An Improved Approximation Algorithm for MULTIWAY CUT

Gruia Călinescu* Howard Karloff† Yuval Rabani‡

Abstract

Given an undirected graph with edge costs and a subset of k nodes called *terminals*, a *multiway cut* is a subset of edges whose removal disconnects each terminal from the rest. MULTIWAY CUT is the problem of finding a multiway cut of minimum cost. Previously, a very simple combinatorial algorithm due to Dahlhaus, Johnson, Papadimitriou, Seymour, and Yannakakis gave a performance guarantee of $2\left(1-\frac{1}{k}\right)$. In this paper, we present a new linear programming relaxation for MULTIWAY CUT and a new approximation algorithm based on it. The algorithm breaks the threshold of 2 for approximating MULTIWAY CUT, achieving a performance ratio of at most $1.5-\frac{1}{k}$. This improves the previous result for every value of k. In particular, for $k=3$ we get a ratio of $\frac{7}{6}<\frac{4}{3}$.

*College of Computing, Georgia Institute of Technology, Atlanta, Georgia 30332-0280, U.S.A. Research supported in part by NSF grant CCR-9319106. E-mail: gruia@cc.gatech.edu.

†College of Computing, Georgia Institute of Technology, Atlanta, Georgia 30332-0280, U.S.A. Research supported in part by NSF grant CCR-9319106. E-mail: howard@cc.gatech.edu.

‡Computer Science Department, Technion — IIT, Haifa 32000, Israel. Work supported by BSF grant 96-00402, and by grants from the S. and N. Grand Research Fund, from the Smoler Research Fund, and from the Fund for the Promotion of Research at the Technion. E-mail: rabani@cs.technion.ac.il.

1 Introduction

We consider the problem Multiway Cut: Given an undirected graph with nonnegative edge costs and a set of k specified nodes in the graph (called terminals), find a cheapest multiway cut, i.e., a subset of the edges whose removal disconnects each terminal from the rest. This is one of several generalizations of the classical undirected s-t cut problem, and it has applications in parallel and distributed computing [24], as well as in chip design.

Dahlhaus, Johnson, Papadimitriou, Seymour, and Yannakakis [8] initiated the study of Multiway Cut. In the published version of their paper [9], they prove that Multiway Cut is MAX SNP-hard even when restricted to instances with three terminals and unit edge costs. Therefore, unless P=NP, there is no polynomial-time approximation scheme for Multiway Cut [3]. For $k = 2$, the problem is identical to the undirected version of the extensively studied s-t min-cut problem of Ford and Fulkerson [11], and thus has polynomial-time algorithms (see, e.g., [21, 1]). Prior to this paper, the best (and essentially the only) approximation algorithm for $k \geq 3$ was due to the above-mentioned paper of Dahlhaus et al. They give a very simple combinatorial *isolation heuristic* that achieves an approximation ratio of $2\left(1 - \frac{1}{k}\right)$. Specifically, for each terminal i, find a minimum-cost cut separating i from the remaining terminals, and then output the union of the $k - 1$ cheapest of the k cuts. For $k = 4$ and for $k = 8$, Alon (see [9]) observed that the isolation heuristic can be modified to give improved ratios of $\frac{4}{3}$ and $\frac{12}{7}$, respectively.

In special cases, far better results are known. For fixed k in planar graphs, the problem is solvable in polynomial time [9]. For trees and 2-trees, there are linear-time algorithms [6]. For dense unweighted graphs, there is a polynomial-time approximation scheme [2, 12].

Chopra and Rao [6] and Cunningham [7] develop a polyhedral approach to Multiway Cut, further extended by Chopra and Owen [5]. These provide useful tips to the implementation of branch-and-cut type heuristics that are reported by the authors to work well in practice. Bertsimas et al. [4] propose a non-linear formulation of Multiway Cut and related problems. They suggest several polynomial time-solvable relaxations, and give a simple randomized rounding argument yielding the same bounds as in [9] (and for essentially the same reasons). Their approach appears incapable of producing better bounds.

In this paper, we present a new approximation algorithm for Multiway Cut. The algorithm is based on a new linear programming relaxation for Multiway Cut, which is derived from a straightforward system of inequalities similar to those of Bertsimas et al., to which we add two sets of valid inequalities. In contrast to previous work on a polyhedral approach, our relaxation provably gives better approximation guarantees. Our algorithm gives a ratio substantially below 2 for all k. We present an algorithm with a performance ratio of at most $1.5 - \frac{1}{k}$. Notice that this algorithm improves upon the approximation guarantee of Dahlhaus et al. for every value of k (including the better bounds for $k = 4$ and $k = 8$ of Alon). In particular, for $k = 3$ the ratio we get is $\frac{7}{6}$, whereas the best previous bound was $\frac{4}{3}$.

