Optimal 3-terminal cuts and linear programming

Kevin K. H. Cheung* William H. Cunningham† Lawrence Tang‡

August 29, 2005

Abstract

Given an undirected graph $G = (V, E)$ and three specified terminal nodes t_1, t_2, t_3, a *3-cut* is a subset A of E such that no two terminals are in the same component of $G \backslash A$. If a non-negative edge weight c_e is specified for each $e \in E$, the *optimal 3-cut problem* is to find a 3-cut of minimum total weight. This problem is $\mathcal{NP}$-hard, and in fact, is max-$\mathcal{SNP}$-hard. An approximation algorithm having performance guarantee $\frac{7}{6}$ has recently been given by Călinescu, Karloff, and Rabani. It is based on a certain linear-programming relaxation, for which it is shown that the optimal 3-cut has weight at most $\frac{7}{6}$ times the optimal LP value. It is proved here that $\frac{7}{6}$ can be improved to $\frac{12}{11}$, and that this is best possible. As a consequence, we obtain an approximation algorithm for the optimal 3-cut problem having performance guarantee $\frac{12}{11}$. In addition, we show that $\frac{12}{11}$ is best possible for this algorithm.

1 Introduction

Given an undirected graph $G = (V, E)$ and k specified terminal nodes $t_1, \ldots, t_k$, a *k-cut* is a subset A of E such that no two terminals are in the same component of $G \backslash A$. If a non-negative edge-weight c_e is specified for each $e \in E$, the *optimal k-cut problem* is to find a k-cut of minimum total weight. This problem was shown by Dahlhaus, Johnson, Papadimitriou, Seymour, and Yannakakis [7] to be $\mathcal{NP}$-hard for $k \geq 3$. (Of course, it is solvable in polynomial time if $k = 2$.) They also gave a simple polynomial-time algorithm having performance guarantee $\frac{2(k-1)}{k}$, that is, one that is guaranteed to deliver a k-cut of weight at most $\frac{2(k-1)}{k}$ times the minimum weight of a k-cut. Later, in [8], the same authors showed that for $k \geq 3$ the problem is max-$\mathcal{SNP}$-hard, which implies that, assuming $\mathcal{P} \neq \mathcal{NP}$, there exists a positive ε such that the problem has no polynomial-time approximation algorithm with performance guarantee $1 + \varepsilon$.

*School of Mathematics and Statistics, Carleton University, 1125 Colonel By Drive, Ottawa, ON, Canada K1S 5B6. E-mail: kcheung@math.carleton.ca. Research of this author was supported by NSERC PGSB.

†Department of Combinatorics & Optimization, Faculty of Mathematics, University of Waterloo, 200 University Avenue West, Waterloo, ON, Canada, N2L 3G1. E-mail: whcunnin@uwaterloo.ca. Research supported by a grant from NSERC of Canada.

‡Hong Kong Monetary Authority, 55/F, Two International Finance Centre, 8 Finance Street, Central, Hong Kong E-mail: ltang77@yahoo.com. Research supported by a grant from NSERC of Canada.

The present paper concentrates on the optimal 3-cut problem. From the above remarks, it follows that this problem is max-$\mathcal{SNP}$-hard, and the approximation algorithm of [8] has a performance guarantee of $\frac{4}{3}$. Later, Călinescu, Karloff, and Rabani [1, 2] gave an approximation algorithm having a performance guarantee of $\frac{7}{6}$. We give a further improvement that is based on their approach.

Chopra and Rao [4] and Cunningham [5] investigated linear-programming relaxations of the 3-cut problem, showing results on classes of facets and separation algorithms. Here are the two simplest relaxations. (By a *T-path* we mean the edge-set of a path joining two of the terminals. By a *wye* we mean the edge-set of a tree having exactly three nodes of degree one, each of which is a terminal. For a set A, a subset B of A, and a vector $z \in \mathbf{R}^A$, $z(B)$ denotes $\sum_{j \in B} z_j$.)

$$(LP1) \qquad \begin{array}{rrcll} \text{minimize} & \sum_{e\in E} c_e x_e & & & \\ & \text{subject to} & & & \\ & x(P) & \geq & 1, & P \text{ a } T\text{-path} \\ & x_e & \geq & 0, & e \in E. \end{array}$$

$$(LP2) \qquad \begin{array}{rrcll} \text{minimize} & \sum_{e\in E} c_e x_e & & & \\ & \text{subject to} & & & \\ & x(P) & \geq & 1, & P \text{ a } T\text{-path} \\ & x(Y) & \geq & 2, & Y \text{ a wye} \\ & x_e & \geq & 0, & e \in E. \end{array}$$

It follows from some simple observations about shortest paths, and the equivalence of optimization and separation, that both problems can be solved in polynomial time. It was proved in [5] that the approximation algorithm of [7] delivers a 3-cut of value at most $\frac{4}{3}$ times the optimal value of $(LP1)$. (In particular, the minimum weight of a 3-cut is at most $\frac{4}{3}$ times the optimal value of $(LP1)$.) It was conjectured that the minimum weight of a 3-cut is at most $\frac{16}{15}$ times the optimal value of $(LP2)$. The examples in Figure 1 (from [5]) show that this conjecture, if true, is best possible. In both examples, the values of a feasible solution x of $(LP2)$ are shown in the figure. The weights c_e are all 2 for the example on the left. For the one on the right they are 1 for the edges of the interior triangle, and 2 for the other edges. In both cases the minimum 3-cut value is 8, but the given feasible solution of $(LP2)$ has value 7.5.

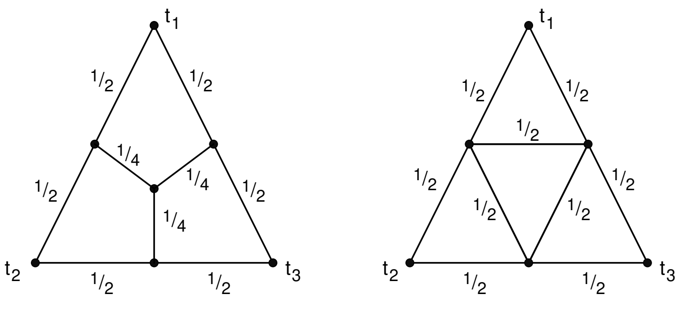

Figure 1: Bad examples for $(LP2)$

Recently, Călinescu, Karloff, and Rabani [1, 2] gave a new linear-programming relaxation. Although their approach applies to any number k of terminals, we continue to restrict attention to the case when $k = 3$. They need to assume that G be a complete graph. (If any missing edges are added with weight zero, the resulting 3-cut problem is equivalent to the given one, so this assumption is not limiting.) The relaxation is based on the following observations. First, every minimal 3-cut is of the form $\beta(R_1, R_2, R_3)$, where $t_i \in R_i$ for all i. Here, where $\mathcal{R}$ is a family of disjoint subsets of V whose union is V, $\beta(\mathcal{R})$ denotes the set of all edges of G joining nodes in different members of the family. Since $c \geq 0$, there is an optimal 3-cut of this form. Second, the incidence vector x of a minimal 3-cut is a kind of distance function, that is, it defines a function $d(v, w) = x_{vw}$ on pairs of nodes of G which is non-negative, symmetric, and satisfies the triangle inequality. Finally, with respect to d the distance between any two terminals is 1, and the sum of the distances from any node v to the terminals is 2. The resulting linear-programming relaxation is:

$$
(LP3) \qquad \begin{array}{lrcl}
\text{minimize} & \sum_{e\in E} c_e x_e & & \\
& \text{subject to} & & \\
& x_{vw} & = & 1, \quad v, w \in T,\ v \neq w \\
& \sum_{v\in T} x_{vw} & = & 2, \quad w \in V \\
& x_{uv} + x_{vw} - x_{uw} & \geq & 0, \quad u, v, w \in V \\
& x_e & \geq & 0, \quad e \in E.
\end{array}
$$

This relaxation is at least as tight as $(LP2)$. To see this, suppose that (after adding missing edges to make G complete), we have a feasible solution x to $(LP3)$. Then for any path P of G joining u to v, $x(P) \geq x_{uv}$, by applying the triangle inequality. It follows that $x(P) \geq 1$ for any T-path P. Moreover, any wye Y is the disjoint union of paths P_1, P_2, P_3 from some node v to the terminals. It follows that $x(Y) \geq \sum_{w\in T} x_{vw} = 2$. Thus every feasible solution of $(LP3)$ gives a feasible solution of $(LP2)$ having the same objective value. The first example of Figure 1 shows that the optimal value of $(LP3)$ can be strictly greater than the optimal value of $(LP2)$. On the other hand, the second example shows that there is no hope to prove in general that the minimum weight of a 3-cut is less than $\frac{16}{15}$ times the optimal value of $(LP3)$.

