Approximability of Capacitated Network Design*

Deeparnab Chakrabarty† Chandra Chekuri‡ Sanjeev Khanna § Nitish Korula¶

Abstract

In the *capacitated* survivable network design problem (Cap-SNDP), we are given an undirected multi-graph where each edge has a capacity and a cost. The goal is to find a minimum cost subset of edges that satisfies a given set of pairwise minimum-cut requirements. Unlike its classical special case of SNDP when all capacities are unit, the approximability of Cap-SNDP is not well understood; even in very restricted settings no known algorithm achieves a $o(m)$ approximation, where m is the number of edges in the graph. In this paper, we obtain several new results and insights into the approximability of Cap-SNDP.

We give an $O(\log n)$ approximation for a special case of Cap-SNDP where the global minimum cut is required to be at least R. (Note that this problem generalizes the min-cost λ-edge-connected subgraph problem, which is the special case of our problem when all capacities are unit.) Our result is based on a rounding of a natural cut-based LP relaxation strengthened with knapsack-cover (KC) inequalities. Our technique extends to give a similar approximation for a new network design problem that captures global minimum cut as a special case. We then show that as we move away from global connectivity, even for the single pair case (that is, when only one pair (s, t) has positive connectivity requirement), this strengthened LP has $\Omega(n)$ integrality gap.

We also consider a variant of Cap-SNDP in which multiple copies of an edge can be bought: we give an $O(\log k)$ approximation for this case, where k is the number of vertex pairs with non-zero connectivity requirement. This improves upon the previously known $O(\min\{k, \log R_{\max}\})$-approximation when $R_{\max}$ is large; here $R_{\max}$ is the largest requirement. On the other hand, we observe that the multiple copy version of Cap-SNDP is $\Omega(\log \log n)$-hard to approximate even for the single-source version of the problem.

*A preliminary version appeared in the 15th Conference on Integer Programming and Combinatorial Optimization (IPCO), 2011 held at the IBM T.J. Watson Centre.

†Microsoft Research, India. 9, Lavelle Road, Bangalore, Karnataka 560025. dechakr@microsoft.com. Work done while at Dept. of CIS, University of Pennsylvania.

‡Dept. of Computer Science, University of Illinois, Urbana, IL 61801. Partially supported by NSF grants CCF-0728782 and CNS-0721899. chekuri@cs.illinois.edu

§Dept. of Computer & Information Science, University of Pennsylvania, Philadelphia, PA 19104. Supported in part by NSF Awards CCF-0635084 and IIS-0904314. sanjeev@cis.upenn.edu

¶Google Inc. 76 Ninth Ave, New York, NY 10011. nitish@google.com. This work was done while at the Dept. of Computer Science, University of Illinois, and was partially supported by NSF grant CCF-0728782 and a University of Illinois Dissertation Completion Fellowship.

1 Introduction

In this paper we consider the *capacitated* survivable network design problem (Cap-SNDP). The input consists of an undirected n-vertex multi-graph $G(V, E)$ and an integer requirement R_{ij} for each unordered pair of nodes (i, j). Each edge e of G has a cost $c(e)$ and an integer capacity $u(e)$. The goal is to find a minimum-cost subgraph H of G such that for each pair of nodes i, j the capacity of the minimum-cut between i and j in H is at least R_{ij}. This generalizes the well-known survivable network design problem (SNDP) in which all edge capacities are 1. SNDP already captures as special cases a variety of fundamental connectivity problems in combinatorial optimization such as min-cost spanning tree, min-cost Steiner tree and forest, as well as min-cost λ-edge-connected subgraph; each of these problems has been extensively studied on its own and several of these special cases are NP-hard and APX-hard to approximate. Jain, in an influential paper [20], obtained a 2-approximation for SNDP via the standard cut-based LP relaxation using the iterated rounding technique.

Although the 2-approximation for SNDP mentioned above has been known since 1998, the approximability of Cap-SNDP has essentially been wide open even in very restricted special cases. Similar to SNDP, Cap-SNDP is motivated by both practial and theoretical considerations. These problems find applications in the design of resilient networks such as in telecommunication infrastructure. In such networks it is often quite common to have equipment with different discrete capacities; this leads naturally to design problems such as Cap-SNDP. At the outset, we mention that a different and somewhat related problem is also referred to by the same name, especially in the operations research literature. In this version the subgraph H has to support *simultaneously* a flow of R_{ij} between each pair of nodes (i, j); this is more closely related to multicommodity flows and buy-at-bulk network design. Our version is more related to connectivity problems such as SNDP.

As far as we are aware, the version of Cap-SNDP that we study was introduced (in the approximation algorithms literature) by Goemans *et al.* [15] in conjunction with their work on SNDP. They made several observations on Cap-SNDP: (i) Cap-SNDP reduces to SNDP if all capacities are the same, (ii) there is an $O(\min(m, R_{\max}))$ approximation where m is the number of edges in G and $R_{\max} = \max_{ij} R_{ij}$ is the maximum requirement, and (iii) if *multiple* copies of an edge are allowed then there is an $O(\log R_{\max})$-approximation. We note that in the capacitated case $R_{\max}$ can be exponentially large in n, the number of nodes of the graph. Carr *et al.* [5] observed that the natural cut-based LP relaxation has an unbounded integrality gap even for the graph consisting of only two nodes s, t connected by parallel edges with different capacities. Motivated by this observation and the goal of obtaining improved approximation ratios for Cap-SNDP, [5] strengthened the basic cut-based LP by using *knapsack-cover* inequalities. (Several subsequent papers in approximation algorithms have fruitfully used these inequalities.) Using these inequalities, [5] obtained a $\beta(G) + 1$ approximation for Cap-SNDP where $\beta(G)$ is the maximum cardinality of a *bond* in the underlying simple graph: a bond is a minimal set of edges that separates some pair of vertices with positive demand. Although $\beta(G)$ could be $\Theta(n^2)$ in general, for certain topologies — for instance, if the underlying graph is a line or a cycle — this gives constant factor approximations.

The above results naturally lead to several questions. What is the approximability of Cap-SNDP? Should we expect a poly-logarithmic approximation or even a constant factor approximation? If not, what are interesting and useful special cases to consider? And do the knapsack cover inequalities help in the general case? What is the approximability of Cap-SNDP if one allows multiple copies of edges? Does this relaxed version of the problem allow a constant factor approximation?

In this paper we obtain several new positive and negative results for Cap-SNDP that provide new insights

into the questions above.

1.1 Our Results

We first discuss results for Cap-SNDP where multiple copies are not allowed. We initiate our study by considering the *global connectivity* version of Cap-SNDP where we want a min-cost subgraph with global min-cut at least R; in other words, there is a "uniform" requirement $R_{ij} = R$ for *all* pairs (i, j). We refer to this as the *Cap-R-Connected Subgraph* problem; the special case when all capacities are unit corresponds to the classical minimum cost λ-edge-connected (spanning) subgraph problem, which is known to be APX-hard [13]. We show the following positive result for arbitrary capacities.

Theorem 1.1. *There is a randomized $O(\log n)$-approximation algorithm for the Cap-R-Connected Subgraph problem. Moreover, for any $\gamma \geq 1$, there is a randomized $O(\gamma \log n)$-approximation algorithm with running time $n^{O(\gamma)}$ for "nearly uniform" Cap-SNDP when all pairwise requirements are in $[R, \gamma R]$.*

To prove Theorem 1.1, we begin with a natural LP relaxation for the problem. Almost all positive results previously obtained for the unit capacity case are based on this relaxation. As remarked already, this LP has an unbounded integrality gap even for a graph with two nodes (and hence for Cap-R-Connected Subgraph). We strengthen the relaxation by adding the valid knapsack cover inequalities. Although we do not know of a polynomial time algorithm to separate over these inequalities, following [5], we find a violated inequality *only if* the current fractional solution does not satisfy certain useful properties. Our main technical tool both for finding a violated inequality and subsequently rounding the fractional solution is Karger's theorem on the number of small cuts in undirected graphs [21].

We believe the approach outlined above may be useful in other network design applications. As a concrete illustration, we use it to solve an interesting and natural generalization of Cap-R-Connected Subgraph, namely, the *k-Way–$\mathcal{R}$-Connected Subgraph* problem. The input consists of $(k-1)$ integer requirements $R_1, \ldots R_{k-1}$, such that $R_1 \leq R_2 \leq \ldots \leq R_{k-1}$. The goal is to find a minimum-cost subgraph H of G such that for each $1 \leq i \leq k-1$, the capacity of any $(i+1)$-way cut of G is at least R_i.[1] It is easy to see that Cap-R-Connected Subgraph is precisely the k-Way–$\mathcal{R}$-Connected Subgraph, with $k = 2$. Note that the k-Way–$\mathcal{R}$-Connected Subgraph problem is not a special case of the general Cap-SNDP as the cut requirements for the former problem are not expressible as pairwise connectivity constraints. Interestingly, our techniques for Cap-R-Connected Subgraph can be naturally extended to handle the multiway cut requirements, yielding the following generalization of Theorem 1.1.

