Improved Approximation Algorithms for Capacitated Network Design and Flexible Graph Connectivity

Ishan Bansal
Amazon, Bellevue, WA, USA

Joe Cheriyan
Department of Combinatorics and Optimization, University of Waterloo, Canada

Sanjeev Khanna
Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA, USA

Miles Simmons
Department of Combinatorics and Optimization, University of Waterloo, Canada

Abstract

We present improved approximation algorithms for some problems in the related areas of Capacitated Network Design and Flexible Graph Connectivity.

In the Cap-k-ECSS problem, we are given a graph $G = (V, E)$ whose edges have non-negative costs and positive integer capacities, and the goal is to find a minimum-cost edge-set F such that every non-trivial cut of the graph $G' = (V, F)$ has capacity at least k. Let $n = |V|$ and let u_{min} (respectively, u_{max}) denote the minimum (respectively, maximum) capacity of an edge; assume that $u_{max} \leq k$. We present an $O(\log(k/u_{min}))$-approximation algorithm for the Cap-k-ECSS problem, asymptotically improving upon the previous best approximation ratio of $\min(O(\log n), k, 2u_{max}, 6 \cdot \lceil k/u_{min} \rceil)$ whenever $\log(k/u_{min}) = o(\log n)$ and u_{max} is sufficiently large.

In the (p, q)-Flexible Graph Connectivity problem, denoted (p, q)-FGC, the input is a graph $G = (V, E)$ where E is partitioned into *safe* and *unsafe* edges, and the goal is to find a minimum-cost edge-set F such that the subgraph $G' = (V, F)$ remains p-edge connected upon removal of any q unsafe edges from F. We present an 8-approximation algorithm for the $(1, q)$-FGC problem that improves upon the previous best approximation ratio of $(q + 1)$.

Both of our results are obtained by using natural LP relaxations strengthened with the knapsack-cover inequalities, and then, during the rounding process, utilizing a recent $O(1)$-approximation algorithm for the Cover Small Cuts problem. In the latter problem, the goal is to find a minimum-cost set of links such that each non-trivial cut of capacity less than a specified value is covered by a link. We also show that the problem of covering small cuts inherently arises in another variant of (p, q)-FGC. Specifically, we give Cook reductions that preserve approximation ratios within $O(1)$ factors between the $(2, q)$-FGC problem and the 2-Cover Small Cuts problem; in the latter problem, each small cut needs to be covered by two links.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis

Keywords and phrases Approximation algorithms, Capacitated network design, Covering small cuts, Edge-connectivity of graphs, f-Connectivity problem, Flexible Graph Connectivity, Knapsack-cover inequalities

Digital Object Identifier 10.4230/LIPIcs.ICALP.2025.20

Category Track A: Algorithms, Complexity and Games

Related Version *Full Version*: `https://doi.org/10.48550/arXiv.2411.18809`

Funding *Ishan Bansal*: This work is external and does not relate to the position at Amazon.
Joe Cheriyan: Supported in part by NSERC grant RGPIN-2024-04473.
Sanjeev Khanna: Supported in part by NSF award CCF-2402284 and AFOSR award FA9550-25-1-0107.

Acknowledgements We thank the anonymous reviewers and the ICALP PC for their comments. We are grateful to David Aleman Espinosa and Sharat Ibrahimpur for reading a preliminary version, and for their detailed comments.

1 Introduction

Given a graph $G = (V, E)$ whose edges have both costs and capacities, a fundamental task in network design is to find a spanning subgraph of minimum cost that satisfies some specified connectivity requirements. In this paper, we present results on two well-studied problems in the area of approximation algorithms pertaining to the design of networks with edge-connectivity requirements.

When all edges have the same capacity, a seminal result by Jain [14] provides a 2-approximation algorithm for the survivable network design problem (SNDP); see section 1.4.3 for further discussion.

In the more general setting, where edges have non-uniform capacities, the problem, referred to as *capacitated network design*, becomes more challenging. Our focus is on designing approximation algorithms for the special case called the Cap-k-ECSS problem where we are given a graph $G = (V, E)$ with a non-negative cost and a positive integer capacity for each edge, and the algorithmic goal is to find a minimum-cost spanning subgraph such that every non-trivial cut has capacity k or more. Let $n = |V|$ and let u_{min} (respectively, u_{max}) denote the minimum (respectively, maximum) capacity of an edge; assume that $u_{max} \leq k$. One of the earliest approximation algorithms for the Cap-k-ECSS problem is due to Goemans et al. [9], and it achieves an approximation ratio of $\min\{2k, |E|\}$. The best approximation ratio known is $\min(O(\log n), k, 2u_{max}, 6 \cdot \lceil k/u_{min} \rceil)$, due to Goemans et al. [9], Chakrabarty et al. [7], Boyd et al. [5], and Bansal [3]. Moreover, there are no known hardness results that rule out the possibility of better asymptotic approximation ratios.

Another line of research called *flexible graph connectivity* (abbreviated as FGC), has emerged recently, motivated by natural questions in network design in the setting of robust optimization. Adjiashvili, Hommelsheim and Mühlenthaler [1] proposed an FGC model that distinguishes between safe (never-failing) and unsafe (failure-prone) edges. The algorithmic goal is to choose a set of edges of minimum cost that satisfies a (global) edge-connectivity requirement, while tolerating failures of up to a specified number of unsafe edges. A basic problem in this setting is the $(1, q)$-FGC problem where the goal is to ensure that the network remains connected even after the failure of up to q unsafe edges. We mention that the $(1, q)$-FGC problem can be modeled as a special case of the Cap-k-ECSS problem.

In the rest of the introduction section, we first discuss related work, and then we formalize the problems studied in this paper. After that, we give an overview of the main tools and techniques underlying our results, followed by our results and discussion on two of our algorithms.

We may use abbreviations for some standard terms, e.g., we may use "$(1, q)$-FGC" as an abbreviation for "the $(1, q)$-FGC problem". For each of the minimization problems (in network design or flexible graph connectivity), we use OPT to denote the optimal value (i.e., the minimum cost of an integer solution), and LP_{opt} to denote the optimal value of an LP relaxation. The context will resolve potential ambiguities.

1.1 Related Work

As mentioned above, research on approximation algorithms for Cap-k-ECSS was initiated by Goemans et al. [9]. Carr et al. [6], in a seminal paper, introduced a key algorithmic tool for capacitated network design based on the Knapsack-Cover Inequalities (KCI); we discuss KCI in more detail in section 1.4.1. More than a decade later, Chakrabarty et al. [7] used

KCI to design an $O(\log n)$ approximation algorithm for Cap-k-ECSS. Boyd et al. [5] gave a $\min\{k, 2u_{max}\}$-approximation algorithm for Cap-k-ECSS, and Bansal [3], based on previous work by Bansal et al. [4] and Williamson et al. [21], gave a $(6 \cdot \lceil k/u_{min} \rceil)$-approximation algorithm for Cap-k-ECSS.

The model of flexible graph connectivity originated from research in the area of robust optimization. Adjiashvili, Stiller and Zenklusen [2] introduced their model of bulk-robust combinatorial optimization, and designed some approximation algorithms. Later, Adjiashvili, Hommelsheim and Mühlenthaler [1] introduced the FGC model. Boyd et al. [5] introduced a generalization called the (p,q)-FGC model. Boyd et al. [5] presented a 4-approximation algorithm for $(p,1)$-FGC based on the primal-dual method of Williamson, Goemans, Mihail & Vazirani (WGMV) [21], and a $(q+1)$-approximation algorithm for $(1,q)$-FGC; moreover, they gave an $O(q \log n)$-approximation algorithm for (p,q)-FGC. Subsequently, several interesting results and approximation algorithms have been presented; we summarize some of these recent papers in chronological order. Chekuri and Jain [8] give $O(p)$-approximation algorithms for, respectively, $(p,2)$, $(p,3)$ and $(2p,4)$-FGC, and an $O(q)$-approximation algorithm for $(2,q)$-FGC. Bansal et al. [4], among other results, give an $O(1)$-approximation algorithm for $(p,2)$-FGC; moreover, they give a 16-approximation algorithm for a related problem called Cover Small Cuts. Nutov [17] improves the approximation ratio for Cover Small Cuts from 16 to 10. Bansal [3], and later Nutov [18], give a 6-approximation algorithm for Cover Small Cuts; also, see [20]. Bansal [3] gives an $O(1)$-approximation algorithm for $(p,3)$-FGC. Hyatt-Denesik et al. [12], among other results, give approximation algorithms for unit-cost FGC problems with edge-connectivity requirements as well as for unit-cost FGC problems with vertex connectivity requirements. Hommelsheim et al. [11] study a model related to a generalization of FGC called the (p,q)-Steiner-Connectivity Preservation problem. Ibrahimpur & Vegh [13] give an $O(\log n)$-approximation algorithm for (p,q)-FGC.

1.2 Capacitated Network Design and the Cap-k-ECSS problem

The Cap-k-ECSS problem is as follows: Given an undirected graph $G = (V,E)$ with edge costs $c \in \mathbb{Q}^E_{\geq 0}$ and edge capacities $u \in \mathbb{Z}^E_{\geq 0}$, find a minimum-cost edge-set $F \subseteq E$ such that the capacity of any cut in (V,F) is at least k. Let u_{min} (respectively, u_{max}) denote the minimum (respectively, maximum) capacity of an edge in E, and assume (w.l.o.g.) that $u_{max} \leq k$.

For a graph $G = (V,E)$ and a set of nodes $S \subseteq V$, the *cut of* S, denoted by $\delta(S)$, refers to the set of edges that have exactly one end-node in S. Whenever we use the term "cut $\delta(S)$" we mean that S is a subset of $V(G)$. We call a cut $\delta(S)$ *non-trivial* if S is a nonempty, proper subset of V, that is, if $\emptyset \neq S \subsetneq V$.

The following integer program formulates the Cap-k-ECSS problem. It can be viewed as the natural "cut covering" formulation of the problem. It has a binary variable x_e for each edge e, with the meaning that an edge e is picked iff $x_e = 1$, and, for each non-trivial cut, it has a constraint stating that the capacity of the picked edges in the cut is $\geq k$.

$$\begin{aligned} \min \quad & \sum_{e \in E} c_e x_e & & \text{(IP: CapkECSS)} \\ \text{s.t.} \quad & \sum_{e \in E \cap \delta(S)} u_e x_e \geq k & & \forall\ \emptyset \subsetneq S \subsetneq V \\ & x_e \in \{0,1\} & & \forall\ e \in E \end{aligned}$$

The LP (linear programming) relaxation of the above integer program is obtained by replacing $x_e \in \{0,1\}$ by $0 \leq x_e \leq 1$, $\forall e \in E$. The following well-known example shows that the LP relaxation has an integrality ratio of $\Omega(k)$; similar examples are given in [6, 7].

