Approximating Fault-Tolerant Group-Steiner Problems

Rohit Khandekar[1], Guy Kortsarz[2], and Zeev Nutov[3]

[1]IBM T.J.Watson Research Center. rohitk@us.ibm.com

[2]Rutgers University, Camden. guyk@camden.rutgers.edu

[3]The Open University of Israel. nutov@openu.ac.il

ABSTRACT. In this paper, we initiate the study of designing approximation algorithms for Fault-Tolerant Group-Steiner (FTGS) problems. The motivation is to protect the well-studied group-Steiner networks from edge or vertex failures. In Fault-Tolerant Group-Steiner problems, we are given a graph with edge- (or vertex-) costs, a root vertex, and a collection of subsets of vertices called groups. The objective is to find a minimum-cost subgraph that has two edge- (or vertex-) disjoint paths from each group to the root. We present approximation algorithms and hardness results for several variants of this basic problem, e.g., edge-costs vs. vertex-costs, edge-connectivity vs. vertex-connectivity, and 2-connecting from each group a single vertex vs. many vertices. Main contributions of our paper include the introduction of very general structural lemmas on connectivity and a charging scheme that may find more applications in the future. Our algorithmic results are supplemented by inapproximability results, which are tight in some cases.

Our algorithms employ a variety of techniques. For the edge-connectivity variant, we use a primal-dual based algorithm for covering an *uncrossable* set-family, while for the vertex-connectivity version, we prove a new graph-theoretic lemma that shows equivalence between obtaining two vertex-disjoint paths from two vertices and 2-connecting a carefully chosen single vertex. To handle large group-sizes, we use a p-Steiner tree algorithm to identify the "correct" pair of terminals from each group to be connected to the root. We also use a non-trivial charging scheme to improve the approximation ratio for the most general problem we consider.

1 Introduction

The fault-tolerant network design problems are well-studied in the theory of combinatorial optimization and approximation algorithms. The basic goal in these problems is to design a minimum-cost network that satisfies some prescribed connectivity requirements. Higher connectivity requirements are usually enforced for fault-tolerance — in order to protect connectivity in the solution against edge or vertex failures. A well-studied fault-tolerant network design problem is the Steiner Network problem. In this problem, we are given edge- (or vertex-) connectivity requirement r_{ij} between every pair $\{i, j\}$ of vertices, and the goal is to design a network with at least r_{ij} edge- or vertex-disjoint paths between i and j for each pair $\{i, j\}$.

In this paper, we study fault-tolerant versions of the Group-Steiner problem. In this problem, we are given a (undirected or directed) graph $G = (V, E)$ with edge- or vertex-costs, a root vertex $r \in V$, and a collection of of subsets (groups) $\mathcal{S} = \{S_1, \ldots, S_q\}$ of $V \setminus \{r\}$.

Foundations of Software Technology and Theoretical Computer Science (Kanpur) 2009.
Editors: Ravi Kannan and K. Narayan Kumar; pp 263–274
Leibniz International Proceedings in Informatics (LIPIcs), Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Germany.
Digital Object Identifier: 10.4230/LIPIcs.FSTTCS.2009.2324

The objective is to find a minimum cost subgraph H of G that contains two edge- or vertex-disjoint paths from each group S_i to the root. This problem models the flexibility, often arising in problems in VLSI design [25], in connecting any vertices from a given group as well as the fault-tolerance which requires the solution to be robust under edge- or vertex-failures.

1.1 Previous work on fault-tolerance problems

Fault tolerance problems have been extensively studied in Combinatorial Optimization and Approximation Algorithms. Consider for example the well known Steiner Network problem. Given an undirected graph $G = (V, E)$ with edge-costs $\{c_e \mid e \in E\}$, and requirement r_{ij} for every pair of vertices $i, j \in V$, the goal is to find a minimum cost subgraph H of G that contains at least r_{ij} edge-disjoint paths between i and j, for all i, j. The network H is fault tolerant in the sense that a pair i, j can sustain $r_{ij} - 1$ link failures and still be connected. The best ratio known for this problem is 2 due to Jain [14].

The internally-disjoint path version of the Steiner Network problem is very hard to approximate [16, 21, 9, 18]. The currently best known ratio for this problem is $O(k^3 \log n)$ for edge-costs due to Chuzhoy and Khanna [8] and $O(k^4 \log^2 n)$ for vertex-costs [20]. The rooted version, where a root s is given and $r_{ij} = 0$ for all pairs i, j so that $i \neq s$ and $j \neq s$, has gotten a significant attention recently on the high level [3, 5, 9, 23, 8, 20]. This problem is at least as hard to approximate as Directed Steiner Tree [18]. The best approximation ratio known for this problem for general rooted demands is $O(k^2)$ for edge-costs and $O(k^2 \log n)$ for vertex-costs [20], where $k = \max r_{ij}$. We mention that prior to the work of [20], a randomized approximation algorithm with ratio $k^{O(k^2)} \log^4 n$ was developed by Chakraborty, Chuzhoy, & Khanna [3], then improved to $k^{O(k)} \log n$ by Chekuri & Korula [5], and then improved to $O(k^2 \log n)$ by Chuzhoy & Khanna [9, 8] and [23]. Note that k can be as large as $\Omega(n)$.

