Cut Problems in Graphs with a Budget Constraint

Roee Engelberg[1], Jochen Könemann[2] ⋆, Stefano Leonardi[3], and Joseph (Seffi) Naor[4] ⋆⋆

[1] Computer Science Department, Technion, Haifa 32000, Israel
roee@cs.technion.ac.il
[2] Department of Combinatorics and Optimization, University of Waterloo, 200 University Avenue West, Waterloo, ON N2L 3G1, Canada
jochen@math.uwaterloo.ca
[3] Dipartimento di Informatica e Sistemistica, Università di Roma “La Sapienza”, Via Salaria 113, 00198 Roma, Italy
leon@dis.uniroma1.it
[4] Microsoft Research, One Microsoft Way, Redmond, WA 98052, USA
naor@cs.technion.ac.il

Abstract. We study budgeted variants of classical cut problems: the *Multiway Cut* problem, the *Multicut* problem, and the k-*Cut* problem, and provide approximation algorithms for these problems. Specifically, for the budgeted multiway cut and the k-cut problems we provide constant factor approximation algorithms. We show that the budgeted multicut problem is at least as hard to approximate as the sparsest cut problem, and we provide a bi-criteria approximation algorithm for it.

1 Introduction

Given an undirected graph $G = (V, E)$ with a positive cost function on the edges $c : E \to \mathbb{Z}^+$, and a subset of vertices $S \subseteq V$, called *terminals*, the well-known *multiway cut* problem is to find a minimum cost subset of edges whose removal disconnects the terminals from each other. The study of the multiway cut problem was initiated by Dahlhaus et al. [6], who proved that it is MAX-SNP-hard even when restricted to instances with 3 terminals and unit edge cost. They also gave a $(2-\frac{2}{k})$-approximation algorithm for the problem, where $|S| = k$.

In [4], Călinescu et al. introduced a $(1.5-\frac{1}{k})$-approximation algorithm. They considered a linear programming relaxation for the multiway cut problem which embeds the given graph into the $(k-1)$-dimensional simplex. The algorithm of [4] rounds an optimal solution to the linear programming relaxation; its bound was later improved to ~ 1.3438 by [11].

In this paper we study two budgeted variants of the multiway cut problem that differ in their objective function. In the budgeted variants, given an instance

⋆ This work was done while being on leave at the Dipartimento di Informatica e Sistemistica at Università di Roma “La Sapienza”, Italy.
⋆⋆ On leave from the Computer Science Department, Technion, Haifa 32000, Israel.

of the multiway cut problem together with an additional positive integer B, the *budget*, the problem is to find a subset of edges whose cost is within the given budget and whose removal maximizes the value of the given objective function.

We say that a pair of terminals (s_i, s_j) is *separated* if there is no path between s_i and s_j, and that a terminal s_i is *isolated* if there is no path between s_i and any other terminal. The number of *isolated* terminals is the objective function of the first *budgeted* variant of the multiway cut problem, referred to as the *budgeted isolating multiway cut* (*BIMC*) problem. In the second *budgeted* variant, referred to as the *budgeted separating multiway cut* (*BSMC*) problem, the objective function is the number of *separated* pairs of terminals. We also consider the *weighted* versions of both *BSMC* and *BIMC*.

An application of the weighted BSMC problem is network design against denial-of-service attacks in networks. In [3], Aura et al. suggest a formal framework for the study of the single-server inhibition attack, which is a common scenario for modelling a denial of service attack. One of the problems they consider is finding the best attack whose cost is within a given budget constraint. In this problem, every client has a non-zero weight denoting its importance. The cost of an attack is the total cost of the disconnected links in the network, and the *value* of the attack is the total weight of the clients separated from the given server. This problem can be considered as a weighted BSMC by setting the weight of every (server, client) pair to be the client's weight.

A well known generalization of the multiway cut problem is the *multicut* problem, which is the problem of finding a minimum cost cut separating a given set of source-sink pairs of vertices. Indeed, the multiway cut problem is a special case of the multicut problem in which the set of source-sink pairs consists of all the pairs of a given set of terminals. Consider the following budgeted variant of the multicut problem. Given is a set of source-sink pairs of vertices together with a budget. Let the source-sink pairs be associated with a non-negative weight. The goal is to find a cut whose cost is within the budget that separates a maximum weight set of source-sink pairs. Thus, this budgeted multicut problem is precisely the weighted version of the BSMC problem.

Finally, given an undirected graph, we consider the problem of finding a set of edges whose cost is within a given budget and whose removal partitions the graph into a maximum number of connected components. This problem, referred to as the *budgeted graph disconnection* (*BGD*) problem, can be thought of as the budgeted version of the *k-cut* problem. In the k-cut problem, an integer k is given and the goal is to find a minimum cost edge set whose removal partitions the graph into at least k connected components.

