

We now give an algorithm for finding the length of an optimal solution to 2EC problem:

2EC Algorithm

Input: Instance $(G, \boldsymbol{w}, S)$ with $G = (V, E)$ planar and $S = (V_S, E_S)$ a tree.

Output: Minimum weight of a set $F \subset E \setminus E_S$ such that every pair of vertices in V_S is 2-edge-connected in $F \cup E_S$.

Initialize: for each $u, v \in V$ compute $\mathbf{d}_S(u, v)$.

for $i = 1, \ldots, m-1$ **do**
  for $j = 0, \ldots, m-1$ **do**
    compute $\mathbf{S}([\sigma_j, \sigma_{j+i}])$ using (1)
    for each $v \in V$ compute $\mathbf{D}([\sigma_j, \sigma_{j+i}], v)$ using (4)
    for each $v \in V$ compute $\mathbf{B}([\sigma_j, \sigma_{j+i}], v)$ using (2)
    for each $v \in V$ compute $\mathbf{C}([\sigma_j, \sigma_{j+i}], v)$ using (3)

compute and return $\mathbf{S}([\sigma_0, \sigma_m])$ using (1).

Theorem 1 *The 2EC Algorithm solves 2EC for instance G planar and S a tree in $O(n^2m^2) = O(n^2k^2)$ time.*

Proof The correctness of the algorithm follows from Proposition 1, noting that order of computing the equations and the choice of index sets for the minimizations always insure that the appropriate terms have been computed at the time each formula is applied (the initial values, when $A[\sigma_\cdot, \sigma_\cdot] = \emptyset$, being 0). For the complexity of the algorithm, first note that $\mathbf{d}_S(u, v)$ can be computed in $O(n^2)$ time as follows: Add large constant M to the weight of each of the edges having one endpoint V_S, and $2M$ to those edges having two endpoints in V_S. Now find the distances $\mathbf{d}(u, v)$ between all vertex pairs, using the $O(n^2)$ algorithm of Fredrickson [11] for planar graphs, and set

$$\mathbf{d}_S(u, v) = \begin{cases} \mathbf{d}(u, v) & \text{neither } u \text{ nor } v \text{ is in } S \\ \mathbf{d}(u, v) - M & \text{exactly one of } u \text{ and } v \text{ is in } S \\ \mathbf{d}(u, v) - 2M & \text{both } u \text{ and } v \text{ are in } S. \end{cases}$$

(Values of $\mathbf{d}(u, v)$ that are M or greater are considered infinite.) Now each of the functions **B**, **C**, and **D** is computed for $m(m-1)n$ values. The evaluations for **B** and **D** in turn involve $O(m)$ terms, and that for **C** involves $O(n)$ terms. The function **S** is computed for $m(m-1)$ values, each evaluation in turn involving $O(m^2)$ terms. Thus the whole algorithm has complexity $O(n^2 + nm^3 + n^2m^2 + m^4) = O(n^2k^2)$. ∎

3 2VC on trees

The algorithm for 2EC given in Section 2 can be modified to solve 2VC on instances $(G, \boldsymbol{w}, S)$ with G planar and S a tree. For any subset B of edges, a *cutvertex* with respect to B is any vertex whose removal disconnects some pair of vertices spanned by B. Analogous to Lemma 1 we have the following result.

Lemma 2 *Let $(G, \boldsymbol{w}, S)$ be an instance of 2VC, with $S = (V_S, E_S)$ a tree, and let $F \subseteq E \setminus E_S$. Then F is an optimal solution to 2VC if and only if F is a minimum weight forest such that no vertex of S is a cutvertex with respect to $F \cup E_S$.*

Proof The first part of the proof is analogous to that given for Lemma 1. To prove that F is a forest, suppose F contains cycle Ω. Since $F^* = F \cup E_S$ is edge-minimal, then it must be that removal of any edge $e = (v, w)$ of Ω produces cutvertex u on F^*, which must therefore also be on Ω. Choose u and e such that the number of edges on Ω between v and u is minimal. We have that $v \neq u$, since v and w are disconnected by removing u. Now consider the other edge $e' = (v, w')$ on Ω adjacent to v, so that removal of e' from F^* causes a vertex u' on Ω to become a cutvertex. We claim that u will also be a cutvertex. For consider any path Γ from w' to w in $F^* \setminus \{e'\}$. From above we know that Γ must contain either e or u. If Γ did not contain u then it contains e, and hence v. It must therefore also pass through u' before it meets v, and hence u' lies between u and w' on Ω. This contradicts the fact that e was chosen so that u is closest to v. Therefore Γ must contain u. But now u is a cutvertex which also lies closer to w' than u did to v, again a contradiction. Therefore the cycle Ω cannot exist, and so F is a forest. ∎