A particular case of fault tolerance problems with 2 disjoint paths has also received an attention on the high level [1, 19, 5]. Lau et al. [19] presented an $O(\log^2 n)$-approximation algorithm for the problem of finding a minimum-cost 2-edge connected subgraph with *at least k* vertices. This problem can be seen as a fault-tolerant generalization of the k-MST problem which requires to find a minimum spanning tree on k vertices. The best known approximation ratio for the k-MST problem is 2 [11]. Chekuri and Korula [6] presented an $O(\log^2 n)$-approximation for the problem of finding a minimum-cost 2-vertex connected subgraph with at least k terminals. In [1, 5], the fault-tolerance version of the Buy-at-Bulk problem was studied, where two edge-disjoint paths are required to be included from every terminal to the root.

In the same spirit, in this paper we consider a generalization of the Group-Steiner Tree problem. In the usual Group-Steiner Tree problem, we are given a graph $G = (V, E)$, edge costs $\{c_e : e \in E\}$, a root $r \in V$, and a collection of subsets (groups) $\mathcal{S} = \{S_1, \ldots, S_q\}$ of $V \setminus \{r\}$. The objective is to find a minimum cost subtree T of G that contains r and at least one vertex from each group $S_i \in \mathcal{S}$. The best known approximation ratio for this problem is $O(\log^3 n)$ [12]. The Fault-Tolerant Group-Steiner Tree problem, on the other hand, requires obtaining two (edge- or vertex-) disjoint paths between each group to the root. We are not aware of any previous work on Fault-Tolerant Group-Steiner problems.

1.2 Problem variants studied in this paper

One can define several variants of the Fault-Tolerant Group-Steiner problem, based on whether we desire 2-edge- or 2-vertex-connectivity, whether we have edge- or vertex-costs, whether we wish to 2-connect to the root a single vertex from each group or two distinct vertices from each group, etc. Below we formally define all the variants studied in this paper. Two paths are said to be *internally-disjoint* if they are vertex-disjoint except for their end-points. Each of the following problems takes a graph $G = (V, E)$ on n vertices with edge-costs $\{c_e \mid e \in E\}$ (or with node-costs $\{c_v \mid v \in V\}$), a root $r \in V$, and groups $\mathcal{S} = \{S_1, \ldots, S_q\}$ as input, and is required to compute a min-cost subgraph H of G with at least two edge/vertex-disjoint paths from each group S_i to the root, so that the endnodes of these paths are distinct. Unless stated otherwise, we consider the edge-cost version and assume that G is undirected. We also assume that the groups $S_1, \ldots, S_q$ are pairwise disjoint.* The vertices in the groups S_i are called *terminals*. We add the prefix EC- for edge-connectivity and the prefix VC- for vertex-connectivity. We add the suffix "-k" after the name of the problem if the instances are restricted to satisfy $|S_i| \leq k$ for all $i = 1, \ldots, q$.

- EC-FTGS: Here for every $i = 1, \ldots, q$, H should contain at least two edge-disjoint $S_i r$-paths; the end-points in S_i of these paths should be distinct.
- VC-FTGS: The same as EC-FTGS, except that the paths should be internally-disjoint.
- EC-FTGS-k and VC-FTGS-k: These are EC-FTGS and VC-FTGS, respectively, restricted to instances with $|S_i| \leq k$ for all $i = 1, \ldots, q$.

In the edge-connectivity case, for both edge and vertex-costs, the version when the end-points in S_i of the two $S_i r$-paths may or may not be distinct, is easily reduced to EC-FTGS as follows. For every terminal s, add a new vertex s' of cost 0 connected to s with an edge of cost 0, and add s' to every group $S \in \mathcal{S}$ that contains s. After this transformation, we can assume, without loss of generality, that the two edge-disjoint paths from each group start from *distinct* terminals in that group. This may double the number of vertices, and cause a constant loss in approximation ratios that depend on n.

We also consider the version when we insist that a single vertex from each group must be 2-edge-connected to the root. Namely, for every $i = 1, \ldots, q$ there should exist a vertex $v_i \in S_i$ such that H contains 2 edge-disjoint rv_i-paths. We call this version Single EC-FTGS.

1.3 The difficulty in approximating Fault-Tolerant problems

When two disjoint paths from every group to the root are required, we cannot use the standard transformation to Bartal trees [2, 10] as done in the approximation of the Group Steiner problem [12]. This is so because disjoint paths from a group to the root in a Bartal tree do not necessarily correspond to disjoint paths in the original graph.

We now give a strong evidence that an approximation that is polylogarithmic in n for either EC-FTGS-k and VC-FTGS-k may be very hard to obtain as it implies solving a long standing open problem. Note that we can reduce the usual Group Steiner problem to EC-FTGS-k or VC-FTGS-k problems by adding a new vertex, which is connected to the root

*This can be typically assumed by making multiple copies of the vertices in multiple groups and adding zero-cost edges connecting the different copies. This reduction, however, increases the number of vertices in the graph, thus possibly affecting the approximation ratio.

Problem	Edge-Connectivity (EC)	Vertex-Connectivity (VC)
FTGS-2	3.55, Vertex-Cover-hard $O(\log n)$, $\Omega(\log n)$-hard (vertex-costs)	$O(\log^2 n)$
FTGS-k	$O(k \log^2 n)$	$O(k \log^2 n)$
FTGS	$O(\sqrt{n} \log n)$ Group Steiner Tree-hard	$O(\sqrt{n} \log n)$ Group Steiner Tree-hard
Single FTGS	Label Cover-hard (directed)	Label Cover-hard (directed)

Table 1: Approximation ratios and hardness results for FTGS variants. The extra assumptions, if any, are given in the parentheses.

by a zero-cost edge, to each group. Since we get one path for "free", any solution to EC-FTGS-k and VC-FTGS-k corresponds to a solution of the usual Group Steiner problem and vice-versa.