From a broader perspective, there are several problems of related interest. Erdős and Székely [10] consider a problem of extending a partial k-coloring of a graph, which is equivalent to Multiway Cut (see [9] for further discussion). In Minimum k-Way Cut, there are no specified terminals, and we are expected to cut the graph into k components. Hochbaum and Shmoys [17] give an algorithm for a special case and Goldschmidt and Hochbaum [15] and Karger and Stein [19] give polynomial-time algorithms for the case of fixed k. Saran and Vazirani [22] give a $2\left(1-\frac{1}{k}\right)$-approximation algorithm for MINIMUM k-WAY CUT.

Garg et al. [13] study a variation of MULTIWAY CUT in which nodes have costs and the goal is to remove a minimum cost set of nodes so as to disconnect each terminal from the others. They give a $2\left(1-\frac{1}{k}\right)$-approximation algorithm for this problem. For the multiway cut problem in directed graphs, Naor and Zosin [20], significantly improving upon previous results of Garg et al. [13], give a 2-approximation algorithm. Garg et al. [13] show that for unbounded k, the approximation guarantees for the node multiway cut and the directed multiway cut problems are at least as large as those for VERTEX COVER. As obtaining a ratio better than two for VERTEX COVER remains a challenging open problem [23], it appears that MULTIWAY CUT in undirected graphs is easier than its node or directed variations.

Finally, Hu [18] proposed MINIMUM MULTICUT as an integral dual to maximum multicommodity flow. In this problem, we have to disconnect a list of pairs of terminals. MULTIWAY CUT is a special case, in which the list of pairs forms a clique. Garg et al. [14] give a $O(\log k)$-approximation algorithm for MINIMUM MULTICUT. As noted in [9], a multiway cut algorithm can be used to approximate minimum multicut by the same ratio with running time polynomial in n and 2^k. Therefore, our algorithm gives better approximation guarantees for MINIMUM MULTICUT when k is $O(\log n)$.

The rest of this paper is organized as follows. In Section 2 we present the necessary notation, define the relaxation, and prove some basic properties, and in Section 3 we describe and analyze the algorithm.

2 Preliminaries

Let $G=(V,E)$ be an undirected graph on $V=\{1,2,...,n\}$ in which each edge $uv\in E$ has a non-negative cost $c(u,v)=c(v,u)$, and let $T=\{1,2,\ldots,k\}\subseteq V$ be a set of *terminals*. MULTIWAY CUT is the problem of finding a minimum cost set $C\subseteq E$ such that in $(V,E\setminus C)$, each of the terminals $1,2,\ldots,k$ is in a different component. Let $MWC=MWC(G)$ be the value of the optimal solution to MULTIWAY CUT.

Notation. Δ_k denotes the $(k-1)$-simplex, i.e., the $(k-1)$-dimensional convex polytope in $\mathbb{R}^k$ given by $\{x\in\mathbb{R}^k|(x\geq 0)\wedge(\sum_i x_i=1)\}$.

For $x\in\mathbb{R}^k$, $\|x\|$ is its L_1 norm: $\|x\|=\sum_i|x_i|$. For $j=1,2,\ldots,k$, $e^j\in\mathbb{R}^k$ denotes the unit vector given by $(e^j)_j=1$ and $(e^j)_i=0$ for all $i\neq j$.

A *semimetric* is a pair (V,d) where V is a set and d is a function $d:V\times V\rightarrow\mathbb{R}$ such that $d(u,v)=d(v,u)\geq 0$ for all u,v; $d(u,u)=0$ for all u; and $d(u,w)\leq d(u,v)+d(v,w)$ for all u,v,w. We sometimes refer to the elements of V as *points*, and to $d(u,v)$ as the *distance* between u and v.

We denote by uv an (undirected) edge with endpoints u and v.