It was proved in [1, 2] that the minimum weight of a 3-cut is at most $\frac{7}{6}$ times the optimal value of $(LP3)$. As a consequence, an approximation algorithm for the optimal 3-cut problem having a performance guarantee of $\frac{7}{6}$ was derived. (It is clear that $(LP3)$ can be solved in polynomial time, since it is of polynomial size.) However, it was left open whether or not this result could be strengthened; the second example of Figure 1 shows an example for which the minimum weight of a 3-cut can be as large as $16/15$ times the optimal value of $(LP3)$, and this is the worst example given in [1, 2]. (To see that x of that example does extend to a feasible solution of $(LP3)$, we simply define x on each missing edge uv to be the minimum length, with respect to lengths x_e, of a path from u to v.)

In this paper we show that the minimum weight of a 3-cut is at most $\frac{12}{11}$ times the optimal value of $(LP3)$, and we show that the constant $\frac{12}{11}$ is best possible. As a consequence, we give an approximation algorithm for the optimal 3-cut problem and prove that it has a performance guarantee of $\frac{12}{11}$. These results were obtained independently by Karger, Klein, Stein, Thorup, and Young [10, 11]. We also provide a more precise bound, which depends on the value of the least common denominator of the components of an optimal solution to $(LP3)$.

The main results above were described in the short paper of the last two authors [6]. The current paper contains more detailed proofs. In addition, we answer one question that was left open in [6]. Namely, we show that the performance guarantee of the algorithm is best possible; that is, no better guarantee holds for this algorithm.

2 Triangle embeddings

Călinescu, Karloff, and Rabani [1, 2] considered an extremely useful geometric relaxation, which they showed was equivalent to the linear-programming relaxation $(LP3)$. Let $\triangle$ denote the convex hull of the three elementary vectors $e^1 = (1,0,0)$, $e^2 = (0,1,0)$, and $e^3 = (0,0,1)$ in $\mathbf{R}^3$. By a *triangle embedding* of G we mean a mapping y from V into $\triangle$ such that $y(t_i) = e^i$ for $i = 1,2,3$. A triangle embedding y determines a vector $x \in \mathbf{R}^E$ as follows. For each edge uv, let x_{uv} be one-half the L_1 distance from $y(u)$ to $y(v)$. It is easy to see that this x is a feasible solution to $(LP3)$. Conversely, a feasible solution x of $(LP3)$ determines a triangle embedding y as follows. For each node v, let $y(v) = (1 - x_{t_1 v}, 1 - x_{t_2 v}, 1 - x_{t_3 v})$.

Given a triangle embedding y we can obtain x as above, and then use x to obtain a triangle embedding y'. It is easy to see that $y = y'$. It is not true, on the other hand, that every feasible solution of $(LP3)$ arises in this way from a triangle-embedding. However, it is "almost true". The following result is implicit in [1, 2], and we include a proof for completeness.

Theorem 1 *Let x be a feasible solution of $(LP3)$, let y be the triangle embedding determined by x and let x' be the feasible solution of $(LP3)$ determined by y. Then $x' \leq x$, and if x is an optimal solution of $(LP3)$, so is x'.*

Proof. First, observe that the second statement is a consequence of the first and the non-negativity of c. Now let $uv \in E$. Both $y(u)$ and $y(v)$ have component-sum 1. Therefore, $y(u) - y(v)$ has component-sum zero, and so one-half of the L_1 distance between $y(u)$ and $y(v)$ is the sum of the non-negative components of $y(u) - y(v)$. Hence we may assume, perhaps by interchanging u with v and relabelling the terminals, that one-half of the L_1 distance between $y(u)$ and $y(v)$ is the sum of the first two components of $y(u) - y(v)$. Therefore,

$$\begin{aligned} x'_{uv} = \frac{1}{2}\|y(u) - y(v)\|_1 &= y_1(u) - y_1(v) + y_2(u) - y_2(v) \\ &= 1 - x_{ut_1} - (1 - x_{vt_1}) + 1 - x_{ut_2} - (1 - x_{vt_2}) \\ &= x_{ut_3} - x_{vt_3} \\ &\leq x_{uv}, \end{aligned}$$

as required. $\square$

The approximation algorithm of Călinescu, Karloff, and Rabani uses the following ideas. Suppose that $(LP3)$ is solved, and an optimal solution x^* that arises from a triangle embedding is found. For a number α between 0 and 1 that is different from x^*_{rv} for every $v \in V$ and $r \in T$, and an ordering r, s, t of T, define $R_r = \{v \in V : x^*_{rv} < \alpha\}$,

$R_s = \{v \in V \backslash R_r : x^*_{sv} < \alpha\}$, $R_t = V \backslash (R_r \cup R_s)$. We call the 3-cut $\beta(R_r, R_s, R_t)$ *uniform* (with respect to this x^*). It is easy to see that there are $O(n)$ uniform 3-cuts. The algorithm of [1, 2] simply chooses the uniform 3-cut having minimum weight. It is proved to have weight at most $\frac{7}{6}$ times the minimum weight of a 3-cut.

We consider a slight generalization of the notion of uniform 3-cut. Let α, α' be two numbers chosen as α was above, and let r, s, t be an ordering of T. Define $R_r = \{v \in V : x^*_{rv} < \alpha\}$, $R_s = \{v \in V \backslash R_r : x^*_{sv} < \alpha'\}$, $R_t = V \backslash (R_r \cup R_s)$. We call the 3-cut $\beta(R_r, R_s, R_t)$ *flat* (with respect to this x^*). Clearly, every uniform 3-cut is flat. It is easy to see that there are $O(n^2)$ flat 3-cuts. Our approximation algorithm simply chooses the flat 3-cut having minimum weight. We will show that it has weight at most $\frac{12}{11}$ times the weight of an optimal 3-cut. This result is based on a tight analysis of the bound for the optimal 3-cut problem given by $(LP3)$.

3 Linear programming again

It is easy to check that if the optimal value of $(LP3)$ is zero, then there is a 3-cut of weight zero. Therefore, we may assume that the optimal value is positive. Define

$$\rho := \inf_{G,c} \frac{\text{optimal value of } (LP3)}{\text{minimum weight of a 3-cut}}.$$

So our problem may be restated as finding the value of ρ. By multiplying c by an appropriate positive number, we may assume that the minimum weight of a 3-cut is 1. It is now more convenient to determine the best lower bound on the value of $(LP3)$.

Assume that G is fixed, and that an optimal solution x^* of $(LP3)$ is also fixed. We may assume that x^* is rational, since it is an optimal solution of a linear-programming problem having rational data. Therefore, there exists a positive integer q such that qx^* is integer-valued. By Theorem 1, we may assume that x^* arises from a triangle-embedding y^*, and it is easy to see that qy^* is integral, as well. Therefore, we can think of y^* as embedding the nodes of G into a finite subset $\triangle_q$ of $\triangle$, consisting of those points $y \in \triangle$ for which qy is integral. We define the planar graph $G_q = (\triangle_q, E_q)$ by $uv \in E_q$ if and only if the L_1 distance between u and v is $\frac{2}{q}$. Figure 2 shows G_9. (Note that the definition of the vertices as points in $\mathbf{R}^3$ and the edges as straight line segments joining their ends provides a natural embedding into the plane defined by $x_1 + x_2 + x_3 = 1$. We make use of this embedding whenever the context assumes G_q to be a plane graph.)

For nodes u, v of G_q, we denote by $d_q(u, v)$ the least number of edges of a path in G_q from u to v. It is easy to see that $d_q(u, v)$ is equal to $\frac{q}{2}$ times the L_1 distance from u to v.