Now since an algorithm for EC-FTGS-k or VC-FTGS-k cannot use Bartal trees, it must solve the Group Steiner problem as well **without** using Bartal trees. The question if Group Steiner problem can be approximated within a polylogarithmic ratio without first reducing the graph into trees is a long standing open question and seemingly a very hard one.

To the best of our knowledge, the best known ratio for Group Steiner problem without using Bartal trees is $O(n^\epsilon)$ for any constant $\epsilon > 0$, with running time $n^{f(1/\epsilon)}$. The *recursive greedy* technique [27, 17, 4], used in this algorithm, is a complex greedy approach with quite delicate analysis that seems completely inappropriate for the requirement of two disjoint paths. Thus even an n^ϵ approximation for every universal constant ϵ seems to be a quite significant challenge for our problems in the current state of knowledge and techniques.

In our opinion, this is a strong evidence that it is quite a challenge to get polylogarithmic ratios for either EC-FTGS-k or VC-FTGS-k in polynomial time.

Remark: The above simple reduction shows that the $\Omega(\log^{2-\epsilon} n)$ hardness for the Group Steiner problem applies also for EC-FTGS-k and VC-FTGS-k. However, from the above evidence, EC-FTGS-k and VC-FTGS-k may in fact be much harder to approximate than $\Omega(\log^{2-\epsilon} n)$, and it may be that a polynomial ratio is the best we can hope for.

1.4 Our results

We start with a definition. For two optimization problems $\mathcal{P}_1$ and $\mathcal{P}_2$, we say that $\mathcal{P}_1$ *is* $\mathcal{P}_2$-*hard* if existence of a ρ-approximation algorithm for $\mathcal{P}_1$ implies existence of a ρ-approximation for $\mathcal{P}_2$. Similarly, $\mathcal{P}_1$ is $\Omega(f(n))$-hard if there exists a constant $\epsilon > 0$ so that $\mathcal{P}_1$ admits no $\epsilon f(n)$-approximation algorithm, unless P=NP.

Our main results are summarized in Theorem 1 and in Table 1.

THEOREM 1.

(i) EC-FTGS-2 *admits the following approximation ratios:*
$(2+\gamma)$ *for edge costs, where* $\gamma < 1.55$ *is the best ratio for the* Steiner Tree *problem,*

and $O(\log n)$ for vertex costs. Moreover, the edge-cost version is Vertex-Cover*-hard and the vertex-cost version is $\Omega(\log n)$-hard.*

(ii) EC-FTGS-k *and* VC-FTGS-k *admit an $O(k \log^2 n)$-approximation algorithm.* EC-FTGS *and* VC-FTGS *admit an $O(\sqrt{n} \log n)$-approximation algorithm.* EC-FTGS *and* VC-FTGS *are* Group Steiner Tree*-hard and thus are $O(\log^{2-\epsilon} n)$-hard for any constant $\epsilon > 0$.*

(iii) *The directed version of* Single EC-FTGS *problem admits no $2^{\log^{1-\epsilon} n}$-approximation for any constant $\epsilon > 0$, unless $\mathrm{NP} \subseteq \mathrm{DTIME}(n^{\mathrm{polylog}(n)})$.*

The results (i) and (ii) are proved in Sections 2 and 3 respectively. The proof of (iii) is omitted due to lack of space.

2 Proof of Part (i)

We start with some definitions. An edge e is said to *cover* a subset $X \subset V$ of vertices if exactly one end-point of e lies in X. Let $\mathcal{F}$ be a collection of subsets of V. We say that an edge-set E' covers $\mathcal{F}$ if for each $X \in \mathcal{F}$, there is an edge $e \in E'$ that covers X. The Set-Family Edge-Cover problem with edge-cost is to find a minimum-cost collection of edges E' that covers $\mathcal{F}$. In the vertex-cost version, we wish to minimize the total cost of vertices incident to E' that covers $\mathcal{F}$. The family $\mathcal{F}$ may not be given explicitly, but we require that certain queries related to $\mathcal{F}$ can be answered in polynomial time. Specifically, we assume that, in the edge-cost version, for any edge-set E', the inclusion minimal members of $\mathcal{F}$ that are *not* covered by E' can be computed in polynomial time; while, in the vertex-cost version, for any $s, t \in V$, one can compute in polynomial time a min-cost cover of all members of $\mathcal{F}$ that separate s and t.

We call a family $\mathcal{F}$ of sets *uncrossable* if $X \cap Y, X \cup Y \in \mathcal{F}$ or $X \setminus Y, Y \setminus X \in \mathcal{F}$ for any $X, Y \in \mathcal{F}$. Our algorithms for EC-FTGS-2 problem with edge-costs or vertex-costs use the following results, respectively.