▶ **Example 1.** The graph G consists of two nodes u, v, and a pair of parallel edges e_1, e_2 between the two nodes. Edge e_1 has cost zero and capacity $k-1$, and edge e_2 has cost one and capacity k. A feasible solution of the integer program has cost one since the edge e_2 must be chosen in any feasible solution. On the other hand, a feasible solution x to the LP relaxation of cost $1/k$ is given by $x_{e_1} = 1, x_{e_2} = 1/k$. Hence, the integrality ratio of the LP relaxation is k for this example. Thus, we face an obstruction for the task of designing any approximation algorithm that achieves approximation ratio $o(k)$ by rounding this LP relaxation.

1.3 Flexible Graph Connectivity and the (p, q)-FGC problem

Adjiashvili, Hommelsheim and Mühlenthaler [1] introduced the model of Flexible Graph Connectivity that we denote by FGC as a way to model network design problems where edges have non-uniform reliability. Boyd, Cheriyan, Haddadan and Ibrahimpur [5] introduced a generalization of FGC, called the (p, q)-Flexible Graph Connectivity problem, denoted (p, q)-FGC, where p is an integer denoting the connectivity requirement, and q is an integer denoting the robustness requirement. An instance of (p, q)-FGC consists of an undirected graph $G = (V, E)$, where E is partitioned into a set of safe edges $\mathcal{S}$ (edges that never fail) and a set of unsafe edges $\mathcal{U}$ (edges that may fail), and nonnegative edge-costs $c \in \mathbb{Q}^E_{\geq 0}$. A subset $F \subseteq E$ of edges is feasible for the (p, q)-FGC problem if for any set F' consisting of at most q unsafe edges, the subgraph $(V, F - F')$ remains p-edge connected. The objective is to find a feasible solution F that minimizes $c(F) = \sum_{e \in F} c_e$.

The following linear program gives a lower bound on the optimal value for (p, q)-FGC. Such LP relaxations are discussed in [5] and [8, Section 2]. To motivate the LP relaxation, consider an auxiliary capacitated graph that has the same set of nodes and the same set of edges as the graph of the (p, q)-FGC instance. Assign a capacity of $(p + q)$ to each safe edge and a capacity of p to each unsafe edge. Let $k = p(p + q)$ and view the capacitated graph as an instance of the Cap-k-ECSS problem. In general, observe that a feasible solution of the (p, q)-FGC instance corresponds to a feasible solution of the Cap-k-ECSS instance, but not vice-versa. (When either $p = 1$ or $q = 1$, then a feasible solution of the Cap-k-ECSS instance corresponds to a feasible solution of the (p, q)-FGC instance.) Each edge $e \in \mathcal{S} \cup \mathcal{U}$ has a variable x_e.

$$\min \sum_{e \in \mathcal{S} \cup \mathcal{U}} c_e x_e \tag{1}$$

$$\text{s.t.} \quad \sum_{e \in \mathcal{S} \cap \delta(S)} (p+q)\, x_e \quad + \sum_{e \in \mathcal{U} \cap \delta(S)} (p)\, x_e \geq p(p+q) \qquad \forall \quad \emptyset \subsetneq S \subsetneq V \tag{2}$$

$$0 \leq x_e \leq 1 \qquad \forall e \in \mathcal{S} \cup \mathcal{U} \tag{3}$$

Unfortunately, similarly to the LP relaxation for Cap-k-ECSS, the above LP relaxation has a large integrality ratio, even for the special case of $(1, q)$-FGC. Example 1 can be modified such that for $p = 1$ and $q > 0$, the above LP relaxation has integrality ratio $(q + 1)$.

1.4 Techniques and Tools

In this subsection, we describe three of the known tools that we apply *together* to obtain our main results. Each of these tools has been used on its own (without the other tools) to obtain improvements in the approximation ratio of the Cap-k-ECSS problem, but, in our opinion, by combining these tools in the right way, we obtain striking improvements in the approximation ratios for the Cap-k-ECSS problem and the $(1, q)$-FGC problem.

The first tool is the algorithmic use of the Knapsack-Cover Inequalities (KCI) for strengthening the LP relaxation of (IP: CapkECSS). This tool was introduced by Carr, Fleischer, Leung & Phillips [6]. The second tool is an $O(1)$ approximation algorithm for the so-called Cover Small Cuts problem. This tool was introduced by Bansal, Cheriyan, Grout & Ibrahimpur [4]. The third tool is Jain's iterative rounding method, [14].

1.4.1 Knapsack-Cover Inequalities (KCI) for Capacitated Network Design

Our approximation algorithms for both Cap-k-ECSS and $(1, q)$-FGC use the LP relaxations highlighted above as the starting point. However, to eliminate the integrality gap, we will strengthen these relaxations using knapsack-cover inequalities. We focus here on illustrating this tool for the Cap-k-ECSS problem, strengthening (IP: CapkECSS).

For any non-trivial cut $\delta(S)$ and a subset of the edges $A \subseteq E$, the following is a valid inequality for all integer solutions of (IP: CapkECSS).

$$\sum_{e \in E \cap \delta(S) - A} u_e(A, S) x_e \geq D(A, S),$$

where $D(A, S) = \max\{0, k - \sum_{e \in \delta(S) \cap A} u_e\}$ and $u_e(A, S) = \min\{u_e, D(A, S)\}$. (By plugging in $A = \emptyset$, we get the constraint $\sum_{e \in E \cap \delta(S)} u_e x_e \geq k$, which is a constraint of (IP: CapkECSS).) Intuitively, the added knapsack-cover inequalities for a cut $\delta(S)$ and edge-set A ensure that if a high-capacity edge is being used to cover $\delta(S)$ and, moreover, $\delta(S)$ is covered by some of the edges in A, then the capacity of the high-capacity edge is reduced to the remaining requirement, namely, $k - u(A \cap \delta(S))$. These inequalities "cut off" poor solutions x of the original LP relaxation (i.e., the one without KCI) such that some high-capacity edge f has a small fractional value for x_f (i.e., $0 < x_f \ll 1$). In particular, for Example 1 (at the end of section 1.2), consider the knapsack-cover inequality for the cut $\delta(S)$ where $S = \{u\}$ and $A = \{e_1\}$. We have $D(A, S) = \max\{0,\ k - (k-1)\} = 1$ and $u_{e_2}(A, S) = \min\{u_{e_2},\ D(A, S)\} = \min\{k, 1\} = 1$, hence, this inequality is $\sum_{e \in \delta(S) - A} u_e(A, S)\, x_e \geq D(A, S)$ which is $x_{e_2} \geq 1$. Clearly, the fractional solution $x_{e_1} = 1, x_{e_2} = 1/k$ is "cut off" by this inequality.

We add these inequalities to (IP: CapkECSS) to obtain the following LP relaxation of the Cap-k-ECSS problem.

$$\begin{aligned} \min \quad & \sum_{e \in E} c_e x_e & & \text{(KCLP: CapkECSS)} \\ \text{s.t.} \quad & \sum_{e \in E \cap \delta(S) - A} u_e(A, S) x_e \geq D(A, S) & & \forall\ \emptyset \subsetneq S \subsetneq V, A \subseteq E \\ & 0 \leq x_e \leq 1 & & \forall\ e \in E \end{aligned}$$

Observe that this LP has a number of constraints that is exponential in the size of the input instance (of Cap-k-ECSS). Moreover, we do not know of any polynomial-time separation oracle for the entire set of knapsack-cover inequalities. Nevertheless, by following the cut-and-round approach employed by Carr, Fleischer, Leung & Phillips [6], one can round this LP in polynomial time to an approximately optimal integer solution via the ellipsoid method by designing efficient subroutines. See sections 2, 3, for details.

Recently, Ibrahimpur & Vegh [13] have presented a polynomial-time separation subroutine for the knapsack-cover inequalities for the (p, q)-FGC problem.

1.4.2 The Cover Small Cuts problem

We follow the notation from [4, Section 1.3]. In an instance of the Cover Small Cuts problem, we are given an undirected capacitated graph $G = (V, E)$ with edge-capacities $u \in \mathbb{Q}^E_{\geq 0}$, a set of links $L \subseteq \binom{V}{2}$ with costs $c \in \mathbb{Q}^L_{\geq 0}$, and a threshold $\lambda \in \mathbb{Q}_{\geq 0}$. A subset $F \subseteq L$ of links is said to *cover* a node-set S if there exists a link $e \in F$ with exactly one end-node in S. The objective is to find a minimum-cost $F \subseteq L$ that covers each non-empty $S \subsetneq V$ with $u(\delta_E(S)) < \lambda$. Let $\mathcal{C} = \{\emptyset \neq S \subsetneq V : u(\delta_E(S)) < \lambda\}$. Then we have the following covering LP relaxation of the problem.

$$\begin{array}{llll}
\min & \displaystyle\sum_{f \in L} c_f x_f & & \text{(LP: Cover Small Cuts)} \\
\text{subject to:} & \displaystyle\sum_{f \in L \cap \delta(S)} x_f \geq 1 & \forall\, S \in \mathcal{C} & \\
& 0 \leq x_f \leq 1 & \forall\, f \in L &
\end{array}$$

The following result is due to Bansal, [3]; also, see Nutov [18].

▶ **Proposition 2.** *Given an instance of* Cover Small Cuts*, the WGMV primal-dual algorithm, [21], finds a feasible solution of cost* $\leq 6\, LP_{opt}$ *in polynomial time, where* LP_{opt} *denotes the optimal value of* (LP: Cover Small Cuts)*.*

In the 2-Cover Small Cuts problem, the inputs are the same as above, namely, $G = (V, E), u, L, c, \lambda$. A subset $F \subseteq L$ of links is said to *two-cover* a node-set S if $|F \cap \delta(S)| \geq 2$, that is, if there exist a pair of (distinct) links $e, e' \in F$ such that each of e and e' has exactly one end-node in S. The objective is to find a minimum-cost $F \subseteq L$ that two-covers each non-empty $S \subsetneq V$ with $u(\delta_E(S)) < \lambda$.

1.4.3 The f-connectivity problem and Jain's iterative rounding algorithm

In the context of approximation algorithms, several connectivity augmentation problems can be formulated in a general framework called f-connectivity. In this problem, we are given an undirected graph $G = (V, E)$ on n nodes with nonnegative costs $c \in \mathbb{Q}^E_{\geq 0}$ on the edges and a requirement function $f : 2^V \to \mathbb{Z}_{\geq 0}$ on subsets of nodes. The algorithmic goal is to find an edge-set $J \subseteq E$ with minimum cost $c(J) := \sum_{e \in J} c_e$ such that for all cuts $\delta(S)$, $S \subseteq V$, we have $|\delta(S) \cap J| \geq f(S)$. A function f is called *weakly supermodular* if $f(V) = 0$, and for all $A, B \subseteq V$, either $f(A)+f(B) \leq f(A-B)+f(B-A)$, or $f(A)+f(B) \leq f(A\cap B)+f(A\cup B)$.

Assuming that the function f is weakly supermodular, integral, and has a positive value for some $S \subset V$, Jain [14] presented a 2-approximation algorithm for the f-connectivity problem.

We will apply Jain's result (or its extension) in most of our algorithms.