1.1 Our Results

The hardness of the multiway cut problem implies that both BIMC and BSMC cannot be efficiently solved unless $P = NP$. Although the problem definitions of BIMC and BSMC are closely related, they capture different aspects of the theory of cuts, and therefore differ in their level of hardness. Thus, we study each of the problems independently.

For the BIMC and weighted BIMC problems we give constant factor approximation algorithms that match some of the lower bounds we prove. Our algorithms basically use a greedy approach. In the weighted case we improve on the greedy approach by using an FPTAS for the *knapsack* problem.

We show that weighted BSMC/budgeted multicut is at least as hard to approximate as the *Sparsest Cut* problem is.[5] We show that a natural linear programming relaxation has an unbounded integrality gap. Nevertheless, based on this relaxation, we introduce a constant factor approximation algorithm for the weighted BSMC on *trees*, which implies a constant integrality gap of the relaxation for tree instances. We further note that a better constant factor approximation can be obtained for the weighted BSMC on *trees* through the work of Sviridenko [15]. We then consider the weighted BSMC problem on general graphs. We achieve a bi-criteria approximation of $(\frac{e}{e-1}, O(\log^2 n \log\log n))$ using a recent hierarchical decomposition of graphs by Räcke (see [13] and [9]).

Interestingly, we show that BSMC is related to the budgeted variant of the *Sparsest Cut* problem. Specifically, we prove that for certain weight functions, an approximation algorithm for BSMC can be used to derive an approximation algorithm for the budgeted sparsest cut problem, and vice versa.

Finally, we give a constant factor approximation algorithm for BGD by using the Gomory-Hu tree (see [8]). Our algorithm uses ideas similar to those of the algorithm of Saran and Vazirani [14] for the k-cut problem.

1.2 Related Work

To the best of our knowledge, all of the above mentioned *budgeted* cut problems are studied for the first time here. Nevertheless, there is a vast literature on budgeted optimization problems and we mention the following relevant works.

Vohra and Hall [16] considered a budgeted variant for the classical *set cover* problem, while Khuller et al. [12] studied its weighted variant. They gave a constant factor approximation algorithm for the problem that is based on the greedy approach, and showed that their result is tight under a (weak) assumption on the hardness of NP. Their result points out the possible gap between the hardness of a problem and the hardness of its budgeted variant, as the *set cover* problem cannot be approximated within a factor of $(1-\epsilon)\ln n$ for any $\epsilon > 0$ under the same assumption on the hardness of NP. By improving a former work by Wolsey [17], Sviridenko [15] generalized the result of Khuller et al. for the problem of maximizing any submodular function subject to a budget constraint. We note that this framework does not capture most of the problems we deal with in this paper, but it does capture the weighted BSMC on *trees*.

[5] For the sake of comparison, we note the recent series of results regarding the sparsest cut problem. In [1] an $O(\sqrt{\log n})$ approximation is presented for the uniform case. For the general sparsest cut problem, [5] gave an $O(\log^{\frac{3}{4}} k)$-approximation, which was improved to an $O(\sqrt{\log k}\log\log k)$-approximation by [2].

2 Preliminaries

In this section we formally define the problems considered in this paper. In all of these problems, we are given an undirected graph $G = (V, E)$ with a positive cost function on the edges $c : E \to \mathbb{Z}^+$, and a positive *budget* B.

Problem 1 (Budgeted Graph Disconnection (BGD)). Find a subset of edges $C \subseteq E$ of cost at most B whose removal partitions the graph into the maximum number of connected components.

In the following problems, we are additionally given a subset of vertices $S \subseteq V$ (let $k = |S|$), called *terminals*.

Definition 1 (*Separation and Isolation*). *Let $S \subseteq V$ be a set of terminals. Given a subset of edges $C \subseteq E$, we say that vertices s and s' ($s' \neq s$) are* separated *by C, or, equivalently, that C is a* separating cut *of (s, s'), if every path between s and s' contains* at least *one edge from C. We say that a vertex $s \in S$ is* isolated *by C, or equivalently, that C is an* isolating cut *of s, if for every $s' \in S$, $s' \neq s$, s and s' are separated by C.*

Definition 2. *Given a weight function on the terminals, $w : S \to \mathbb{Z}^+$, the* isolation weight *of a given subset of edges $C \subseteq E$, is the sum of the weights of the terminals isolated by C. Given a weight function on the pairs of terminals, $w : S \times S \to \mathbb{Z}^+$, the* separation weight *of a given subset of edges $C \subseteq E$, is the sum of the weights of the pairs of terminals separated by C.*

Problem 2 (Weighted Budgeted Isolating Multiway Cut (weighted BIMC)). Given a weight function on the terminals, $w : S \to \mathbb{Z}^+$, find a subset of edges $C \subseteq E$ of cost at most B whose isolation weight is maximized.

Without loss of generality we assume that there exists $s \in S$ such that the cost of the minimum cost isolating cut of s is at most B. We denote by *BIMC* the special case of weighted BIMC where $w(s) = 1$ for every $s \in S$.