THEOREM 2. (Goemans et al. [13]) *The* Set-Family Edge-Cover *problem with edge-costs and with uncrossable set-family $\mathcal{F}$ admits a 2-approximation algorithm.*

THEOREM 3.([22]) *The* Set-Family Edge-Cover *problem with vertex-costs and with uncrossable set-family $\mathcal{F}$ admits an $O(\log n)$-approximation algorithm.*

2.1 Algorithmic results

Since the problem insists that the two edge-disjoint paths from each group must start at distinct terminals in the group, the optimum solution contains a Steiner tree containing all the terminals and the root. Our algorithm first finds a Steiner tree T that connects all the terminals to the root. If we use an α-approximation algorithm for this step ($\alpha < 1.55$ for the edge-costs [26] and $\alpha = O(\log n)$ for the vertex-costs [15]), we get $\mathrm{COST}(T) \leq \alpha \cdot \mathrm{OPT}$ where OPT denotes the cost of the optimum solution.

For $X \subset V$, let $\mathtt{deg}_T(X)$ denote the number of edges in T from X to $V \setminus X$. Define an instance of Set-Family Edge-Cover by setting

$$\mathcal{F} = \{X \subseteq V \setminus \{r\} \mid \mathtt{deg}_T(X) = 1, S \subseteq X \text{ for some } S \in \mathcal{S}\}.$$

We now present two important observations.

LEMMA 4. *For $I \subseteq E \setminus E(T)$, the set $T \cup I$ is a feasible solution to* EC-FTGS-2 *if, and only if, I covers $\mathcal{F}$.*

PROOF. Note that T has a path from each terminal to the root. Thus by Menger's Theorem, $H = T \cup I$ is a feasible solution to EC-FTGS-2 if, and only if, $\texttt{deg}_H(X) \geq 2$ for every set $X \subseteq V \setminus \{r\}$ that contains some group $S \in \mathcal{S}$. As $\texttt{deg}_T(X) \geq 1$ for any $X \subseteq V \setminus \{r\}$ that contains at least one vertex from some group, we obtain that the latter condition is equivalent to $\texttt{deg}_I(X) \geq 1$ for every $X \in \mathcal{F}$. ∎

LEMMA 5. *The set family $\mathcal{F}$ is uncrossable.*

PROOF. Note that by the definition of $\mathcal{F}$, $X \in \mathcal{F}$ if, and only if, X is a union of a rooted proper subtree of T that contains a group $S \in \mathcal{S}$ and any subset of vertices not in T. Let $X, Y \in \mathcal{F}$. Then $X \cap T, Y \cap T$ are disjoint, or one of them contains the other. In the former case, we have $X \setminus Y, Y \setminus X \in \mathcal{F}$; e.g., $X \setminus Y \in \mathcal{F}$ since $(X \setminus Y) \cap T = X \cap T$, hence $X \setminus Y$ is a union of the subtree contained in X and the vertex subset $X \setminus (T \cup Y)$ disjoint to T. In the latter case, $X \cap Y, X \cup Y \in \mathcal{F}$; e.g., if $X \subseteq Y$, then $X \cap Y$ is a union of the subtree contained in X and some vertices not in T, while $X \cup Y$ is the union of the subtree contained in Y and some vertices not in T. ∎

It is easy to check that for any edge set $I \subset E \setminus E(T)$, the minimal members of the family $\mathcal{F}$ not covered by I can be computed in polynomial time. Moreover, for any $s, t \in V$, a min-cost cover of all members in $\mathcal{F}$ that separate s and t can also be computed in polynomial time. Thus, we can use the algorithms in Theorems 2 and 3 respectively to complete the solutions for the edge- and vertex-cost versions.

2.2 Hardness of approximation results

We now show that EC-EFTGS-2 is Vertex-Cover hard in the case of edge-costs, and that it is Hitting-Set-hard, i.e., $\Omega(\log n)$-hard, in the case of vertex-costs.

Let $J = (V_J, E_J)$ be an instance of Vertex-Cover. Define an instance $\{G = (V, E), \{c_e : e \in E\}, r, \mathcal{S}\}$ of 2-EC-FTGS-2 as follows. Set $V = V_J \cup \{a, r\}$, connect every vertex in V_J to a with an edge of cost 0 and connect a to r with an edge of cost 0. Let T denote the set of these zero-cost edges. Connect each vertex in V_J to r with an edge of cost 1 each. The set $\mathcal{S}$ of pairs is defined by edges of E_J, namely, $\mathcal{S} = \{\{u, v\} \mid (u, v) \in E_J\}$. Note that the optimum solution, without loss of generality, picks all the edges in T. It is now easy to see that $T + H$ is a feasible solution to the obtained instance of EC-FTGS-2 if and only if the set of end-points of the edges in H is a vertex-cover in J.

In the case of vertex-costs, EC-FTGS-2 is easily reduced to the Steiner Tree problem with vertex-costs which is known to be Hitting-Set-hard [15]. Given an instance $\{J = (V_J, E_J), r, S\}$ of Steiner Tree with vertex-costs, for every $s \in S$ add a copy s' of cost 0 and connect s' to r. The set of pairs is $\mathcal{S} = \{\{s, s'\} \mid s \in S\}$. It is easy to see that T is a feasible solution to Steiner Tree with vertex-costs if and only if $T \cup \{(r, s') \mid s \in S\}$ is a feasible solution to the constructed 2-EC-FTGS-2 instance.

The proof of Part (i) of Theorem 1 is thus complete.

3 Proof of Part (ii)

3.1 Algorithm for VC-FTGS-2

In this section, we introduce our main technical ideas. We present an $O(\log^2 n)$-approximation algorithm for VC-FTGS-2.