1.5 Our results

Our first result is an $O(\log k/u_{min})$ approximation algorithm for the Cap-k-ECSS problem. Thus, we asymptotically improve upon the previous best approximation ratio of $\min(O(\log n), k, 2u_{max}, 6 \cdot \lceil k/u_{min} \rceil)$ whenever $\log(k/u_{min}) = o(\log n)$ and u_{max} is sufficiently large.

▶ **Theorem 3.** *There is a polynomial-time algorithm that, given an instance of Cap-k-ECSS, computes a vector x^* of cost at most* OPT *that possibly satisfies only a subset of the constraints of* (KCLP: CapkECSS)*, and rounds it to a feasible integer solution of cost at most $O(\log(k/u_{min})) \cdot$* OPT*.*

Before sketching our approximation algorithm for general instances of Cap-k-ECSS, let us focus on the special case of two capacities 1 and k. Let $E^{(1)}$ be the set of unit-capacity edges, and let $E^{(k)}$ be the set of edges of capacity k. For expository reasons (and glossing over some critical points), let us assume that we can find, in polynomial time, an optimal solution x^* of (KCLP: CapkECSS), the LP with KCI. Thus, we have $c(x^*) = \mathrm{LP}_{\mathrm{opt}}$, where $c(x^*) = \sum_e c_e x_e$ is the cost of x^*. Let $E^{(1)}_{high} = \{e \in E^{(1)} : x^*_e \geq 1/2\}$; this is the set of unit-capacity edges that are assigned values of 1/2 or more by the LP solution x^*. Let $E^{(1)}_{low} = E^{(1)} - E^{(1)}_{high}$ (the set of unit-capacity edges with x^*-values less than 1/2). We call a non-trivial cut $\delta(S)$ a small cut if

$$|E^{(1)}_{high} \cap \delta(S)| + \sum_{e \in E^{(1)}_{low} \cap \delta(S)} 2x^*_e < k; \qquad \text{(definition of small cut)}$$

in other words, a non-trivial cut is defined to be small if the fractional capacity contributed by the unit-capacity edges is less than the requirement of k even after rounding up edges in $E^{(1)}_{high}$ to have x-value one and scaling up the x-value of each edge in $E^{(1)}_{low}$ by factor 2. We construct an instance of Cover Small Cuts with small cuts as defined above, and we define the link-set of this instance to be $E^{(k)}$. To handle the small cuts, we apply the knapsack-cover inequalities to show that the solution x^* restricted to edges of capacity k and scaled up by a factor of 2 (i.e., $2x^*_{E^{(k)}}$) constitutes a feasible solution to the LP relaxation of the Cover Small Cuts instance. This is the critical point where our algorithm and analysis relies on the knapsack-cover inequalities. We can thus pick a set $E^{(k)}_{picked}$ of edges of capacity k by applying the 6-approximation algorithm of [3, 4, 21] to this instance of the Cover Small Cuts problem; note that the cost of $E^{(k)}_{picked}$ is at most 6 times the cost of $2x^*_{E^{(k)}}$. After this step, we contract the connected components formed by the edges in $E^{(k)}_{picked}$, and get a new instance that does not have any small cuts. Since there are no small cuts, every non-trivial cut $\delta(S)$ satisfies the inequality $|E^{(1)}_{high} \cap \delta(S)| + \sum_{e \in E^{(1)}_{low} \cap \delta(S)} 2x^*_e \geq k$. Therefore, we get a feasible solution for the LP relaxation of the f-connectivity problem where $f(S) = k$ for every non-empty set $S \subsetneq V$, by picking all edges in $E^{(1)}_{high}$ and scaling up x^* restricted to $E^{(1)}_{low}$ by a factor of 2. Since f is weakly supermodular, we can apply Jain's iterative rounding method [14] to solve this f-connectivity problem and obtain a 2-approximate solution, giving us an integral solution whose cost is at most 4 times the cost of x^* restricted to unit-capacity edges. Thus, we get an integral solution whose total cost is at most $6 \cdot 2 \cdot \mathrm{LP}_{\mathrm{opt}}$.

Let us recap the new algorithmic lever we deployed above. We partitioned the edges according to their capacities into the set of low-capacity edges and the set of high-capacity edges; let E^{big} denote the latter set. Then we defined an instance of Cover Small Cuts whose small cuts were defined using the low-capacity edges and rounding up (and/or scaling up) the fractional capacity contribution $u_e x^*_e$ of each of these edges e; moreover, we defined the links (of the Cover Small Cuts instance) to be the high-capacity edges. The small cuts identified above are precisely the cuts where the LP solution x^* has not invested sufficient fractional capacity in the low-capacity edges to cover the cuts. For these cuts, the knapsack-cover inequalities ensured that x^* restricted to the high-capacity edges and scaled up by a small constant, say, η, (i.e., $\eta x^*_{E^{big}}$) forms a feasible solution to the LP relaxation of the Cover Small Cuts instance. Based on this, we applied the 6-approximation algorithm of [3, 4, 21] to find a set of high-capacity edges of cost $\leq 6\,\eta c(x^*_{E^{big}})$ that covers all the small cuts.

Now, let us sketch an extension of the above method that gives an $O(\log k)$-approximation algorithm for Cap-k-ECSS. As above, let us assume that we can find, in polynomial time, an optimal solution x^* of (KCLP: CapkECSS), the LP with KCI. Thus, we have $c(x^*) = \mathrm{LP}_{\mathrm{opt}}$. Throughout the execution of the algorithm, we maintain a set of edges E_{cur} acting as our current solution. We begin with $E_{cur} = \{e \in E : x^*_e \geq 1/2\}$. Define $T = \lceil \log k \rceil$ and for $j = 1, 2, \ldots, T$, define $E_j = \{e \in E : x^*_e < 1/2 \text{ and } u_e \leq 2^j\}$. Ideally, we wish to apply T iterations, and, in each iteration, we wish to apply the above algorithmic lever $O(1)$ times. In more detail, in the ith-iteration, we could take the edges of capacity $\leq 2^{T-i}$ to be the low-capacity edges and the edges of capacity $> 2^{T-i}$ to be the high-capacity edges; that is, E_{T-i} is the set of low-capacity edges and $E - E_{T-i}$ is the set of high-capacity edges. Then, we could define an instance of Cover Small Cuts where we could define the small-cuts to be the non-trivial cuts $\delta(S)$ such that $(\sum_{e \in E_{cur} \cap \delta(S)} u_e) + (\sum_{e \in E_{T-i} \cap \delta(S)} 2u_e x^*_e) < k$, and we could take the link-set (for the Cover Small Cuts instance) to be the set of high-capacity edges, $E - E_{T-i}$. Although we can apply the algorithmic lever and find an integral cover of the small cuts using the edges in $E - E_{T-i}$ such that the cost of the integral cover is $O(1)c(x^*_{E-E_{T-i}})$, we run into a difficulty. The capacity of an edge in $E - E_{T-i}$ could be as small as $\Theta(2^{T-i})$, hence, by adding just one of these edges to a small cut $\delta(S)$ we cannot guarantee that the capacity of $\delta(S)$ is augmented to become $\geq k$. Thus, our algorithm and analysis, presented in section 3, have further steps; a deeper analysis is required to show that our algorithm finds an integral solution of cost $O(\log k) \cdot \mathrm{LP}_{\mathrm{opt}}$. Moreover, we improve the approximation ratio from $O(\log k)$ to $O(\log(k/u_{min}))$.

Recall that the $(1, q)$-FGC problem can be formulated as a special case of Cap-k-ECSS with two capacities (the unsafe edges have unit capacity and the safe edges have capacity $k = (q + 1)$). Clearly, the $O(1)$-approximation algorithm sketched above applies to the $(1, q)$-FGC problem. Our next result improves the approximation ratio to 8, thus improving on the previous best $(q + 1)$-approximation algorithm for $(1, q)$-FGC, [5].

▶ **Theorem 4.** *There is a polynomial-time algorithm that, given an instance of* $(1, q)$-FGC*, computes a vector* x^* *of cost at most* OPT *that possibly satisfies only a subset of the constraints of* (KCLP:$(1, q)$-FGC)*, and rounds it to a feasible integer solution of cost at most* 8 OPT.

Moreover, we present $O(1)$-approximate reductions between the $(2, q)$-FGC problem and the 2-Cover Small Cuts problem. The following two results summarize the two reductions; the details are omitted due to space constraints; see the arXiv version of this paper.

▶ **Theorem 5.** *Suppose an LP relative ρ-approximation algorithm for* 2-Cover Small Cuts *that runs in polynomial time is available. Then there is an algorithm for* $(2, q)$-FGC *that runs in polynomial time and returns a feasible (integer) solution of cost* $\leq (4(\rho + 1) + 8)$ OPT.

▶ **Theorem 6.** *Suppose a ρ'-approximation algorithm for* $(2, q)$-FGC *that runs in polynomial time is available. Then there is an algorithm for* 2-Cover Small Cuts *that runs in polynomial time and returns a feasible (integer) solution of cost* $\leq (\rho' + 2)$ OPT.

1.6 Organization of the Paper

For the sake of readability and accessibility, we present our 8-approximation algorithm for $(1, q)$-FGC in the next section, and we defer the presentation of our improved approximation algorithm for Cap-k-ECSS to section 3. Due to space constraints, our remaining results are omitted but are presented in the arXiv version of this paper.

2 An 8-Approximation Algorithm for $(1, q)$-FGC

This section presents a 8-approximation algorithm for the $(1, q)$-FGC problem. For the convenience of the reader, the presentation in this section is independent of the rest of the paper. For a graph $G = (V, E)$ and $S \subseteq V$, the cut $\delta(S)$ refers to the set of edges that have exactly one end-node in S; $\delta(S)$ is called a non-trivial cut if $\emptyset \neq S \subsetneq V$. (Whenever we use the term "cut $\delta(S)$" we mean that S is a subset of $V(G)$.) We use the term *small cut* to mean a non-trivial cut $\delta(S)$ with capacity below a specified threshold-value, say, λ.

Our starting point is the natural LP relaxation that follows from taking a capacitated network design view of the problem where each unsafe edge $e \in \mathcal{U}$ has capacity $u_e = 1$, and each safe edge $e \in \mathcal{S}$ has capacity $u_e = (q + 1)$. The natural LP relaxation then seeks to minimize the total cost of edges subject to the constraint that $\forall\ \emptyset \subsetneq S \subsetneq V$, we have $\sum_{e \in \mathcal{S} \cap \delta(S)} (q + 1)x_e + \sum_{e \in \mathcal{U} \cap \delta(S)} x_e \geq (q + 1)$.

It is easy to see that any feasible solution to this LP with zero/one values is a valid solution to a given instance of $(1, q)$-FGC, and vice versa. However, it is also easy to show that this LP has integrality ratio $(q+1)$ by adapting Example 1 given at the end of section 1.2.