Problem 3 (Weighted Budgeted Separating Multiway Cut (weighted BSMC)). Given a weight function on the pairs of terminals, $w : S \times S \to \mathbb{Z}^+$, find a subset of edges $C \subseteq E$ of cost at most B whose separation weight is maximized.

Without loss of generality we assume that for every pair $s, s' \in S$, the cost of the minimum cost separating cut of s and s' is at most B. We denote by *BSMC* the special case of weighted BSMC where $w(s, s') = 1$ for every $s, s' \in S$.

With respect to the same input, we define the Sparsest Cut problem. Given a non-empty subset of vertices $U \subset V$, the cut *associated with* U, denoted by $(U, \overline{U})$, is $\{e = (u, v) \in E : u \in U, v \notin U\}$. The *Sparsity* of the cut $(U, \overline{U})$ is given by $\frac{c(U,\overline{U})}{w(U,\overline{U})}$, where $w(\cdot)$ is the separation weight.

Problem 4 (Sparsest Cut). Find a non-empty subset of vertices $U \subset V$ such that the sparsity of its associated cut is minimized.

Lastly, we say that an algorithm ALG is a *bi-criteria approximation with parameters* (α, β) for a given maximization budget problem Π, or simply an (α, β)-*approximation for* Π, if for every instance of Π with budget B, ALG outputs a solution whose value is at least $|OPT|/\alpha$ and whose cost is at most βB, where $|OPT|$ is the value of the optimal solution with respect to the *given budget* B.

3 The Budgeted Isolating Multiway Cut Problem

In this section, we study BIMC and weighted BIMC problems. First we show some hardness results, including integrality gaps of two possible linear relaxations. These integrality gaps suggest that an approximation algorithm which is based on them cannot outperform the constant factor approximation algorithm we give for BIMC. Finally, we give two approximation algorithms for weighted BIMC, the second of which matches one of the lower bounds we show.

3.1 Hardness Results

The proof of the next two propositions is given in the full version of this paper.

Proposition 1. *Unless $P = NP$, there is no α-approximation for the BIMC problem for all $\alpha > 1/3$.*

Proposition 2. *Unless $P = NP$ there is no α-approximation for the BIMC problem for every $\alpha > 1 - 2/OPT$, where $OPT > 2$ is the number of isolated terminals in an optimal solution. Moreover, there is no α-approximation for every $\alpha > 1 - 2/k$, when the number of terminals is a fixed k.*

Integrality Gap of Linear Programming Relaxations We consider two linear programming relaxations for the BIMC problem, and show in the full version of this paper that their integrality gap is at least 2. Hence, we argue that using these relaxations, one cannot achieve an approximation factor for BIMC better than the constant factor approximation presented in the next subsection.

In what follows we assume that for every $s \in S$, the cost of the minimum cost isolating cut of s is at most B (if not, a slight modification can be made in the relaxations and the relevant claims still hold). The first relaxation is a straight forward formulation.

$$\begin{aligned} &\max \textstyle\sum_{s \in S} x_s && \text{(N-ISO-LP)} \\ &\text{s.t.} \\ &\quad x_s - \textstyle\sum_{e \in P_{s,s'}} y_e \le 0 && \text{for every } s, s' \in S \ (s \neq s') \\ &&& \text{and path } P_{s,s'} \text{ from } s \text{ to } s' \\ &\quad \textstyle\sum_{e \in E} c(e) \cdot y_e \le B \\ &\quad 0 \le x_s \le 1 && \text{for every } s \in S \\ &\quad 0 \le y_e && \text{for every } e \in E \end{aligned}$$

The second formulation we consider is derived from the linear programming relaxation of the multiway cut problem presented in [4]. We assume that $S =$

$\{s_1, \ldots, s_k\}$, and embed the given graph into the k-dimensional simplex. We "reserve" the 0-coordinate for the connected component that contains all the terminals *not* isolated by the solution, and the ith coordinate for the connected component that contains terminal s_i, if terminal s_i is isolated by the solution. Thus, we only allow terminal s_i to be mapped to either the "0" component, or the ith component.

$$
\begin{aligned}
&\max \textstyle\sum_{s_i \in S} x^i_{s_i} && \text{(CKR-ISO-LP)}\\
&\text{s.t.}\\
&\quad x^i_{s_i} + x^0_{s_i} = 1 && \text{for } 1 \le i \le k\\
&\quad \textstyle\sum_{i=0}^{k} x^i_v = 1 && \text{for every } v \in V \setminus S\\
&\quad x^i_v \ge 0 && \text{for every } v \in V \text{ and } 0 \le i \le k\\
&\quad y_e = \tfrac{1}{2} \textstyle\sum_{i=0}^{k} |x^i_u - x^i_v| && \text{for every } e = (u,v) \in E\\
&\quad \textstyle\sum_{e \in E} c(e) \cdot y_e \le B
\end{aligned}
$$

3.2 A Greedy Approximation Algorithm for BIMC

The following greedy algorithm for BIMC is a variant of the algorithm presented in [6] for the multiway cut problem. As [6] mentioned, note that a minimum cost isolating cut for $s_i \in S$ can be computed efficiently by merging the terminals in $S \setminus \{s_i\}$ into a single node r and computing a minimum cut separating r and s_i.