The relaxation. MULTIWAY CUT with edge costs can be formulated as the following integer program. The variables in the program are $d(u,v)$ for all $u,v \in V$.

$$\text{Minimize} \sum_{uv \in E} c(u,v)d(u,v) \text{ subject to}$$

$$\begin{aligned}
&(V,d) \text{ is a semimetric} && && (1)\\
&d(t_1,t_2) = 1 && \forall t_1,t_2 \in T,\ t_1 \neq t_2, && (2)\\
&d(u,v) \in \{0,1\} && \forall u,v \in V. && (3)
\end{aligned}$$

By relaxing the integrality constraints (3) to

$$0 \leq d(u,v) \leq 1 \ \ \forall u,v \in V,$$

we obtain a linear programming relaxation for Multiway Cut with edge costs, which we denote by $LP1$. The integrality ratio for $LP1$ is precisely the Dahlhaus et al. guarantee. To see this, consider a k-leaf star, with the leaves as the terminals (all edge costs are 1). The optimal integral solution places $k-1$ edges in the multiway cut, but a feasible (and optimal) fractional solution assigns length $\frac{1}{2}$ to each of the k edges. A simple rounding argument gives an algorithm with identical performance guarantee: for each terminal $t \in T$, pick $\rho_t \in (0, \frac{1}{2})$ such that the cost of edges crossing the boundary of $B_d(t,\rho_t)$ (the ball around t having radius ρ_t in metric d) is minimized. Notice that by removing these edges, we isolate t from the other terminals. Take the $k-1$ smallest such cuts as the multiway cut.

In order to do better, we strengthen the relaxation. We add the following valid inequalities:

$$\begin{aligned}
\textstyle\sum_{t \in T} d(u,t) &= k-1 && \forall u \in V, && (4)\\
d(u,v) &\geq \textstyle\sum_{t \in S}[d(u,t) - d(v,t)] && \forall u,v \in V,\ \forall S \subseteq T. && (5)
\end{aligned}$$

We denote this stronger relaxation by $LP2$. Notice that constraint (5) implies that $d(u,v) \geq |\sum_{t \in S}[d(u,t) - d(v,t)]|$, because $d(u,v) = d(v,u)$. In this formulation, there is an exponential number of constraints (5). However, $\sum_{t \in S}[d(u,t) - d(v,t)]$ is maximized at the set S of all terminals t for which $d(u,t) - d(v,t) > 0$. Therefore, we have a polynomial time separation oracle for $LP2$. This implies that we can find the optimum to $LP2$ in polynomial time [16]. In fact, this observation leads to a polynomial size formulation: For every $u,v \in V$ and $t \in T$, add the pair of constraints $d'(u,v,t) \geq d(u,t) - d(v,t)$ and $d'(u,v,t) \geq 0$. Replace the constraints (5) by $d(u,v) \geq \sum_t d'(u,v,t)$. Thus we can use interior-point algorithms to solve $LP2$.

Another possible relaxation for Multiway Cut with edge costs is the following:

$$\text{Minimize } \frac{1}{2} \sum_{uv \in E} c(u,v) \cdot \|x^u - x^v\| \text{ subject to}$$

$$\begin{aligned}
x^u &\in \Delta_k && \forall u \in V && (6)\\
x^t &= e^t && \forall t \in T. && (7)
\end{aligned}$$

In other words, we place the terminals at the vertices of the $(k-1)$-simplex, and the other nodes anywhere in the simplex, and measure an edge's length by the total variation distance between its endpoints. Clearly, placing all nodes at simplex vertices gives an integral solution: the lengths of edges are either 0 (if both endpoints are at the same vertex) or 1 (if the endpoints are at different vertices), and the removal of all unit length edges disconnects the graph into at least k components, each containing at most one terminal. We denote this relaxation by SLP.

Proposition 1. $LP2$ and SLP are equivalent.

Proof. Given a feasible solution d to $LP2$ we compute a feasible solution x to SLP with no greater value as follows. For all $t \in T$, set $x^t = e^t$. For all $u \in V \setminus T$, for all $i \in T$, set $x_i^u = 1 - d(u,i)$. As $\sum_{i\in T} d(u,i) = k-1$, we have $\sum_{i\in T} x_i^u = 1$. Since $d(u,i) \leq 1$, $x_i^u \geq 0$. Thus, $x^u \in \Delta_k$. Furthermore, for all $u, v \in V$, $\frac{1}{2}\|x^u - x^v\| = \sum_i \max\{0, x_i^u - x_i^v\}$. Put $S = \{i | x_i^u - x_i^v > 0\}$. We get $\frac{1}{2}\|x^u - x^v\| = \sum_i \max\{0, x_i^u - x_i^v\} = \sum_{i\in S}(x_i^u - x_i^v) = \sum_{i\in S}(d(v,i) - d(u,i)) \leq d(u,v)$.