We can now modify the functions $\mathbf{S}$, $\mathbf{B}$, $\mathbf{C}$, and $\mathbf{D}$ and equations (1), (2), (3), and (4) to solve 2VC. Let $[\sigma_i, \sigma_j]$ be an interval, and let F be a one-sided set of edges with respect to $[\sigma_i, \sigma_j]$, with F^* and ∂F^* defined as in Section 2. We say that F is a $[\sigma_i, \sigma_j]$-*2VC-feasible* set if S contains no cutvertex with respect to F^*. The functions $\mathbf{S}'([\sigma_i, \sigma_j])$, $\mathbf{B}'([\sigma_i, \sigma_j], v)$, $\mathbf{C}'([\sigma_i, \sigma_j], v)$, and $\mathbf{D}'([\sigma_i, \sigma_j], v)$ are now defined analogously to $\mathbf{S}([\sigma_i, \sigma_j])$, $\mathbf{B}([\sigma_i, \sigma_j], v)$, $\mathbf{C}([\sigma_i, \sigma_j], v)$, and $\mathbf{D}([\sigma_i, \sigma_j], v)$, referring now to $[\sigma_i, \sigma_j]$-2VC-feasible sets. To give the recursive formulae, define

$$[\sigma_i, \sigma_j]' = \{\sigma_i, \sigma_j\} \cup \{\sigma_r \in [\sigma_i, \sigma_j] : \langle\sigma_i\rangle \neq \langle\sigma_r\rangle \neq \langle\sigma_j\rangle\}$$

and define $(\sigma_i, \sigma_j]' = [\sigma_i, \sigma_j]' \setminus \{\sigma_i\}$ and $[\sigma_i, \sigma_j)' = [\sigma_i, \sigma_j]' \setminus \{\sigma_j\}$. Finally, set

$$\begin{aligned}[\sigma_i, \sigma_j]'_+ = \ & \{[\sigma_r, \sigma_s] \text{: } \sigma_r, \sigma_s \in [\sigma_i, \sigma_j]', [\sigma_r, \sigma_s] \neq [\sigma_i, \sigma_j], \sigma_r \neq \sigma_s, \text{ and} \\ & P[\sigma_r, \sigma_s] \text{ and } P[\sigma_i, \sigma_j] \text{ have at least one } \textit{edge} \text{ in common}\}\end{aligned}$$

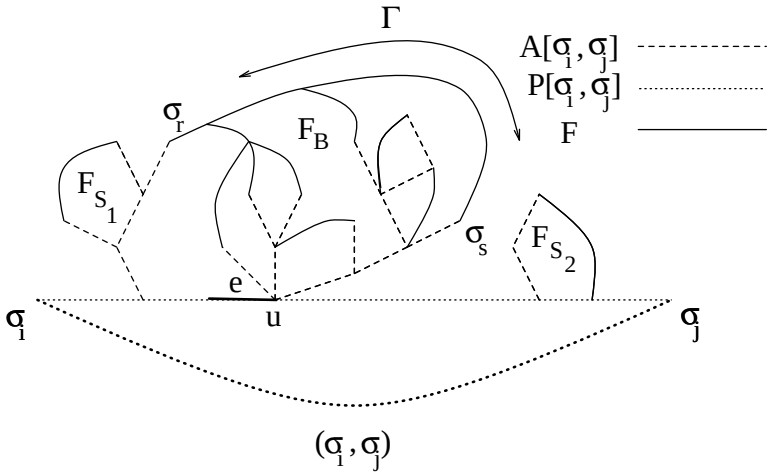

Figure 3: Forest decomposition for $\mathbf{S}'([\sigma_i, \sigma_j])$

Proposition 2 *Let $[\sigma_i, \sigma_j]$ be an interval such that $\langle\sigma_i\rangle \neq \langle\sigma_j\rangle$. Then $\mathbf{S}'([\sigma_i, \sigma_j])$ can be computed using the following set of equations:*

$$\mathbf{S}'([\sigma_i, \sigma_j]) = \min_{[\sigma_r, \sigma_s] \in [\sigma_i, \sigma_j]'_+} \{\mathbf{S}'([\sigma_i, \sigma_r]) + \mathbf{B}'([\sigma_r, \sigma_s], \sigma_r) + \mathbf{S}'([\sigma_s, \sigma_j])\} \quad (5)$$

$$\mathbf{B}'([\sigma_r, \sigma_s], v) = \min_{\sigma_p \in [\sigma_r, \sigma_s)'} \{\mathbf{C}'([\sigma_r, \sigma_p], v) + \mathbf{D}'([\sigma_p, \sigma_s], v)\} \quad (6)$$

$$\mathbf{C}'([\sigma_r, \sigma_p], v) = \min_{u \in V} \{\mathbf{d}_S(v, u) + \mathbf{B}'([\sigma_r, \sigma_p], u)\} \quad (7)$$

$$\mathbf{D}'([\sigma_p, \sigma_s], v) = \min_{\sigma_q \in (\sigma_r, \sigma_s]'} \{\mathbf{S}'([\sigma_p, \sigma_q]) + \mathbf{C}'([\sigma_q, \sigma_s], v)\} \quad (8)$$

with these values being 0 whenever $A[\sigma_\cdot, \sigma_\cdot] = \emptyset$.