As in the edge-connectivity case, we first compute a Steiner tree T of cost $\text{COST}(T) \leq \alpha \cdot \text{OPT}$ connecting all terminals to the root. We have $\alpha < 1.55$ for edge-costs and $\alpha = O(\log n)$ for vertex-costs. Our algorithm uses set-cover like approach in which we iteratively add partial solutions with low *density*, i.e., low cost to profit ratio, till we complete the solution. We get one logarithmic factor in the approximation from the set-cover analysis (and since the number of groups is $O(n)$) and another logarithmic factor from the fact that we can only compute $O(\log n)$ approximation to the minimum-density subproblem.

Given a partial solution $I \subset E \setminus E(T)$, let the *deficiency* of I be the number of groups in $\mathcal{S}$ that are not 2-vertex-connected to r in $T \cup I$. Let the *density* of an edge set $F \subset E \setminus (E(T) \cup I)$ be $\text{COST}(F)$ divided by the decrease in the deficiency caused by adding F to $T \cup I$. The following two lemmas captures the essence of our algorithm for VC-FTGS-2.

LEMMA 6. *Given a partial solution $T \cup I$, the problem of computing a minimum density augmenting edge set $F \subset E \setminus (E(T) \cup I)$ for* VC-FTGS-2 *admits an $O(\log n)$-approximation algorithm.*

LEMMA 7. *The algorithm in Lemma 6 can be used to obtain $O(\log^2 n)$ approximation for* VC-FTGS-2.

As mentioned, the proof of Lemma 7 follows from the standard set-cover like analysis and is omitted. In the rest of this section we prove Lemma 6. We ignore the groups in $\mathcal{S}$ that are already connected in $T \cup I$ to the root via 2 vertex-disjoint paths starting from distinct vertices.

We start by recalling some definitions. A vertex $v \in V$ is a cut-vertex of a graph H if $H \setminus \{v\}$ has more connected components than H. A cut-vertex v of H is said to *separate* vertex r and set $S \subset V \setminus \{r, v\}$ if r and S belong to the same connected component of H but $H \setminus \{v\}$ does not contain a path from r to any vertex in S. Consider a group $\{s_1, s_2\} \in \mathcal{S}$. By Menger's Theorem we have:

PROPOSITION 8. *A subgraph that contains an rs_1-path and an rs_2-path, contains such paths that are internally vertex disjoint if and only if it has no cut-vertex that separates r and $\{s_1, s_2\}$.*

Now think of the tree T as being rooted at r. For any two vertices $s_1, s_2 \in T$, we define $\text{LCA}(s_1, s_2)$ to be the least common ancestor of s_1 and s_2 in T. Consider $S = \{s_1, s_2\} \in \mathcal{S}$ with $u = \text{LCA}(s_1, s_2)$ in T. Note that $u \neq r$ since S is not 2-vertex connected to r via paths starting from s_1 and s_2. Let $\mathcal{U}(S) = \{u_1, u_2\}$ be two (possibly identical) vertices defined as follows.

- If $u \notin \{s_1, s_2\}$, then let u_1 (resp. u_2) be the child of u that lies on the rs_1- (resp. rs_2-) path in T.
- If $u \in \{s_1, s_2\}$, then let $u_1, u_2 = u$.

Define a family $\mathcal{U} = \mathcal{U}(\mathcal{S}, T)$ of pairs (groups) as $\mathcal{U} = \{\mathcal{U}(S) \mid S \in \mathcal{S}\}$. Note that the pairs in $\mathcal{U}$ may not be disjoint, and that some "pairs" in $\mathcal{U}$ are in fact singletons. The following lemma captures the "equivalence" between covering pairs in $\mathcal{S}$ and pairs in $\mathcal{U}$.

LEMMA 9. *For any edge set $F \subset E \setminus (E(T) \cup I)$, the solution $H = T \cup I \cup F$ contains 2-vertex disjoint paths (with distinct starting points) between r and S if, and only if, it contains 2-vertex disjoint paths (with distinct starting points) between r and $\mathcal{U}(S)$.*

PROOF. Let $u = \text{LCA}(s_1, s_2)$ where $S = \{s_1, s_2\}$. By Proposition 8, S is 2-connected to r in $H = T \cup I \cup F$ if and only if H has no cut-vertex separating S from r; namely, no vertex on the ur-path in T is a cut-vertex of H. By the definition of $\mathcal{U}(S) = \{u_1, u_2\}$ and Proposition 8, this is equivalent to the property that $\mathcal{U}(S)$ is 2-connected to r in H. ∎

The above lemma implies that the densities of F w.r.t. $\mathcal{S}$ and w.r.t. $\mathcal{U}$ are equal. Furthermore, $T \cup I \cup F$ is a feasible solution w.r.t. $\mathcal{S}$ if and only if it is w.r.t. $\mathcal{U}$. Note that the groups $\mathcal{U}$ satisfy a special property, which is crucial in rest of the analysis.

`Property P`: For all groups $\{u_1, u_2\} \in \mathcal{U}$, either $u_1 = u_2$ or u_1, u_2 have the same parent in T.

LEMMA 10. *Let $U = \{u_1, u_2\} \in \mathcal{U}$. For $F \subset E \setminus (E(T) \cup I)$, the graph $H = T \cup I \cup F$ contains a $u_1 r$-path and a $u_2 r$-path that are internally vertex-disjoint if and only if H contains 2 internally-disjoint paths to r from either u_1 or u_2.*

PROOF. The proof uses Menger's Theorem and `Property P` of the groups in $\mathcal{U}$.