Conversely, given a feasible solution x to SLP, we compute a feasible solution d to $LP2$ with the same value as follows. For all $u, v \in V$, set $d(u,v) = \frac{1}{2}\|x^u - x^v\|$. Feasibility and equality of the objective function value are obvious. ∎

Subdivisions. Let $uv \in E$. Let G' be the graph obtained from G by subdividing the edge uv at a point w. Formally, let $w \notin V$, and define $V' = V \cup \{w\}$ and $E' = (E \setminus \{uv\}) \cup \{uw, wv\}$; w is a nonterminal of G'. The new edges uw and wv have $c(u,w) = c(w,v) = c(u,v)$, while the edge uv disappears.

Proposition 2. Given a multiway cut $C' \subseteq E'$ in G' of cost Z, one can construct a multiway cut $C \subseteq E$ in G of cost at most Z.

Proof. If $C' \subseteq E$ then let $C = C'$, and otherwise let $C = (C' \setminus \{uw, wv\}) \cup \{uv\}$. ∎

We need some special properties of solutions to SLP. We obtain these properties using subdivisions.

Lemma 3. Let x be a feasible solution to SLP for a weighted graph $G = (V, E)$. We can construct a graph $\tilde{G} = (\tilde{V}, \tilde{E})$, derived from G by a sequence of at most $k|E|$ subdivisions, and a corresponding feasible solution $\tilde{x}$, such that
(i) the value of $\tilde{x}$ is at most the value of x, and
(ii) for all edges $uv \in \tilde{E}$, the k-vectors x^u and x^v differ in at most two coordinates.

Proof. We exploit the additivity of the L_1 norm. Let uv be an edge in E such that the k-dimensional vectors x^u and x^v differ in more than two coordinates. Let i be a terminal such that $x_i^u < x_i^v$. As $\sum_{l=1}^k x_l^u = 1 = \sum_{l=1}^k x_l^v$, there is a terminal $j \neq i$ such that $x_j^u > x_j^v$. Put $\alpha = \min\{x_i^v - x_i^u, x_j^u - x_j^v\}$. Let $w \notin V$ and define $x_l^w = x_l^u$ for $l \in \{1, 2, \ldots, k\} \setminus \{i, j\}$, $x_i^w = x_i^u + \alpha$ and $x_j^w = x_j^u - \alpha$. It is immediate that $\sum_{l=1}^k x_l^w = 1$ and that $0 \leq x_i^w \leq 1$. We have $\frac{1}{2}\|x^u - x^w\| = \alpha$ and that x^u and x^w differ in only two coordinates. Also, it is easy to verify that $\frac{1}{2}\|x^v - x^w\| = \frac{1}{2}\|x^u - x^v\| - \alpha$ and that the number of coordinates in which x^v and x^w differ is smaller than the number of coordinates in which x^u and x^v differ. As described above, we subdivide the edge uv into uw and wv, and extend the solution to SLP to include the new node. The value of the extended solution does not increase. We continue, if necessary, to subdivide the new edge wv. We need to do this at most $k-2$ times, since each time the number of coordinates in which the new node differs from x^v decreases. We stop when all edges yz obtained from subdividing uv have the property that x^y and x^z differ in at most two coordinates.

We repeat the process described above for all edges of G. In this process we create at most $k|E|$ new nodes and associated k-vectors. At the end, we have a graph $\tilde{G}$ with at most $n + k|E|$ nodes. We also have a feasible solution $\tilde{x}$, which extends x to all the new nodes of $\tilde{G}$ and has the same cost as x. ■

3 The Algorithm

Here we present our algorithm which finds a multiway cut with cost within a factor of $1.5 - 1/k$ of optimal. After presenting a randomized algorithm, we provide a simple derandomization.

We begin by computing an optimal solution to SLP, for instance by solving the linear program $LP2$ and then transforming the solution. Clearly, the value Z^* of this solution is a lower bound on the cost of the minimum multiway cut MWC. In fact, assume that we have a feasible (optimal) solution x to SLP of value Z, such that for all $uv \in E$, x^u and x^v differ in at most two coordinates. The rounding proceedure will construct a multiway cut $C \subseteq E$ of expected cost at most $(1.5 - 1/k)Z$. Then, using Lemma 3 and Proposition 2, one can (trivially) extend the multiway cut construction to the general case.

We then apply a rounding algorithm to get an integral solution in which all the nodes are at the vertices of Δ_k. The rounding algorithm iteratively assigns some nodes to terminal 1, then some of the remaining nodes to terminal 2, then some of the remaining ones to terminal 3, and so on, and then some to terminal $k-1$. Any nodes left over are assigned to terminal k, which acts as an "overflow" bin. (In fact, the algorithm does this only with probability 1/2. In the complementary case, the algorithm first assigns some nodes to terminal $k-1$, then some of the others to terminal $k-2$, then some of the rest to terminal $k-3$, ..., and then some to terminal 1. Terminal k again acts as an overflow bin, taking any nodes that remain at the end.) An edge $\{u, v\}$ ends up in the multiway cut if and only if u and v are assigned to different terminals.