Algorithm *GR-ISO*: First, for each $s \in S$, find a minimum cost isolating cut for s, and denote it by C_s. Then, sort the cuts in a non-decreasing order of their cost. Output the maximal sequence of cuts, starting from the cheapest, whose total cost is at most B.

Lemma 1. *Let l denote the value of an optimal solution. Algorithm GR-ISO achieves an approximation factor of $\frac{1}{2}$ if l is even, and $\frac{1}{2} - \frac{1}{2l}$ if l is odd.*[6]

Proof. Let OPT be an optimal solution, and let I denote the set of terminals isolated by OPT. We assume without loss of generality that there is no edge in OPT that can be removed without changing the set of isolated terminals. Let $G' = (V, E \setminus OPT)$. For $s \in I$, let OPT_s be the edges in OPT that have an endpoint in the connected component of s in G'.

Consider the following charging scheme for the terminals in I. Charge the cost of every edge $e \in OPT$ as follows: if there exist two distinct terminals $s \in I$ and $s' \in I$ such that $e \in OPT_s$ and $e \in OPT_{s'}$, charge each of the terminals with $c(e)/2$; else, charge the terminal $s \in I$ such that $e \in OPT_s$ with $c(e)$. Denote by $c(s)$ the total cost charged to terminal s. Obviously, $\sum_{s \in I} c(s) = c(OPT) \le B$ (every edge in OPT is clearly paid for by the charging scheme) and $c(C_s) \le c(OPT_s) \le 2c(s)$ for every $s \in I$ (OPT_s is an isolating cut for s). Let A_l be the set of the first l terminals as sorted by the algorithm. Notice that $\sum_{s \in A_l} c(C_s) \le \sum_{s \in I} c(C_s) \le 2 \sum_{s \in I} c(s) \le 2B$. Thus, the cost of the first $\lfloor l/2 \rfloor$ terminals is $\le B$, and the lemma follows from the definition of *GR-ISO*. $\square$

The above analysis is tight as we show in the full version of this paper.

[6] For the trivial case in which $l = 1$ the algorithm finds an optimal solution.

3.3 Approximation Algorithms for the Weighted BIMC Problem

We present two algorithms for the weighted BIMC problem. The first one is a generalization of algorithm *GR-ISO*.

Algorithm $\boldsymbol{GR\text{-}ISO_w}$: First, for each $s \in S$, find a minimum cost isolating cut for s, and denote it by C_s. Then, sort the cuts with $c(C_s) \leq B$ in a non-decreasing order of the ratio between their cost and their terminal's weight ($c(C_s)/w(s)$). Let $\{C_i\}_{i=1}^{k}$ be the resulting sequence of cuts. Let $\{C_i\}_{i=1}^{m}$ be the maximal prefix of $\{C_i\}_{i=1}^{k}$ with a total cost of at most B. Output the heavier cut (with respect to isolation weight) between $\bigcup_{i=1}^{m} C_i$ and C_{m+1} (if $m = k$ then $\bigcup_{i=1}^{m} C_i$ is an optimal solution). In the full version of this paper we prove that Algorithm *GR-ISO*$_w$ achieves an approximation factor of $\frac{1}{4}$.

A $(\frac{1}{3} - \epsilon)$-Approximation The analysis of algorithm *GR-ISO*$_w$ hints that improving the approximation factor requires an efficient use of the given budget. To this end, we use the FPTAS for the *Knapsack* problem [10], denoted by $A(\pi, \epsilon)$, where π is the Knapsack instance.

Algorithm $\boldsymbol{PACK_w(\epsilon)}$: For each $s \in S$, find a minimum cost isolating cut for s, and denote it by C_s. Construct an instance of the *Knapsack* problem, π: treat each terminal $s \in S$ such that $c(C_s) \leq B$ as an item whose profit is $w(s)$ and whose size is $c(C_s)$, and let B be the "knapsack capacity". Run $A(\pi, \epsilon)$ and denote by P the resulting subset of terminals. Finally, Output $\bigcup_{s \in P} C_s$.

Let OPT be an optimal solution for the weighted BIMC instance. Since every terminal s with $c(C_s) > B$ cannot be isolated by either OPT or $PACK_w(\epsilon)$, we ignore such terminals in what follows. Let I denote the set of the terminals isolated by OPT and l be the isolation weight of OPT, i.e., the value of the optimal solution. Denote by $|OPT(\pi)|$ the value of the optimal solution for the *Knapsack* instance π.