To get around this obstruction, we strengthen the LP relaxation of $(1, q)$-FGC using the knapsack-cover inequalities to obtain the following stronger LP. Intuitively, the added knapsack-cover inequalities for a cut $\delta(S)$ ensure that if a safe edge is being used to cover $\delta(S)$ and, moreover, $\delta(S)$ is partly covered by unsafe edges, say, by ℓ of them, then the capacity of the safe edge is reduced to $(q + 1) - \ell$.

$$\begin{aligned}
\min \quad & \sum_{e \in E} c_e x_e & & \text{(KCLP:}(1,q)\text{-FGC)} \\
\text{s.t.} \quad & \sum_{e \in \mathcal{S} \cap \delta(S)} (q + 1)x_e + \sum_{e \in \mathcal{U} \cap \delta(S)} x_e \geq (q + 1) & & \forall\ \emptyset \subsetneq S \subsetneq V \\
& \sum_{e \in E \cap \delta(S) - A} u_e(A, S)x_e \geq D(A, S) & & \forall\ \emptyset \subsetneq S \subsetneq V, A \subseteq E \\
& 0 \leq x_e \leq 1 & & \forall\ e \in E,
\end{aligned}$$

where $D(A, S) = \max\{0,\ (q+1) - \sum\{u_e \mid e \in A \cap \delta(S)\}\}$, and $u_e(A, S) = \min\{u_e,\ D(A, S)\}$. We mentioned above that Ibrahimpur & Vegh [13] have presented a polynomial-time separation subroutine for the knapsack-cover inequalities for the (p, q)-FGC problem. Instead of using their method, we follow the algorithmic scheme of [6, 7], because we will use a similar algorithmic scheme in section 3 (to the best of our knowledge, there is no polynomial-time separation subroutine for the knapsack-cover inequalities in section 3). Our plan is to identify a subset of unsafe edges A and a collection of sets $\mathcal{C}$ (in polynomial time) such that, as long as the knapsack-cover inequalities hold for A and all $S \in \mathcal{C}$, we will be able to execute our rounding algorithm. Using this, we will design a polynomial-time approximation algorithm for $(1, q)$-FGC.

In this section, let α denote the best approximation ratio known for the Cover Small Cuts problem. We will design an $(\alpha + 2)$-approximation algorithm for the $(1, q)$-FGC problem. As of now, we have $(\alpha + 2) = 8$, since $\alpha = 6$, due to [3, 4, 21]; also see [18].

Let LP_{opt} denote the optimal value of (KCLP:$(1, q)$-FGC), the LP with the knapsack-cover inequalities. Using binary search for LP_{opt} together with the ellipsoid algorithm and our polynomial-time subroutines, we will find a vector x^* with cost $c(x^*) = \sum_e c_e x_e$ such that $c(x^*) \leq \text{LP}_{\text{opt}}$ and x^* satisfies the constraints of (KCLP:$(1, q)$-FGC) specified in the next lemma (though x^* could violate other constraints of (KCLP:$(1, q)$-FGC)). Then, we will "round" x^* to an integer solution of cost $\leq (\alpha + 2)c(x^*)$. Thus, we will find an integer solution of cost $\leq (\alpha+2)\mathrm{LP}_{\mathrm{opt}}$ (even though we will not compute the precise value of $\mathrm{LP}_{\mathrm{opt}}$). We discuss our overall algorithm and the outer loop of binary search at the end of this section.

▶ **Lemma 7.** *There is a polynomial-time algorithm that given a vector x^* (that is a candidate solution of* (KCLP:$(1,q)$-FGC)*) and a value z either finds a violated constraint of the LP or else verifies that $c(x^*) \leq z$ and, moreover, x^* satisfies the following two properties:*

(P1) $\sum_{e\in \mathcal{S}\cap\delta(S)}(q+1)x^*_e + \sum_{e\in\mathcal{U}\cap\delta(S)} x^*_e \geq (q+1) \qquad \forall\, \emptyset \neq S \subsetneq V.$

(P2) *Let $A = \{e \in \mathcal{U} \mid x^*_e \geq \dfrac{2}{(\alpha+2)}\}$. For any non-empty $S \subsetneq V$, if $\sum_{e\in\mathcal{S}\cap\delta(S)}(q+1)x^*_e + \sum_{e\in\mathcal{U}\cap\delta(S)} x^*_e \leq 2(q+1)$, then $\displaystyle\sum_{e\in E\cap\delta(S)-A} u_e(A,S)x_e \geq D(A,S)$.*

Proof. Essentially, we will describe a polynomial-time separation oracle that identifies any violations of properties (P1) and (P2). Given a vector x^*, we first check that $\sum_{e\in E} c_e x^*_e \leq z$. If not, we return this as a violated constraint. Otherwise, let $\hat{G} = (\hat{V}, \hat{E})$ be the capacitated graph where $\hat{V} = V, \hat{E} = E$, and each edge $e \in \hat{E}$ is assigned a capacity of $u_e x^*_e$. We can now check that the capacity of a minimum-cut of $\hat{G}$ is at least $(q+1)$ using a polynomial-time global minimum-cut algorithm [19]. If not, we return a global minimum cut in $\hat{G}$ as a violated constraint. Otherwise, we know that (P1) is satisfied, and we proceed to verify (P2) with respect to the set $A = \{e \in \mathcal{U} \mid x^*_e \geq \dfrac{2}{(\alpha+2)}\}$.

By Karger's result [15], there are at most $O(n^4)$ cuts of capacity at most $2(q+1)$ (i.e., at most twice the capacity of a minimum cut), and, moreover, we can enumerate all such cuts of $\hat{G}$ in polynomial time [16]. By iterating over each of the $O(n^4)$ cuts, we can now verify, in polynomial time, that the knapsack-cover inequalities are satisfied w.r.t. the set A. If not, we have found a violated constraint. ◀

The next corollary follows from the above lemma and the well-known fact that the ellipsoid algorithm terminates after $n^{O(1)}$ iterations of feasibility verification [10]. The outer loop of our algorithm runs a binary search for $\mathrm{LP}_{\mathrm{opt}}$ (see the discussion at the end of this section).

▶ **Corollary 8.** *There is a polynomial-time algorithm that computes a vector x^* of cost $c(x^*) \leq LP_{opt}$ such that x^* satisfies properties (P1) and (P2); possibly, x^* violates some of the other constraints of* (KCLP:$(1,q)$-FGC).

The Rounding Algorithm. Given a value z and a vector x^* of cost $c(x^*) \leq z$ that satisfies properties (P1) and (P2) (see Lemma 7), Algorithm 1, presented below, rounds it to an integer solution of cost at most $(\alpha+2)z$. Below, we describe the main idea of our rounding scheme.

Recall that our LP relaxation assigns capacity $u_e = 1$ for each unsafe edge $e \in \mathcal{U}$ and capacity $u_e = (q+1)$ to each safe edge $e \in \mathcal{S}$. We call a non-trivial cut $\delta(S)$ a small cut if $|\mathcal{U}_1 \cap \delta(S)| + \sum_{e\in\mathcal{U}_2\cap\delta(S)} \frac{(\alpha+2)}{2} x^*_e < (q+1)$ where $\mathcal{U}_1 = \{e \in \mathcal{U} : x^*_e \geq \dfrac{2}{(\alpha+2)}\}$, and $\mathcal{U}_2 = \mathcal{U} - \mathcal{U}_1$. To handle the presence of small cuts, we first construct an instance of Cover Small Cuts with the specified small cuts and with link-set the set of safe edges; then, via Lemma 9, we show that the vector x^* restricted to safe edges and scaled up by a factor of $\dfrac{(\alpha+2)}{\alpha}$ (i.e., $\frac{(\alpha+2)}{\alpha} x^*_{\mathcal{S}}$) constitutes a feasible solution to the LP relaxation of the Cover Small Cuts instance. We can thus pick a set $\mathcal{S}_1$ of safe edges by applying the α-approximation algorithm to the Cover Small Cuts instance; note that the cost of

$\mathcal{S}_1$ is at most α times the cost of $\frac{(\alpha+2)}{\alpha}x^*_{\mathcal{S}}$. After this step, we contract the connected components formed by the edges in $\mathcal{S}_1$, and get a new instance that does not have any small cuts. Since there are no small cuts, every non-trivial cut $\delta(S)$ satisfies the inequality $|\mathcal{U}_1 \cap \delta(S)| + \sum_{e\in\mathcal{U}_2\cap\delta(S)} \frac{(\alpha+2)}{2}x^*_e \geq (q+1)$. Therefore, we get a feasible solution for the LP relaxation of the f-connectivity problem where $f(S) = q+1$ for every non-empty set $S \subsetneq V$, by picking all edges in $\mathcal{U}_1$ and scaling up x^* restricted to $\mathcal{U}_2$ by a factor of $(\alpha+2)/2$. Since f is weakly supermodular, we can apply Jain's iterative rounding method [14] to solve this f-connectivity problem and obtain a 2-approximate solution, giving us an integral solution whose cost is at most $(\alpha+2)$ times the cost of $x^*_{\mathcal{U}}$ (see Lemma 10). Thus, we get an integral solution whose total cost of safe and unsafe edges is at most $(\alpha+2)$ times $\mathrm{LP}_{\mathrm{opt}}$.

Algorithm 1 $(\alpha+2)$-approximate solution to $(1,q)$-FGC.

```
Require: Graph $G = (V,E)$ where $E$ is partitioned into $\mathcal{S} \bigcup \mathcal{U}$, with edge costs $\{c_e\}_{e\in E}$.
 A solution $x^*$ to (KCLP:$(1,q)$-FGC) as promised by Lemma 7.
1. $\mathcal{U}_1 \leftarrow \{e \in \mathcal{U} : x^*_e \geq \frac{2}{(\alpha+2)}\}$, and $\mathcal{U}_2 \leftarrow \mathcal{U} - \mathcal{U}_1$.
2. (a) Apply the approximation algorithm for Cover Small Cuts on the instance with
       $G' = (V, E' = \mathcal{U})$, where each edge of $\mathcal{U}_1$ is given unit capacity, each edge $e \in \mathcal{U}_2$ is
       given capacity $\frac{(\alpha+2)}{2}x^*_e$, and with link set $\mathcal{S}$. Define the threshold $\lambda$ (for small cuts)
       to be $(q+1)$. Thus, we have:
           $|\mathcal{U}_1 \cap \delta(S)| + \sum_{e\in\mathcal{U}_2\cap\delta(S)} \frac{(\alpha+2)}{2}x^*_e < (q+1)$      (definition of small cuts)
   (b) Let the output of the call in step (a) be denoted $\mathcal{S}_1$.
3. Construct a graph $G'' = (V, \mathcal{U}_2)$ where each edge $e \in \mathcal{U}_2$ has cost $c_e$. Define the
   requirement of a non-trivial cut $\delta(S)$ to be
       $f(S) = \max\{0,\ \ (q+1) - ((q+1)|\delta(S) \cap \mathcal{S}_1| + |\delta(S) \cap \mathcal{U}_1|)\}$.
   This function $f$ is weakly supermodular, so a 2-approximate solution for this instance of
   $f$-connectivity can be computed using Jain's iterative rounding method [14].
4. Return the union of the set of unsafe edges picked by the previous step (via the iterative
   rounding algorithm) and $\mathcal{S}_1 \cup \mathcal{U}_1$.
```

The next two lemmas formalize the key properties of the solution x^* that are used in the rounding scheme above, allowing us to show that it returns a feasible integral solution of cost at most $(\alpha+2)c(x^*)$.