Theorem 1.2. *There is a randomized $O(k \log n)$-approximation algorithm for the k-Way–$\mathcal{R}$-Connected Subgraph problem with $n^{O(k)}$ running time.*

We remark that even for the unit-capacity case of this problem, it is not clear how to obtain a better ratio than that guaranteed by the above theorem. We discuss this further in Section 2.4.

Once the pairwise connectivity requirements are allowed to vary arbitrarily, Cap-SNDP seems to become distinctly harder. Surprisingly, the difficulty of the general case starts to manifest even for the simplest representative problem in this setting, where there is only one pair (s, t) with $R_{st} > 0$; we refer to this as the *single pair* problem. The only known positive result for this seemingly restricted case is a polynomial-factor approximation that follows from the results in [15, 5] for general Cap-SNDP. We give several negative

[1] An i-way cut $\mathcal{C}$ of a graph $G(V, E)$ is a partition of its vertices into i non-empty sets $V_1, \ldots, V_i$; we use $\delta(\mathcal{C})$ to denote the set of edges with endpoints in different sets of the partition $\mathcal{C}$. The *capacity* of an i-way cut $\mathcal{C}$ is the total capacity of edges in $\delta(\mathcal{C})$.

results to suggest that this special case may capture the essential difficulty of Cap-SNDP. In particular, we start by observing that the LP with knapsack cover inequalities has an $\Omega(n)$ integrality gap even for the single-pair problem.[2] Next we show that the single pair problem is $\Omega(\log \log n)$-hard to approximate.

Theorem 1.3. *The single pair Cap-SNDP problem cannot be approximated to a factor better than* $\Omega(\log \log n)$ *unless* $NP \subseteq DTIME(n^{\log \log \log n})$.

The above theorem is a corollary of the results in Chuzhoy *et al.* 's work on the hardness of related network design problems [9]. For completeness, we provide a brief proof in Section 3.1.

Allowing Multiple Copies: Given the negative results above for even the special case of the single-pair Cap-SNDP, it is natural to consider the relaxed version of the problem where multiple copies of an edge can be chosen. Specifically, for any integer $\alpha \geq 0$, α copies of e can be bought at a cost of $\alpha \cdot c(e)$ to obtain a capacity $\alpha \cdot u(e)$. In some applications, such as in telecommunication networks, this is a reasonable model. As we discussed, this model was considered by Goemans *et al.* [15] who gave an $O(\log R_{\max})$ approximation for Cap-SNDP. This follows from a simple $O(1)$ approximation for the case when all requirements are in $\{0, R\}$. The advantage of allowing multiple copies is that one can group request pairs into classes and separately solve the problem for each class while losing only the number of classes in the approximation ratio. For instance, one easily obtains a 2-approximation for the single pair problem even in directed graphs, in contrast to the difficulty of the problem when multiple copies are not allowed. Note that this also implies an easy $2k$ approximation where k is the number of pairs with $R_{ij} > 0$. We address the approximability of Cap-SNDP with multiple copies of edges allowed. When $R_{\max}$ is large, we improve the $\min\{2k, O(\log R_{\max})\}$-approximation discussed above via the following.

Theorem 1.4. *In undirected graphs, there is an* $O(\log k)$*-approximation algorithm for Cap-SNDP with multiple copies, where* k *is the number of pairs with* $R_{ij} > 0$.

Both our algorithm and analysis are inspired by the $O(\log k)$-competitive online algorithm for the Steiner forest problem by Berman and Coulston [4], and the subsequent adaptation of these ideas for the priority Steiner forest problem by Charikar *et al.* [7]. However, we believe the analysis of our algorithm is more transparent (although it gets weaker constants) than the original analysis of [4].

We complement our algorithmic result by showing that the multiple copy version is $\Omega(\log \log n)$-hard to approximate. This hardness holds even for the *single-source* Cap-SNDP where we are given a source node $s \in V$, and a set of terminals $T \subseteq V$, such that $R_{ij} > 0$ iff $i = s$ and $j \in T$. Observe that single-source Cap-SNDP is a simultaneous generalization of the classical Steiner tree problem ($R_{ij} \in \{0, 1\}$) as well as both Cap-R-Connected Subgraph and single-pair Cap-SNDP.

Theorem 1.5. *In undirected graphs, single source Cap-SNDP with multiple copies cannot be approximated to a factor better than* $\Omega(\log \log n)$ *unless* $NP \subseteq DTIME(n^{\log \log \log n})$.

The above theorem, like Theorem 1.3, follows easily from the results of [9] and is sketched in Section 3.1. We note that the hardness reduction above creates instances with super-polynomially (in k) large capacities. For such instances, our $O(\log k)$-approximation strongly improves on the previously known approximation guarantees.

[2] In [5] it is mentioned that there is a series-parallel graph instance of Cap-SNDP such that the LP with knapsack-cover inequalities has an integrality gap of at least $\lfloor \beta(G)/2 \rfloor + 1$. However, no example is given; it is not clear if the gap applied to a single pair instance or if $\beta(G)$ could be as large as n in the construction.

Related Work: Network design has a large literature in a variety of areas including computer science and operations research. Practical and theoretical considerations have resulted in numerous models and results. Due to space considerations it is infeasible even to give a good overview of closely related work. We briefly mention some work that allows the reader to compare the model we consider here to related models. As we mentioned earlier, our version of Cap-SNDP is a direct generalization of SNDP and hence is concerned with (capacitated) connectivity between request node pairs. We refer the reader to the survey [22] and some recent and previous papers [15, 20, 14, 10, 11, 24] for pointers to literature on network design for connectivity. A different model arises if one wishes to find a min-cost subgraph that supports multicommodity flow for the request pairs; in this model each node pair (i, j) needs to routes a flow of R_{ij} in the chosen graph and these flows simultaneously share the capacity of the graph. We observe that if multiple copies of an edge are allowed then this problem is essentially equivalent to the non-uniform buy-at-bulk network design problem. Buy-at-bulk problems have received substantial attention; we refer the reader to [8] for several pointers to this work. If multiple copies are not allowed, the approximability of this flow version is not well-understood; for example, if the flow for each pair is only allowed to be routed on a single path, then even checking feasibility of a given subgraph is NP-Hard since the problem captures the well-known edge-disjoint paths and unsplittable flow problems. Andrews *et al.* [2] have recently considered special cases of this problem with uniform capacities obtaining a polylogarithmic approximation while allowing polylogarithmic congestion (that is, a few copies) on the chosen edges. A more general version of the problem includes a 'per-unit-flow' cost f_e for each edge; this problem is called the (capacitated) *fixed charge network flow* (FCNF) problem. This problem has been studied extensively in the Operations Research community (see [25, 19, 3, 26, 18] and references within), although most results have been computational in nature. At this point, we should mention that in directed graphs, even the single pair Cap-SNDP problem is hard to approximate to a factor $2^{\log^{1-\varepsilon} n}$ for any $\varepsilon > 0$ unless $NP \subseteq DTIME(n^{\mathtt{poly}\log n})$ [12], and so the interesting case (from a theoretical computer science viewpoint) is that of undirected graphs. The k-Way–$\mathcal{R}$-Connected Subgraph problem that we consider seems not to have been studied previously even in the unit-capacity case.

Remark 1.6. *There have been significant improvements on the hardness front since the publication of the conference version of this article. Hajiaghayi* et al. *[16] show that the single pair Cap-SNDP problem in undirected graphs is as hard as the group Steiner tree problem, and therefore via [17], there is no $O(\log^{2-\varepsilon} n)$-approximation for any $\varepsilon > 0$ unless $NP \subseteq ZTIME(n^{\mathtt{poly}\log n})$. More recently, [6] show that single pair Cap-SNDP in undirected graphs is as hard as in directed graphs, and therefore, via [12], it is hard to approximate to within a factor $2^{\log^{1-\varepsilon} n}$ for any fixed $\varepsilon > 0$ unless $NP \subseteq DTIME(n^{\mathtt{poly}\log n})$.*

2 The Cap-R-Connected Subgraph problem

In this section, we prove Theorem 1.1, giving an $O(\log n)$-approximation for the Cap-R-Connected Subgraph problem. For ease of exposition, we first describe the proof assuming each $R_{ij} = R$; the extension to the case when requirements are "nearly uniform" is deferred to Subsection 2.3. We start by writing a natural linear program relaxation for the problem; the integrality gap of this LP can be arbitrarily large. To deal with this, we introduce additional valid inequalities, called the *knapsack cover* inequalities, that must be satisfied by any integral solution. We show how to round this strengthened LP, obtaining an $O(\log n)$-approximation.