Lemma 2. $|OPT(\pi)| \geq \frac{1}{3} l$

Proof. Let $U = \{X \subseteq I \mid \sum_{s \in X} w(s) \geq \frac{1}{3} l\}$, i.e., U is the set of the subsets of I of profit $\geq \frac{1}{3} l$. Let Y be a set in U of minimum *size in* π (notice that there must exist such a subset). Assume to the contrary that $|OPT(\pi)| < \frac{1}{3} l$, and in particular that $\sum_{s \in Y} c(C_s) > B$. It follows from our assumption, that for every $s \in S$, $w(s) < \frac{1}{3} l$. Thus, there are at least two terminals in Y, and moreover, $\sum_{s \in Y} w(s) < \frac{2}{3} l$ (otherwise, by taking off a terminal from Y we get a contradiction for the minimality of Y in U with respect to size). By similar arguments to those used in the proof of Lemma 1, we get that $\sum_{s \in I} c(C_s) \leq 2B$. Thus, $\sum_{s \in I \setminus Y} c(C_s) < B$ and $\sum_{s \in I \setminus Y} w(s) > \frac{1}{3} l$ and thus $I \setminus Y$ is a feasible solution to π with the desired value. $\square$

It follows from Lemma 2 and the FPTAS for *Knapsack* that Algorithm $PACK_w(\epsilon)$ achieves an approximation factor of $\frac{1}{3} - \epsilon$. We can show that Lemma 2 is tight for arbitrarily large values of k by constructing appropriate examples.

4 Weighted Budgeted Separating Multiway Cut

In this section, we study weighted BSMC. We show that approximating it is at least as hard as the sparsest cut problem. We present a natural linear programming relaxation for the problem and show that it has an unbounded integrality gap for general graphs. However, we give a constant factor approximation algorithm for weighted BSMC on *trees*, which is based on this relaxation. We further note that a better approximation is achieved by Sviridenko's [15] framework. Finally, we use a hierarchical decomposition of graphs by Räcke [13, 9] to obtain a bi-criteria approximation of $(\frac{e}{e-1}, O(\log^2 n \log\log n))$ for arbitrary graphs.

4.1 Hardness Results

Hardness with respect to the Sparsest Cut Problem We firstly prove a lemma and a corollary whose proofs are given in the full version of this paper.

Lemma 3. *Given a non-empty cut $C \subseteq E$ that partitions G into $r > 2$ connected components, there is an algorithm that finds a cut $C' \subset C$ such that $c(C')/w(C') \le c(C)/w(C)$ and C' partitions G into $r-1$ connected components.*

Corollary 1. *Given a non-empty cut $C \subseteq E$, there is an algorithm that finds a non-empty subset of vertices $U \subseteq V$ such that the sparsity of the cut associated with U is at most $c(C)/w(C)$.*

The following theorem shows that the weighted BSMC problem is at least as hard to approximate as the sparsest cut problem is (up to a constant).

Theorem 1. *Let ALG be an (α, β)-approximation for weighted BSMC. Then, there exists a $(1+\epsilon)\alpha\beta$-approximation for Sparsest Cut, for every $\epsilon > 0$.*

Proof. Assume we are given an instance of the Sparsest Cut problem, denote it by π, and let OPT_π denote its optimal solution, and $|OPT_\pi| = \frac{c(OPT_\pi, \overline{OPT_\pi})}{w(OPT_\pi, \overline{OPT_\pi})}$ denote the optimal solution's sparsity. Denote by (π, B) the input for the weighted BSMC problem that consists of the instance π and the budget B, and let $OPT_{\pi,B}$ be a corresponding optimal solution. Then, since $(OPT_\pi, \overline{OPT_\pi})$ is a feasible solution for the weighted BSMC problem on (π, B) for every $B \ge c(OPT_\pi, \overline{OPT_\pi})$, then $w(OPT_\pi, \overline{OPT_\pi}) \le w(OPT_{\pi,B})$ for every $B \ge c(OPT_\pi, \overline{OPT_\pi})$.

For $\lceil \log_{1+\epsilon} c(C_{min}) \rceil \le i \le \lceil \log_{1+\epsilon} c(E) \rceil$, where C_{min} is the minimum cost cut in G, let C_{B_i} be the cut returned by $ALG(\pi, B_i = (1+\epsilon)^i)$. Then, by applying Corollary 1 on each C_{B_i} we can obtain a non-empty subset of vertices $U_i \subseteq V$ where the sparsity of the cut associated with U_i is at most $\frac{c(C_{B_i})}{w(C_{B_i})} \le \frac{\beta B_i}{w(OPT_{\pi,B_i})/\alpha} = \alpha\beta \frac{B_i}{w(OPT_{\pi,B_i})}$. Let $j = \lceil \log_{1+\epsilon} c(OPT_\pi, \overline{OPT_\pi}) \rceil$. Then, $\frac{B_j}{w(OPT_{\pi,B_j})} \le (1+\epsilon) \frac{c(OPT_\pi, \overline{OPT_\pi})}{w(OPT_\pi, \overline{OPT_\pi})} = (1+\epsilon)|OPT_\pi|$. We conclude that the sparsity of $(U_j, \overline{U_j})$ is at most $(1+\epsilon)\alpha\beta|OPT_\pi|$, and the theorem follows by choosing the sparsest cut among $\{(U_i, \overline{U_i})\}_{\lceil \log_{1+\epsilon} c(C_{min}) \rceil \le i \le \lceil \log_{1+\epsilon} c(E) \rceil}$. □