Proof The proof for (5) proceeds similarly to that for (1); refer to Figure 3. Let F be a minimum weight $[\sigma_i, \sigma_j]$-2VC-feasible forest, with F^* and ∂F^* as defined in Section 2. Let u be any vertex on $P[\sigma_i, \sigma_j]$ adjacent to some edge of $A[\sigma_i, \sigma_j]$. Since u cannot be a cutvertex of F^*, then it cannot be one for ∂F^* either, and so in particular one of the S-paths Γ in ∂F^* must create a cycle with S that also encloses u along with at least one edge e of $P[\sigma_i, \sigma_j]$ adjacent to u. It follows that for the endpoints σ_r and σ_s of Γ, $P[\sigma_r, \sigma_s]$ must also contain e. Further, neither σ_r nor σ_s can be coincident with σ_i or σ_j, respectively, without the index being the same, since otherwise that vertex would be a cutvertex of F^*. Thus $[\sigma_r, \sigma_s]$ is in $[\sigma_r, \sigma_s]_+$. Again we can unambiguously partition the edges of F into $[\sigma_i, \sigma_r]$-,

$[\sigma_r, \sigma_s]$-, and $[\sigma_s, \sigma_j]$-2VC-feasible subforests F_{S_1}, F_B and F_{S_2}, respectively corresponding to those S-components of F incident to vertices in $[\sigma_i, \sigma_r]$, $[\sigma_r, \sigma_s]$, and $[\sigma_s, \sigma_j]$.

Conversely, let F_{S_1}, F_B and F_{S_2} be minimum weight $[\sigma_i, \sigma_r]$-, $[\sigma_r, \sigma_s]$- and $[\sigma_s, \sigma_j]$-2VC-feasible forests, respectively, with $P[\sigma_i, \sigma_j]$ and $P[\sigma_r, \sigma_s]$ sharing a common edge. Since $F^*_{S_1}$, F^*_B and $F^*_{S_2}$ (when they exist) contain no cutvertices of S and also contain at least one edge in common with either $P[\sigma_i, \sigma_j] \cup \{(\sigma_i, \sigma_j)\}$ or $P[\sigma_r, \sigma_s] \cup \{(\sigma_r, \sigma_s)\}$, then their union likewise contains no cutvertices of S.

The proofs for Equations (6), (7), and (8) are analogous to that for Equations (2), (3), and (4) in Proposition 1, noting that neither σ_p nor σ_q can be coincident to its corresponding endpoint without creating a cycle in F. ∎

Modifying the 2EC Algorithm by substituting $\mathbf{S}'([\sigma_i, \sigma_j])$, $\mathbf{B}'([\sigma_i, \sigma_j], v)$, $\mathbf{C}'([\sigma_i, \sigma_j], v)$, and $\mathbf{D}'([\sigma_i, \sigma_j], v)$ for $\mathbf{S}([\sigma_i, \sigma_j])$, $\mathbf{B}([\sigma_i, \sigma_j], v)$, $\mathbf{C}([\sigma_i, \sigma_j], v)$, and $\mathbf{D}([\sigma_i, \sigma_j], v)$, using (5), (6), (7), (8) in place of (1), (2), (3), (4), we can compute all of the intermediate values of $\mathbf{S}'$, $\mathbf{B}'$, $\mathbf{C}'$, $\mathbf{D}'$ correctly. The final value $\mathbf{S}'([\sigma_0, \sigma_m])$ to be returned by the algorithm is not defined, since Equation (5) requires that $\langle \sigma_i \rangle \neq \langle \sigma_j \rangle$. We can extend Equation (5) to handle the case where $\langle \sigma_i \rangle = \langle \sigma_j \rangle$ by defining in this case

$$[\sigma_i, \sigma_j]'_+ = \{[\sigma_r, \sigma_s] : \sigma_r, \sigma_s \in [\sigma_i, \sigma_j]' : P[\sigma_r, \sigma_s] \text{ contains edges } (\sigma_r, \sigma_{r+1}) \text{ and } (\sigma_{s-1}, \sigma_s)\}.$$

The proof is analogous to the case when $\langle \sigma_i \rangle \neq \langle \sigma_j \rangle$.

By modifying the 2EC algorithm as given above, we obtain the *2VC Algorithm*, and the following result.

Theorem 2 *2VC can be solved in $O(n^2k^2)$ time when G is planar and S a tree.*

4 ECAP and VCAP on Trees

In this section we solve the ECAP and VCAP problems in the case where G is planar, S is a tree, and $r_{ij} \leq 2$ for all $i, j \in S$, by extending or modifying the 2EC and 2VC Algorithms. In the case of the ECAP the extension can be established easily by means of the following lemma.