▶ **Lemma 9.** *In step 2 of Algorithm 1, a feasible fractional solution to the* Cover Small Cuts *instance is given by* $\hat{x}_e = \min\{1, \frac{(\alpha+2)}{\alpha}x^*_e\}$ *for* $e \in \mathcal{S}$.

Proof. Consider any small cut $\delta(S)$. We will establish the lemma by considering two cases.

First, consider the case that $\sum_{e\in\mathcal{S}\cap\delta(S)}(q+1)x^*_e + \sum_{e\in\mathcal{U}\cap\delta(S)} x^*_e > 2(q+1)$. Since $\delta(S)$ is a small cut, we have

$$\sum_{e\in\mathcal{U}_1\cap\delta(S)} x^*_e + \sum_{e\in\mathcal{U}_2\cap\delta(S)} x^*_e \leq |\mathcal{U}_1 \cap \delta(S)| + \sum_{e\in\mathcal{U}_2\cap\delta(S)} \frac{(\alpha+2)}{2}x^*_e < (q+1).$$

Since $\sum_{e\in\mathcal{S}\cap\delta(S)}(q+1)x^*_e + \sum_{e\in\mathcal{U}\cap\delta(S)} x^*_e > 2(q+1)$, it follows that $\sum_{e\in\mathcal{S}\cap\delta(S)}(q+1)x^*_e \geq (q+1)$, hence, $\sum_{e\in\mathcal{S}\cap\delta(S)} x^*_e \geq 1$. Thus, $\sum_{e\in\mathcal{S}\cap\delta(S)} \hat{x}_e \geq 1$.

Now suppose that $\sum_{e\in\mathcal{S}\cap\delta(S)}(q+1)x^*_e + \sum_{e\in\mathcal{U}\cap\delta(S)} x^*_e \le 2(q+1)$. Then, by (P2), the cut $\delta(S)$ satisfies the knapsack-cover inequality w.r.t. the set $A = \mathcal{U}_1$:

$$\sum_{e\in\mathcal{S}\cap\delta(S)} u_e(A,S)x_e + \sum_{e\in\mathcal{U}_2\cap\delta(S)} u_e(A,S)x_e \ge D(A,S),$$

where $D(A,S) = \max\{0,\ (q+1) - \sum\{u_e \mid e \in A\cap\delta(S)\}\} = \max\{0,\ (q+1) - |\mathcal{U}_1\cap\delta(S)|\}$, and $u_e(A,S) = \min\{u_e,\ D(A,S)\}$.

Moreover, since $\delta(S)$ is a small cut, we have $|\mathcal{U}_1\cap\delta(S)| + \displaystyle\sum_{e\in\mathcal{U}_2\cap\delta(S)} \frac{(\alpha+2)}{2}x^*_e < (q+1)$. We rewrite this inequality as $\displaystyle\sum_{e\in\mathcal{U}_2\cap\delta(S)} \frac{(\alpha+2)}{2}u_e(A,S)x^*_e < D(A,S)$, which is the same as $\displaystyle\sum_{e\in\mathcal{U}_2\cap\delta(S)} u_e(A,S)x^*_e < \frac{2}{(\alpha+2)}D(A,S)$. Thus, by the knapsack-cover inequality, we have $\displaystyle\sum_{e\in\mathcal{S}\cap\delta(S)} u_e(A,S)x^*_e \ge \frac{\alpha}{(\alpha+2)}D(A,S)$. By definition, $u_e(A,S) \le D(A,S)$, hence, we have $\displaystyle\sum_{e\in\mathcal{S}\cap\delta(S)} \frac{(\alpha+2)}{\alpha}D(A,S)x^*_e \ge D(A,S)$. Therefore, $\sum_{e\in\mathcal{S}\cap\delta(S)} \hat{x}_e \ge 1$, completing the proof. ◀

▶ **Lemma 10.** *In step 3 of Algorithm 1, a feasible fractional solution to the f-connectivity problem is given by $x'_e = \frac{(\alpha+2)}{2}x^*_e$ for $e \in \mathcal{U}_2$.*

Proof. By way of contradiction, suppose that the claim does not hold. Then for some non-trivial cut $\delta(S)$, we have $(q+1)|\mathcal{S}_1\cap\delta(S)| + |\mathcal{U}_1\cap\delta(S)| + \displaystyle\sum_{e\in\mathcal{U}_2\cap\delta(S)} \frac{(\alpha+2)}{2}x^*_e < (q+1)$. This implies that the cut $\delta(S)$ is a small cut in step 2 of Algorithm 1. Hence, step 2 ensures (via Cover Small Cuts) that $|\mathcal{S}_1\cap\delta(S)| \ge 1$ and so $(q+1)|\mathcal{S}_1\cap\delta(S)| \ge (q+1)$. This is a contradiction. ◀

The output of Algorithm 1 is feasible for the $(1,q)$-FGC problem by Lemmas 9, 10. The cost of the edges in $\mathcal{U}_1$ is $\le \frac{(\alpha+2)}{2}\displaystyle\sum_{e\in\mathcal{U}_1} c_e x^*_e$ since $x^*_e \ge \frac{2}{(\alpha+2)}$ for each edge e in $\mathcal{U}_1$. Additionally, the cost of the edges in $\mathcal{S}_1$ is $\le (\alpha+2)\sum_{e\in\mathcal{S}_1} c_e x^*_e$ by Lemma 9 and our definition of α. Lastly, the cost of the edges returned by Jain's iterative rounding algorithm (in step 3 of Algorithm 1) is at most $\displaystyle\sum_{e\in\mathcal{U}_2} (2)((\alpha+2)/2)c_e x^*_e = (\alpha+2)\sum_{e\in\mathcal{U}_2} c_e x^*_e$, by Lemma 10. Therefore, the cost of the solution returned by Algorithm 1 is at most $(\alpha+2)c(x^*)$.

The outer loop of our algorithm runs a binary search for $\mathrm{LP}_{\mathrm{opt}}$, but note that we are not using a "true" polynomial-time separation subroutine. Given a vector x^* (a candidate solution to (KCLP:$(1,q)$-FGC)), our subroutine either finds that x^* violates one of the constraints specified in Lemma 7 or else it rounds x^* to an integer solution of cost $\le (\alpha+2)c(x^*)$, where $c(x^*) = \sum_e c_e x^*_e$. The binary search for $\mathrm{LP}_{\mathrm{opt}}$ starts with the interval $[0, c(E)]$, where $c(E) = \sum_e c_e$. Assume that the instance has a feasible integer solution, let OPT denote the cost of an optimal integer solution, and assume that $0 < \text{OPT} \le c(E)$.

In an arbitrary iteration, the binary search calls the ellipsoid algorithm with the additional constraint $\sum_e c_e x_e \le z$, where the current interval is $[\ell_c, h_c]$ and $z = \frac{\ell_c + h_c}{2}$. (The binary search maintains the invariant: $\mathrm{LP}_{\mathrm{opt}} > \ell_c$ and there exists an integer solution of cost $\le (\alpha+2)h_c$.) The ellipsoid algorithm calls our subroutine one or more times, and either

(1) reports that the LP (with the additional constraint) is infeasible or else (2) it finds a vector x^* with $c(x^*) \leq z$ and an integer solution of cost $\leq (\alpha+2)c(x^*)$. The binary search continues as usual, that is, in case (1) it replaces the current interval $[\ell_c, h_c]$ by the upper half-interval $[\frac{\ell_c+h_c}{2}, h_c]$, and in case (2) it replaces the current interval by the lower half-interval $[\ell_c, \frac{\ell_c+h_c}{2}]$. The binary search terminates when the current interval $[\ell_{final}, h_{final}]$ is sufficiently small. Clearly, the LP with the additional constraint $\sum_e c_e x_e \leq \ell_{final}$ is infeasible, and the algorithm found an integer solution of cost $\leq (\alpha+2)h_{final}$. Hence, $\mathrm{LP_{opt}} > \ell_{final}$ and the last integer solution found by the algorithm has cost $\leq (\alpha+2)(\mathrm{LP_{opt}} + (h_{final} - \ell_{final}))$.

▶ **Theorem 4.** *There is a polynomial-time algorithm that, given an instance of* $(1, q)$-FGC*, computes a vector* x^* *of cost at most* OPT *that possibly satisfies only a subset of the constraints of* (KCLP:$(1, q)$-FGC)*, and rounds it to a feasible integer solution of cost at most* $8\,$OPT*.*

3 An $O(\log \frac{k}{u_{min}})$-Approximation Algorithm for Cap-k-ECSS

In this section, we present an $O(\log(k/u_{min}))$-approximation algorithm for Cap-k-ECSS that runs in polynomial time assuming $k/u_{min} \leq |V(G)| = n$. Note that when $k/u_{min} > n$, then the previously known approximation algorithm of [7] for Cap-k-ECSS achieves an approximation ratio of $O(\log n) \leq O(\log(k/u_{min}))$.

Let us recall a few terms & notation from previous sections. For a graph $G = (V, E)$ and $S \subseteq V$, the cut $\delta(S)$ refers to the set of edges that have exactly one end-node in S; $\delta(S)$ is called a non-trivial cut if $\emptyset \neq S \subsetneq V$. (Whenever we use the term "cut $\delta(S)$" we mean that S is a subset of $V(G)$.) We use the term *small cut* to mean a non-trivial cut $\delta(S)$ with capacity below a specified threshold-value, say, λ.

Let $\mathrm{LP_{opt}}$ denote the optimal value of (KCLP: CapkECSS), the LP with the knapsack-cover inequalities. Similarly to section 2, we use binary search for $\mathrm{LP_{opt}}$ together with the ellipsoid algorithm and our polynomial-time subroutines to find a vector x^* with cost $c(x^*) = \sum_e c_e x_e$ such that $c(x^*) \leq \mathrm{LP_{opt}}$ and x^* satisfies the constraints of (KCLP: CapkECSS) specified in the proof of Lemma 11 (though x^* could violate other constraints of (KCLP: CapkECSS)). Then, we will round x^* to an integer solution of cost $\leq O(\log(k/u_{min}))c(x^*)$. Thus, we will find an integer solution of cost $\leq O(\log(k/u_{min}))\mathrm{LP_{opt}}$, even though we will not compute the precise value of $\mathrm{LP_{opt}}$. At the end of this section, the proof of Lemma 11 discusses our overall algorithm in more detail.

Assumption. In what follows, assume that the vector x^* satisfies all the constraints of (KCLP: CapkECSS). In section 3.6 below, we explain that we can easily remove this assumption.