Suppose that H contains 2 internally vertex-disjoint paths from u_1 to r. Then H has no cut-vertex separating r and u_1, by Menger's Theorem. In particular, there is no cut-vertex separating r and $\{u_1, u_2\}$. Thus H contains a $u_1 r$-path and a $u_2 r$-path that are internally vertex-disjoint, by Proposition 8.

Suppose now that H contains a $u_1 r$-path and a $u_2 r$-path that are internally vertex-disjoint. Now we use `Property P`. If $u_1 = u_2$, the proof is complete. Assume therefore that $u_1 \neq u_2$ and that they have a common parent u in T and assume to the contrary that H has no pair of internally vertex-disjoint $u_i r$-paths for $i = 1, 2$. Then by Menger's Theorem, there are cut-vertices v_1, v_2 in H, where v_i separates u_i from r. As u_1, u_2 have a common parent $u \neq r$, any vertex separating r from one of u_1, u_2 must lie on the ur-path in T. If $v_1 = v_2 = v$, then v separates both u_1, u_2 from r contradicting the assumption (by Proposition 8). Thus $v_1 \neq v_2$, so one of v_1, v_2 is distinct from u, say $v_1 \neq u$. The graph $H \setminus \{v_1\}$ contains a $u_2 r$-path. As $T \setminus \{v_1\}$ contains a $u_1 u_2$-path, this implies that $H \setminus \{v_1\}$ contains a $u_1 r$-path. This contradicts that v_1 separates u_1 and r. ∎

Thus the original density problem can be reduced to the following problem. Given a collection of groups $\{u_1^i, u_2^i\}$ for $i = 1, 2, \ldots$, find a subset $F \subset E \setminus (E(T) \cup I)$ such that the ratio of $\text{COST}(F)$ to the number of groups i such that at least one of u_1^i or u_2^i has 2-vertex-disjoint paths to r in $E(T) \cup I \cup F$.

The problem of finding a subgraph that minimizes the ratio of its cost over the total profit of vertices that are 2-vertex-connected to the root was studied by Chekuri and Korula [6], who gave an $O(\log n)$-approximation for the problem. We use their algorithm to compute an $O(\log n)$-approximation for the density version of our problem, as follows. The input to the algorithm of Chekuri and Korula is the original graph with root r and with *profits* on vertices defined as follows. Let the *profit* $p(u)$ of a vertex u be the number of groups

```
1. Initialize subgraph $\mathcal{H} \leftarrow \emptyset$ and $q' \leftarrow q$ to be the number of uncovered groups.
2. If $q' > 0$, begin a phase:
    (a) Assign a cost of zero to all vertices in $\mathcal{H}$.
    (b) Find Twin-pairs: Compute a subgraph $H$ of cost $O(\text{OPT} \cdot k \log n)$ that contains twin
        pairs (Definition 11) from at least $q'/2$ uncovered groups.
    (c) Cover: Compute a subgraph $I$ of cost $O(\text{OPT} \cdot k \log n)$ that covers at least $q'/2$ uncovered
        groups.
    (d) Update $\mathcal{H} \leftarrow \mathcal{H} \cup H \cup I$ and update $q'$ to be the number of uncovered groups in $\mathcal{H}$.
3. Output $\mathcal{H}$.
```

Figure 1: An outline of our algorithm for VC-FTGS-k with vertex-costs.

i such that $u \in \{u_1^i, u_2^i\}$. Thus $p(u)$ denotes the number of new groups that would get connected to the root via 2 vertex-disjoint paths provided u gets connected to the root via 2 vertex-disjoint paths. Note that since both u_1^i or u_2^i may claim profit for covering group i, we may overestimate the profit of 2-vertex connecting a subset of vertices to r by a factor of 2. This introduces another factor 2 in the approximation. Using Chekuri-Korula algorithm, we compute a subgraph which yields a $O(\log n)$ approximation for minimizing the ratio of its cost over the total profit of its vertices that are 2-vertex-connected to the root. This subgraph yields $O(\log n)$ approximation to the problem defined in Lemma 6.

Thus the proof is complete.

3.2 Algorithm for EC-FTGS-k and VC-FTGS-k with edge/vertex costs

We present an algorithm for VC-FTGS-k with vertex-costs, which is more general than the case of edge-costs. Adaptation of this algorithm to EC-FTGS-k is easy.

Fix an optimum solution OPT with cost also denoted by OPT. To simplify the presentation, the algorithm given below is assumed to know the value of OPT. In reality, the algorithm tries all possible guesses for the power of 2 that is closest to OPT and picks the cheapest solution among those computed for each of these guesses.

Our algorithm has logarithmic number of phases. In each phase, it covers at least half of the remaining groups. A high-level pseudo-code of the algorithm is given in Figure 1. At the beginning of each phase, we make the cost of the vertices that are already picked in the solution $\mathcal{H}$ *zero*. In what follows, we analyze a single phase which begins with q' uncovered groups overall. The groups in $\mathcal{S}$ that are already covered are ignored. In what follows, we explain how to implement steps `Find Twin-pairs` and `Cover` respectively.