Lemma 3 *Let $(G, \boldsymbol{w}, S, \boldsymbol{r})$ be an instance of ECAP, with G is planar, S is a tree, and $r_{ij} \leq 2$ for all $i, j \in S$. Let $G^2 = (V^2, E^2)$ be the graph obtained by contracting all edges of S not lying on at least one path between vertices u, v with $r_{uv} = 2$, and let $\boldsymbol{w}^2$ and S^2 correspond to $\boldsymbol{w}$ and S restricted to G^2. Then any optimal solution to 2EC for instance $(G^2, \boldsymbol{w}^2, S^2)$ is an optimal solution to ECAP for instance $(G, \boldsymbol{w}, S, \boldsymbol{r})$.*

Proof Let F be any set of edges, and set $F^* = F \cup S$. Since S is a tree, then contracting the given subset of edges of S in F^* will not change the set of bridges of F^*. Since no noncontracted edge of S can now be a bridge, then F is feasible for the ECAP problem on $(G, \boldsymbol{w}, S, \boldsymbol{r})$ if and only if it is feasible for the 2EC problem on $(G^2, \boldsymbol{w}^2, S^2)$. Thus the optimal solutions are the same for both problems. ∎

Unfortunately, there does not appear to be an obvious modification like the one given in Lemma 3 that will reduce the corresponding VCAP instance to 2VC. VCAP can be solved in this case, however, by a modification of the 2VC Algorithm. We say that a vertex u is an *S-separating vertex* if it lies on some path of S between vertices v and w with $r_{vw} = 2$, and specifically, we say that it S-separates v from w. The following result is an immediate generalization of Lemma 2.

Lemma 4 *Let $(G, \boldsymbol{w}, S, \boldsymbol{r})$ be an instance of VCAP, with $S = (V_S, E_S)$ a tree, and let $F \subseteq E \setminus E_S$. Then F is an optimal solution to VCAP if and only if F is a minimum weight forest such that no S-separating vertex is a cutvertex with respect to $F \cup E_S$.*

It follows from Lemma 4 that an optimal solution to VCAP is a minimum weight set F of edges such that there are no S-separation vertices that are cutvertices of $F \cup E_S$. Now let $[\sigma_i, \sigma_j]$ be an interval, and let F be a one-sided set of edges with respect to $[\sigma_i, \sigma_j]$, with F^* and ∂F^* defined as in Section 2. We say that F is $[\sigma_i, \sigma_j]$-*VCAP-feasible* set if F^* contains no cutvertices that are S-separation vertices. It follows from Lemma 2 that any minimum weight $[\sigma_0, \sigma_m]$-VCAP-feasible set is an optimal solution to VCAP.

Define the functions $\mathbf{S}''([\sigma_i, \sigma_j])$, $\mathbf{B}''([\sigma_i, \sigma_j], v)$, $\mathbf{C}''([\sigma_i, \sigma_j], v)$, and $\mathbf{D}''([\sigma_i, \sigma_j], v)$ analogously to $\mathbf{S}'([\sigma_i, \sigma_j])$, $\mathbf{B}'([\sigma_i, \sigma_j], v)$, $\mathbf{C}'([\sigma_i, \sigma_j], v)$, and $\mathbf{D}'([\sigma_i, \sigma_j], v)$, now referring to $[\sigma_i, \sigma_j]$-VCAP-feasible sets rather than $[\sigma_i, \sigma_j]$-2VC-feasible sets. The main difficulty in modifying equations (5), (6), (7), (8) to apply to the VCAP problem is that the set F^* may contain a cutvertex so long as it is not a S-separation vertex. To determine an optimal solution F to VCAP, therefore, we first need to break F^* recursively into 2-connected components from the "outside in". With this in mind define

$\mathbf{S}''_0([\sigma_i, \sigma_j]) =$ minimum weight of a $[\sigma_i, \sigma_j]$-VCAP-feasible forest F such that ∂F^* is 2-connected (*i.e.* a cycle).

In Figure 4, for example, F is not a feasible forest for $\mathbf{S}''_0([\sigma_i, \sigma_a])$, whereas F_0 is a feasible forest for $\mathbf{S}''_0([\sigma_i, \sigma_j])$ so long as u does not S-separate a vertex in the subtree above it from any vertex not in this subtree. Finally, for interval $[\sigma_i, \sigma_a]$ define