Theorem 2 *Let G, c be a 3-cut instance, let x^* be a rational-valued optimal solution of $(LP3)$, with corresponding triangle-embedding y^*, and let q be a positive integer such that qx^* is integral. Then there is a 3-cut instance on graph $\hat{G}$ with nodeset $\triangle_q$ and edge-weights $\hat{c}$ such that:*

(a) $\hat{x}$ defined by $q\hat{x}_{uv} = d_q(u, v)$ for all $uv \in E$ is a feasible solution of $(LP3)$ (for $\hat{G}, \hat{c}$), and $\hat{c}\hat{x} \le cx^*$;

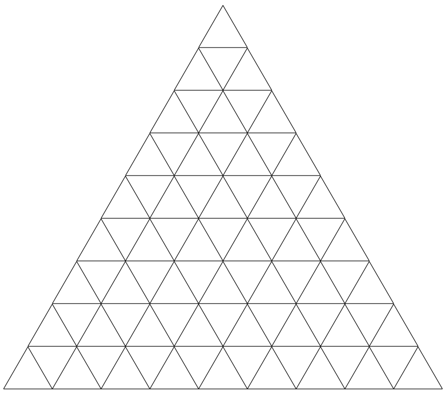

Figure 2: G_9

(b) The optimal 3*-cut value for* $\hat{G}, \hat{c}$ *is at least that for* G, c*;*

(c) $\hat{c}_e = 0$ *for all* $e \notin E_q$*;*

(d) For every flat 3*-cut of* $\hat{G}$ *with respect to* $\hat{x}$*, there is a flat* 3*-cut of* G *with respect to* x^* *having no larger weight.*

Proof. We use the mapping y^* from V to $\triangle_q$, and we assume that x^* arises from y^*. Suppose that two nodes u, v of G are mapped to the same point of $\triangle_q$ by y^*. Form G' by identifying u with v and, where multiple edges are formed, replacing the pair by a single edge whose weight is their sum. Then every 3-cut of G' determines a 3-cut of G having the same weight, so the minimum weight of a 3-cut of G' is at least the minimum weight of a 3-cut of G. Moreover, x^* also determines a triangle-embedding of G', so there is a feasible solution of $(LP3)$ for G' having value cx^*. Finally, every flat cut of G' gives a flat cut of G of the same weight. Thus the theorem is true for G if it is true for G', and so we may assume that y^* is one-to-one.

Now suppose that y^* is not onto, that is, that there is an element z of $\triangle_q$ such that $y^*(v) \neq z$ for all $v \in V$. We can form a graph G' from G by adding a node v and an edge uv of weight zero for every $u \in V$. It is easy to see that the minimum weight of a 3-cut of G' is the same as that of G. Also, if we map the new node to z, we get a triangle embedding of G', and it corresponds to a feasible solution of $(LP3)$ on G' having value equal to cx^*. Finally, every flat cut of G' corresponds to a flat cut of G of the same weight. So the theorem is true for G if it is true for G'. It follows that we may assume that y^* is onto. Therefore, we may assume that $V = \triangle_q$, and that y^* is the identity mapping.

Now suppose that there exists $uv \in E \backslash E_q$, such that $c_{uv} = \varepsilon > 0$. Let P be the edge-set of a path in G_q from u to v such that $|P| = d_q(u, v)$. Decrease c_{uv} to zero, and increase c_e by ε for all $e \in P$. We denote the new c by c'. Then, since every 3-cut using e uses an edge from P, the minimum weight of a 3-cut with respect to c' is not less than that with respect to c. (Similarly, every flat 3-cut has value with respect to c' not less than that with respect to c.) Now $c'x^* = cx^* - \varepsilon d_q(u, v) + \varepsilon d_q(u, v) = cx^*$. This argument can be repeated as long as there is such an edge uv. □

(Remark: It can be shown that $\hat{x}$ of Theorem 2 is an extreme point of the feasible region of $(LP3)$.)

For each positive integer q, let $F(q)$ be the optimal value of the following linear-programming problem.

$$(P_q) \qquad \begin{array}{lrcll} \text{minimize} & \multicolumn{4}{l}{\frac{1}{q}\sum_{e\in E} c_e} \\ & \multicolumn{4}{l}{\text{subject to}} \\ & c(S) & \geq & 1, & S \text{ a 3-cut of } G_q \\ & c_e & \geq & 0, & e \in E_q. \end{array}$$

The dual problem is

$$(D_q) \qquad \begin{array}{lrcll} \text{maximize} & \multicolumn{4}{l}{\sum z_S} \\ & \multicolumn{4}{l}{\text{subject to}} \\ & \sum_{e\in S} z_S & \leq & \frac{1}{q}, & e \in E_q \\ & z_S & \geq & 0, & S \text{ a 3-cut of } G_q. \end{array}$$

Proposition 3 $\rho = \inf_q F(q)$.

Proof. Let ρ' denote $\inf_q F(q)$. It is easy to see from Theorem 2 that ρ' is a lower bound for ρ.

Now, consider an optimal solution $\bar{c}$ to (P_q) for some q. Clearly, the optimal weight of a 3-cut in the weighted graph $(G_q, \bar{c})$ is 1. However, $\hat{x}$ as defined in Theorem 2 is a feasible solution to $(LP3)$ for $(G_q, \bar{c})$ with objective value $F(q)$. Thus, $\rho \leq F(q)$. Since q is arbitrary, it follows that $\rho \leq \rho'$. The result follows. $\square$

We used CPLEX to solve (P_q) and (D_q) for all values of q up to 50, and then were able to find solutions for general q.

Theorem 4 *For $q \geq 1$,*

$$F(q) = \begin{cases} \frac{11}{12} + \frac{1}{12(q+1)}, & \text{if } q \equiv 0 \bmod 3 \\ \frac{11}{12} + \frac{1}{12q}, & \text{if } q \equiv 1 \bmod 3 \\ \frac{11}{12} + \frac{1}{12q} - \frac{1}{12q^2}, & \text{if } q \equiv 2 \bmod 3 \end{cases}$$

Moreover, there is an optimal solution of (D_q) for which z_S is positive only if S is a flat 3-cut.

It is easy to see that Proposition 3 and Theorem 4 have the following consequence.

Theorem 5 *For any 3-cut instance, the minimum weight of a 3-cut is at most $\frac{12}{11}$ times the optimal value of $(LP3)$, and the constant $\frac{12}{11}$ is best possible.* $\square$

Theorem 5 has been proved independently by Karger *et al.* [10], whose approach is somewhat different, but also uses a linear-programming analysis of triangle-embedding.

4 An improved approximation algorithm

Algorithm 3-CUT

1. Find a rational-valued optimal solution x^* of $(LP3)$.
2. Find the triangle embedding y^* determined by x^*.
3. Return the flat 3-cut of minimum weight.

As pointed out before, the first step can be performed in polynomial time. The polynomial-time algorithms for linear programming can be modified to return a rational-valued optimal solution, and one of polynomial size. The second is easy. So is the third step, using the observation made earlier that there are only $O(n^2)$ flat 3-cuts of G.

Theorem 6 *Algorithm 3-CUT returns a 3-cut of weight at most $\frac{1}{F(q)}cx^*$ where q is a common denominator for the components of x^*.*

Proof. We may assume that the optimal value of a 3-cut is 1. Consider an optimal solution z^* of (D_q) as given by Theorem 4. Then

$$\sum_S \frac{1}{F(q)} z_S^* \geq 1,$$

and $z_S^* > 0$ only if S is a flat 3-cut of G_q. Obtain $\hat{c}$ from Theorem 2. Then,

$$\begin{aligned}
& \text{min weight of a flat 3-cut of } (G, c) \\
\leq\; & \text{min weight of a flat 3-cut of } (G_q, \hat{c}) \qquad \text{by part (d) of Theorem 2} \\
\leq\; & \min_{z_S^* > 0} \hat{c}(S) \\
\leq\; & \sum_S \frac{1}{F(q)} z_S^* \hat{c}(S) \\
=\; & \frac{1}{F(q)} \sum_S z_S^* \hat{c}(S) \\
=\; & \frac{1}{F(q)} \sum_{e \in E(G_q)} \hat{c}_e \sum_{e \in S} z_S^* \\
\leq\; & \frac{1}{F(q)} \sum_{e \in E(G_q)} \hat{c}_e \hat{x}_e \\
\leq\; & \frac{1}{F(q)} \sum_{e \in E} c_e x_e^* \qquad \text{by part (a) of Theorem 2.}
\end{aligned}$$

□

Corollary 7 *Algorithm 3-CUT returns a 3-cut of weight at most $\frac{12}{11}$ times the minimum weight of a 3-cut.*

Proof. Since $\frac{1}{F(q)} < \frac{12}{11}$ and the optimal value of $(LP3)$ is at most the minimum weight of a 3-cut, the result follows immediately from Theorem 6. □

5 Proof of Theorem 4

To prove Theorem 4, it is enough to give feasible solutions of (P_q) and of (D_q) having the claimed common objective value. We shall give the details only for the case when $q = 3m$ for some integer $q \geq 2$. Note that this is sufficient to obtain Theorem 5 and Corollary 7, since a common denominator for the components of x^* can always be chosen to have this property. (In fact, to prove Corollary 7 and all but the "best possible" part of Theorem 5, such a solution of (D_q) is enough.) The remaining cases are similar and do not add much more insight to the problem. Complete details of the other cases can be found in [3].