Throughout the execution of the rounding algorithm, we will maintain a set of edges E_{cur} acting as our current solution. We begin with $E_{cur} = \{e \in E : x^*_e \geq 1/2\}$. Define $T = \lceil \log k \rceil$ and for $j = 1, 2, \ldots, T$, define $E_j = \{e \in E : x^*_e < 1/2 \text{ and } u_e \leq 2^j\}$; thus, the edge-sets $E_T - E_{T-1}, E_{T-1} - E_{T-2}, \ldots, E_3 - E_2, E_2 - E_1, E_1$ form a partition of the edges in $E - E_{cur}$ into T buckets based on the capacities; let us call the set $E_{T-i+1} - E_{T-i}$ the i-th bucket (and E_1 is the T-th bucket). See Figure 1 for an illustration.

Our algorithm will have T iterations and each iteration (except for the first and the last) will have two phases. During phase 1 of iteration i, we will round some of the edges in the i-th bucket, i.e., some of the edges in the set $E_{T-i+1} - E_{T-i}$. Note that an edge e in the i-th bucket has capacity $2^{T-i} < u_e \leq 2^{T-i+1}$. Informally speaking, in phase 1, we want to augment cuts of very small capacity with edges of capacity $\approx 2^{T-i}$, and, in

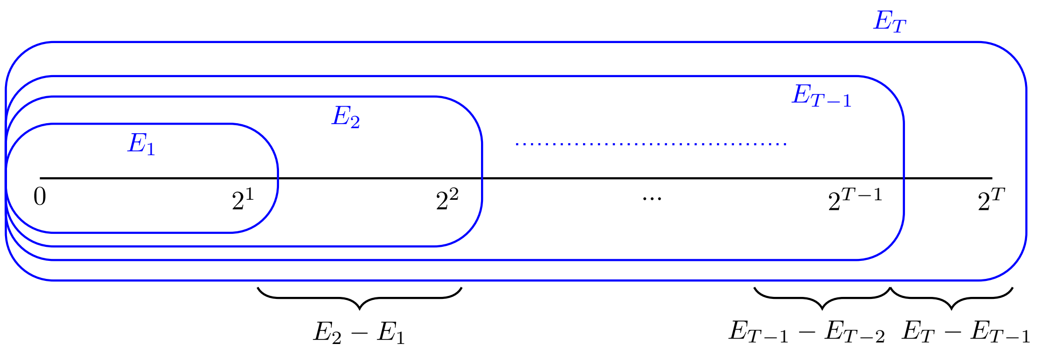

Figure 1 Illustration of buckets. Note that $T = \lceil \log k \rceil$.

general, we need $\Theta(k/2^{T-i})$ rounds of augmentation to achieve capacity k; thus, phase 1 has $\Theta(k/2^{T-i})$ sub-iterations. During phase 2 of iteration i, we will round some of the edges in $E - (E_{T-i} \cup E_{cur})$.

Next, we present pseudo-code for the rounding algorithm, followed by explanation and analysis of the main steps.

Algorithm 2 $O(\log(k/u_{min}))$-approximate solution to Cap-k-ECSS.

```
Require: Graph $G = (V, E)$ with capacities $\{u_e\}_{e \in E}$ and costs $\{c_e\}_{e \in E}$. A vector $x^*$
  satisfying some constraints of (KCLP: CapkECSS), set $E_{cur} = \{e \in E : x^*_e \geq 1/2\}$, and
  sets $E_j \subseteq E, j = 1, \ldots, \lceil \log k \rceil$ as defined above.
1. Iteration 1:
   (a) Let $\mathcal{C} = \{\emptyset \neq S \subsetneq V \ : \ \sum_{e \in E_{cur} \cap \delta(S)} u_e + \sum_{e \in E_{T-1} \cap \delta(S)} 2u_e x^*_e < k\}$.
   (b) Apply the approximation algorithm for Cover Small Cuts to select edges from
       $E - E_{T-1} - E_{cur}$ to cover the cuts in $\mathcal{C}$. Add the selected edges to $E_{cur}$.
   (c) Repeat (a), (b) once.
2. For $i = 2, \ldots, T - 1$, Iteration $i$:
   (a) For $\ell = 1, \ldots, \lfloor (k - 2^{T-i+1})/2^{T-i} \rfloor$:
       (i) Let $\mathcal{C} = \{\emptyset \neq S \subsetneq V \ : \ \sum_{e \in E_{cur} \cap \delta(S)} u_e + \sum_{e \in E_{T-i} \cap \delta(S)} 2u_e x^*_e < \ell \, 2^{T-i}\}$.
       (ii) Apply the approximation algorithm for Cover Small Cuts to select edges from
            $E_{T-i+1} - E_{T-i} - E_{cur}$ to cover the cuts in $\mathcal{C}$. Add the selected edges to $E_{cur}$.
   (b) Let $\mathcal{C} = \{\emptyset \neq S \subsetneq V \ : \ \sum_{e \in E_{cur} \cap \delta(S)} u_e + \sum_{e \in E_{T-i} \cap \delta(S)} 2u_e x^*_e < k\}$.
   (c) Apply the approximation algorithm for Cover Small Cuts to selected edges from
       $E - E_{T-i} - E_{cur}$ to cover the cuts in $\mathcal{C}$. Add the selected edges to $E_{cur}$.
   (d) Repeat (b), (c) **two** additional times.
3. Iteration $T$:
   (a) At this point, we have that $\sum_{e \in E_{cur} \cap \delta(S)} u_e + \sum_{e \in E_1 \cap \delta(S)} 2u_e x^*_e \geq k$ for all
       $S \subsetneq V, S \neq \emptyset$. Apply Jain's iterative rounding method to round the ($x$ variables of
       the) edges in $E_1$ to an integer solution $E^*_1$, such that $E_{cur} \cup E^*_1$ is a feasible solution
       to Cap-$k$-ECSS.
4. Return $E_{cur} \cup E^*_1$.
```

For every non-trivial cut $\delta(S)$, we will maintain the following invariants for all iterations $i = 2, \ldots, (T - 1)$ (i.e., except the first and the last iteration):

(1) At the beginning of iteration i, $E_{cur} \cap E_{T-i+1} = \emptyset$ and

$$\sum_{e \in E_{cur} \cap \delta(S)} u_e + \sum_{e \in E_{T-i+1} \cap \delta(S)} 2\ u_e x_e^* \geq k.$$

We note that iteration 1 ensures that this invariant holds at the start of iteration 2.

(2) At the end of phase 1 of iteration i,

$$\sum_{e \in E_{cur} \cap \delta(S)} u_e + \sum_{e \in E_{T-i} \cap \delta(S)} 2\ u_e x_e^* \geq k - 2^{T-i+1} - 2^{T-i}.$$

(3) At the end of iteration i, which is also the end of phase 2 of iteration i, $E_{cur} \cap E_{T-i} = \emptyset$, and

$$\sum_{e \in E_{cur} \cap \delta(S)} u_e + \sum_{e \in E_{T-i} \cap \delta(S)} 2\ u_e x_e^* \geq k.$$

Observe that invariant (3) for iteration i is the same as invariant (1) for iteration $i+1$.

3.1 Iteration 1

In this iteration, we consider the family of small cuts $\delta(S)$ where

$$\sum_{e \in E_{cur} \cap \delta(S)} u_e + \sum_{e \in E_{T-1} \cap \delta(S)} 2\ u_e x_e^* < k \qquad \text{(definition of small cuts)}$$

We will cover these cuts using edges in $E - E_{T-1} - E_{cur}$. Consider any one of these small cuts $\delta(S)$. Since $\delta(S)$ is a small cut, we have $\sum_{e \in E_{T-1} \cap \delta(S)} 2\ u_e x_e^* < k - (\sum_{e \in E_{cur} \cap \delta(S)} u_e) = k - u(E_{cur} \cap \delta(S))$, hence, we have $\sum_{e \in E_{T-1} \cap \delta(S)} u_e x_e^* < (k - u(E_{cur} \cap \delta(S)))/2$. Consider the knapsack-cover inequality for one of these small cuts $\delta(S)$ and the set $A = E_{cur}$, $\sum_{e \in E \cap \delta(S) - A} u_e(A,S)\, x_e \geq D(A,S)$, where $D(A,S) = (k - u(E_{cur} \cap \delta(S)))$ and $u_e(A,S) = \min\{u_e, D(A,S)\}$. By the above inequality and the knapsack-cover inequality, each of these small cuts $\delta(S)$ satisfies the inequality $\sum_{e \in (E - E_{T-1} - E_{cur}) \cap \delta(S)} \min\{u_e, D(A,S)\} x_e^* > D(A,S)/2$, which implies the inequality $\sum_{e \in (E - E_{T-1} - E_{cur}) \cap \delta(S)} D(A,S) x_e^* > D(A,S)/2$. Thus $2\, x^*_{E - E_{T-1} - E_{cur}}$ is feasible for the Cover Small Cuts problem implying that we incur a cost of at most $6 \cdot 2 \cdot c(x^*_{E_T - E_{T-1}})$ here. We run the 6-approximation algorithm for Cover Small Cuts, [3, 4, 21], and use the edge-set returned by that algorithm to augment E_{cur}. We repeat one more time, i.e., we again consider all cuts $\delta(S)$ where $\sum_{e \in E_{cur} \cap \delta(S)} u_e + \sum_{e \in E_{T-1} \cap \delta(S)} 2u_e x_e^* < k$ and cover these cuts using edges in $E - E_{T-1} - E_{cur}$, incurring a further cost of $6 \cdot 2 \cdot c(x^*_{E - E_{T-1}})$. Now, we will have necessarily satisfied invariant (3) at the end of this iteration. To see this, observe that if some cut violated this invariant, then this cut participated as a small cut in both instances of Cover Small Cuts considered in this step. This means we would have added at least two edges that cover this cut, each of capacity at least $k/2$, ensuring that invariant (3) holds.

3.2 Iteration i Phase 1 (Step 2 (a) in Algorithm 2)

We are starting with invariant (1) at the beginning of this iteration (as this corresponds to the invariant (3) that holds at the end of the previous iteration). Hence we have $E_{cur} \cap E_{T-i+1} = \emptyset$ and $\sum_{e \in E_{cur}} u_e + \sum_{e \in E_{T-i+1}} 2u_e x_e^* \geq k$. We will run multiple sub-iterations within this phase. The first sub-iteration is described below.