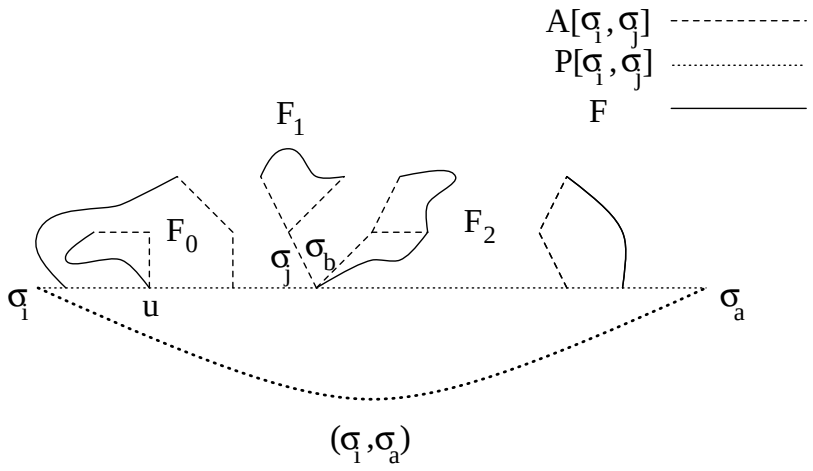

Figure 4: A $[\sigma_i, \sigma_j]$-VCAP-feasible set

$$[\sigma_i, \sigma_a]_0'' = \{[\sigma_a, \sigma_a]\} \cup \{[\sigma_j, \sigma_b] : \sigma_j, \sigma_b \in [\sigma_i, \sigma_a],\ [\sigma_i, \sigma_a] \neq [\sigma_j, \sigma_b],\ \sigma_j \neq \sigma_b,\ \langle\sigma_j\rangle = \langle\sigma_b\rangle,$$
and $\langle\sigma_j\rangle$ does not S-separate two vertices exactly one of which is in $[\sigma_j, \sigma_b]\}$.

Proposition 3 *Let $[\sigma_i, \sigma_a]$ be an interval. If $A[\sigma_i, \sigma_a]$ contains no vertex u with $r_{uv} = 2$ for some v, then $\mathbf{S}''([\sigma_i, \sigma_a]) = 0$; otherwise $\mathbf{S}''([\sigma_i, \sigma_a])$ can be computed using the following formula:*

$$\mathbf{S}''([\sigma_i, \sigma_a]) = \min_{(\sigma_j, \sigma_b) \in [\sigma_i, \sigma_a]_0''} \{\mathbf{S}_0''([\sigma_i, \sigma_j]) + \mathbf{S}''([\sigma_j, \sigma_b]) + \mathbf{S}''([\sigma_b, \sigma_a])\} \tag{9}$$

Proof Refer to Figure 4 for the proof. Let F be a $[\sigma_i, \sigma_a]$-VCAP-feasible forest. If ∂F^* has no cutvertices, then F is feasible for $\mathbf{S}_0''([\sigma_i, \sigma_a])$, and so it corresponds to $[\sigma_j, \sigma_b] = [\sigma_a, \sigma_a]$. Otherwise, let σ_j be the first element of $[\sigma_i, \sigma_a]$ in the clockwise traversal of ∂F^* such that $\langle\sigma_j\rangle$ is a cutvertex of ∂F^*, and let σ_b be the next element in the traversal at which $\langle\sigma_b\rangle = \langle\sigma_j\rangle$. Since $\langle\sigma_j\rangle$ is a cutvertex for F^*, it cannot S-separate two vertices exactly one of which is in $[\sigma_j, \sigma_a]$, and hence $[\sigma_j, \sigma_b] \in [\sigma_i, \sigma_a]_0''$. Further, from the construction of σ_j and σ_b there can be no S-paths of F connecting vertices in different ones of $[\sigma_i, \sigma_j]$, $[\sigma_j, \sigma_b]$, and $[\sigma_b, \sigma_a]$. It follows that F can be partitioned into subforests F_0, F_1 and F_2, respectively, with ∂F_0^* having no cutvertices by choice of σ_j. Further, any S-cutvertex for F_0^*, F_1^* or F_2^* would have to S-separate those same vertices in F^*. It follows that F_0, F_1 and F_2 are $[\sigma_i, \sigma_j]$-VCAP-feasible, $[\sigma_j, \sigma_b]$-VCAP-feasible, and $[\sigma_b, \sigma_a]$-VCAP-feasible sets.

Conversely, let F_0, F_1 and F_2 be $[\sigma_i, \sigma_j]$-, $[\sigma_j, \sigma_b]$-, and $[\sigma_b, \sigma_a]$-VCAP-feasible forests, respectively, with ∂F_0^* 2-connected. Set $F = F_0 \cup F_1 \cup F_2$. Suppose F^* had cutvertex $\langle\sigma_l\rangle$

that is an S-separating vertex of F^*. Since F_0, F_1 and F_2 are VCAP-feasible, then $\langle\sigma_l\rangle$ cannot S-separate any pair of vertices both of which are in one of these sets. If σ_l were in $[\sigma_i, \sigma_a]$, then it would lie on or inside the cycle ∂F_0^*, and so $\langle\sigma_l\rangle$ could not be a cutvertex between a vertex of $[\sigma_i, \sigma_j]$ and one in either $[\sigma_j, \sigma_b]$ or $[\sigma_b, \sigma_a]$. Finally, if σ_l were in $[\sigma_j, \sigma_b]$ then $\langle\sigma_l\rangle$ could not be an S-separating vertex between a point in $[\sigma_j, \sigma_b]$ and one in $[\sigma_b, \sigma_a]$, by choice of $[\sigma_j, \sigma_b]$. Thus F is a $[\sigma_i, \sigma_a]$-VCAP-feasible set. ∎