2.1 The Standard LP Relaxation and Knapsack-Cover Inequalities

We assume without any loss of generality that the capacity of any edge is at most R. For each subset $S \subseteq V$, we use $\delta(S)$ to denote the set of edges with exactly one endpoint in S. For a set of edges A, we use $u(A)$ to denote $\sum_{e \in A} u(e)$. We say that a set of edges A satisfies (the cut induced by) S if $u(A \cap \delta(S)) \geq R$. Note that we wish to find the cheapest set of edges which satisfies every subset $\emptyset \neq S \subset V$. The following is the LP relaxation of the standard integer program capturing the problem.

$$\min \sum_{e \in E} c(e) x_e \qquad \text{(Std LP)}$$

$$\forall S \subseteq V, \quad \sum_{e \in \delta(S)} u(e) x_e \geq R$$

$$\forall e \in E, \quad 0 \leq x_e \leq 1$$

The following example shows that (2.1) can have integrality gap as bad as R.

Example 1: Consider a graph G on three vertices p, q, r. Edge pq has cost 0 and capacity R; edge qr has cost 0 and capacity $R - 1$; and edge pr has cost C and capacity R. To achieve a global min-cut of size at least R, any integral solution must include edge pr, and hence must have cost C. In contrast, in (2.1) one can set $x_{pr} = 1/R$, and obtain a total cost of C/R.

In the previous example, any integral solution in which the mincut separating r from $\{p, q\}$ has size at least R must include edge pr, even if qr is selected. The following valid inequalities are introduced precisely to enforce this condition. More generally, let S be a set of vertices, and A be an arbitrary set of edges. Define $R(S, A) = \max\{0, R - u(A \cap \delta(S))\}$ be the *residual* requirement of S that must be satisfied by edges in $\delta(S) \setminus A$. That is, any feasible solution has $\sum_{e \in \delta(S) \setminus A} u(e) x_e \geq R(S, A)$. However, any integral solution also satisfies the following stronger requirement

$$\sum_{e \in \delta(S) \setminus A} \min\{R(S, A), u(e)\} x_e \geq R(S, A)$$

and thus these inequalities can be added to the LP to strengthen it. These additional inequalities are referred to as *Knapsack-Cover* inequalities, or simply KC inequalities, and were first used by [5] in design of approximation algorithms for Cap-SNDP.

Below, we write a LP relaxation, (2.1), strengthened with the knapsack cover inequalities. Note that the original constraints correspond to KC inequalities with $A = \emptyset$; we simply write them explicitly for clarity.

$$\min \sum_{e \in E} c(e) x_e \qquad \text{(KC LP)}$$

$$\forall S \subseteq V, \quad \sum_{e \in \delta(S)} u(e) x_e \geq R \qquad \text{(Original Constraints)}$$

$$\forall A \subseteq E, \forall S \subseteq V, \quad \sum_{e \in \delta(S) \setminus A} \min(u(e), R(S, A)) x_e \geq R(S, A) \qquad \text{(KC-inequalities)}$$

$$\forall e \in E, \quad 0 \leq x_e \leq 1$$

The Linear Program (2.1), like the original (2.1), has exponential size. However, unlike the (2.1), we do not know of the existence of an efficient separation oracle for this. Nevertheless, as we show below, we do not need to solve (2.1); it suffices to get to what we call a *good* fractional solution.

Definition 2.1. *Given a fractional solution x, we say an edge e is* nearly integral *if $x_e \geq \frac{1}{40 \log n}$, and we say e is* highly fractional *otherwise.*

Definition 2.2. *For any $\alpha \geq 1$, a cut in a graph G with capacities on edges, is an* α-mincut *if its capacity is within a factor α of the minimum cut of G.*

Theorem 2.3. [Theorems 4.7.6 and 4.7.7 of [21]] *The number of α-mincuts in an n-vertex graph is at most $n^{2\alpha}$. Moreover, the set of all α-mincuts can be found in $O(n^{2\alpha} \log^2 n)$ time with high probability.*

Given a fractional solution x to the edges, we let A_x denote the set of nearly integral edges, that is, $A_x := \{e \in E : x_e \geq \frac{1}{40 \log n}\}$. Define $\hat{u}(e) = u(e)x_e$ to be the fractional capacity on the edges. Let $\mathcal{S} := \{S \subseteq V : \hat{u}(\delta(S)) \leq 2R\}$. A solution x is called *good* if it satisfies the following three conditions:

(a) The global mincut in G with capacity $\hat{u}$ is at least R, i.e. x satisfies the original constraints.

(b) The KC inequalities are satisfied for the set A_x and the sets in $\mathcal{S}$. Note that if (a) is satisfied, then by Theorem 2.3, $|\mathcal{S}| \leq n^4$.

(c) $\sum_{e \in E} c(e)x_e$ is at most the value of the optimum solution to (2.1).

Note that a good solution need not be *feasible* for (2.1) as it is required to satisfy only a subset of KC-inequalities. We use the ellipsoid method to get such a solution. Such a method was also used in [5].

Lemma 2.4. *There is a randomized algorithm that computes a good fractional solution with high probability.*

Proof: We start by guessing the optimum value M of (2.1) and add the constraint $\sum_{e \in E} c(e)x_e \leq M$ to the constraints of (2.1). If the guessed value is too small, a good solution may not exist; however, a simple binary search suffices to identify the smallest feasible value of M. With this constraint in place, we will use the ellipsoid method to compute a solution that satisfies (a), (b), and (c) with high probability. Since we do not know of a polynomial-time separation oracle for KC inequalities, we will simulate a separation oracle that verifies condition (b), a subset of KC inequalities, in polynomial time. Specifically, we give a randomized polynomial time algorithm such that given a solution x that violates condition (b), the algorithm detects the violation with high probability and outputs a violated KC inequality. We now describe the entire process.

Given a solution x we first check if condition (a) is satisfied. This can be done in polynomial time by $O(n)$ max-flow computations. If (a) is not satisfied, we have found a violated constraint. Once we have a solution that satisfies (a), we know that $|\mathcal{S}| \leq n^4$. By Theorem 2.3, the set $\mathcal{S}$ can be computed in polynomial time with high probability. Thus we can check condition (b) in polynomial-time, and with high-probability find a violating constraint for (b) if one exists. Once we have a solution that satisfies both (a) and (b), we check if $\sum_{e \in E} c(e)x_e \leq M$. If not, we have once again found a violated constraint for input to the ellipsoid algorithm. Thus in polynomially many rounds, where each round runs in polynomial-time, the ellipsoid algorithm combined with the simulated separation oracle, either returns a solution x that satisfies (a), (b), and $\sum_{e \in E} c(e)x_e \leq M$, with high probability, or proves that the system is infeasible. Using binary search, we find the smallest M for which a solution x is returned satisfying conditions (a), (b) and $\sum_{e \in E} c(e)x_e \leq M$. Since M is less than the optimum value of (2.1), we get that the returned x is a good fractional solution with high probability. □

2.2 The Rounding and Analysis

Given a good fractional solution x, we now round it to get a $O(\log n)$ approximation to the Cap-R-Connected Subgraph problem. A useful tool for our analysis is the following Chernoff bound (see [23], for instance):

Lemma 2.5. *Let $X_1, X_2, \ldots X_k$ be a collection of independent random variables in $[0,1]$, let $X = \sum_{i=1}^{k} X_i$, and let $\mu = \mathbb{E}[X]$. The probability that $X \leq (1-\delta)\mu$ is at most $e^{-\mu\delta^2/2}$.*

We start by selecting A_x, the set of all nearly integral edges. Henceforth, we lose the subscript and denote the set as simply A. Let $F = E \setminus A$ denote the set of all highly fractional edges; for each edge $e \in F$, select it with probability $(40 \log n \cdot x_e)$. Let $F^* \subseteq F$ denote the set of selected highly fractional edges. The algorithm returns the set of edges $E_A := A \cup F^*$.

It is easy to see that the expected cost of this solution E_A is $O(\log n) \sum_{e \in E} c(e)x_e$, and hence by condition (c) above, within $O(\log n)$ times that of the optimal integral solution. Thus, to prove Theorem 1.1, it suffices to prove that with high probability, E_A satisfies every cut in the graph G; we devote the rest of the section to this proof. We do this by separately considering cuts of different capacities, where the capacities are w.r.t $\hat{u}$ (recall that $\hat{u}(e) = u(e)x_e$). Let $\mathcal{L}$ be the set of cuts of capacity at least $2R$, that is, $\mathcal{L} := \{S \subseteq V : \hat{u}(\delta(S)) > 2R\}$.

Lemma 2.6. $\mathbf{Pr}[\ \forall S \in \mathcal{L} :\ u(E_A \cap \delta(S)) \geq R]\ \geq\ 1 - \frac{1}{2n^{10}}$.