Rounding. Set $B(i, \rho) = \{u \in V \mid x_i^u > 1 - \rho\}$, the set of nodes suitably "close" to terminal i in the simplex. Choose a permutation $\sigma = \langle \sigma_1, \sigma_2, ..., \sigma_k \rangle$ to be either $\langle 1, 2, 3, \ldots, k-1, k \rangle$ or $\langle k-1, k-2, k-3, \ldots, 1, k \rangle$ with probability $\frac{1}{2}$ each. Independently, choose $\rho \in (0, 1)$ uniformly at random. Then, process the terminals in the order $\sigma(1), \sigma(2), \sigma(3), \ldots, \sigma(k)$. For each j from 1 to $k-1$, place the nodes that remain in $B(\sigma_j, \rho)$ at e^{σ_j}. Place whatever nodes remain at the end at e^k. The following code specifies the rounding procedure more formally. We use $\bar{x}$ to denote the rounded solution.

The Rounding Procedure

Let $\sigma = \langle 1, \ldots, k-3, k-2, k-1, k \rangle$ or $\langle k-1, k-2, k-3, \ldots, 1, k \rangle$, each with probability 1/2
Let ρ be a random real in $(0, 1)$ /* See note below. */
for $j = 1$ **to** $k-1$ **do**
    for all u such that $x^u \in B(\sigma_j, \rho) \setminus \cup_{i:i<j} B(\sigma_i, \rho)$ **do**
        $\bar{x}^u := e^{\sigma_j}$ /* assign node u to terminal σ_j */

    endfor
endfor
for all u such that $x^u \notin \cup_{i:i<k} B(\sigma_i, \rho)$ **do**
    $\bar{x}^u := e^k$
endfor

We can implement this algorithm to run in random polynomial time, as follows. (Because the algorithm can easily be derandomized (see Section 3.2), we ignore the fact that in reality one cannot choose ρ uniformly from $(0, 1)$.) After choosing σ and ρ, we maintain k lists of nodes, list i eventually containing the nodes assigned to terminal i. Initially, the kth list contains all the nodes, and the other lists are empty. For each terminal we scan each node in the kth list and check if it is close enough to the terminal. If so, we move it to that terminal's list. Otherwise, it remains in the kth list.

Let C be the set of edges whose endpoints are at different vertices of Δ_k in the rounded solution $\bar{x}$. Clearly, the value of this solution is exactly the sum of costs of the edges in C. Furthermore, these edges form a multiway cut. In what follows we relate the expected total cost of edges in C to the value Z of the fractional solution x.

3.1 Analysis

We are assuming that for every edge $uv \in E$, the k-vectors x^u and x^v differ in at most two coordinates. Let $uv \in E$. It is impossible for x^u and x^v to differ in exactly one coordinate. Therefore, we can partition the edges of G into $E = (\cup_{i<j} E_{ij}) \cup E_0$, where

$$E_{ij} = E_{ji} = \{uv \in E \mid (x_i^u \neq x_i^v) \wedge (x_j^u \neq x_j^v)\}$$

for $i \neq j$, and

$$E_0 = \{uv \in E \mid x^u = x^v\}.$$

Let uv be an edge in E. If $uv \notin E_0$, then there are two terminals i, j such that $uv \in E_{ij}$. Assume, without loss of generality, that $x_i^u = \max\{x_i^u, x_j^u, x_i^v, x_j^v\}$, i.e., the smallest distance between one of the terminals i, j and one of the nodes u, v is achieved between i and u. (The smallest distance is not necessarily unique.)

Proposition 4. For any edge $uv \in E_{ij}$, either $x_i^u \geq x_i^v \geq x_j^v \geq x_j^u$ or $x_i^u \geq x_j^v \geq x_i^v \geq x_j^u$.

Proof. $\sum_t x_t^u = 1 = \sum_t x_t^v$. As $x_t^u = x_t^v$ for all $t \notin \{i, j\}$, we have $x_i^u + x_j^u = x_i^v + x_j^v$. As x_i^u is the largest among these four coordinates, it follows that $x_i^v \geq x_j^u$ and $x_j^v \geq x_j^u$. ∎

Define $d(u, v) = \frac{1}{2}\|x^u - x^v\|$.