How to implement step `Find Twin-pairs`**.**

DEFINITION 11. *For a group $S \in \mathcal{S}$, we say that the terminals $s_1, s_2 \in S$ form a* twin pair *if* OPT *contains two internally-disjoint paths from s_1 and s_2 to r. If there are more than one such pairs for a group, we designate exactly one of these pairs as* twin pair *arbitrarily.*

We do not know which terminals form twin pairs a-priori. Nevertheless, we can compute a low-cost tree that connects the twin pairs from at least half of the remaining groups to the root, as shown below. We iteratively use p-Steiner tree algorithm for $p = q'$. Recall that the p-Steiner Tree problem is to compute a minimum-cost tree that connects at least p terminals to the root. Let H denote the union of the p-Steiner trees computed so far. Assume that the number of uncovered groups is at least $q'/2$.

LEMMA 12. *Assign a cost of zero vertices in H. Now apply $(c \log n)$-approximation algorithm [7] for the p-Steiner tree problem (where c is a constant) for the instance given by H, root r, $p = q'$ and terminals as the vertices in the the union of uncovered groups in $\mathcal{S}$ but not in H: $\{v \in S \mid S \in \mathcal{S}$ is uncovered, $v \notin H\}$. If the cost of the computed Steiner tree is more than $(c \log n) \cdot \text{OPT}$, then H contains the twin pairs for at least $q'/2$ uncovered groups.*

PROOF. Assume on the contrary that H does not contain twin pairs for at least $q'/2$ uncovered groups. Thus the OPT solution connects at least $2 \cdot q'/2 = q'$ terminals in this p-Steiner tree instance to the root. Since we use a $(c \log n)$-approximation, the cost of the computed Steiner tree is at most $(c \log n) \cdot \text{OPT}$. This is a contradiction, and the lemma follows. ∎

We run the p-Steiner tree algorithm iteratively while the cost of the new tree computed is at most $c \log n \cdot \text{OPT}$. Since H contains at least $p = q'$ new terminals in each iteration, the total number of invocations of such p-Steiner tree algorithm is at most $q' \cdot k/q'$. This holds since the size of each group is at most k. Thus the total cost of the step `Find Twin-pairs` is $\text{COST}(H) \leq O(\text{OPT} \cdot k \log n)$.

How to implement step `Cover`.

Even if H contains the twin pairs of at least $q'/2$ uncovered groups, we still do not know which of the terminals in H form twin pairs. We therefore need one more definition.

DEFINITION 13. *Let T be a spanning tree of H. We say that a vertex u_1* can help *an uncovered group S if there exist distinct vertices $s_1, s_2 \in S \cap T$ and another vertex u_2 so that u_1, u_2 have the same parent $u = \text{LCA}(s_1, s_2)$ in the tree T. The* profit *$p(u)$ of vertex u is defined as the number of uncovered groups in $\mathcal{S}$ that u can help.*

The intuition of the above definition comes from our algorithm for VC-FTGS-2, and in particular, from Lemma 10. Note that the profit of a vertex u is the number of uncovered groups that will get covered if u gets connected to the root via 2 internally-disjoint paths.

Since more than one vertex can claim a profit for covering a single group, it is important to understand how many vertices can help a particular group. Since there are at most k terminals in any group $S \in \mathcal{S}$, it is obvious that there can be at most $O(k^2)$ vertices that can help S. This crude upper bound comes from the fact that there are $O(k^2)$ pairs $s_1, s_2 \in S$ that can give rise to such vertices. However the following lemma presents a careful counting of such vertices.

LEMMA 14. *There are $O(k)$ vertices that can help any single group $S \in \mathcal{S}$.*

PROOF. Consider tree T with terminals in group S marked as s (see Figure 2). Further consider a subtree T' restricted to terminals in S. The vertices $u \in T'$ that can play a role of $\text{LCA}(s_1, s_2)$ for $s_1, s_2 \in S$ (these are shown as red squares in Figure 2) have a degree of at least 3 in T', i.e., $\texttt{deg}_{T'}(u) \geq 3$. Thus if a vertex v can help group S, it must be a child of a vertex u with $\texttt{deg}_{T'}(u) \geq 3$. The number of children of a vertex u is $\texttt{deg}_{T'}(u) - 1$. Therefore, the number of vertices v that can help S is at most $\sum_{u:\texttt{deg}_{T'}(u) \geq 3} (\texttt{deg}_{T'}(u) - 1)$.

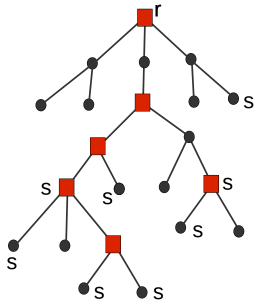

Figure 2: The vertices marked with s are terminals in a group S. The vertices marked with red squares can take a role of $\text{LCA}(s_1, s_2)$ for a pair $\{s_1, s_2\}$ in group S. The children of red squares that lie on a path from some $s \in S$ to r can help S. There are $O(k)$ such vertices.

Since T' is a tree induced on at most k terminals of S, the tree has at most k leaves. By a simple counting argument, it therefore follows that the desired sum $\sum_{u:\texttt{deg}_{T'}(u)\geq 3}(\texttt{deg}_{T'}(u) - 1)$ is $O(k)$. The lemma thus holds. ∎

Now we have all the ingredients to present the step `Cover`. We again use the $O(\log n)$-approximation algorithm of Chekuri and Korula [6] that was also used in Section 3.1, for the problem of finding a subgraph that minimizes the ratio of its cost over the total profit of vertices that are 2-vertex-connected to the root. Recall that the subgraph induced by OPT has cost at most OPT and profit at least $q'/2$ (since T contains twin pairs from at least $q'/2$ uncovered groups). Thus the cost-to-profit ratio of the solution computed by our algorithm is at most $O(\log n \cdot \text{OPT}/q')$. From Lemma 14, we get that the ratio of the cost of this subgraph to the number of uncovered groups covered is at most $O(k \log n \cdot \text{OPT}/q')$.