Integrality Gap of a Linear Programming Relaxation We present a natural linear programming relaxation for the weighted BSMC problem, and in the full version of the paper we show that its integrality gap of is $\Omega(n)$. This implies that an algorithm based on this linear relaxation would have poor performance. Nevertheless, in what follows we show an approximation algorithm for the special case of trees based on the same relaxation. In what follows we assume w.l.o.g. that $c(e) \leq B$ for every $e \in E$.

$$\begin{aligned}
&\max \textstyle\sum_{s_i,s_j\in S} w(s_i,s_j)\cdot x_{ij} && \text{(SEP-LP)}\\
&\text{s.t.}\\
&\quad x_{ij} - \textstyle\sum_{e\in P_{i,j}} y_e \leq 0 && \text{for every } s_i,s_j \in S\\
&&& \text{and path } P_{i,j} \text{ from } s_i \text{ to } s_j\\
&\quad \textstyle\sum_{e\in E} c(e)\cdot y_e \leq B\\
&\quad 0 \leq x_{ij} \leq 1 && \text{for every } s_i,s_j \in S\\
&\quad 0 \leq y_e && \text{for every } e \in E
\end{aligned}$$

4.2 Approximation Algorithms for Weighted BSMC in Trees

Let P_{ij} denote the unique path in the tree between s_i and s_j. The dual LP of SEP-LP is:

$$\begin{aligned}
&\min B\cdot\gamma + \textstyle\sum_{s_i,s_j\in S}\beta_{ij} && \text{(SEP-DLP)}\\
&\text{s.t.}\\
&\quad c(e)\cdot\gamma - \textstyle\sum_{i,j:e\in P_{ij}}\alpha_{ij} \geq 0 && \text{for every } e \in E\\
&\quad \alpha_{ij} + \beta_{ij} \geq w(s_i,s_j) && \text{for every } s_i,s_j \in S\\
&\quad \gamma,\alpha_{ij},\beta_{ij} \geq 0 && \text{for every } s_i,s_j \in S
\end{aligned}$$

We define the *worthiness of an edge e with respect to C*, a feasible solution, as $\Gamma_C(e) = \frac{\sum_{i,j:e\in P_{ij}} w(s_i,s_j)\cdot(1-x_{ij})}{c(e)}$, where x is the corresponding solution of SEP-LP. The following algorithm greedily updates the solution as long as the budget is not exceeded, while maintaining the corresponding solution of SEP-LP.

Algorithm *GR-SEP*: Initialize: $h = 0$, $C_0 = \emptyset$, $\forall\ s_i,s_j \in S$, $x_{ij} = 0$, and $\forall\ e \in E$, $y_e = 0$. While there exists an edge $e \in E \setminus C_h$, execute the following loop: Let e_h be a lowest cost edge among the edges with the maximum value of Γ_{C_h}. If $c(e_h) > B - c(C_h)$, output the better solution between $\{e_h\}$ and $C = C_h$. Otherwise, $C_{h+1} \leftarrow C_h \cup \{e_h\}$, set $y_{e_h} = 1$ and $x_{ij} = 1$ for all the pairs (s_i,s_j) separated by e_h, and let $h \leftarrow h+1$. If $E \setminus C_h$ is empty, output $C = C_h$.

Observation 1. *By the definition of the worthiness of an edge, and the fact that edges are only added to the solution during the algorithm, for every $e \in E$ and $0 < h \leq |C|$, $\Gamma_{C_h}(e) \leq \Gamma_{C_{h-1}}(e)$, i.e. the worthiness of an edge can only decrease during the algorithm.*

Corollary 2. *If $C \neq E$, then $c(e_{|C|})\cdot\Gamma_C(e_{|C|}) \leq c(e_{|C|})\cdot\Gamma_{C_0}(e_{|C|}) = w(\{e_{|C|}\})$, i.e., adding the edge $e_{|C|}$ to C increases its separation weight by at most the separation weight of $\{e_{|C|}\}$.*

Theorem 2. *Algorithm GR-SEP achieves an approximation factor of* $\frac{1}{3}$.