Lemma 5. For any edge $uv \in E \setminus \cup_{t:t<k} E_{tk}$,

$$\Pr[uv \in C] \leq 1.5\, d(u, v).$$

Proof. For $i \neq j$, by $i \prec j$ denote the fact that i precedes j in σ.

If $x^u = x^v$, then regardless of σ and ρ, $\bar{x}^u = \bar{x}^v$ (i.e., the two nodes get assigned to the same terminal). Therefore, $\Pr[uv \in C] = 0$.

So we may assume that $uv \in E_{ij}$ for a pair of terminals $i \neq j$. Assume, without loss of generality, that $x_i^u = \max\{x_i^u, x_j^u, x_i^v, x_j^v\}$.

For any $l \in \{1, 2, \ldots, k\} \setminus \{i, j\}$, $x_l^u = x_l^v$, and therefore either both $u, v \in B(l, \rho)$ or both $u, v \notin B(l, \rho)$. So uv is cut only if one of its endpoints is assigned to either i or j. Thus the only way we can possibly put uv in C is to have either $\rho \in I_L = (1 - x_i^u, 1 - x_i^v]$ or $\rho \in I_R = (1 - x_j^v, 1 - x_j^u]$ (each interval is open on the left and closed on the right). I_L and I_R are not necessarily disjoint (see Proposition 4). We have $1 - x_i^v \leq 1 - x_j^u$. Both intervals I_L and I_R have the same length $d(u, v)$. Clearly $I_L \cup I_R = I_L \cup (I_R \setminus I_L)$, so that for uv to be in C, either $\rho \in I_L$ or $\rho \in I_R \setminus I_L$.

Now suppose that $i \prec j$ and that by the time i is processed, neither u nor v has been assigned to a vertex. For uv to be cut, we must have $\rho \in I_L \cup (I_R \setminus I_L)$. The crux of the whole proof is that, if $\rho \in I_R \setminus I_L$, then in both cases of Proposition 4, $\rho > 1 - x_i^v \geq 1 - x_i^u$. Therefore, when i is processed, *both u and v are assigned to terminal i, ensuring that uv is not in C.* Using this crucial fact and the independence of ρ and σ, we have

$$\begin{aligned}
\Pr[uv \in C] &\leq \Pr[(j \prec i) \wedge (\rho \in I_L \cup I_R)] + \Pr[(i \prec j) \wedge (\rho \in I_L)] \\
&\leq \frac{1}{2} 2d(u, v) + \frac{1}{2} d(u, v) \\
&= 1.5\, d(u, v). \qquad \blacksquare
\end{aligned}$$

Lemma 6. For any edge $uv \in \cup_{t:t<k} E_{tk}$, $Pr[uv \in C] \leq d(u, v)$.

Proof. Similar to the previous lemma. The difference is that terminal k is always processed last, so that for $uv \in E_{ik}$, the only way that u and v are placed at different terminals is if one of them, but not the other, is placed at i when i is processed. Thus

$$\Pr[uv \in C] = \Pr[\rho \in (1 - x_i^u, 1 - x_i^v]] = d(u, v). \qquad \blacksquare$$

Theorem 7. The expected weight of the multiway cut found by the algorithm is at most $(1.5 - 1/k)Z^*$.

Proof. Let x^* be an optimal solution to SLP for the graph G, and let Z^* be its value. By Lemma 3, we can construct in polynomial time a graph $\tilde{G} = (\tilde{V}, \tilde{E})$ and a feasible solution $\tilde{x}$ to SLP for $\tilde{G}$ with the following properties: (i) $\tilde{G}$ is derived from G through a sequence of subdivisions (so it has the same set of terminals); (ii) the value of $\tilde{x}$ is Z^*; and (iii) $\tilde{x}$ satisfies our simplifying assumption (i.e., for every edge uv, $\tilde{x}^u$ and $\tilde{x}^v$ differ in at most two coordinates).

Rename the terminals so that for

$$Z_i = \sum_{uv \in \cup_{t:t \neq i} E_{ti}} c(u, v) d(u, v),$$

$i \in \{1, 2, ..., k\}$, Z_i is maximized at $i = k$. Let

$$Z' = \sum_{uv \in E \setminus E_0} c(u, v) d(u, v) \leq Z^*.$$

As $Z' = \sum_i \frac{1}{2} Z_i$, we have $Z_k \geq \frac{2}{k} Z'$. Let $\bar{x}$ be the (random) solution output by the rounding procedure given $\tilde{x}$. Combining Lemmas 5 and 6, and using linearity of expectation, the expected value of $\bar{x}$ is at most $1.5Z' - 0.5Z_k \leq \left(1.5 - \frac{1}{k}\right) Z'$, and we are done, since $Z' \leq Z^*$. $\blacksquare$

3.2 Derandomization

Instead of choosing ρ from a continuous distribution in line 2, we show in Theorem 8 that it is sufficient to choose ρ from a small finite sample space. Therefore, we can enumerate over all possible choices of σ and ρ.