Proof: We partition $\mathcal{L}$ into sets $\mathcal{L}_2, \mathcal{L}_3, \cdots$ where $\mathcal{L}_j := \{S \subseteq V : jR < \hat{u}(\delta(S)) \leq (j+1)R\}$. Note that Theorem 2.3 implies $|\mathcal{L}_j| \leq n^{2(j+1)}$ by condition (a) above. Fix j, and consider an arbitrary cut $S \in \mathcal{L}_j$. If $u(A \cap \delta(S)) \geq R$, then S is clearly satisfied by E_A. Otherwise, since the total $\hat{u}$-capacity of S is at least jR, we have $\hat{u}(F \cap \delta(S)) \geq \hat{u}(\delta(S)) - u(A \cap \delta(S)) \geq (j-1)R$. Thus

$$\sum_{e \in F \cap \delta(S)} \frac{u(e)}{R} x_e \geq (j-1)$$

Recall that an edge $e \in F$ is selected in F^* with probability $(40 \log n \cdot x_e)$. Thus, for the cut S, the expected value of $\sum_{e \in F^* \cap \delta(S)} \frac{u(e)}{R} \geq 40(j-1)\log n$. Since $u(e)/R \leq 1$, we can apply Lemma 2.5 to get that the probability that S is not satisfied is at most $e^{-16 \log n (j-1)} = 1/n^{16(j-1)}$. Applying the union bound, the probability that there exists a cut in $\mathcal{L}_j$ not satisfied by E_A is at most $n^{2(j+1)}/n^{16(j-1)} = n^{18-14j}$. Thus probability that some cut in $\mathcal{L}$ is not satisfied is bounded by $\sum_{j \geq 2} n^{18-14j} \leq 2n^{-10}$ if $n \geq 2$. Hence with probability at least $1 - 1/2n^{10}$, $A \cup F^*$ satisfies all cuts in $\mathcal{L}$. □

One might naturally attempt the same approach for the cuts in $\mathcal{S}$ (recall that $\mathcal{S} = \{S \subseteq V : \hat{u}(\delta(S)) \leq 2R\}$) modified as follows. Consider any cut S, which is partly satisfied by the nearly integral edges A. The fractional edges contribute to the residual requirement of S, and since x_e is scaled up for fractional edges by a factor of $40 \log n$, one might expect that F^* satisfies the residual requirement, with the $\log n$ factor providing a high-probability guarantee. This intuition is correct, but the KC inequalities are crucial. Consider Example 1; edge pr is unlikely to be selected, even after scaling. In the statement of Lemma 2.5, it is important that each random variable takes values in $[0,1]$; thus, to use this lemma, we need the expected capacity from fractional edges to be large compared to the maximum capacity of an individual edge. But the KC inequalities, in which edge capacities are "reduced", enforce precisely this condition. Thus we get the following lemma using a similar analysis as above.

Lemma 2.7. $\mathbf{Pr}[\ \forall S \in \mathcal{S} :\ u(\delta(E_A \cup \delta(S))) \geq R]\ \geq\ 1 - \frac{1}{n^{12}}$.

The $O(\log n)$-approximation guarantee for the Cap-R-Connected Subgraph problem stated in Theorem 1.1 follows from the previous two lemmas.

2.3 Proof of Theorem 1.1: Near Uniform Cap-SNDP

The algorithm described above can be extended to the case where requirements are *nearly* uniform, that is, if $R_{pq} \in [R, \gamma R]$ for all pairs $(p, q) \in V \times V$. We obtain an $O(\gamma \log n)$-approximation, while increasing the running time by a factor of $O(n^{4\gamma})$. We work with a similar LP relaxation; for each set $S \subseteq 2^V$, we use $R(S) = \max_{p \in S, q \notin S}\{R_{pq}\}$ to denote the requirement of S. Now, the original constraints are of the form

$$\sum_{e \in \delta(S)} u(e)x_e \geq R(S)$$

for each set S, and we define the residual requirement for a set as $R(S, A) = \min\{0, R(S) - u(A \cap \delta(S))\}$. The KC inequalities use this new definition of $R(S, A)$.

Given a fractional solution x to the KC LP, we modify the definitions of highly fractional and nearly integral edges: An edge e is said to be nearly integral if $x_e \geq \frac{1}{40\gamma \log n}$, and highly fractional otherwise. Again, for a fractional solution x, we let A_x denote the set of nearly integral edges; the set $\mathcal{S}$ of small cuts is now $\{S \subseteq V : \hat{u}(\delta(S)) \leq 2\gamma R\}$. From the cut-counting theorem, $|\mathcal{S}| \leq n^{4\gamma}$. We use $\mathcal{L}$ to denote the set of *large* cuts, the sets $\{S \subseteq V : \hat{u}(\delta(S)) > 2\gamma R\}$.

As before, a fractional solution x is *good* if the original constraints are satisfied, and the KC Inequalities are satisfied for the set of edges A_x and the sets in $\mathcal{S}$. These constraints can be checked in time $O(n^{4\gamma+2} \log^2 n)$, so following the proof of Lemma 2.4, for constant γ, we can find a good fractional solution in polynomial time.

The rounding and analysis proceed precisely as before: For each highly fractional edge e, we select it for the final solution with probability $40\gamma \log n \cdot x_e$. The expected cost of this solution is at most $O(\gamma \log n)$ times that of the optimal integral solution, and analogously to the proofs of Lemmas 2.6 and 2.7, one can show that the solution satisfies all cuts with high probability. This completes the proof of Theorem 1.1.

2.4 The k-Way–$\mathcal{R}$-Connected Subgraph Problem

The k-Way–$\mathcal{R}$-Connected Subgraph problem that we define is a natural generalization of the well-studied min-cost λ-edge-connected subgraph problem. The latter problem is motivated by applications to fault-tolerant network design where any $\lambda - 1$ edge failures should not disconnect the graph. However, there may be situations in which global λ-connectivity may be too expensive or infeasible. For example the underlying graph G may have a single cut-edge but we still wish a subgraph that is as close to 2-edge-connected as possible. We could model the requirement by k-Way–$\mathcal{R}$-Connected Subgraph (in the unit-capacity case) by setting $R_1 = 1$ and $R_2 = 3$; that is, at least 3 edges have to be removed to partition the graph into 3 disconnected pieces.

To prove Theorem 1.2, we work with the generalization of (2.1) given below. For any i-way cut $\mathcal{C}$ and for any set of edges A, we use $R(\mathcal{C}, A)$ to be $\max\{0, R_i - u(A \cap \delta(C))\}$.[3]

[3]For ease of notation, we assume that for any edge e, $u(e) \leq R_1$. This is not without loss of generality, but the proof can be trivially generalized: In the constraint for each $i+1$-way cut $\mathcal{C}$ such that $e \in \delta(\mathcal{C})$, simply use the minimum of $u(e)$ and R_i.

$$\min \quad \sum_{e \in E} c(e) x_e \qquad \text{(k-way KC LP)}$$

$$\forall i, \forall i\text{-way cuts } \mathcal{C}, \qquad \sum_{e \in \delta(\mathcal{C})} u(e) x_e \geq R_i \qquad \text{(Original Constraints)}$$

$$\forall A \subseteq E, \forall i, \forall i\text{-way cuts } \mathcal{C}, \qquad \sum_{e \in \delta(\mathcal{C}) \setminus A} \min\{u(e), R(\mathcal{C}, A)\} x_e \geq R(\mathcal{C}, A) \qquad \text{(KC-inequalities)}$$

$$\forall e \in E, \qquad 0 \leq x_e \leq 1$$

As before, given a fractional solution x to this LP, we define A_x (the set of nearly integral edges) to be $\{e \in E \colon x_e \geq \frac{1}{40k \log n}\}$. Define $\hat{u}(e) = u(e)x_e$ to be the fractional capacity on the edges. Let $\mathcal{S}_i := \{\mathcal{C} : \mathcal{C} \text{ is an } i+1\text{-way cut and } \hat{u}(\delta(C)) \leq 2R_i\}$. The solution x is said to be *good* if it satisfies the following three conditions:

(a) If the capacity of e is $\hat{u}(e)$, the capacity of any $i+1$-way cut in G is at least R_i; equivalently x satisfies the original constraints.

(b) The KC inequalities are satisfied for the set A_x and the sets in $\mathcal{S}_i$, for each $1 \leq i \leq k-1$. Note that if (a) is satisfied, then by Lemma 2.8, $|\mathcal{S}_i| \leq n^{4i}$.

(c) $\sum_{e \in E} c(e) x_e$ is at most the value of the optimum solution to the linear program (k-way KC LP).

Following the proof of Lemma 2.4, it is straightforward to verify that there is a randomized algorithm that computes a good fractional solution with high probability in $n^{O(k)}$ time.

Once we have a good fractional solution, our algorithm is to select A_x, the set of nearly integral edges, and to select each highly fractional edge $e \in E \setminus A_x$ with probability $40k \log n \cdot x_e$. If F^* denotes the highly fractional edges that were selected, we return the solution $A_x \cup F^*$. As before, it is trivial to see that the expected cost of this solution is $O(k \log n)$ times that of the optimal integral solution.

We show below that for any $i \leq k-1$, we satisfy all $i+1$-way cuts with high probability; taking the union bound over the $k-1$ choices of i yields the theorem. We will need the following lemma due to Karger.