For a terminal t and an integer j, let $R_t(j)$ denote the set $\{v \in V_q : d_q(t, v) < j\}$. If a face triangle of G_q has the same orientation as $\triangle$, it is called *upright*; otherwise, it is *inverted*.

A solution to (D_q)

First we show a feasible solution of (D_q) having objective value $\frac{11}{12} + \frac{1}{12(3m+1)}$. This requires assigning dual variables to flat 3-cuts of G_q. We need some terminology.

We use the term *row* in the following technical sense. A row is defined by a straight line through the centre of a face triangle and parallel to one of its three sides. The terminal *opposite* to the row is the terminal separated by the straight line from the other two terminals. When we speak of the face triangles *in* the row, we mean all of the face triangles that are intersected by the line. When we speak of the edges *in* the row, we mean all of the edges that are intersected by the line. The *distance* between the row and its opposite terminal is defined as the shortest graph distance from the terminal to a vertex of one of the triangles in the row. Some of the above definitions are illustrated on the left in Figure 3.

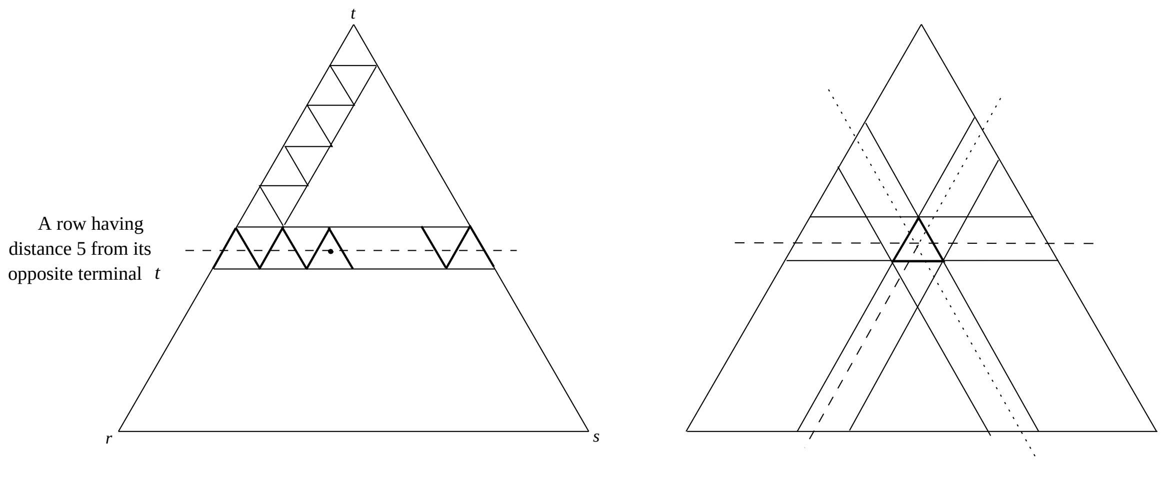

Figure 3: Illustrations for technical definitions

We assign positive dual variables to two kinds of flat 3-cuts. The values assigned to the first type of 3-cut are determined by a weighting of the face triangles of G_q. Actually, we assign weights only to upright face triangles. Figure 4 shows weightings of the face triangles for G_6 and G_9. (The weight of any face triangle containing no number is understood to be zero.)

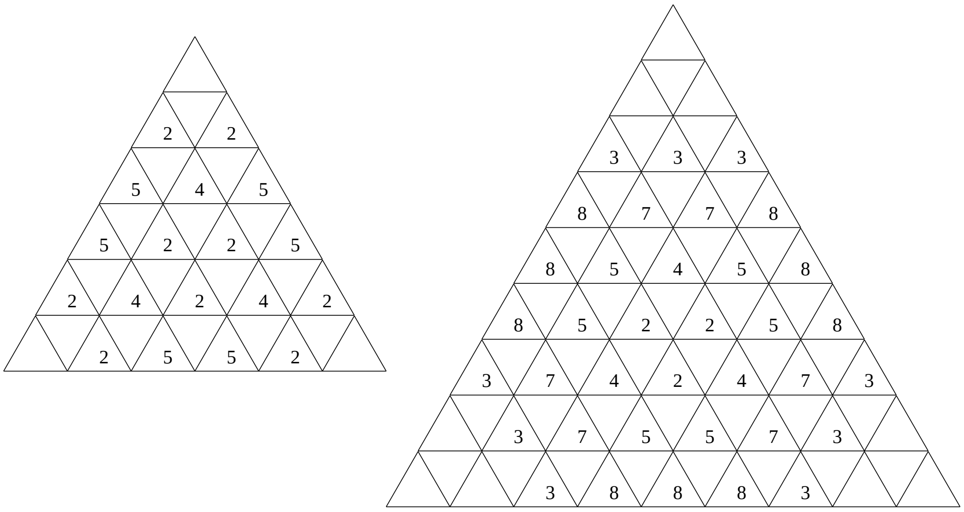

Figure 4: Weightings of the face triangles of G_6 and G_9

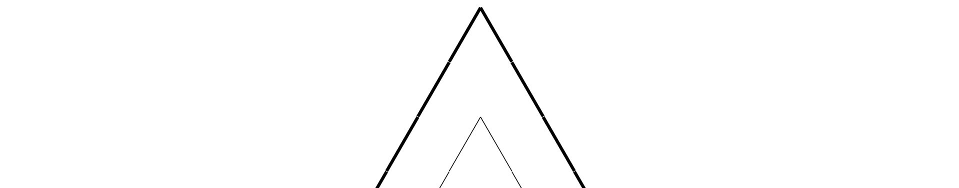

Figure 5: G_{3m} with $G_{3(m-1)}$ inside

A weighting for the general case can be defined inductively. Think of G_{3m} as consisting of the rows at distance $3m-1$ from the terminals together with $G_{3(m-1)}$ (see Figure 5) and use the face weighting for $G_{3(m-1)}$ with the following changes. In each row at distance $m-1$ from its opposite terminal, each upright triangle is assigned weight m. In each row at distance $3m-1$ from its opposite terminal, each upright triangle between the two ones assigned weight m above, is assigned weight $3m-1$. Finally, each upright triangle in a row at distance m from its opposite terminal, which was assigned weight $m-1$ in $G_{3(m-1)}$, is assigned weight $3m-2$. Clearly, the sum of the weights of the face triangles in each row of distance exactly $m-1$ from its opposite terminal is m^2. It is an easy induction to show that the sum of the weights of the face triangles in each row of distance at least m from its opposite terminal is $m(3m+1)$.

Given an upright face triangle, consider the set of all edges in the three rows containing the triangle. Choose two flat 3-cuts of G_q whose union is this set, and whose intersection is the set of edges of the face triangle. (There is more than one way to do this. See the illustration on the right in Figure 3.) For each of these two 3-cuts, assign a z-value equal to the weight of the face triangle divided by B, where $B = 6m^2(3m+1)$.