Theorem 8. There is a deterministic polynomial time algorithm that finds a multiway cut of cost at most $(1.5 - 1/k)Z^*$.

Proof. By the proof of Theorem 7, there exists a choice of σ, ρ that gives an integral solution of value at most the expectation. There are two possible choices for σ. For a given permutation σ, two different values of ρ, $\rho_1 < \rho_2$, produce combinatorially distinct solutions only if there is a terminal i and a node u such that $x_i^u \in (1-\rho_2, 1-\rho_1]$. Thus we may enumerate over at most $k|\tilde{V}|$ "interesting" values of ρ. We can determine these values easily, by sorting the nodes according to each coordinate separately. The resulting discrete sample space for (σ, ρ) has size at most $2k|\tilde{V}|$, so we can search it exhaustively to find a point that produces a solution of cost at most the expectation. Thus we can construct, in polynomial time, a multiway cut for $\tilde{G}$ of cost at most $\left(1.5 - \frac{1}{k}\right) Z'$. By Proposition 2, we can use this multiway cut to construct a multiway cut for G in polynomial time. ∎

4 Concluding Remarks

We do not know the integrality ratio for the relaxation we proposed. It is possible that a better rounding procedure can be discovered. Here are the worst examples of which we are aware. For $k = 3$, the following example (which also appeared in [7]) satisfies all the new constraints and shows that the integrality ratio is at least $\frac{16}{15}$. Consider the graph $G = (V, E)$. $V = \{S \subseteq \{1, 2, 3\} |\ 1 \leq |S| \leq 2\}$, where $\{1\}, \{2\}, \{3\}$ are the terminals. $E = \{\{S, T\} |\ S \neq T, |S \cap T| = 1\}$. The edges $\{S, T\}$ with S or T of size 1 (between a terminal and a nonterminal) have cost 2, and the edges with $|S| = |T| = 2$ (between two nonterminals) have cost 1. It is not hard to see that the optimum multiway cut has cost 8 (by enumerating over all distinct assignments of the nonterminals to terminals). On the other hand, assigning length $\frac{1}{2}$ to all the edges is a feasible (optimal) solution to the relaxation (i.e., place nonterminal $\{i, j\}$ midway between terminals $\{i\}$ and $\{j\}$ in the 2-simplex). This solution has value 7.5. Thus, the ratio of the integral optimum to the fractional optimum for this example is at least $\frac{16}{15}$.

The example can be generalized to $k = 4$. The graph has node set $V = \{S \subseteq \{1, 2, 3, 4\} |\ 1 \leq |S| \leq 2\}$, where $\{1\}, \{2\}, \{3\}, \{4\}$ are the terminals. The edge set is defined as in the previous example: $E = \{\{S, T\} |\ S \neq T, |S \cap T| = 1\}$. The 12 edges $\{S, T\}$ such that $|S| = 1$ or $|T| = 1$ (between a terminal and a nonterminal) have cost 3, and the 12 edges $\{S, T\}$ such that $|S| = |T| = 2$ (between two nonterminals) have cost 1. By exhaustive search, one can verify that the optimum multiway cut has cost 26. There is a feasible fractional solution in which all edges have length $\frac{1}{2}$. Its cost is 24, so the integrality ratio in this case is at least $\frac{13}{12}$.

We also do not know the exact performance ratio (as opposed to integrality ratio) for the algorithm we presented. For $k = 3$, the following example (based on the gadget used in [9] for proving the NP-hardness of the problem) shows that the performance ratio of the algorithm is at least $\frac{16}{15}$. Consider the complete graph with vertex set $V = \{1, 2, 3\} \times \{1, 2, 3\}$, where $(1, 1), (2, 2), (3, 3)$ are the terminals. For pairwise distinct $i, j, k \in \{1, 2, 3\}$, the edges of type $\{(i, i), (i, j)\}$ and $\{(i, i), (j, i)\}$ have cost 2, the edges of type $\{(i, k), (i, j)\}$ and $\{(k, i), (j, i)\}$ have cost 1, and the remaining edges have cost 0. Using the dual, one can verify that $x^{(i,j)} = (1/2)(e_i + e_j) \in \mathbb{R}^3$ for all i, j is an optimal solution to SLP. If the rounding procedure starts with this optimal solution to SLP, then all the multiway cuts it produces have cost 16. However, a multiway cut of cost 15 exists: the nodes that get assigned to terminal i are $(i, 1), (i, 2), (i, 3)$.