The recursive equations for $\mathbf{S}_0''$, $\mathbf{B}''$, $\mathbf{C}''$, and $\mathbf{D}''$ are analogous to those for $\mathbf{S}'$, $\mathbf{B}'$, $\mathbf{C}'$, and $\mathbf{D}'$:

$$\mathbf{S}_0''([\sigma_i, \sigma_j]) = \min_{(\sigma_r, \sigma_b) \in [\sigma_i, \sigma_j]'_+} \{\mathbf{S}_0''([\sigma_i, \sigma_r]) + \mathbf{B}'([\sigma_r, \sigma_s], \sigma_r) + \mathbf{S}_0''([\sigma_s, \sigma_j])\} \tag{10}$$

$$\mathbf{B}''([\sigma_r, \sigma_s], v) = \min_{\sigma_p \in [\sigma_r, \sigma_s)'} \{\mathbf{C}''([\sigma_r, \sigma_p], v) + \mathbf{D}''([\sigma_p, \sigma_s], v)\} \tag{11}$$

$$\mathbf{C}''([\sigma_r, \sigma_p], v) = \min_{u \in V} \{\mathbf{d}_S(v, u) + \mathbf{B}''([\sigma_r, \sigma_p], u)\} \tag{12}$$

$$\mathbf{D}''([\sigma_p, \sigma_s], v) = \min_{\sigma_q \in (\sigma_r, \sigma_s]'} \{\mathbf{S}''([\sigma_p, \sigma_q]) + \mathbf{C}''([\sigma_q, \sigma_s], v)\} \tag{13}$$

with these values being 0 whenever $A[\sigma., \sigma.]$ contains no vertex u with $r_{uv} = 2$ for some v. The proofs are analogous to those of Theorem 1 and 2, noting for Equation (10) that since ∂F^* is 2-connected, then the partitioning can be done as if F is 2VC-feasible. We can therefore modify the 2EC algorithm analogously as for 2VC, computing (9) after (10), to solve VCAP. The above discussion, along with Lemma 4, is summarized in the following result.

Theorem 3 *The VCAP and ECAP problems can be solved in in $O(n^2k^2)$ time when G planar, and S a tree, and all connectivity requirements are at most 2.*

5 ECAP and VCAP on Connected Subgraphs

This section completes the proof of the Main Theorem by showing how to generalize all of the problems in this paper to cover instances where where S is a general connected subgraph.

Lemma 5 *Let $(G, \boldsymbol{w}, S, \boldsymbol{r})$ be an instance of ECAP (VCAP) with S connected. Let $T = (V_S, E_T)$ be any spanning tree for S, and let $\boldsymbol{w}'$ be obtained from $\boldsymbol{w}$ by changing the weights of edges of $E_S \setminus E_T$ to zero (or a sufficiently small positive number). If F' is an optimal solution for ECAP (VCAP) on $(G, \boldsymbol{w}', T, \boldsymbol{r})$ then $F = F' \setminus E_S$ is an optimal solution for ECAP (VCAP) on $(G, \boldsymbol{w}, S, \boldsymbol{r})$.*

Proof First, let F_0' be feasible for ECAP (VCAP) on instance $(G, \boldsymbol{w}', T, \boldsymbol{r})$, so that the connectivity in $F_0' \cup E_T$ between every pair i, j of vertices in V_S is r_{ij}. Now since the edges of $E_S \setminus E_T$ connect vertices of V_S, then adding any of these edges to $F_0' \cup E_T$ will not decrease the connectivity between any pair of vertices. Thus $F_0' \cup E_S$ also satisfies the connectivity requirements of $\boldsymbol{r}$, and so $F_0' \setminus E_S$ is also feasible for ECAP (VCAP) on instance $(G, \boldsymbol{w}, S, \boldsymbol{r})$ and has the same weight. Conversely, let F_0 be feasible for ECAP (VCAP) on instance $(G, \boldsymbol{w}, S, \boldsymbol{r})$, so that the connectivity in $F_0 \cup E_S$ between every pair i, j of vertices in V_S is r_{ij}. But then $F_0' = F_0 \cup E_S \setminus E_T$ is also feasible for ECAP (VCAP) on $(G, \boldsymbol{w}', T, \boldsymbol{r})$ — since $F_0' \cup E_T = F_0 \cup E_S$ — and has the same weight. The optimal solutions for these two problems will therefore also have the same weight, and the lemma follows. ∎

The Main Theorem follows from Theorem 3 and Lemma 5.

6 Two Applications

The algorithm given in Section 2 has important applications in the areas of network design and network vulnerability. In this paper we outline two of these applications.

6.1 Heuristic for a Network Design Problem

The construction of networks with specified connectivity levels between vertices has been of considerable interest [20], recently with regard to telephone system design [4, 16, 21]. The version of ECAP or VCAP considered in these papers was introduced in [19], and involves instance $(G, \boldsymbol{w}, S, \boldsymbol{r})$ where the subgraph S is not connected, and the connectivity requirements are of the form $r_{ij} = \min\{r_i^0, r_j^0\}$, where $\boldsymbol{r}^0$ is a vector of *node connectivity requirements*. This version remains NP-hard even when G is planar and all $r_i^0 = 2$, as mentioned in Section 1. The paper [21] gives some good polynomial time heuristics for the problem in the special case where requirement values are restricted to 1 or 2, including one called "two-trees dense" which finds a minimum spanning tree T on G (connecting all pairs i, j with $r_{ij} \geq 1$) and then finds a second spanning tree edge-disjoint from T on the set of vertices having $r_i^0 = 2$. We can improve the second phase of this heuristic considerably on planar graphs by using 2EC or 2VC to actually find the minimum weight subgraph containing T and 2-connecting the required set of vertices. The method is as follows.