Let I_e denote the constraint of (D_q) corresponding to an edge e. The contribution to the left-hand side of I_e by the variables whose values we have just assigned is the sum of the weights of the face triangles in the two rows containing e divided by B. We now consider three types of edges e:

(a) Those for which the two rows containing e are at distance at least m from their respective opposite terminals, in which case this sum is twice $m(3m+1)/B = \frac{1}{3m}$;

(b) Those for which one of the rows containing e is at distance less than $m-1$ from the opposite terminal (so the other is at distance at least $m+1$ from its opposite terminal), in which case this sum is $m(3m+1)/B = \frac{1}{6m}$;

(c) Those for which one of the rows containing e is at distance exactly $m-1$ from the opposite terminal (so the other is at distance at least $m+1$ from its opposite terminal), in which case this sum is $m(3m+1)/B + m^2/B$.

Note that for edges of type (a) above, the dual variables already defined satisfy I_e with equality. We now assign positive z-values to some uniform 3-cuts, which will contribute to I_e only for edges e of types (b) and (c). For each uniform 3-cut S of the form $\beta(R_r(j), R_s(j), V\backslash(R_r(j) \cup R_s(j)))$ where r and s are two distinct terminals and $j \in \{1, 2, ..., m-1\}$, we set $z_S = \frac{1}{12m}$. These contribute to I_e only for edges of type (b), and it is easy to see that those inequalities are now satisfied with equality. Finally, for each uniform 3-cut S of the form $\beta(R_r(m), R_s(m), V\backslash(R_r(m) \cup R_s(m)))$ where r and s are two distinct terminals, we set $z_S = \frac{2m+1}{12m(3m+1)}$. Note that these variables contribute to I_e only for edges e of type (c), and it is easy to check that those inequalities are now satisfied with equality.

Hence we have defined a feasible solution to (D_q). It remains to compute its objective value. There are $3(m-1)$ variables corresponding to uniform 3-cuts with value $\frac{1}{12m}$ and three variables corresponding to uniform 3-cuts with value $\frac{2m+1}{12m(3m+1)}$. The contribution of the other variables is two times the sum of the weights of the face triangles divided by B.

Therefore the objective value is

$$\frac{3(m-1)}{12m} + \frac{3(2m+1)}{12m(3m+1)} + \frac{2(m^2 + 2m^2(3m+1))}{B} = \frac{11}{12} + \frac{1}{12(3m+1)},$$

as required.

A solution to (P_q)

We describe a feasible solution c of (P_q) having objective value $\frac{11}{12} + \frac{1}{12(3m+1)}$. The solution is given in terms of the integral vector $c' = 4(3m+1)c \in \mathbf{R}^{E_q}$. Figure 6 (ignoring the dotted edges) shows G_6. The numbers beside the edges are the values of c', except that values equal to 1 are omitted.

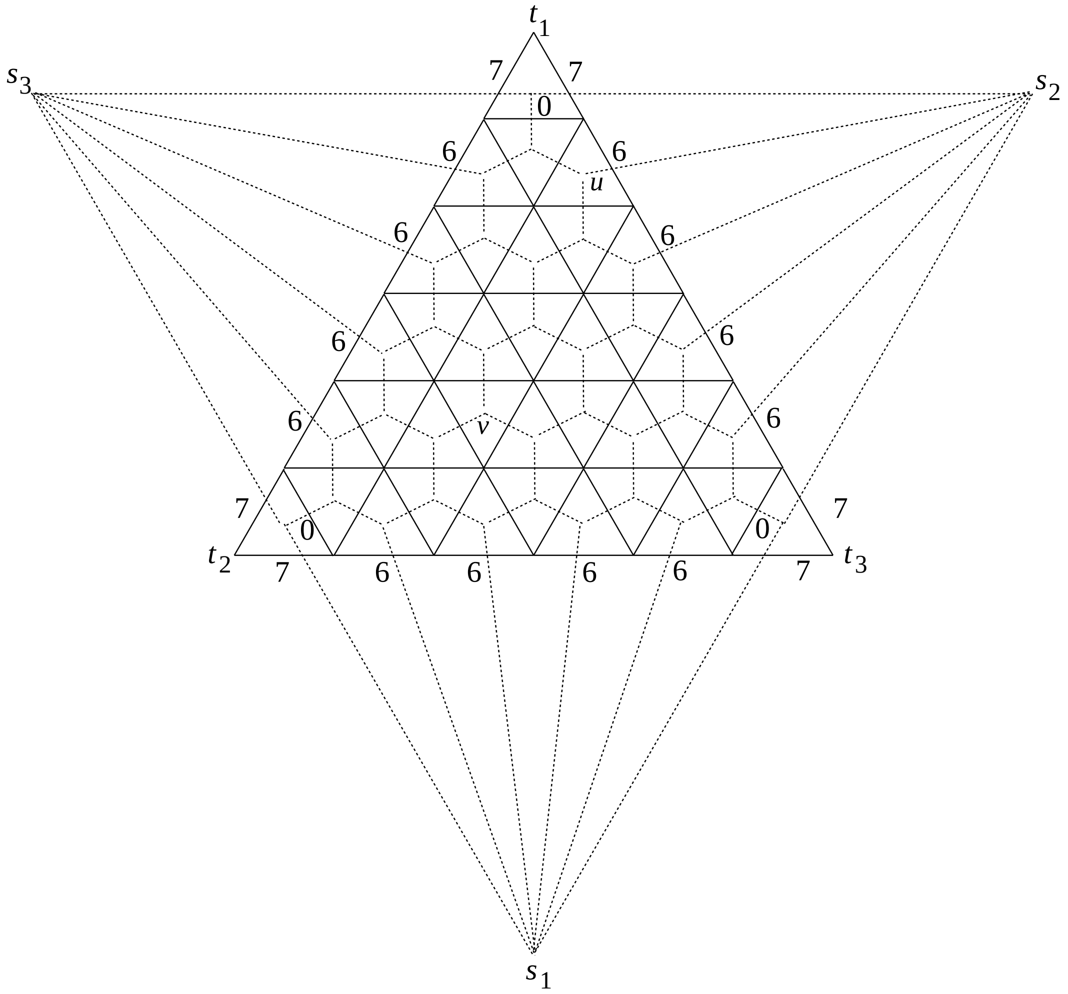

Figure 6: G_6 and (G_6', c')

Here is the general construction. (The construction described in [6] contained an error.) Divide G_{3m} into three *corner triangles* of side m together with the *middle hexagon*. An edge in a corner triangle is called a *peel edge* if it is parallel to some edge on the boundary of G_{3m} and of distance 1 from it. The *corner subtriangle* in a corner triangle is the triangle bounded by the peel edges and the boundary edges of the middle hexagon. (Note that when $m = 2$, the corner subtriangle is a single point.) In each corner triangle, the vertex on the corner subtriangle closest to the terminal is called an *apex*. Put $c'_e = 3m + 1$ for all edges incident with the terminals. Put $c'_e = 2m + 2$ for all other edges on the boundary of G_{3m}. Put $c'_e = m - 1$ for each peel edge incident with an apex and a vertex on the boundary of G_{3m}. In each corner subtriangle, put $c'_e = m - i - 1$ if e is a peel edge of distance i from the apex and put $c'_e = 1$ for all other edges parallel to a peel edge. Put $c'_e = 1$ for all other edges in the middle hexagon (including its boundary). Put $c'_e = 0$ for all other edges. Figure 7 (ignoring the values in italics) illustrates the definition of c'. The key result is the following.

Lemma 8 *The minimum weight of a 3-cut with respect to c' is $4(3m + 1)$.*

It follows that c is a feasible solution to (P_q). Its objective value is the total c'-weight of all edges, divided by $4(3m + 1)(3m)$. There are 6 edges e having $c'_e = 3m + 1$, $3(3m - 2)$ edges e having $c'_e = 2m + 2$, 6 edges e having $c'_e = m - 1$, 6 edges e having $c'_e = m - i - 1$ for $i = 0, ..., m - 3$, and $3(m - 2)(m - 3) + 9m^2$ edges having $c'_e = 1$, from which we compute the total c'-weight to be $33m^2 + 12m$. It follows that the objective value of c is $\frac{11}{12} + \frac{1}{12(3m+1)}$, as required.

The ideas for the proof of Lemma 8 come, essentially, from the result of Dahlhaus, *et al.* [7], showing that there is a polynomial-time algorithm to solve the optimal multiterminal cut problem when G is planar and the number of terminals is fixed. Any minimal 3-cut of G_q has the form $\beta(R_1, R_2, R_3)$. There are two kinds of such 3-cuts, corresponding to the case in which there is a pair i, j for which there is no edge joining a node in R_i to a node in R_j, and the other one where this is not true. We call these Type I cuts and Type II cuts, respectively.