Proof. If the algorithm outputs $C = E$, the solution is optimal. Otherwise, it is the case that $\sum_{h=0}^{|C|-1} c(e_h) + c(e_{|C|}) > B$. Denote by $w(GR\text{-}SEP)$ the value of the solution output by *GR-SEP*. Consider the following dual solution: $\beta_{ij} = w(s_i, s_j) \cdot x_{ij}, \alpha_{ij} = w(s_i, s_j) \cdot (1 - x_{ij}), \gamma = \Gamma_C(e_{|C|})$. Since $\Gamma_C(e_{|C|}) \geq \Gamma_C(e)$ for every $e \notin C$, this is a feasible dual solution. Let z denote its value. Then,

$$
\begin{aligned}
z = B \cdot \gamma + \sum_{s_i, s_j \in S} \beta_{ij} &= B \cdot \Gamma_C(e_{|C|}) + \sum_{s_i, s_j \in S} w(s_i, s_j) \cdot x_{ij} && (1)\\
&< \left(\sum_{h=0}^{|C|-1} c(e_h) + c(e_{|C|}) \right) \cdot \Gamma_C(e_{|C|}) + w(C) && (2)\\
&\leq \sum_{h=0}^{|C|-1} c(e_h) \cdot \Gamma_C(e_{|C|}) + w(\{e_{|C|}\}) + w(C) && (3)\\
&\leq \sum_{h=0}^{|C|-1} c(e_h) \cdot \Gamma_{C_h}(e_h) + w(\{e_{|C|}\}) + w(C) && (4)\\
&= w(C) + w(\{e_{|C|}\}) + w(C) \leq 3w(GR\text{-}SEP), && (5)
\end{aligned}
$$

Where: Inequality (2) follows from the definition of the algorithm *GR-SEP*, (3) follows from Corollary 2, and (4) follows by noticing that for every $0 < h \leq |C|$, $\Gamma_{C_h}(e_h) \leq \Gamma_{C_{h-1}}(e_{h-1})$. Thus, the theorem follows by weak duality. $\square$

Remark 1. In [15], Sviridenko introduces a greedy $\frac{e-1}{e}$-approximation algorithm for the problem of maximizing a submodular function subject to a budget constraint. We note that on tree instances (unlike general graphs), the separation weight is a submodular function and thus the weighted BSMC problem on trees can be solved using Sviridenko's algorithm. We note that Sviridenko's algorithm's running time is $\Omega(n^3)$, while the running time of *GR-SEP* is $O(n^2)$.

4.3 Weighted BSMC - General Graphs

In this subsection we introduce an $(\frac{e}{e-1}, O(\log^2 n \log\log n))$-approximation algorithm for weighted BSMC. Since the weighted budgeted variant of Multicut is equivalent to weighted BSMC, we conclude that a $(\frac{e}{e-1}, O(\log^2 n \log\log n))$-approximation exists for this problem as well.

In [13], Räcke describes a hierarchical decomposition of any undirected graph $G = (V, E)$ into a tree T_G, where there is a $1-1$ correspondence between V and the leaves of T_G. T_G has the property that any feasible multi-commodity flow function in T_G can be routed in G causing a congestion bounded by a function of G's parameters, denoted by β. By min-cut-max-flow theorems this implies a corresponding bounded ratio between the cost of cuts in G and the cost of cuts in T_G. In [9], Harrelson et al. give a polynomial-time construction of T_G with $\beta = O(\log^2 n \log\log n)$, which we use in the following algorithm.

Algorithm *SEP*: Let $B' = 2\beta B$. Construct a decomposition tree, T_G, of G. For every $e = (u, v) \in T_G$ with a cost $> B'$, merge the vertices u and v. Let T'_G be the resulted tree. Run Sviridenko's algorithm on T'_G with budget B', and output the associated cut in G.

Theorem 3. *Algorithm SEP is a $(\frac{e}{e-1}, O(\log^2 n \log\log n))$-approximation for the weighted BSMC problem.*

Proof. Let OPT be an optimal solution, and let I denote the set of pairs of terminals separated by OPT. Let OPT_{T_G} be a minimum cost cut separating I in T_G. By [7], $c(OPT_{T_G}) \leq 2MCF_I(T_G)$, where $MCF_I(T_G)$ is the value of the maximum multi-commodity flow in T_G between the pairs in I. By the construction of T_G and its property, $MCF_I(T_G) \leq \beta MCF_I(G)$. Since $MCF_I(G)$ lower bounds the cost of any cut separating I in G, $MCF_I(G) \leq c(OPT)$, and thus we get $c(OPT_{T_G}) \leq 2\beta c(OPT) \leq 2\beta B = B'$. Particularly, OPT_{T_G} does not contain any edge with cost more than B', and thus OPT_{T_G} is a feasible solution for the weighted BSMC problem on T'_G with budget B', with value $w(OPT_{T_G}) \geq w(OPT)$. From [15], running Sviridenko's algorithm will return a solution C whose cost is at most B' and whose value is at least $\frac{e-1}{e} w(OPT_{T_G})$. By the properties of the decomposition tree, the associated cut in G has a cost of at most B' and a separation weight of at least $w(C)$ and the theorem follows. $\square$

4.4 Further Discussion

Although it remains an open question whether it is possible to improve upon the bi-criteria approximation, or even achieve a uni-criteria approximation, in this subsection we review some related ideas and point out some possible directions towards solving the problem. First, consider the following budget problem, whose input is the same as the input for the weighted BSMC problem.