We apply such algorithm iteratively till end of the phase, i.e., at least $q'/2$ previously uncovered groups are covered in this phase, and call this subgraph I. From the above analysis, it is clear that the total cost of step `Cover` is $\text{COST}(I) = q' \cdot O(k \log n \cdot \text{OPT}/q') = O(k \log n \cdot \text{OPT})$. Thus the total cost of a phase is $\text{COST}(H) + \text{COST}(I) = O(\text{OPT} \cdot k \log n)$. Since there are $O(\log q) = O(\log n)$ phases overall, the total cost of the algorithm is $O(\text{OPT} \cdot k \log^2 n)$ as desired.

As was mentioned in the Introduction, an $O(\sqrt{n} \log n)$-approximation algorithm for VC-FTGS immediately follows from the $O(k \log^2 n)$-approximation for VC-FTGS-k, and thus the proof of Part (ii) of Theorem 1 is now complete.

References

[1] S. Antonakopoulos, C. Chekuri, F. B. Shepherd, and L. Zhang. Buy-at-bulk network design with protection. *FOCS* 2007.

[2] Y. Bartal. On approximating arbitrary metrices by tree metrics. *STOC* 1998.

[3] T. Chakraborty, J. Chuzhoy, and S. Khanna. Network design for vertex connectivity. *STOC* 2008.

[4] M. Charikar, C. Chekuri, T.-Y. Cheung, Z. Dai, A. Goel, S. Guha, and M. Li. Approximation algorithms for directed steiner problems. *J. Algorithms*, 33(1):73–91, 1999.

[5] C. Chekuri and N. Korula. Single-sink network design with vertex connectivity requirements. *FSTTCS* 2008.

[6] C. Chekuri and N. Korula. Pruning 2-Connected Graphs. *FSTTCS* 2008.

[7] F. Chudak, T. Roughgarden, and D. Williamson. Approximate k-MSTs and k-Steiner Trees via the Primal-Dual Method and Lagrangean Relaxation. *IPCO* 2001.

[8] J. Chuzhoy and S. Khanna. An $O(k^3 \log n)$-approximation algorithm for vertex-connectivity survivable network design. *FOCS* 2009.

[9] J. Chuzhoy and S. Khanna. Algorithms for single-source vertex-connectivity. *FOCS* 2008.

[10] J. Fakcharoenphol, S. Rao, and K. Talwar. A tight bound on approximating arbitrary metrics by tree metrics. *J. Comput. Syst. Sci.*, 69(3):485–497, 2004.

[11] N. Garg. Saving an epsilon: a 2-approximation for the k-MST problem in graphs. *STOC* 2005.

[12] N. Garg, G. Konjevod, and R. Ravi. A polylogarithmic approximation algorithm for the group steiner tree problem. *J. Algorithms*, 37(1):66–84, 2000.

[13] M. X. Goemans, A. V. Goldberg, S. Plotkin, D. B. Shmoys, E. Tardos, and D. P. Williamson. Improved approximation algorithms for network design problems. *SODA* 1994.

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

[15] P. N. Klein and R. Ravi. A nearly best-possible approximation algorithm for node-weighted steiner trees. *J. Algorithms*, 19(1):104–115, 1995.

[16] G. Kortsarz, R. Krauthgamer, and J. R. Lee. Hardness of approximation for vertex-connectivity network design problems. *SIAM J. Computing*, 33(3):704–720, 2004.

[17] G. Kortsarz and D. Peleg. Approximating the weight of shallow steiner trees. *Discrete Applied Mathematics*, 93(2-3):265–285, 1999.

[18] Y. Lando and Z. Nutov. Inapproximability of survivable networks. *Theor. Comput. Sci.*, 410(21-23):2122-2125, 2009.

[19] L. C. Lau, J. Naor, M. R. Salavatipour, and M. Singh. Survivable network design with degree or order constraints. *STOC* 2007.

[20] Z. Nutov. Approximating minimum cost connectivity problems via uncrossable bifamilies and spider-cover decompositions. *FOCS* 2009.

[21] Z. Nutov. Approximating connectivity augmentation problems. *SODA* 2005.

[22] Z. Nutov. Approximating Steiner Networks with Node Weights. *LATIN* 2008.

[23] Z. Nutov. An almost $O(\log k)$-approximation for k-connected subgraphs. *SODA* 2009.

[24] R. Raz. A parallel repitition theorem. *SIAM J. Computing*, 27(3):763–803, 1998.

[25] G. Reich and P. Widmayer. Beyond steiner's problem: a VLSI oriented generalization. In *Proceedings of the workshop on Graph-theoretic concepts in computer science*, 1990.

[26] G. Robins and A. Zelikovsky. Tighter bounds for graph Steiner tree approximation. *SIAM J. Discrete Math.*, 1(19):122–134, 2005.

[27] A. Zelikovsky. A series of approximation algorithms for the acyclic directed steiner tree problem. *Algorithmica*, 18(1):99–110, 1997.