From (G_q, c'), define a 3-terminal Steiner Tree Problem instance (G'_q, c') as follows: We take the planar dual of (G_q, c') and split $\mathcal{O}$, the vertex that corresponds to the outside face, into three vertices s_1, s_2, s_3, which we call the *terminals* of G'_q. We also split the edges incident with $\mathcal{O}$ as follows: an edge e is incident with s_i if e crosses an edge of G_q opposite terminal t_i. G'_6 is shown in Figure 6.

Observe that a Type I cut corresponds to a Steiner tree of (G'_q, c') with no degree-3 vertex. It is easy to see that, in order to show that such a Steiner tree has weight at least $4(3m + 1)$, it suffices to show the following.

Proposition 9 *The weight of a path in (G'_q, c') joining two distinct terminals is at least $2(3m + 1)$.*

Now, observe that a Type II cut corresponds to a Steiner tree of (G'_q, c') with a degree-3 vertex in $V(G'_q)\backslash\{s_1, s_2, s_3\}$. For each $v \in V(G'_q)\backslash\{s_1, s_2, s_3\}$, let $l_i(v)$ denote the length, with respect to c', of a shortest path from v to s_i in (G'_q, c') for each $i \in \{1, 2, 3\}$. To show that such a Steiner tree has weight at least $4(3m + 1)$, it suffices to show the following.

Proposition 10 *For each $v \in V(G'_q)\backslash\{s_1, s_2, s_3\}$, $\sum_{i=1}^{3} l_i(v) \geq 4(3m + 1)$.*

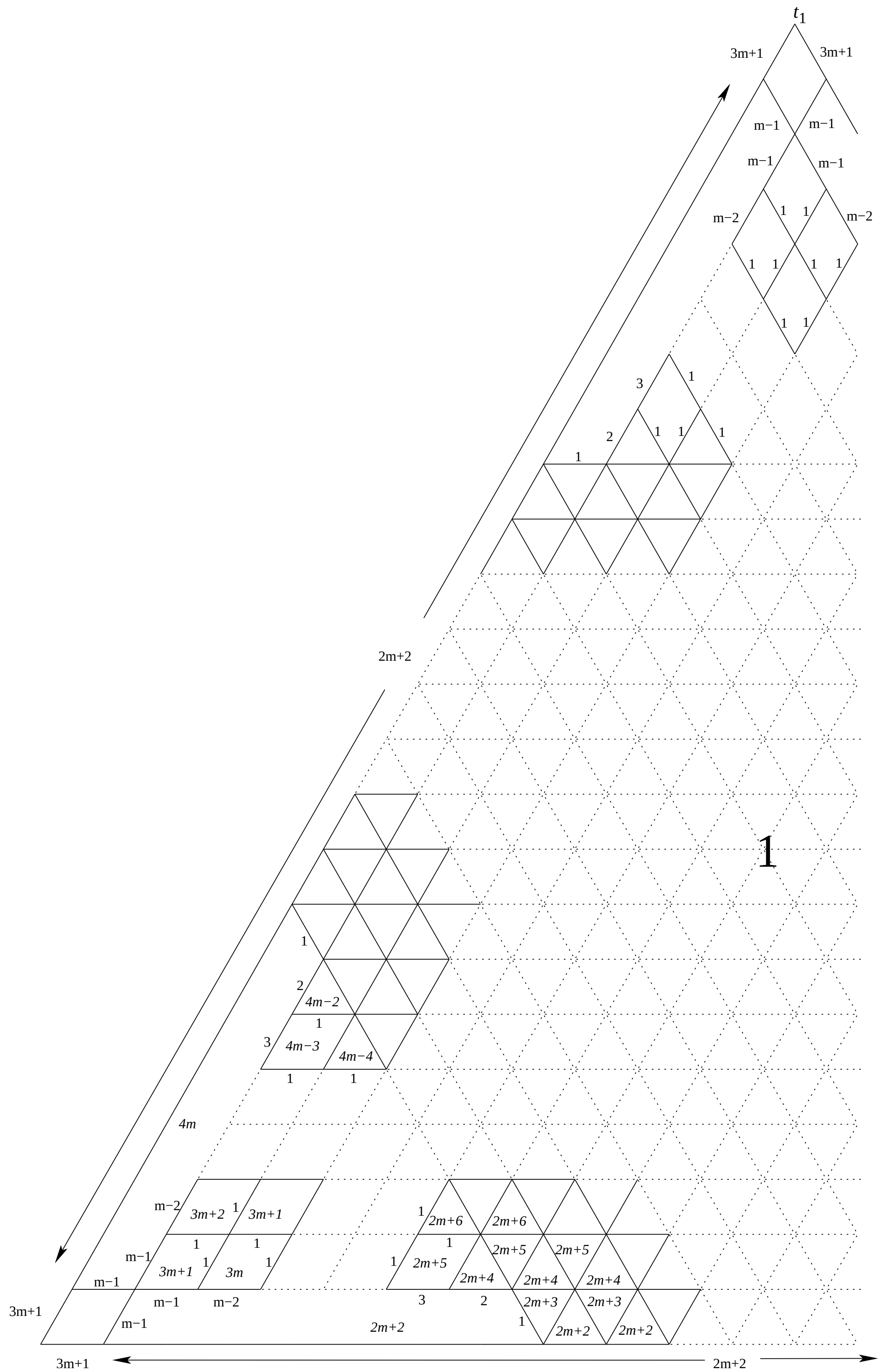

Figure 7: G_q for $q = 3m$

Hence, Lemma 8 follows from Propositions 9 and 10.

Before we prove Propositions 9 and 10, we need some further notation and technical results. Let $v \in V(G'_q)\backslash\{s_1, s_2, s_3\}$. Let $f(v)$ denote the face-triangle to which v corresponds. There is a natural coordinate system for the elements of $V(G'_q)\backslash\{s_1, s_2, s_3\}$. For each $v \in V(G'_q)\backslash\{s_1, s_2, s_3\}$, define $p^v \in \mathbf{R}^3$ as follows: For each $i \in \{1, 2, 3\}$, p^v_i is the smallest number that is the (graph) distance in G_q between a vertex of $f(v)$ and a vertex on the side of $\triangle$ opposite terminal t_i. For example, in Figure 6, we have $p^u = (4, 0, 1)$ and $p^v = (1, 2, 1)$. The following is immediate.

Lemma 11 *Let $v \in V(G'_q)\backslash\{s_1, s_2, s_3\}$. If $f(v)$ is upright, then $p^v_1 + p^v_2 + p^v_3 = 3m - 1$. If $f(v)$ is inverted, then $p^v_1 + p^v_2 + p^v_3 = 3m - 2$.*

For each $i \in \{1, 2, 3\}$, we are going to define $\pi^i \in \mathbf{R}^{\{s_i\}\cup(V(G'_q)\backslash\{s_1,s_2,s_3\})}$ with $\pi^i_{s_i} = 0$ such that π^i gives a feasible potential for the shortest-path problem from s_i to all the non-terminal vertices of G'_q. (This will certify the optimality of shortest paths.) We consider two cases.
Case 1: $f(v)$ is in the middle hexagon. If $f(v)$ is upright, then set $\pi^i_v = 2m + 2 + 2p^v_i$; otherwise, set $\pi^i_v = 2m + 2 + 2p^v_i + 1$.
Case 2: $f(v)$ is in a corner triangle. If $p^v_i \geq 2m$, then set $\pi^i_v = 6m + 2$. If $p^v_i < 2m$, then note that $0 \leq p^v_i \leq m - 1$ and there exist j, k such that $\{i, j, k\} = \{1, 2, 3\}$ with $p^v_j \leq m - 1$ and $p^v_k \geq m$.

Suppose $p^v_i = 0$. If $p^v_j = 0$, then set $\pi^i_v = 3m + 1$; otherwise, set $\pi^i_v = 2m + 2$.

Suppose $p^v_i \geq 1$. If $p^v_j = 0$, then set $\pi^i_v = 4m$; otherwise, set $\pi^i_v = 3m + 1 + p^v_i - p^v_j$.