Problem 5 (Budgeted Sparsest Cut). Find a non-empty subset of vertices $U \subset V$ such that $c(U, \overline{U}) \leq B$ and the sparsity of $(U, \overline{U})$ is minimized.

In order to understand the relationship between weighted BSMC and Budgeted Sparsest Cut, we look for results similar to those of Subsection 4.1. Notice that the algorithm of Corollary 1 actually finds a cut whose cost is at most $c(C)$. Hence, Theorem 1 can be easily generalized to obtain the following.

Theorem 4. *Let ALG be an (α, β)-approximation for weighted BSMC. Then, there exists a $((1+\epsilon)\alpha\beta, \beta)$-approximation for the Budgeted Sparsest Cut problem for every $\epsilon > 0$.*

Specifically, notice that a uni-criteria approximation for weighted BSMC implies a uni-criteria approximation for the Budgeted Sparsest Cut problem. This result suggests that the budgeted sparsest cut is not harder than BSMC. Nevertheless, it seems that the budgeted sparsest cut is not much easier, as we argue in the full paper, and we prove for certain weight functions that this is indeed the case.

5 The Budgeted Graph Disconnection Problem

The following algorithm for BGD is a variant of the algorithm presented in [14] for the k-cut problem.

Algorithm GR-PAR: First, compute a Gomory-Hu tree T for G. Then, sort the edges of T in a non-decreasing order of their cost. Finally, choose the maximal sequence of edges starting from the cheapest whose cost is at most B, and output the union of the cuts in G corresponding to these edges. Let l be the value of an optimal solution. We can prove that Algorithm *GR-PAR* achieves an approximation factor of $\frac{1}{2} + \frac{1}{l}$ if l is even, and of $\frac{1}{2} + \frac{1}{2l}$ if l is odd.

References

[1] S. Arora, S. Rao, and U. Vazirani. Expander flows, geometric embeddings, and graph partitionings. In *Proc. of STOC'04*, pages 222–231, 2004.

[2] S. Arora, J. R. Lee and A. Naor. Euclidean distortion and the sparsest cut. In *Proc. of STOC'05*, pages 553–562, 2005.

[3] T. Aura, M. Bishop and D. Sniegowski. Analyzing single-server network inhibition. In *Proc. of CSFW'00*, p. 108–117, 2000.

[4] G. Călinescu, H. Karloff and Y. Rabani. An Improved Approximation Algorithm for Multiway Cut. *JCSS*, 60(3):564–574, 2000.

[5] S. Chawla, A. Gupta and H. Räcke. Embeddings of negative-type metrics and an improved approximation to generalized sparsest cut. In *Proc. of SODA'05*, p. 102–111, 2005.

[6] E. Dahlhaus, D. Johnson, C. Papadimitriou, P. Seymour and M. Yannakakis. The Complexity of Multiterminal Cuts. *SIAM J. on Computing*, 23:864–894, '94. Preliminary version appeared in *Proc. of the 24th ACM Symposium on Theory of Computing*, p. 241–251, 1992.

[7] N. Garg, V.V.Vazirani and M. Yannakakis. Primal-dual approximation algorithms for integral flow and multicut in trees. *Algorithmica*, 18:3–20, 1997.

[8] R. Gomory and T. Hu. Multiterminal network flows. *J. of SIAM*, 9:551–570, '61.

[9] C. Harrelson, K. Hidrum and S. Rao. A polynomial time tree decomposition to minimize congestion. In *Proc. of SPAA'03*, p. 34–43, 2003.

[10] O.H. Ibarra and C.E. Kim. Fast approximation algorithms for the knapsack and sum of subset problems. *J. of the ACM*, 22:463–468, 1975.

[11] D. Karger, P. Klein, C. Stein, M. Thorup and N. Young. Rounding algorithms for a geometric embedding of minimum multiway cut. *STOC'99*, p. 668–678, '99.

[12] S. Khuller, A. Moss and J. Naor. The Budgeted Maximum Coverage Problem. *Information Processing Letters*, 70(1):39–45, '99.

[13] H. Räcke. Minimizing congestion in general networks. *In FOCS'02*, p. 43–52, '02.

[14] H. Saran and V.V. Vazirani. Finding k-cuts within twice the optimal. *SIAM J. on Comp.*, 24:101–108, 1995.

[15] M. Sviridenko. A note on maximizing a submodular set function subject to knapsack constraint. *Operations Research Letters*, 32:41–43, 2004.

[16] R.V. Vohra and N.G. Hall. A probabilistic analysis of the maximal covering location problem. *Discrete Applied Mathematics*, 43(2):175–183, 1993.

[17] L. Wolsey. Maximizing real-valued submodular functions: primal and dual heuristics for location problems. *Mathematics of Operations Research*, 7:410–425, '82.