Lemma 2.8 (Lemma 11.2.1 of [21])**.** *In an n-vertex graph, the number of k-way cuts with capacity at most α times that of a minimum k-way cut is at most $n^{2\alpha(k-1)}$.*

As in Lemmas 2.6 and 2.7, we separately consider the "large" and "small" $i+1$-way cuts. First, consider any small cut $\mathcal{C}$ in $\mathcal{S}_i$. From the Chernoff bound (Lemma 2.5) and the KC inequality for $\mathcal{C}$ and A_x, it follows that the probability we fail to satisfy $\mathcal{C}$ is at most $1/n^{19k}$. From the cut-counting Lemma 2.8, there are at most $n^{4i} < n^{4k}$ such small cuts, so we satisfy all the small $i+1$ way cuts with probability at least $1 - \frac{1}{n^{15k}}$.

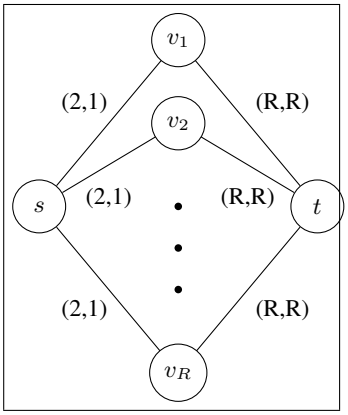
For the large $i+1$-way cuts $\mathcal{L}$, we separately consider cuts of differing capacities. For each $j \geq 2$, let $\mathcal{L}(j)$ denote the $i+1$-way cuts $\mathcal{C}$ such that $jR_i \leq \hat{u}(\mathcal{C}) \leq (j+1)R_i$. Consider any cut $\mathcal{C} \in \mathcal{L}_j$; if $u(A_x \cap \delta(C)) \geq R_i$, then the cut $\mathcal{C}$ is clearly satisfied. Otherwise, $\hat{u}(\delta(\mathcal{C}) \setminus A_x) \geq (j-1)R_i$. But since we selected each edge e in $\delta(\mathcal{C}) \setminus A_x$ for F^* with probability $40k \log n \cdot x_e$, the Chernoff bound implies that we do not satisfy $\mathcal{C}$ with probability at most $\frac{1}{n^{19k(j-1)}}$. The cut-counting Lemma 2.8 implies there are most $n^{2i(j+1)} < n^{2k(j+1)}$ such cuts, so we fail to satisfy any cut in $\mathcal{L}(j)$ with probability at most n^{21-17j}. Taking the union bound over all j, the failure probability is at most $2n^{-13}$.

It would be interesting to explore algorithms and techniques for other more general variants of the k-Way–$\mathcal{R}$-Connected Subgraph problem that we consider here.

2.5 Integrality Gap for the Single Pair Cap-SNDP

We show that for any positive integer R, there exists a single-pair Cap-SNDP instance G with $(R+2)$ vertices such that the integrality gap of the natural LP relaxation strengthened with KC inequalities is $\Omega(R)$. The instance G consists of a source vertex s, a sink vertex t, and R other vertices $v_1, v_2, \ldots, v_R$.

There is an edge of capacity 2 and cost 1 (call these *small* edges) between s and each v_i, and an edge of capacity R and cost R between each v_i and t (*large* edges). We have $R_{st} = R$. Clearly, an optimal integral solution must select at least $R/2$ of the large edges (in addition to small edges), and hence has cost greater than $R^2/2$. The instance is depicted in the accompanying figure: Label (u, c) on an edge denotes capacity u and cost c.



We now describe a feasible LP solution: set $x_e = 1$ on each small edge e, and $x_{e'} = 2/R$ on each large edge e'. The cost of this solution is R from the small edges, and $2R$ from the large edges, for a total of $3R$. This is a factor of $R/6$ smaller than the optimal integral solution, proving the desired integrality gap.

It remains only to verify that this is indeed a feasible solution to (2.1). Consider the constraint corresponding to sets S, A. As edges in $A \setminus \delta(S)$ play no role, we may assume $A \subseteq \delta(S)$. If A includes a large edge, or at least $R/2$ small edges, the residual requirement $R(S, A)$ that must be satisfied by the remaining edges of $\delta(S)$ is 0, and so the constraint is trivially satisfied. Let A consist of $a < R/2$ small edges; the residual requirement is thus $R-2a$. Let $\delta(S)$ contain i large edges and thus $R-i$ small edges. Now, the contribution to the left side of the constraint from small edges in $\delta(S) \setminus A$ is $2(R-i-a) = (R-2a)+(R-2i)$. Therefore, the residual requirement is satisfied by small edges alone unless $i > R/2$. But the contribution of large edges is $i \cdot \frac{2}{R} \cdot (R-2a)$ which is greater than $R-2a$ whenever $i > R/2$. Thus, we satisfy each of the added KC inequalities.

3 Cap-SNDP with Multiple Copies Allowed

We now consider the version of Cap-SNDP when multiple copies of any edge e can be chosen; that is, for any integer $\alpha \geq 0$, α copies of e can be bought at a cost $\alpha \cdot c(e)$ to obtain a capacity of $\alpha \cdot u(e)$. Allowing multiple copies makes the problem easier, and Goemans *et al.* [15] give a $O(\log R_{max})$ factor approximation algorithm for the problem. In this section, we complement this result with a $O(\log k)$ factor approximation algorithm, where k is the number of (i, j) pairs with $R_{ij} > 0$.[4] Our algorithm is inspired by the work of Berman and Coulston [4] on online Steiner Forest. For notational convenience, we rename the pairs $(s_1, t_1), \cdots, (s_k, t_k)$, and denote the requirement R_{s_i,t_i} as R_i; the vertices s_i, t_i are referred to as *terminals*. We also assume that the pairs are so ordered that $R_1 \geq R_2 \geq \cdots \geq R_k$.

We first give an intuitive overview of the algorithm. The algorithm considers the pairs in decreasing order of requirements, and maintains a *forest solution* connecting the pairs that have been already been

[4]Note that we overload the letter 'k', previously used in the definition of the k-Way–$\mathcal{R}$-Connected Subgraph problem; this should cause no ambiguity as we discuss only pairwise connectivity requirements in this section.

processed; that is, if we retain a single copy of each edge in the partial solution constructed so far, we obtain a forest F. For any edge e on the path in F between s_j and t_j, the total capacity of copies of e will be at least R_j. When considering s_i, t_i, we connect them as cheaply as possible, assuming that edges previously selected for F have 0 cost. (Note that this can be done since we are processing the pairs in decreasing order of requirements and for each edge already present in F, the capacity of its copies is at least R_i.) The key step of the algorithm is that *in addition* to connecting s_i and t_i, we also connect the pair to certain other components of F that are "nearby". The cost of these additional connections can be bounded by the cost of the direct connection costs between the pairs. These additional connections are useful in allowing subsequent pairs of terminals to be connected cheaply. In particular, they allow us to prove a $O(\log k)$ upper bound on the approximation factor.

We now describe the algorithm in more detail. The algorithm maintains a forest F of edges that have already been bought; F satisfies the invariant that, after iteration $i-1$, for each $j \leq i-1$, F contains a unique path between s_j and t_j. In iteration i, we consider the pair s_i, t_i. We define the cost function $c_i(e)$ as $c_i(e) := 0$ for edges e already in F, and $c_i(e) := c(e) + \frac{R_i}{u(e)}c(e)$, for edges $e \notin F$. Note that for an edge $e \notin F$, the cost $c_i(e)$ is sufficient to buy enough copies of e to achieve a total capacity of R_i. Thus it suffices to connect s_i and t_i and pay cost $c_i(e)$ for each edge; in the Cap-SNDP solution we would pay at most this cost and get a feasible solution. However, recall that our algorithm also connects s_i and t_i to other "close" components; to describe this process, we introduce some notation:

For any vertices p and q, we use $d_i(p, q)$ to denote the distance between p and q according to the metric given by edge costs $c_i(e)$. We let $\ell_i := d_i(s_i, t_i)$ be the cost required to connect s_i and t_i, given the current solution F. We also define the *class* of a pair (s_j, t_j), and of a component:

- For each $j \leq i$, we say that pair (s_j, t_j) is in *class* h if $2^h \leq \ell_j < 2^{h+1}$.
  Equivalently, $\texttt{class}(j) = \lfloor \log \ell_j \rfloor$.
- For each connected component X of F, $\texttt{class}(X) = \max_{(s_j,t_j)\in X} \texttt{class}(j)$.

Now, the algorithm connects s_i (respectively t_i) to component X if $d_i(s_i, X)$ (resp. $d_i(t_i, X)$) $\leq 2^{\min\{\texttt{class}(i),\texttt{class}(X)\}}$. That is, if X is close to the pair (s_i, t_i) compared to the classes they are in, we connect X to the pair. As we show in the analysis, this extra connection cost can be charged to some pair (s_j, t_j) in the component X. The complete algorithm description is given below.