Tree Augmentation Heuristic

1. Find a minimum spanning tree T on G using any minimum spanning tree algorithm.
2. Let T' be the unique minimal subtree of T containing those vertices v having $r_v = 2$.
3. Set the weights of all edges in $T \setminus T'$ to 0, and apply the 2VC or 2EC Algorithm. The resulting optimal set S, together with the original tree T, will be the minimum weight subgraph B containing T and having the appropriate connectivity between its vertices.

The Tree Augmentation Heuristic is also appropriate when the initial tree is constructed using edge weights that are different from those used for the 2-connections, and particularly when the tree weights are significantly greater than the weights for the secondary connections. In this case it would be appropriate to construct the optimal spanning tree first without regard for the secondary connections, and then make the additional connections based on the choice of the initial tree. In fact, if (a) the minimum spanning tree is unique, and (b) the first stage weights are such that the difference in weight between the smallest and second-smallest tree is larger than the sum of the weights for the secondary connections, then it can be proven that the Tree Augmentation Heuristic actually finds the correct solution for ECAP or VCAP.

The heuristic applies as well when vertices v with $r_v^0 = 0$ are allowed, meaning that the solution is not necessarily required to span G. In this case Step 1 of the Tree Augmentation Heuristic will require finding a Steiner tree on the set K of vertices with $r_v^0 \geq 1$. The Steiner tree problem is also NP-hard, even on planar graphs, but there are a large number of good heuristics, approximations, and polynomial time special case algorithms that can be applied to solve this step of the algorithm. In particular, for the planar case, if the vertices of K lie entirely on the exterior boundary (or even on a fixed number of different faces) of G, then the Steiner tree problem can be solved in polynomial time [6], and hence this heuristic can be applied efficiently.

Finally, we note that this heuristic applies as well when r_{ij} is generally defined, so long as the set of pairs i, j with $r_{ij} \geq 1$ are "connected" in the sense that V cannot be partitioned into two sets V_1 and V_2 such that

(i) $r_{ij} \geq 1$ for at least one pair $i, j \in V_1$ and at least one pair $i, j \in V_2$;

(ii) $r_{ij} = 0$ for all pairs $i \in V_1$, $j \in V_2$.

This insures that the first stage of the heuristic — finding the set of edges for the connections among pairs with $r_{ij} \geq 1$ — again requires finding a Steiner tree on the set $K = \{i : \ r_{ij} \geq 1 \text{ for at least one } j \in V\}$. The second stage — making the appropriate connections between pairs i, j with $r_{ij} = 2$ — is an application of the general ECAP or VCAP algorithm.

6.2 The Multicommodity Cut Problem

The *(Full) Multicommodity Cut Problem* has as input graph $G = (V, E)$, edge weights $\boldsymbol{w} = (w_e : \ e \in E)$, and set $(s_1, t_1), \ldots, (s_k, t_k)$ of terminal pairs of vertices of G. It is desired to find the minimum weight set of edges $S \subseteq E$ whose removal disconnects each s_i-t_i pair. This problem has been studied in the second author's PhD thesis [1], and a restricted version of the problem, called the *Multiway Cut Problem* (or *Multiterminal Cut Problem* or *k-Terminal Cut Problem*), has also been studied quite extensively [2, 3, 5]. The Multicommodity Cut Problem has been shown to be NP-hard, even if there are only three terminal pairs [5], G is a tree [15] or G is a grid graph with all terminals on the outside boundary [1]. In this section we consider a further restriction on the problem. An instance of the Multicommodity Cut Problem is called *noncrossing* if the G has a plane layout with all terminal pairs lying on the exterior face of G, and further, when the edges $(s_1, t_1), \ldots, (s_k, t_k)$ are added to the exterior face the graph remains planar.

Theorem 4 *Let $(G, \boldsymbol{w}, (s_1, t_1), \ldots, (s_k, t_k))$ be a noncrossing instance of the Multicommodity Cut Problem, let G^d be the graph obtained by adding the edges (s_1, t_1), $\ldots, (s_k, t_k)$, and let G^* be the planar dual of G^d with dual edge weights $\boldsymbol{w}^*$ corresponding to those for the corresponding edges of G. Then the edges which are dual to $(s_1, t_1), \ldots, (s_k, t_k)$ form a tree T^* in G^*, and any solution to 2EC on $(G^*, \boldsymbol{w}^*, T^*)$ will have its corresponding dual set of edges a solution to the Multicommodity Cut Problem on $G, (s_1, t_1), \ldots, (s_k, t_k)$.*

Proof For the first part of the theorem, note that the edges of T^* form a tree, since

(a) G^d has no vertices lying in the exterior of G, so that T^* contains no cycles, and

(b) the exterior face G is a connected region, so that T^* is connected.