Figure 7 illustrates some of the entries of π^1 in italics. From the picture, it is obvious that $\pi^i_u + c'_{uw} \geq \pi^i_w$ for all $uw \in E(G'_q - (\{s_1, s_2, s_3\}\backslash\{s_i\}))$. Hence, we have the next two lemmas. The first is immediate.

Lemma 12 *For each $i \in \{1, 2, 3\}$, $l_i(v) \geq \pi^i_v$ for all v in $V(G'_q)\backslash\{s_1, s_2, s_3\}$.*

Lemma 13 *Let $i, j \in \{1, 2, 3\}$ be distinct. The weight of the shortest path between s_i and s_j is at least the minimum value of $\pi^i_v + \pi^j_v$ over all v such that $p^v_i = 0$.*

Proof. This follows from the previous lemma and that $p^v_i = 0$ for every neighbour v of s_i.
□

Proof of Proposition 9.
By symmetry, we may assume that the terminals are s_1 and s_2. By Lemma 13, it suffices to show that for every $v \in V(G'_q)\backslash\{s_1, s_2, s_3\}$ such that $p^1_v = 0$, we have $\pi^1_v + \pi^2_v \geq 2(3m + 1)$. From the definition of π^1 and π^2, we see that if $p^2_v = 0$, then $\pi^1_v = \pi^2_v = 3m + 1$, giving $\pi^1_v + \pi^2_v \geq 2(3m + 1)$; otherwise, $\pi^1_v \geq 2m + 2$ and $\pi^2_v \geq 4m$, again giving $\pi^1_v + \pi^2_v \geq 2(3m + 1)$.
□

Proof of Proposition 10.
We consider two cases.
Case 1: $f(v)$ is in the middle hexagon.

By Lemma 12, it suffices to show that $\sum_{i=1}^{3} \pi_v^i \geq 4(3m+1)$. Suppose $f(v)$ is upright. Then $\pi_v^i = 2m+2+2p_i^v$ for $i = 1,2,3$. By Lemma 11,

$$\sum_{i=1}^{3} \pi_v^i = \sum_{i=1}^{3}(2m+2+2p_i^v) = 6m+6+2(3m-1) = 4(3m+1).$$

Now, suppose $f(v)$ is inverted. Then $\pi_v^i = 2m+2+2p_i^v+1$ for $i = 1,2,3$. By Lemma 11,

$$\sum_{i=1}^{3} \pi_v^i = \sum_{i=1}^{3}(2m+2+2p_i^v+1) = 6m+9+2(3m-2) = 4(3m+1)+1 > 4(3m+1)$$

as desired.

Case 2: $f(v)$ is in a corner triangle.

By symmetry, we may assume that $p_1^v \geq 2m$. Hence, $\pi_v^1 = 6m+2$ by construction. Now, it follows from Proposition 9 that $l_v^2 + l_v^3 \geq 2(3m+1)$. Therefore, $\sum_{i=1}^{3} l_v^i \geq 6m+2+2(3m+1) = 4(3m+1)$. □

6 Bad Examples for Algorithm 3-CUT

Since the constant $\frac{12}{11}$ is best possible in Theorem 5, it is natural to ask if it is best possible in Corollary 7. The two issues are different. By Theorem 6, the weight of the flat 3-cut delivered by Algorithm 3-CUT is at most $1/F(q)$ times the optimal value of $(LP3)$. It follows that, if that value is close to $F(q)$ times the weight of an optimal 3-cut, then Algorithm 3-CUT will deliver a 3-cut that has weight close to the weight of an optimal 3-cut. Therefore, bad examples for Theorem 5 do not directly provide bad examples for Corollary 7. However, such examples do exist.

Theorem 14 *For each $q \equiv 0 \pmod 6$, there exist a weighted graph (H_q, c) and an embedding of H_q determining an optimal solution for $(LP3)$, such that Algorithm 3-CUT delivers a flat 3-cut (with respect to the embedding) having weight $\frac{12(q+1)}{11q+12}$ times the weight of some 3-cut.*

Here is the class of graphs that we will use to prove Theorem 14. Let $q = 6m$ where m is a positive integer. Construct the weighted graph (H_q, c) as follows. Take (G_q, c'). For each outside edge e on the line joining t_2 and t_3, reduce the weight on e by $2m+2$. Let $\tilde{v}$ be the vertex at the midpoint between t_2 and t_3. Let $\tilde{u}$ and $\tilde{w}$ be the two neighbours of $\tilde{v}$ that lie on the line joining t_2 and t_3 with $\tilde{u}$ closer to t_2. Remove the edges $\tilde{u}\tilde{v}$ and $\tilde{v}\tilde{w}$. Add the edge $\tilde{u}\tilde{w}$ with weight $2m$ and the edges $t_2\tilde{v}$ and $t_3\tilde{v}$, each with weight $2m+2$. The resulting weighted graph (H_q, c) is depicted in Figure 8.

Before proving Theorem 14, we briefly describe the origin of the above construction. When seeking bad examples, there are two main issues to consider. First, we need to identify weighted graphs such that the ratio of the weight of the best flat 3-cut with respect to some

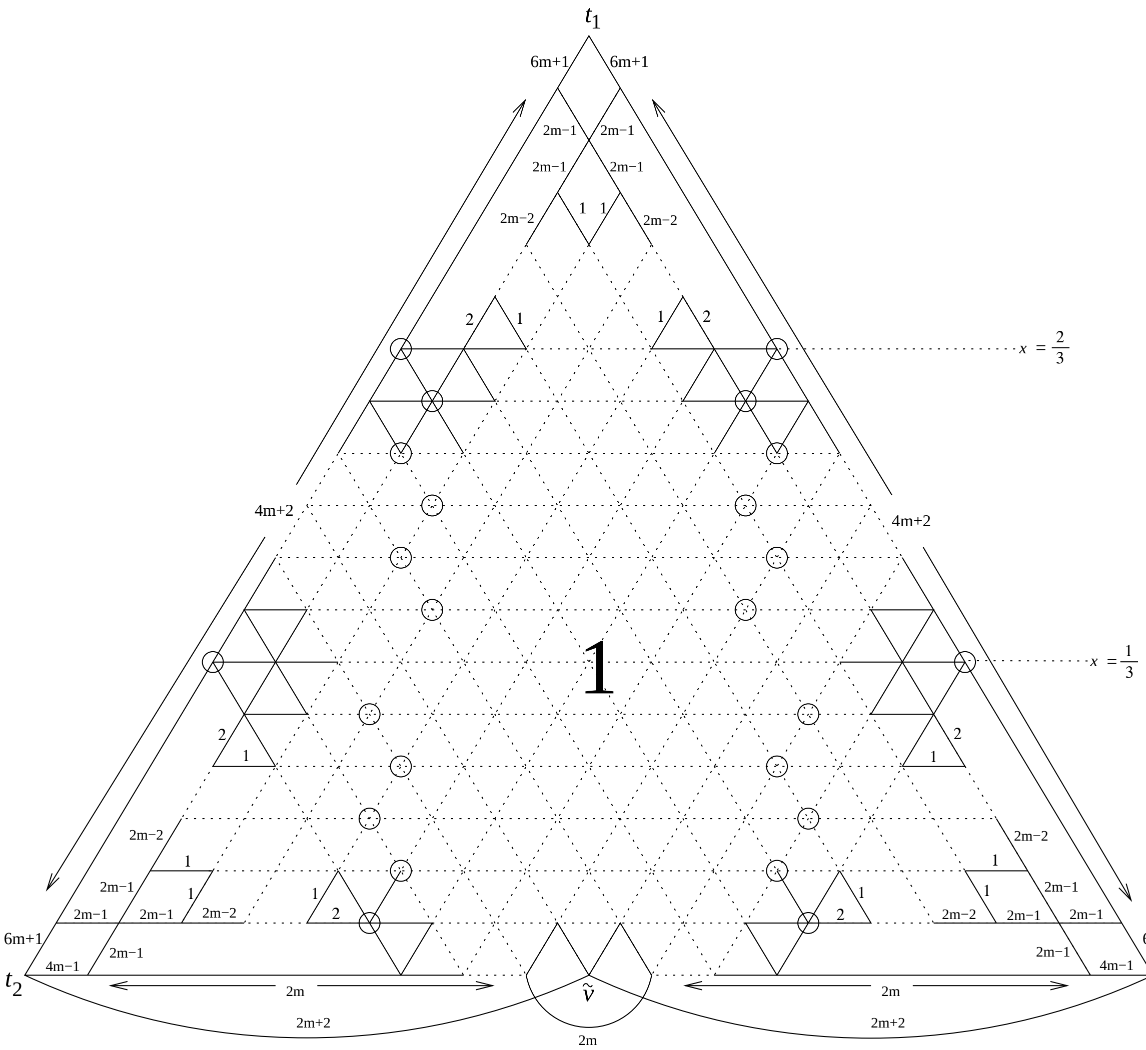

Figure 8: H_q with edge weights

embedding of the graph into $\triangle$ and the weight of the optimal 3-cut is close to $\frac{12}{11}$. Second, given such a weighted graph, we need to make sure the embedding does yield an optimal solution to $(LP3)$. It is not clear how to resolve both issues at the same time. What follows is an outline of our approach.