Consider the family of small cuts $\delta(S)$ where $\sum_{e\in E_{cur}\cap\delta(S)} u_e + \sum_{e\in E_{T-i}\cap\delta(S)} 2u_e x_e^* < 2^{T-i}$. We will cover these cuts using edges in $E_{T-i+1} - E_{T-i} - E_{cur}$. For these small cuts $\delta(S)$, we have

$$\sum_{e\in(E_{T-i+1}-E_{T-i}-E_{cur})\cap\delta(S)} 2u_e x_e^* \geq k - 2^{T-i}$$

$\big($this inequality is obtained by subtracting the inequality defining the small cuts from the inequality of invariant (1), namely, $\sum_{e\in E_{cur}\cap\delta(S)} u_e + \sum_{e\in E_{T-i+1}\cap\delta(S)} 2\ u_e x_e^* \geq k\big)$. Observe that $\sum_{e\in(E_{T-i+1}-E_{T-i}-E_{cur})\cap\delta(S)} 2x_e^* \geq (k-2^{T-i})/2^{T-i+1}$, because $u_e \leq 2^{T-i+1}$ for all edges in E_{T-i+1}. Thus, $2x^*_{E_{T-i+1}-E_{T-i}-E_{cur}} \cdot 2^{T-i+1}/(k-2^{T-i})$ is feasible for the Cover Small Cuts problem. We run the 6-approximation algorithm for Cover Small Cuts, [3, 4, 21], and use the edge-set returned by that algorithm to augment E_{cur}. After this, there are no non-trivial cuts $\delta(S)$ with $\sum_{e\in E_{cur}\cap\delta(S)} u_e + \sum_{e\in E_{T-i}\cap\delta(S)} 2u_e x_e^* < 2^{T-i}$ since we would have covered any such cut by an edge from $E_{T-i+1} - E_{T-i} - E_{cur}$, and all these edges have capacity at least 2^{T-i}. Next, we shift the threshold in the definition for small cuts to $2\cdot 2^{T-i}$, and then to $3\cdot 2^{T-i}$, ..., all the way until $\hat{\ell}\cdot 2^{T-i}$ where $\hat{\ell} = \lfloor (k - 2^{T-i+1})/2^{T-i}\rfloor$. This would imply that $\hat{\ell}\cdot 2^{T-i} \geq k - 2^{T-i+1} - 2^{T-i}$. We describe these sub-iterations in more detail now.

For $\ell = 1, 2, \ldots, \hat{\ell}$, consider the family of small cuts $\delta(S)$ where

$$\sum_{e\in E_{cur}\cap\delta(S)} u_e + \sum_{e\in E_{T-i}\cap\delta(S)} 2u_e x_e^* < \ell\cdot 2^{T-i}. \qquad \text{(definition of small cuts)}$$

Since invariant (1) is true (and we have only increased the LHS of invariant (1) during the phase), we have $\displaystyle\sum_{e\in(E_{T-i+1}-E_{T-i}-E_{cur})\cap\delta(S)} 2u_e x_e^* \geq k - \ell\cdot 2^{T-i}$. Since $u_e \leq 2^{T-i+1}$ for all edges in E_{T-i+1}, we have $\displaystyle\sum_{e\in(E_{T-i+1}-E_{T-i}-E_{cur})\cap\delta(S)} 2x_e^* \geq (k-\ell\cdot 2^{T-i})/2^{T-i+1}$. Thus, $2x^*_{E_{T-i+1}-E_{T-i}-E_{cur}} \cdot 2^{T-i+1}/(k-\ell\cdot 2^{T-i})$ is feasible for our instance of Cover Small Cuts. We run the 6-approximation algorithm for Cover Small Cuts, [3, 4, 21], and use the edge-set returned by that algorithm to augment E_{cur}. Then, we move on to the next sub-iteration. At the end of the last sub-iteration (with $\ell = \hat{\ell}$), we have

$$\sum_{e\in E_{cur}\cap\delta(S)} u_e + \sum_{e\in E_{T-i}\cap\delta(S)} 2u_e x_e^* \geq (\hat{\ell})2^{T-i} \geq k - 2^{T-i+1} - 2^{T-i},$$

and so invariant (2) is maintained. Let us analyze the cost we incurred in this phase.

The cost we incur is at most $6\cdot 2\cdot c(x^*_{E_{T-i+1}-E_{T-i}})2^{T-i+1}(\frac{1}{k-2^{T-i}} + \frac{1}{k-2\cdot 2^{T-i}} + \cdots + \frac{1}{k-\hat{\ell}2^{T-i}})$. We bound this last sum as follows. Note that $\hat{\ell}2^{T-i} \leq k - 2^{T-i+1}$ and so $k - \hat{\ell}2^{T-i} \geq 2^{T-i+1}$

$$\begin{aligned}
&\frac{1}{k-2^{T-i}} + \frac{1}{k-2\cdot 2^{T-i}} + \cdots + \frac{1}{k-\hat{\ell}2^{T-i}} \\
&= \frac{1}{k-\hat{\ell}2^{T-i}} + \frac{1}{k-\hat{\ell}2^{T-i}+2^{T-i}} + \frac{1}{k-\hat{\ell}2^{T-i}+2\cdot 2^{T-i}} + \cdots \frac{1}{k-\hat{\ell}2^{T-i}+(\hat{\ell}-1)\cdot 2^{T-i}} \\
&\leq \frac{1}{2^{T-i+1}} + \sum_{\ell=1}^{\hat{\ell}-1} \frac{1}{k-\hat{\ell}2^{T-i}+2^{T-i}\ell} \\
&\leq \frac{1}{2^{T-i+1}} + \int_0^{\hat{\ell}-1} \frac{1}{k-\hat{\ell}2^{T-i}+2^{T-i}\ell} d(\ell) \\
&= \frac{1}{2^{T-i+1}} + \frac{1}{2^{T-i}}\left(\log(k-2^{T-i}) - \log(k-\hat{\ell}2^{T-i})\right) \\
&\leq \frac{1}{2^{T-i+1}} + \frac{1}{2^{T-i}}\left(\log(k) - \log(2^{T-i+1})\right) && \text{(using the inequality } k-\hat{\ell}2^{T-i} \geq 2^{T-i+1}) \\
&= O\left(\frac{\log(k/2^{T-i+1})}{2^{T-i}}\right)
\end{aligned}$$

Thus, the cost incurred in this phase is
$\leq 6 \cdot 2(2^{T-i+1}/2^{T-i})O(\log(k/2^{T-i+1}))c(x^*_{E_{T-i+1}-E_{T-i}})$.

3.3 Iteration i Phase 2 (Step 2 (b)-(d) in Algorithm 2)

We are beginning with invariant (2), which is valid at the end of phase 1, and thus for all non-trivial cuts $\delta(S)$, we have $\sum\limits_{e \in E_{cur} \cap \delta(S)} u_e + \sum\limits_{e \in E_{T-i} \cap \delta(S)} 2u_e x^*_e \geq k - 2^{T-i+1} - 2^{T-i}$. We will add more edges from $E - E_{T-i} - E_{cur}$ to these cuts, if needed, to increase the capacity to k. Note that all edges in $E - E_{T-i} - E_{cur}$ have capacity at least 2^{T-i} and so at most three more edges need to be added. To do so, we employ the method we used in iteration 1.

Consider the family of small cuts $\delta(S)$ where $\sum\limits_{e \in E_{cur} \cap \delta(S)} u_e + \sum\limits_{e \in E_{T-i} \cap \delta(S)} 2u_e x^*_e < k$. Then, by the knapsack-cover inequalities, $2\,x^*_{E-E_{T-i}-E_{cur}}$ is feasible for the Cover Small Cuts instance. Indeed for each of these small cuts $\delta(S)$, we have $\sum_{e \in E_{T-i} \cap \delta(S)} u_e x^*_e < (k - \sum_{e \in E_{cur} \cap \delta(S)} u_e)/2 = (k - u(E_{cur} \cap \delta(S)))/2$. The knapsack-cover inequality then implies that $\sum_{e \in (E - E_{T-i} - E_{cur}) \cap \delta(S)} D(A,S)x^*_e > D(A,S)/2$, where $A = E_{cur}$ and $D(A,S) = k - u(E_{cur} \cap \delta(S))$. We run the 6-approximation algorithm for Cover Small Cuts, [3, 4, 21], and use the edge-set returned by that algorithm to augment E_{cur}. This incurs a cost of at most $6 \cdot 2 \cdot c(x^*)$. We repeat three times, adding the approximate solution of the Cover Small Cuts instance to E_{cur} and incur a cost of at most $3 \cdot 6 \cdot 2 \cdot c(x^*)$. At the end of this phase, for every non-trivial cut $\delta(S)$, we have $\sum\limits_{e \in E_{cur} \cap \delta(S)} u_e + \sum\limits_{e \in E_{T-i} \cap \delta(S)} 2u_e x^*_e \geq k$. This is precisely invariant (3) and we have completed this phase.

3.4 Iteration T

At the beginning of the last iteration, by invariant (1), we have for any non-trivial cut $\delta(S)$:

$$\sum_{e \in E_{cur} \cap \delta(S)} u_e + \sum_{e \in E_1 \cap \delta(S)} 2u_e x^*_e \geq k.$$

Now, we apply Jain's iterative rounding method to round the edges in E_1, incurring a cost of at most $(2\max\{u_e : e \in E_1\}) \cdot c(x^*_{E_1}) = 4c(x^*_{E_1})$.

3.5 Total Cost

We calculate the total cost incurred separately for each iteration. In iteration 1, we incur a cost of $O(1) \cdot c(x^*)$; this includes the term $\sum_{\{e \in E \,:\, x^*_e \geq 1/2\}} 2c_e x^*_e$ (due to the initial E_{cur}). In phase 1 of iterations $i = 2, \ldots, T-1$, we incur a total cost of

$$O(1) \cdot \sum_{i=2}^{T-1} \log(k/2^{T-i+1})c(x^*_{E_{T-i+1}-E_{T-i}}) \quad \leq \quad O(\log k)c(x^*).$$

In phase 2 of iterations $i = 2, \ldots, T-1$, we incur a cost of $\sum_{i=2}^{T-1} O(1) \cdot c(x^*) \leq O(T)c(x^*) \leq O(\log k)c(x^*)$. Finally the cost incurred in iteration T is $O(1) \cdot c(x^*)$. Thus the total cost incurred is $O(\log k)c(x^*)$.

Observe that if the minimum capacity u_{min} over all edges in E is greater than 2, then the algorithm stops at an earlier iteration. In fact, it stops at an iteration i_{final} where $2^{T-i_{\mathsf{final}}} < u_{min} \leq 2^{T-i_{\mathsf{final}}+1}$. Since $T = \lceil \log k \rceil$, we obtain that $i_{\mathsf{final}} = O(\log(k/u_{min}))$. In such a scenario, the total cost incurred in phase 1 of iterations $i = 2, \dots, i_{\mathsf{final}}$, is

$$\begin{aligned}
&O(1) \cdot \sum_{i=2}^{i_{\mathsf{final}}} \log(k/2^{T-i+1}) c(x^*_{E_{T-i+1}-E_{T-i}}) \\
&\leq O(1) \cdot \sum_{i=2}^{i_{\mathsf{final}}} \log(k/u_{min}) c(x^*_{E_{T-i+1}-E_{T-i}}) && (\text{since } u_{min} \leq 2^{T-i_{\mathsf{final}}+1}) \\
&\leq O(\log(k/u_{min})) c(x^*).
\end{aligned}$$

Similarly the total cost incurred in phase 2 of iterations $i = 2, \dots, i_{\mathsf{final}}$ is $\sum_{i=2}^{i_{\mathsf{final}}} O(1) \cdot c(x^*) \leq O(i_{\mathsf{final}}) c(x^*) \leq O(\log(k/u_{min})) c(x^*)$. Thus the overall cost is $O(\log(k/u_{min})) c(x^*)$.