For the second part of the theorem, let F^* be any set of edges in $G^* \setminus T^*$, and let F be the corresponding set of edges of G. Let (s_i, t_i) be a terminal pair, and (s_i^*, t_i^*) the corresponding edge of T^*. It follows from the properties of planar dual graphs that F disconnects s_i from t_i if and only if there is a path in F^* from s_i^* to t_i^*. But this is saying

that the edge (s_i^*, t_i^*) is contained in a cycle of $F^* \cup T^*$. The second part of the theorem now follows from Lemma 1. ∎

Corollary 1 *The Multicommodity Cut Problem can be solved for noncrossing instances in* $O(n^2k^2)$ *time.*

References

[1] R.C. Burk, "Full and Partial Multicommodity Cuts," Ph.D. thesis, University of North Carolina, Chapel Hill, 1993.

[2] S. Chopra and M.R. Rao, On the multiway cut polyhedron, *Networks* **21** (1991), 51–89.

[3] W.H. Cunningham, The optimal multiterminal cut problem, *DIMACS Series in Disc. Math. and Theor. Comp. Sci.* **5** (1991), 105–120.

[4] R.H. Cardwell, C.L. Monma, and T.H. Wu, Computer-aided design procedures for survivable fiber optic networks, *IEEE Select. Areas Commun.* **7** (1989), 1188–1197.

[5] E. Dahlhaus, D.S. Johnson, C.H. Papadimitriou, P.D. Seymour, and M. Yannakakis, The complexity of multiway cuts, in "Proceedings of the Twenty-fourth Annual ACM Symposium on the Theory of Computing," pp. 241–251, ACM Press, New York 1992.

[6] R.E. Erickson, C.L. Monma, and A.F. Veinott, The send-and-split method for minimum-concave-cost network flows, *Math. of O.R.* **12** (1987), 634–644.

[7] K.P. Eswaran and R.E. Tarjan, Augmentation problems, *SIAM J. Comput.* **5** (1976), 653–665.

[8] A. Frank, Augmenting graphs to meet edge-connectivity requirements, *SIAM J. Disc. Math.* **5** (1992), 25–53.

[9] A. Frank, Connectivity augmentation problems in network design, in "Mathematical Programming: State of the Art 1994," J.R. Birge and K.G. Murty (Eds.), pp. 34–63, The University of Michigan, Ann Arbor 1994.

[10] H. Frank and W. Chou, Connectivity considerations in the design of survivable networks, *IEEE Trans. Circuit Theory* **CT-17** (1970), 486–490.

[11] Fredrickson, G.N. Fast algorithms for shortest paths in planar graphs, with applications, *SIAM Jour. Comp.* **16** (1987), 1004–1022.

[12] Fredrickson, G.N. and J. JáJá, Approximation algorithms for several graph augmentation problems, *SIAM Jour. Comp.* **10** (1982), 189–201.

[13] M.R. Garey and D.S. Johnson, The rectilinear Steiner tree problem is NP-complete, *SIAM J. Appl. Math.* **32** (1977), 826–834.

[14] M.R. Garey, D.S. Johnson and R.E. Tarjan, The planar Hamilonian circuit problem is NP-complete, *SIAM Jour. Comp.* **5** (1976), 704–714.

[15] N. Garg, V.V. Vazirani and M. Yannakakis, Primal-dual approximation algorithms for integral flow in trees, with applications to matching and set cover, in "Proceedings of the Twentieth International Colloquium on Automata, Languages and Programming," A. Lingas, R. Karlsson, and S. Carlsson (Eds.), pp. 64–75, Lecture Notes in Computer Science 700, Springer-Verlag, 1993.

[16] M. Grötschel and C.L. Monma, Integer polyhedra arising from certain network design problems with connectivity constraints, *SIAM J. Disc. Math.* **3** (1990), 502–523.

[17] M. Grötschel, C.L. Monma, and M. Stoer, Design of survivable networks, in "Network Models," M.O. Ball, T.L. Magnanti, C.L. Monma, and G.L. Nemhauser (Eds.), pp. 617–672, Handbooks in Operations Research and Managements Science, Volume 7, North-Holland, New York, 1995.

[18] F.K. Hwang, D.S. Richards, and P. Winter, "The Steiner Tree Problem," North-Holland, New York 1992.

[19] J. Krarup, *Generalized Steiner problem*, unpublished note, 1978.

[20] T.L. Magnanti and R.T. Wong, Network Design and transportation planning: models and algorithms, *Trans. Sci.* **18** (1984), 1–55.

[21] C.L. Monma and D.F. Shallcross, Methods for designing communications networks with certain two-connected survivability constraints, *Oper. Res.* **37** (1989), 531–541.

[22] J.S. Provan, On finding two-connected subgraphs in planar graphs, Technical Report UNC/OR TR94-15, Department of Operations Research, University of North Carolina, Chapel Hill, 1995.

[23] P. Winter, Generalized Steiner problem in series-parallel networks, *J. Algorithms* **7** (1986), 549–566.