CAP-SNDP-MC:
$F \leftarrow \emptyset$ $\langle\langle F$ *is the forest solution returned* $\rangle\rangle$
For $i \leftarrow 1$ to k
  For each edge $e \in F$, $c_i(e) \leftarrow 0$
  For each edge $e \notin F$, $c_i(e) \leftarrow c(e) + (R_i/u(e))c(e)$
  $\ell_i \leftarrow d_i(s_i, t_i)$
  Add to F a shortest path (of length ℓ_i) from s_i to t_i under distances $c_i(e)$
  $\texttt{class}(i) \leftarrow \lfloor \log \ell_i \rfloor$
  For each connected component X of F
    If $d_i(s_i, X) \leq 2^{\min\{\texttt{class}(i),\texttt{class}(X)\}}$
      Add to F a shortest path connecting s_i and X
  For each connected component X of F
    If $d_i(t_i, X) \leq 2^{\min\{\texttt{class}(i),\texttt{class}(X)\}}$
      Add to F a shortest path connecting t_i and X
  Buy $\lceil R_i/u_e \rceil$ copies of each edge e added during this iteration.

The structure of our proof is as follows: Recall that ℓ_i was the direct connection cost between s_i and t_i; in addition to paying ℓ_i to connect these vertices, the algorithm also buys additional edges connecting s_i and t_i to existing components. We first show (in Lemma 3.1) that the total cost of extra edges bought can be charged to the direct connection costs; thus, it suffices to show that $\sum_i \ell_i \leq O(\log k)\text{OPT}$, where OPT is the cost of an optimal solution. To prove this (Lemma 3.2), we bucket the pairs (s_i, t_i) into $O(\log k)$ groups based on $\texttt{class}(i)$, and show that in each bucket h, $\sum_{i:\texttt{class}(i)=h} \ell_i \leq O(\text{OPT})$.

Lemma 3.1. *The total cost of all edges bought by* CAP-SNDP-MC *is at most* $9\sum_{i=1}^{k} \ell_i$.

Proof: Let F_i denote the set of edges added to F during iteration i. First, note the total cost paid for copies of edge $e \in F_i$ is $\lceil \frac{R_i}{u(e)} \rceil c(e) < c(e) + \frac{R_i}{u_e} c(e) = c_i(e)$. Thus, it suffices to show:

$$\sum_{i=1}^{k} \sum_{e \in F_i} c_i(e) \leq 9 \sum_{i=1}^{k} \ell_i$$

We prove that the total cost of the *additional* edges bought is at most $8\sum_{i=1}^{k} \ell_i$; this clearly implies the desired inequality. It is *not* true that for each i, the total cost of additional edges bought during iteration i is at most $8\ell i$. Nonetheless, a careful charging scheme proves the needed bound on total cost. In iteration i, suppose we connect the pair (s_i, t_i) to the components $X_1, \ldots, X_r$. We charge the cost of connecting (s_i, t_i) and component X_j to the connection cost ℓ_j of a pair (s_j, t_j) in X_j. This is possible since we know the additional connection cost is at most $2^{\texttt{class}(X_j)}$. Care is required to ensure no pair is overcharged. To do so, we introduce some notation.

At any point during the execution of the algorithm, for any current component X of F, we let $\texttt{Leader}(X)$ be a pair $(s_i, t_i) \in X$ such that $\texttt{class}(i) = \texttt{class}(X)$. For integers $h \leq \texttt{class}(X)$, $h\text{-}\texttt{Leader}(X)$ will denote a pair (s_j, t_j) in X; we explain how this pair is chosen later. (Initially, $h\text{-}\texttt{Leader}(X)$ is undefined for each component X.)

Now, we have to account for additional edges bought during iteration i; these are edges on a shortest path connecting s_i (or t_i) to some other component X; we assume w.l.o.g. that the path is from s_i to X. Consider any such path P connecting s_i to a component X; we have $\sum_{e \in P} c_i(e) = d_i(s_i, X) \leq 2^{\min\{\texttt{class}(i), \texttt{class}(X)\}}$. Let $h = \lfloor \log d_i(s_i, X) \rfloor$: Charge all edges on this path to $h\text{-}\texttt{Leader}(X)$ if it is defined; otherwise, charge all edges on the path to $\texttt{Leader}(X)$. In either case, the pair (s_i, t_i) becomes the $h\text{-}\texttt{Leader}$ of the new component just formed. Note that a pair (s_i, t_i) could simultaneously be the h_1-$\texttt{Leader}$, h_2-$\texttt{Leader}$, etc. for a component X if (s_i, t_i) connected to many components during iteration i. However, it can never be the $h\text{-}\texttt{Leader}$ of a component for $h > \texttt{class}(i)$, and once it has been charged as $h\text{-}\texttt{Leader}$, it is never charged again as $h\text{-}\texttt{Leader}$. Also observe that if a pair is in a component X whose $h\text{-}\texttt{Leader}$ is defined, subsequently, it always stays in a component in which the $h\text{-}\texttt{Leader}$ is defined.

For any i, we claim that the total charge to pair (s_i, t_i) is at most $8\ell_i$, which completes the proof. Consider any such pair: any charges to the pair occur when it is either $\texttt{Leader}$ or $h\text{-}\texttt{Leader}$ of its current component. First, consider charges to (s_i, t_i) as $\texttt{Leader}$ of a component. Such a charge can only occur when connecting some s_j (or t_j) to X. Furthermore, if $h = \lfloor \log d_j(s_j, X) \rfloor \leq \texttt{class}(X) = \texttt{class}(i)$, the $h\text{-}\texttt{Leader}(X)$ must be *currently undefined*, for otherwise the $h\text{-}\texttt{Leader}(X)$ would have been charged. Subsequently, the $h\text{-}\texttt{Leader}$ of the component containing (s_i, t_i) is always defined, and so (s_i, t_i) will never again be charged as a $\texttt{Leader}(X)$ by a path of length in $[2^h, 2^{h+1})$. Therefore, the total charge to (s_i, t_i) as $\texttt{Leader}$ of a component is at most $\sum_{h=1}^{\texttt{class}(i)} 2^{h+1} < 2^{\texttt{class}(i)+2} \leq 4\ell_i$.

Finally, consider charges to (s_i, t_i) as h−Leader of a component. As observed above, $h \leq \texttt{class}(i)$. Also for a fixed h, a pair is charged at most once as h−Leader. Since the total cost charged to (s_i, t_i) as h−Leader is at most 2^{h+1}; summing over all $h \leq \texttt{class}(i)$, the total charge is less than $2^{\texttt{class}(i)+2} = 4\ell_i$.

Thus, the total charge to (s_i, t_i) is at most $4\ell_i + 4\ell_i = 8\ell_i$, completing the proof. □

Lemma 3.2. *If* OPT *denotes the cost of an optimal solution to the instance of Cap-SNDP with multiple copies, then* $\sum_{i=1}^{k} \ell_i \leq 64(\lceil \log k \rceil + 1)$OPT.

Proof: Let C_h denote $\sum_{i:\texttt{class}(i)=h} \ell_i$. Clearly, $\sum_{i=1}^{k} \ell_i = \sum_h C_h$. The lemma follows from the two sub-claims below:

Sub-Claim 1: $\sum_h C_h \leq (2(\lceil \log k \rceil + 1)) \cdot \max_h C_h$

Sub-Claim 2: For each h, $C_h \leq 32$OPT.

Proof of Sub-Claim 1: Let $h' = \max_i \texttt{class}(i)$. We have $C_{h'} \geq 2^{h'}$, and for any terminal i such that $\texttt{class}(i) \leq h' - (\lceil \log k \rceil + 1)$, we have $\ell_i \leq \frac{2^{h'+1}}{2k}$. Thus, the total contribution from such classes is at most $\frac{2^{h'}}{k} \cdot k = 2^{h'}$, and hence:

$$\sum_{h=h'-\lceil \log k \rceil}^{h'} C_h \geq \frac{\sum_h C_h}{2}, \text{ which implies}$$

$$\max_{h'-\lceil \log k \rceil \leq h \leq h'} C_h \geq \frac{\sum_h C_h}{2(\lceil \log k \rceil + 1)}.$$

□

It remains to show Sub-Claim 2, that for each h, $C_h \leq 32$OPT. Fix h. Let $\mathcal{S}_h$ denote the set of pairs s_i, t_i such that $\texttt{class}(i) = h$. Our proof will go via the natural primal and dual relaxations for the Cap-SNDP problem. In particular, we will exhibit a solution to the dual relaxation of cost $C_h/32$. To do so we will require the following claim. Define $\texttt{ball}(s_i, r)$, a *ball* of radius r around s_i as containing the set of vertices v such that $d_i(s_i, v) \leq r$ and the set of edges $e = uv$ such that $d_i(s_i, \{u, v\}) + c_i(e) \leq r$. An edge e is *partially* within the ball if $d_i(s_i, \{u, v\}) < r < d_i(s_i, \{u, v\}) + c_i(e)$. Subsequently, we assume for ease of exposition that no edges are partially contained within the balls we consider; this can be achieved by subdividing the edges as necessary. Similarly, we define $\texttt{ball}(t_i, r)$, the ball of radius r around t_i. Two balls are said to be *disjoint* if they contain no common vertices.