References

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows*. Prentice-Hall, Upper Saddle River, NJ, 1993.

[2] S. Arora, D. Karger, and M. Karpinski. Polynomial-time Approximation Schemes for Dense Instances of NP-hard Problems. In *Proc. 27th ACM STOC*, pages 284-293, 1995.

[3] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof Verification and Hardness of Approximation Problems. In *Proc. 33rd IEEE FOCS*, pages 14-23, 1992.

[4] D. Bertsimas, C. Teo, and R. Vohra. Nonlinear Formulations and Improved Randomized Approximation Algorithms for Multicut Problems. In *Proc. 4th Conference on Integer Programming and Combinatorial Optimization*, *Lecture Notes in Computer Science* 920, Springer-Verlag, pages 29–39, 1995.

[5] S. Chopra and L. H. Owen. Extended Formulations for the A-cut Problem. Preprint, 1994.

[6] S. Chopra and M. R. Rao. On the Multiway Cut Polyhedron. *Networks*, 21:51–89, 1991.

[7] W. H. Cunningham. The Optimal Multiterminal Cut Problem. In *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, volume 5, pages 105–120, 1991.

[8] E. Dahlhaus, D. S. Johnson, C. H. Papadimitriou, P. D. Seymour, and M. Yannakakis. The Complexity of Multiway Cuts. Extended abstract, 1983.

[9] E. Dahlhaus, D. S. Johnson, C. H. Papadimitriou, P. D. Seymour, and M. Yannakakis. The Complexity of Multiterminal Cuts. *SIAM Journal on Computing* 23:864-894, 1994. Preliminary version appeared in *STOC '92*.

[10] P. Erdős and L. A. Székely. On Weighted Multiway Cuts in Trees. *Math Programming* 65:93–105, 1994.

[11] L. R. Ford and D. R. Fulkerson. *Flows in Networks*. Princeton University Press, Princeton, NJ, 1962.

[12] A. Frieze and R. Kannan. The Regularity Lemma and Approximation Schemes for Dense Problems. In *Proc. 37th IEEE FOCS*, pages 12–20, 1996.

[13] N. Garg, V. V. Vazirani, and M. Yannakakis. Multiway Cuts in Directed and Node Weighted Graphs. Manuscript. Preliminary version appeared in *Proc. 21st ICALP*, *Lecture Notes in Computer Science* 820, Springer-Verlag, pages 487-498, 1994.

[14] N. Garg, V. V. Vazirani, and M. Yannakakis. Approximate Max-Flow Min-(multi)cut Theorems and Their Applications. *SIAM Journal on Computing* 25:235-251, 1996. Preliminary version appeared in *STOC '93*.

[15] O. Goldschmidt and D. S. Hochbaum. Polynomial Algorithm for the k-Cut Problem. In *Proc. 29th IEEE FOCS*, pages 444-451, 1988.

[16] M. Grötschel, L. Lovász, and A. Schrijver. Geometric Algorithms and Combinatorial Optimization. Springer-Verlag, 1993 (2nd edition).

[17] D. S. Hochbaum and D. B. Shmoys. An $O(|V|^2)$ Algorithm for the Planar 3-Cut Problem. *SIAM J. Algebraic and Discrete Methods* 6:707–712, 1985.

[18] T. C. Hu. *Integer Programming and Network Flows*. Addison-Wesley, Reading, MA, 1969.

[19] D. R. Karger and C. Stein. A New Approach to the Minimum Cut Problem. *Journal of the ACM* 43:601–640, 1996.

[20] J. Naor and L. Zosin. A 2-Approximation Algorithm for the Directed Multiway Cut Problem. In *Proc. 38th IEEE FOCS*, pages 548–553, 1997.

[21] G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization*. John Wiley & Sons, New York, NY, 1988.

[22] H. Saran and V. V. Vazirani. Finding k-Cuts Within Twice the Optimal. In *Proc. 32nd IEEE FOCS*, pages 743-751, 1991.

[23] D. B. Shmoys. Computing Near-Optimal Solutions to Combinatorial Optimization Problems. In *Combinatorial Optimization*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science (20), W. Cook and L. Lovász (eds.), AMS Publications, 1995.

[24] H. S. Stone. Multiprocessor Scheduling with the Aid of Network Flow Algorithms. In *IEEE Transactions on Software Engineering* SE-3, pages 85-93, 1977.