3.6 Solving the LP Relaxation

Clearly, our rounding algorithm runs in polynomial time, provided an optimal (and feasible) solution x^* to (KCLP: CapkECSS) is given. As in section 2, we would like solve (KCLP: CapkECSS) using the ellipsoid method, but, unfortunately, we do not know of any polynomial-time separation oracle for the entire set of knapsack-cover inequalities. Instead, we will iteratively (in polynomial time) identify a subset of edges A and a collection of sets $\mathcal{C}$ such that, as long as the knapsack-cover inequalities hold for A and all $S \in \mathcal{C}$, we will be able to execute our rounding algorithm.

▶ **Lemma 11.** *There is a polynomial-time algorithm that, given a vector x^* (that is a candidate solution of* (KCLP: CapkECSS)*) and a value z, either finds a violated constraint of the LP or else verifies that $c(x^*) \leq z$ and, moreover, for every iteration i, $i = 1, \dots, (T-1)$, x^* satisfies the property that $2x^*_{E-E_{T-i}-E_{cur}}$ is feasible for the LP relaxations of the* Cover Small Cuts *instances created in steps 1(b) and 2(c) of Algorithm 2.*

Proof. Given a candidate vector x^* and a candidate objective value z, we first check that $\sum_{e \in E} c_e x^*_e \leq z$ (see the discussion on binary search at the end of section 2). If not, we return this as a violated constraint. Otherwise, let $\hat{G} = (\hat{V}, \hat{E})$ be the capacitated graph where $\hat{V} = V, \hat{E} = E$, and each edge $e \in \hat{E}$ is assigned a capacity of $u_e x^*_e$. We can now check that the capacity of a minimum cut in $\hat{G}$ is at least k using a polynomial-time global minimum-cut algorithm [19]. If not, we return a global minimum cut in $\hat{G}$ as a violated constraint.

By Karger's result [15], we know that there are at most $O(n^4)$ cuts of capacity at most $2k$ (i.e., at most twice the capacity of a minimum cut), and, moreover, we can enumerate all such cuts of $\hat{G}$ in polynomial time [16]. By iterating over each of the $O(n^4)$ cuts, we can then verify in polynomial time that the knapsack-cover inequalities are satisfied w.r.t. the set $A = E_{cur}$ in each of the steps 1(b) and 2(c) for cuts whose capacity is at most $2k$. If not, we have found a violated constraint. It remains then to handle the case when we are at step 1(b) or 2(c), and we have a small cut $\delta(S)$ such that $\sum_{e \in \delta(S)} u_e x^*_e > 2k$.

In this case, we note that in step 1(b), by the definition of small cuts, we have $\sum_{e \in E_{cur} \cap \delta(S)} u_e + \sum_{e \in E_{T-1} \cap \delta(S)} 2u_e x^*_e < k$. But then since the total capacity of this cut is at least $2k$, it follows that $\sum_{e \in (E-E_{T-1}-E_{cur}) \cap \delta(S)} u_e x^*_e > k$. Since $u_e \leq k$ for every edge $e \in E$, it follows that $\sum_{e \in (E-E_{T-1}-E_{cur}) \cap \delta(S)} x^*_e > 1$. Thus $2\,x^*_{E-E_{T-1}-E_{cur}}$ is feasible on this cut for the Cover Small Cuts instance. A similar argument can be used to show that in step 2(c), if we have a small cut $\delta(S)$ with $\sum_{e \in \delta(S)} u_e x^*_e > 2k$, then $2\,x^*_{E-E_{T-i}-E_{cur}}$ is feasible for the Cover Small Cuts instance. Specifically, by the definition of small cuts, we have $\sum_{e \in E_{cur} \cap \delta(S)} u_e + \sum_{e \in E_{T-i} \cap \delta(S)} 2u_e x^*_e < k$. As the total capacity is at least $2k$, it follows that $\sum_{e \in (E-E_{T-i}-E_{cur}) \cap \delta(S)} u_e x^*_e > k$, and since $u_e \leq k$ for every edge $e \in E$, $\sum_{e \in (E-E_{T-i}-E_{cur}) \cap \delta(S)} x^*_e > 1$. Thus $2x^*_{E-E_{T-i}-E_{cur}}$ is feasible on this cut for the Cover Small Cuts instance.

Finally, if at any step of the rounding algorithm, we identify a violated constraint, then we *restart* the rounding algorithm from the very beginning. It is worth highlighting that the verification of knapsack-cover inequalities identified in steps 1(b) and 2(c) of the algorithm, is always done with respect to the solution x^* given by the Ellipsoid algorithm (without any modification). As the rounding progresses, the only thing that changes is the definition of the set $A = E_{cur}$ with respect to which we verify the knapsack-cover inequalities. So whenever a violated constraint is identified, it contributes to the iteration count of the ellipsoid algorithm. Since the ellipsoid algorithm terminates after $n^{O(1)}$ iterations of feasibility verification [10], it must be the case that after at most $n^{O(1)}$ re-starts of the rounding process, we arrive at a solution x^* to (KCLP: CapkECSS) of value at most $\mathrm{LP_{opt}}$ such that the solution satisfies the property that $2x^*_{E-E_{T-i}-E_{cur}}$ is feasible for the Cover Small Cuts instances created in steps 1(b) and 2(c). ◀

▶ Remark 12. We mention that the analysis of phase 1 of iteration i ($i = 2, \ldots, (T-1)$) (i.e., step 2(a) of iteration i) does *not* use the knapsack-cover inequalities, hence, Lemma 11 does not address step 2(a).

▶ **Theorem 3.** *There is a polynomial-time algorithm that, given an instance of Cap-k-ECSS, computes a vector x^* of cost at most* OPT *that possibly satisfies only a subset of the constraints of* (KCLP: CapkECSS)*, and rounds it to a feasible integer solution of cost at most $O(\log(k/u_{min})) \cdot$* OPT*.*

References

1 David Adjiashvili, Felix Hommelsheim, and Moritz Mühlenthaler. Flexible Graph Connectivity. *Mathematical Programming*, 192:409–441, 2022. `doi:10.1007/s10107-021-01664-9`.

2 David Adjiashvili, Sebastian Stiller, and Rico Zenklusen. Bulk-Robust combinatorial optimization. *Math. Program.*, 149(1-2):361–390, 2015. `doi:10.1007/s10107-014-0760-6`.

3 Ishan Bansal. A Global Analysis of the Primal-Dual Method for Pliable Families. *CoRR*, abs/2308.15714, 2024. `doi:10.48550/arXiv.2308.15714`.

4 Ishan Bansal, Joseph Cheriyan, Logan Grout, and Sharat Ibrahimpur. Improved approximation algorithms by generalizing the primal-dual method beyond uncrossable functions. *Algorithmica*, 86(8):2575–2604, 2024. `doi:10.1007/s00453-024-01235-2`.

5 Sylvia C. Boyd, Joseph Cheriyan, Arash Haddadan, and Sharat Ibrahimpur. Approximation algorithms for flexible graph connectivity. *Mathematical Programming*, 2023. `doi:10.1007/s10107-023-01961-5`.

6 Robert D. Carr, Lisa K. Fleischer, Vitus J. Leung, and Cynthia A. Phillips. Strengthening integrality gaps for capacitated network design and covering problems. In *Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '00, pages 106–115, USA, 2000. Society for Industrial and Applied Mathematics. URL: `http://dl.acm.org/citation.cfm?id=338219.338241`.

7 Deeparnab Chakrabarty, Chandra Chekuri, Sanjeev Khanna, and Nitish Korula. Approximability of Capacitated Network Design. *Algorithmica*, 72(2):493–514, 2015. `doi:10.1007/s00453-013-9862-4`.

8 Chandra Chekuri and Rhea Jain. Approximation Algorithms for Network Design in Non-Uniform Fault Models. In *Proceedings of the 50th International Colloquium on Automata, Languages, and Programming*, volume 261, article 36, pages 1–20, 2023. `doi:10.4230/LIPIcs.ICALP.2023.36`.

9 Michel X. Goemans, Andrew V. Goldberg, Serge A. Plotkin, David B. Shmoys, Éva Tardos, and David P. Williamson. Improved Approximation Algorithms for Network Design Problems. In *Proceedings of the 5th Symposium on Discrete Algorithms*, pages 223–232, 1994. URL: `http://dl.acm.org/citation.cfm?id=314464.314497`.

10 Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer Berlin, 1993. doi:10.1007/978-3-642-78240-4.

11 Felix Hommelsheim, Zhenwei Liu, Nicole Megow, and Guochuan Zhang. Protecting the Connectivity of a Graph under Non-uniform Edge Failures. *CoRR*, abs/2501.04540, 2025. doi:10.48550/arXiv.2501.04540.

12 Dylan Hyatt-Denesik, Afrouz Jabal Ameli, and Laura Sanità. Improved Approximations for Flexible Network Design. In Timothy M. Chan, Johannes Fischer, John Iacono, and Grzegorz Herman, editors, *32nd Annual European Symposium on Algorithms, ESA 2024, September 2-4, 2024, Royal Holloway, London, United Kingdom*, volume 308 of *LIPIcs*, pages 74:1–74:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.ESA.2024.74.

13 Sharat Ibrahimpur and László A. Végh. An $O(\log n)$-Approximation Algorithm for (p, q)-Flexible Graph Connectivity via Independent Rounding. *CoRR*, abs/2501.12549, 2025. doi:10.48550/arXiv.2501.12549.

14 Kamal Jain. A Factor 2 Approximation Algorithm for the Generalized Steiner Network Problem. *Combinatorica*, 21(1):39–60, 2001. doi:10.1007/s004930170004.

15 David R. Karger. Global min-cuts in RNC, and other ramifications of a simple min-cut algorithm. In Vijaya Ramachandran, editor, *Proceedings of the Fourth Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms, 25-27 January 1993, Austin, Texas, USA*, pages 21–30. ACM/SIAM, 1993. URL: http://dl.acm.org/citation.cfm?id=313559.313605.

16 Hiroshi Nagamochi, Kazuhiro Nishimura, and Toshihide Ibaraki. Computing All Small Cuts in an Undirected Network. *SIAM Journal on Discrete Mathematics*, 10(3):469–481, 1997. doi:10.1137/S0895480194271323.

17 Zeev Nutov. Improved approximation ratio for covering pliable set families. *CoRR*, 2024. doi:10.48550/arXiv.2404.00683.

18 Zeev Nutov. Tight analysis of the primal-dual method for edge-covering pliable set families. *CoRR*, abs/2504.03910, 2025. doi:10.48550/arXiv.2504.03910.

19 Alexander Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*, volume 24 of *Algorithms and Combinatorics*. Springer, Berlin Heidelberg New York, 2003.

20 Miles Simmons, Ishan Bansal, and Joe Cheriyan. A Bad Example for Jain's Iterative Rounding Theorem for the Cover Small Cuts Problem. *CoRR*, abs/2504.13105, 2025. doi:10.48550/arXiv.2504.13105.

21 David P. Williamson, Michel X. Goemans, Milena Mihail, and Vijay V. Vazirani. A Primal-Dual Approximation Algorithm for Generalized Steiner Network Problems. *Combinatorica*, 15(3):435–454, 1995. doi:10.1007/BF01299747.