Claim 3.3. *There exists a subset of pairs,* $\mathcal{S}'_h \subseteq \mathcal{S}_h$, $|\mathcal{S}'_h| \geq |\mathcal{S}_h|/2$, *and a collection of* $|\mathcal{S}'_h|$ *disjoint balls of radius* $2^h/4$ *centred around either* s_i *or* t_i, *for every pair* $(s_i, t_i) \in \mathcal{S}'_h$.

We prove this claim later; we now use it to complete the proof of Sub-Claim 2. First we describe the LP. Let the variable x_e denote whether or not edge e is in the Cap-SNDP solution. Let $\mathcal{P}_i$ be the set of paths from s_i to t_i. For each $P \in \mathcal{P}_i$, variable f_P denotes how much flow t sends to the root along path P. We use $u_i(e)$ to refer to $\min\{R_i, u(e)\}$, the effective capacity of edge e for pair (s_i, t_i).

$$\begin{array}{|rcll|}
\multicolumn{4}{|l|}{\textbf{Primal}\quad \min \sum_{e\in E} c_e x_e} \\
\sum_{P\in\mathcal{P}_i} f_P & \geq & R_i & (\forall i \in [k]) \\
\sum_{P\in\mathcal{P}_t | e\in P} f_P & \leq & u_i(e)x_e & (\forall i \in [k], e \in E) \\
x_e, f_P & \geq & 0 & \\
\hline
\end{array}
\qquad
\begin{array}{|rcll|}
\multicolumn{4}{|l|}{\textbf{Dual}\quad \max \sum_{t\in T} R_i \alpha_i} \\
\sum_i u_i(e)\beta_{i,e} & \leq & c_e & (\forall e \in E) \\
\alpha_i & \leq & \sum_{e\in P} \beta_{i,e} & (\forall i \in [k], P \in \mathcal{P}_i) \\
\alpha_i, \beta_{i,e} & \geq & 0 & \\
\hline
\end{array}$$

We now describe a feasible dual solution of value at least $C_h/32$ using Claim 3.3. For $(s_i, t_i) \in \mathcal{S}'_h$, if there is a ball B around s_i (or equivalently t_i), we define $\beta_{i,e} = c(e)/u_i(e)$ for each edge in the ball. Since the balls are disjoint, the first inequality of the dual is clearly satisfied. Set $\alpha_i = 2^h/8R_i$. For any path $P \in \mathcal{P}_i$, we have

$$\sum_{e\in P} \beta_{i,e} = \frac{1}{R_i} \sum_{e\in P\cap B} \frac{R_i c(e)}{u_i(e)} \geq \frac{1}{2R_i} \sum_{e\in P\cap B} \frac{R_i c(e)}{u(e)} + c(e) \geq \frac{1}{2R_i} \sum_{e\in P\cap B} c_i(e) \geq \frac{1}{2R_i}\frac{2^h}{4} = \alpha_i$$

where the first inequality used $u_i(e) \leq R_i$, the second follows from the definition of $c_i(e)$, and the last inequality follows from the definition of $\texttt{ball}(s_i, 2^h/4)$. Thus, $\alpha_i = 2^h/8R_i$ is feasible along with these $\beta_{i,e}$'s. This gives a total dual value of

$$\frac{2^h}{8} \cdot |\mathcal{S}'_h| \geq \frac{2^h}{16} \cdot |\mathcal{S}_h| \geq \frac{1}{32} \sum_{i\in\mathcal{S}_h} \ell_i = \frac{C_h}{32}$$

where the last inequality follows from the fact that $\texttt{class}(i) = h$. This proves the lemma modulo Claim 3.3, which we now prove.

Proof of Claim 3.3: We process the pairs in $\mathcal{S}_h$ in the order they are processed by the original algorithm and grow the balls. We abuse notation and suppose these pairs are $(s_1, t_1), \ldots, (s_p, t_p)$. We maintain a collection of disjoint balls of radius $r = 2^h/4$, initially empty.

At stage i, we try to grow a ball of radius r around either s_i or t_i. If this is not possible, the ball around s_i intersects that around some previous terminal in $\mathcal{S}'_h$, say s_j; similarly, the ball around t_i intersects that of a previous terminal, say t_ℓ. Let v be a vertex in $\texttt{ball}(s_i, r)$ and $\texttt{ball}(s_j, r)$. We have $d_i(s_i, s_j) \leq d_i(s_i, v) + d_i(v, s_j) \leq d_i(s_i, v) + d_j(v, s_j) < 2^h/2$. (The second inequality follows because for any $j < i$ and any edge e, $c_i(e) \leq c_j(e)$.) Similarly, we have $d_i(t_i, t_\ell) < 2^h/2$.

Now, we observe that s_j and t_ℓ could not have been in the same component of F at the beginning of iteration i of CAP-SNDP-MC; otherwise $d_i(s_i, t_i) \leq d_i(s_i, s_j) + d_i(t_i, t_\ell) < 2^h$, contradicting that $\texttt{class}(i) = h$. But since $d_i(s_i, s_j) \leq 2^h/2$ and $\texttt{class}(i) = \texttt{class}(j) = h$, we connect s_i to the component of s_j during iteration i; likewise, we connect t_i to the component of t_ℓ during this iteration. Hence, at the end of the iteration, s_i, t_i, s_j, t_ℓ are all in the same component. As a result, the number of components of F containing pairs of $\mathcal{S}_h$ *decreases* by at least one during the iteration.

It is now easy to complete the proof: During any iteration of F corresponding to a pair $(s_i, t_i) \in \mathcal{S}_h$, the number of components of F containing pairs of $\mathcal{S}_h$ can go up by at most one. Say that an iteration *succeeds* if we can grow a ball of radius r around either s_i or t_i, and *fails* otherwise. During any iteration that fails, the number of components decreases by at least one; as the number of components is always non-negative, the number of iterations which fail is no more than the number which succeed. That is, $|\mathcal{S}'_h| \geq |\mathcal{S}_h - \mathcal{S}'_h|$. □

□

Theorem 1.4 is now a straightforward consequence of Lemmas 3.1 and 3.2. We recall the statement.

Theorem 1.4. *In undirected graphs, there is an $O(\log k)$-approximation algorithm for Cap-SNDP with multiple copies, where k is the number of pairs with $R_{ij} > 0$.*

Proof of Theorem 1.4: The total cost of edges bought by the algorithm is at most $\sum_{i=1}^{k} \sum_{e \in F_i} c_i(e) \leq 9 \sum_{i=1}^{k} \ell_i$, by Lemma 3.1. But $\sum_{i=1}^{k} \ell_i \leq 64(\lceil \log k \rceil + 1)\text{OPT}$, by Lemma 3.2, and hence the total cost paid by CAP-SNDP-MC is at most $O(\log k)\text{OPT}$. □

3.1 Hardness of Approximation for Cap-SNDP in Undirected Graphs

In this section, we prove Theorem 1.5, $\Omega(\log \log n)$ hardness for the single-source version of Cap-SNDP with multiple copies. We use a reduction from the *Priority Steiner Tree* problem. A similar hardness also applies to the basic Cap-SNDP problem, as the copies of edges do not play a significant role in the reduction; we omit the details here.

In the Priority Steiner Tree problem, the input is an undirected graph $G(V, E)$ with a cost $c(e)$ and a priority $P(e) \in \{1, 2, \ldots, k\}$ for each edge e. (We assume k is the highest and 1 the lowest priority.) We are also given a root r and a set of terminals $T \subseteq V - \{r\}$; each terminal $t \in T$ has a desired priority $P(t)$. The goal is to find a minimum-cost Steiner Tree in which the unique path from each terminal t to the root consists only of edges of priority $P(t)$ or higher.[5]

Chuzhoy *et al.* [9] showed that one cannot approximate the Priority Steiner Tree problem within a factor better than $\Omega(\log \log n)$ unless $NP \subseteq DTIME(n^{\log \log \log n})$, even when all edge costs are 0 or 1. We now show an approximation-preserving reduction from this problem to Cap-SNDP with multiple copies.

Given an instance $\mathcal{I}_{pst}$ of Priority Steiner Tree on graph $G(V, E)$ with edge costs in $\{0, 1\}$, we construct an instance $\mathcal{I}_{cap}$ of Cap-SNDP defined on the graph G as the underlying graph. Fix R to be any integer greater than $2m^3$ where m is the number of edges in the graph G. We now assign a capacity of $u(e) = R^i$ to each edge e with priority $P(e) = i$ in $\mathcal{I}_{pst}$. Each edge e of cost 0 in $\mathcal{I}_{pst}$ has cost $c(e) = 1$ in $\mathcal{I}_{cap}$, and each edge e of cost 1 in $\mathcal{I}_{pst}$ has cost $c(e) = m^2$ in $\mathcal{I}_{cap}$. Finally, for each terminal t, set $R_{tr} = R^i$ if $P(t) = i$; for every other pair of vertices (p, q), $R_{pq} = 0$.

Let C denotes the cost of an optimal solution to $\mathcal{I}_{pst}$; note that $C \leq m$; we now argue that $\mathcal{I}_{pst}$ has an optimal solution of cost C iff $\mathcal{I}_{cap}$ has an optimal solution of of cost between Cm^2 and $Cm^2 + m < (C + 1)m^2$. Given a solution E^* to $\mathcal{I}_{pst}$ of cost C, simply select the same edges for $\mathcal{I}_{cap}$; the cost in $\mathcal{I}_{cap}$ is at most $Cm^2 + m$ since in $\mathcal{I}_{cap}$, we pay 1 for each edge in E^* that has cost 0 in $\mathcal{I}_{pst}$. This is clearly a feasible solution to $\mathcal{I}_{cap}$ as each terminal t has a path to r in E^* containing only edges with priority at least $P(t)$, which is equivalent to having capacity at least R_{tr}. Conversely, given a solution E' to $\mathcal{I}_{cap}$ with cost in $[Cm^2, (C + 1)m^2)$, select a single copy of each edge in E' as a solution to $\mathcal{I}_{pst}$; clearly the total cost is at most C. To see that this is a feasible solution, suppose that E' did not contain a path from some terminal t to the root r using edges of priority $P(t)$ or more. Then there must be a cut separating t from r in which all edges of E' have capacity at most $R^{P(t)-1}$. But since E' supports a flow of $R^{P(t)}$ from t to r, it must use at least R edges (counting with multiplicity); this implies that the cost of E' is at least $R \geq (C + 1)m^2$, a contradiction.

[5] It is easy to see that a minimum-cost subgraph containing such a path for each terminal is a tree; given any cycle, one can remove the edge of lowest priority.

The reduction above proved $\Omega(\log \log n)$ hardness for the *single-source* version of Cap-SNDP (with or without multiple copies). A similar reduction also proves $\Omega(\log \log n)$ hardness for the *single-pair* Cap-SNDP problem if multiple copies are not allowed: One can effectively encode an instance of the single-source Fixed-Charge Network Flow (FCNF, [9]), very similar to single-source Cap-SNDP with multiple copies, as an instance of single-pair Cap-SNDP *without* multiple copies: Create a new sink t^*, and connect t^* to each original terminal t with a single edge of cost 0 and capacity R_{tr}. The only way to send flow $\sum_{t \in T} R_{tr}$ flow from t^* to the source s is for each terminal t to send R_{tr} to s. Thus, $\Omega(\log \log n)$ hardness for single-pair Cap-SNDP (Theorem 1.3) is a simple consequence of the $\Omega(\log \log n)$ hardness for single-source FCNF [9].

4 Conclusions

In this paper we made progress on addressing the approximability of Cap-SNDP. We gave an $O(\log n)$ approximation for the Cap-R-Connected Subgraph problem, which is a capacitated generalization of the well-studied min-cost λ-edge-connected subgraph problem. Can we improve this to obtain an $O(1)$ approximation or prove super-constant hardness of approximation? We also highlighted the difficulty of Cap-SNDP by focusing on the single pair problem, and showing both super-constant hardness and an $\Omega(n)$ integrality gap example, even for the LP with KC inequalities. Recent results (see Remark 1.6) show that the single pair problem is essentially as hard as the label cover problem. It may be useful to examine special cases or assumptions that allow one to bypass these strong negative results. As we noted, allowing multiple copies of edges makes the problem easier; in practice, however, it may be desirable to not allow too many copies of an edge to be used. It is therefore of interest to examine the approximability of Cap-SNDP if we allow only a small number of copies of an edge. Does the problem admit a non-trivial approximation if we allow $O(1)$ copies or, say, $O(\log n)$ copies? This investigation may further serve to delineate the easy versus difficult cases of Cap-SNDP.

Acknowledgements: CC's interest in capacitated network design was inspired by questions from Matthew Andrews. He thanks Mathew Andrews and Lisa Zhang for several useful discussions on their work on capacitated network design for multi-commodity flows.

References

[1] M. Andrews. Hardness of Buy-at-Bulk Network Design. In *Proceedings, IEEE Foundations of Computer Science (FOCS)*, pp 115–124, 2004.

[2] M. Andrews, S. Antonakopoulos, and L. Zhang. Minimum-Cost Network Design with (Dis)economies of Scale. In *Proceedings, IEEE Foundations of Computer Science (FOCS)*, pp 585– 592, 2010.

[3] A. Atamturk. On Capacitated Network Design Cut-Set Polyhedra. Mathematical Programming 92, 425-437, 2002.

[4] P. Berman and C. Coulston. On-Line Algorithms for Steiner Tree Problems (Extended Abstract). In *Proceedings, ACM Symposium on Theory of Computation (STOC)*, pp. 344–353, 1997.

[5] R. D. Carr, L. K. Fleischer, V. J. Leung, and C. A. Phillips. Strengthening integrality gaps for capacitated network design and covering problems. In *Proceedings, ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 106–115, 2000.

[6] D. Chakrabarty, R. Krishnaswamy, S. Li, and S. Narayanan. Capacitated Network Design in Undirected Graphs. *Unpublished Manuscript as of April 2013.*

[7] M. Charikar, J. Naor, and B. Schieber. Resource optimization in QoS multicast routing of real-time multimedia. *IEEE/ACM Trans. Netw.* 12(2):340–348, 2004.

[8] C.Chekuri, M. T. Hajiaghayi, G. Kortsarz, and M. R. Salavatipour. Approximation Algorithms for Non-Uniform Buy-at-Bulk Network Design. *SIAM Journal of Computing*, 39(5):1772–1798, 2009.

[9] J. Chuzhoy, A. Gupta, J. Naor and A. Sinha. On the approximability of some network design problems. *ACM Transactions on Algorithms*, 4(2), 2008.

[10] J. Chuzhoy and S. Khanna. Algorithms for single-source vertex connectivity. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 105–114, 2008.

[11] J. Chuzhoy and S. Khanna. An $O(k^3 \log n)$-Approximation Algorithm for Vertex-Connectivity Survivable Network Design. *Theory of Computing*, 8(1):401–413, 2012.

[12] G. Even, G. Kortsarz, and W. Slany. On network design: fixed charge flows and the covering Steiner problem. *ACM Transaction on Algorithms*, 1(1):74-101, 2005.

[13] C. G. Fernandes. A better approximation ratio for the minimum k-edge-connected spanning subgraph problem. In *Proceedings, ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 629–638, 1997.

[14] L. Fleischer, K. Jain, and D.P. Williamson. Iterative rounding 2-approximation algorithms for minimum-cost vertex connectivity problems. *Journal of Computer and System Sciences*, 72(5):838–867, 2006.

[15] M. X. Goemans, A. V. Goldberg, S. A. Plotkin, D. B. Shmoys, É. Tardos, and D. P. Williamson. Improved Approximation Algorithms for Network Design Problems In *Proceedings, ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 223–232, 1994.

[16] M. T. Hajiaghayi, R. Khandekar, G. Kortsarz, and G. Nutov. Capacitated Network Design Problems: Hardness, Approximation Algorithms, and Connections to Group Steiner Tree. *Unpublished manuscript as of April 2013.*

[17] E. Halperin, and R. Krauthgamer. Polylogarithmic Inapproximability. In *Proceedings, ACM Symposium on Theory of Computation (STOC)*, pp. 585–594, 2003.

[18] M. Hewitt, G. Nemhauser, and M. W. P. Savelsbergh. Combining Exact and Heuristic Approaches for the Capacitated Fixed-Charge Network Flow Problem INFORMS Journal on Computing, Volume 22 Issue 2, Spring 2010

[19] D. Hochbaum, and A. Segev. Analysis of a flow problem with fixed charges. *Networks*, 19(3): 291–312, 1989.

[20] K. Jain. A Factor 2 Approximation Algorithm for the Generalized Steiner Network Problem *Combinatorica*, 21(1): 39–60, 2001.

[21] David Karger. Random Sampling in Graph Optimization Problems. Ph.D. Thesis, Stanford University, 1994.

[22] G. Kortsarz and Z. Nutov. Approximating minimum cost connectivity problems. In T.F. Gonzalez, editor, *Handbook of Approximation algorithms and Metaheuristics*. CRC Press, 2007.

[23] R. Motwani, and P. Raghavan. Randomized Algorithms. *Cambridge University Press*, 1995.

[24] Z. Nutov. Approximating minimum cost connectivity problems via uncrossable bifamilies and spider-cover decompositions. In *Proceedings of the fiftieth Annual IEEE Symposium on Foundations of Computer Science*, pages 417–426. IEEE, 2009.

[25] G. Nemhauser, and L. A. Wolsey. Integer and Combinatorial Optimization, Section II.6.4. *John Wiley and Sons.*, 1988.

[26] F. Ortega, and L. A. Wolsey. A branch-and-cut algorithm for the single-commodity, uncapacitated, fixed-charge network flow problem. *Networks*, 41(3): 143–158, 2003.