We focused on graphs that have vertex-set $\triangle_q$ for small values of q. (That is, we assumed that the graphs were already embedded into $\triangle$.) Initially, the graphs were assumed to be complete. We did the following for each graph G we chose. For each (non-flat) 3-cut C in G, we generated a linear-programming problem. For each edge, there is a variable representing its unknown weight. For each flat 3-cut, we have a constraint that ensures that it has weight at least 1. The objective function is to minimize the weight of the 3-cut C.

We went through all the 3-cuts of G and identified candidates that gave the best possible ratio. We then went through the process once again for these candidates, each time gradually reducing the number of edges that are not in E_q. We then experimented with the weight vector to see if the embedding we started with actually gave an optimal solution to the linear-programming problem with the given weight vector. A pattern gradually emerged and it allowed us to guess which graphs to consider for higher values of q. After performing the computations on the candidates for higher values of q, we formulated a conjecture on what would be an infinite family of bad examples for $q \equiv 0 \pmod 6$. Finally, we proved the conjecture using analytical methods.

Proof of Theorem 14.

Since $V(H_q) = \triangle_q$, the 3-tuples of the vertices give an embedding of H_q into $\triangle$. Note that a flat 3-cut in this embedding has the same weight as the corresponding flat 3-cut in (G_q, c'). By Lemma 8, every 3-cut in (G_q, c') has weight at least $4(q+1)$. Thus every flat 3-cut in (H_q, c) has weight at least $4(q+1)$.

Now, the 3-cut

$$\beta(\{t_3\}, \{t_2, \tilde{v}\}, V(H_q)\backslash\{t_2, t_3, \tilde{v}\}).$$

has weight $(6m+1)+(4m-1)+2+(2m+2)+(4m-1)+(6m+1) = 22m+4 = \frac{11q+12}{3}$. Thus the ratio of the value of an optimal flat 3-cut in the embedding of H_q to the value of this 3-cut is $4(q+1)/((11q+12)/3) = 12(q+1)/(11q+12)$. As q approaches infinity, the ratio approaches $\frac{12}{11}$. Hence, it remains to show that the embedding given by the 3-tuples of the vertices of H_q determines an optimal solution to $(LP3)$.

By Theorem 1, it suffices to show that the embedding is optimal for

$$\begin{array}{ll} \min \displaystyle\sum_{uv \in E(H_q)} \frac{1}{2} c_{uv} \|\mathbf{x}^u - \mathbf{x}^v\|_1 & \\ \text{subject to} & \\ \mathbf{x}^u \in \triangle, & u \in V(H_q)\backslash\{t_1, t_2, t_3\} \\ \mathbf{x}^{t_i} = e_i, & i = 1, 2, 3. \end{array}$$

Observe that the objective value given by the embedding is $22m+4$. (This can also be seen using the calculation in the paragraph following Lemma 8.) We show that $22m+4$ is the optimal value using linear-programming duality. Since $c \geq 0$, writing $\mathbf{x}^u$ as $(x_u, y_u, z_u)^T$, we can rewrite the above minimization problem as the following linear-programming problem. This problem was introduced by Calinescu *et al.* [1, 2].

$$
(SLP)\qquad
\begin{array}{l}
\min \displaystyle\sum_{uv\in E(H_q)} \frac{1}{2} c_{uv}(X_{uv}+Y_{uv}+Z_{uv}) \\
\quad \text{subject to} \\
X_{uv}-x_u+x_v \geq 0, \quad uv\in E(H_q) \\
X_{uv}-x_v+x_u \geq 0, \quad uv\in E(H_q) \\
Y_{uv}-y_u+y_v \geq 0, \quad uv\in E(H_q) \\
Y_{uv}-y_v+y_u \geq 0, \quad uv\in E(H_q) \\
Z_{uv}-z_u+z_v \geq 0, \quad uv\in E(H_q) \\
Z_{uv}-z_v+z_u \geq 0, \quad uv\in E(H_q) \\
x_u+y_u+z_u = 1, \quad u\in V(H_q)\backslash\{t_1,t_2,t_3\} \\
x_{t_1}=1,\ y_{t_1}=0,\ z_{t_1}=0 \\
x_{t_2}=0,\ y_{t_2}=1,\ z_{t_2}=0 \\
x_{t_3}=0,\ y_{t_3}=0,\ z_{t_3}=1 \\
x,y,z\geq 0.
\end{array}
$$

Notice that, while there is one variable X_{uv} for each edge $uv\in E(H_q)$—that is, X_{uv} is the same as X_{vu}—there is a constraint $X_{uv}-x_u+x_v\geq 0$ for each *ordered* pair (u,v) such that $uv\in E(H_q)$. A similar observation holds for Y and Z. Therefore, it is convenient to introduce the digraph H_q' obtained from H_q by replacing each edge by a pair of oppositely directed edges. Now, we write the dual of (SLP). We make use of the notation $f_z(u)$ to denote the "net outflow" from vertex u in H_q', with respect to $z\in \mathbf{R}^{E(H_q')}$, namely, $f_z(u)=\sum_{w:uw\in E(H_q')} z_{uw}-\sum_{w:uw\in E(H_q')} z_{wu}$.

$$
(DSLP)\qquad
\begin{array}{l}
\max \quad \delta_{t_1}+\varepsilon_{t_2}+\phi_{t_3}+\sum_{u\in V(H_q)\backslash\{t_1,t_2,t_3\}}\theta_u \\
\qquad\qquad \text{subject to} \\
\alpha_{uv}+\alpha_{vu}=\frac{c_{uv}}{2}, \quad uv\in E(H_q) \\
\beta_{uv}+\beta_{vu}=\frac{c_{uv}}{2}, \quad uv\in E(H_q) \\
\gamma_{uv}+\gamma_{vu}=\frac{c_{uv}}{2}, \quad uv\in E(H_q) \\
\theta_u\leq f_\alpha(u), \quad u\in V(H_q)\backslash\{t_1,t_2,t_3\} \\
\theta_u\leq f_\beta(u), \quad u\in V(H_q)\backslash\{t_1,t_2,t_3\} \\
\theta_u\leq f_\gamma(u), \quad u\in V(H_q)\backslash\{t_1,t_2,t_3\} \\
\delta_{t_i}\leq f_\alpha(t_i), \quad i=1,2,3 \\
\varepsilon_{t_i}\leq f_\beta(t_i), \quad i=1,2,3 \\
\phi_{t_i}\leq f_\gamma(t_i), \quad i=1,2,3 \\
\alpha,\beta,\gamma\geq 0.
\end{array}
$$

We now give a feasible solution to $(DSLP)$ having objective value $22m+4$. We do this in two steps. First, we fix the values of the components of $\theta,\delta,\varepsilon,\phi$ as follows. Let Let U denote the set of points $\frac{1}{q}(x,y.z)$ of $\triangle_q$ such that
$x=\frac{q}{3}$ and $y=0$ or $z=0$, or
$x=4m-2i$, and $y=i$ or $z=i$ for some $i\in\{0,...,2m-1\}, i\neq m$, or
$x=4m-2i+1$, and $y=i$ or $z=i$ for some $i\in\{1,...,m\}$, or
$x=4m-2i+1$, and $y=i-1$ or $z=i-1$ for some $i\in\{m+1,...,2m\}$.

The circled vertices in Figure 8 are the vertices in U. Set

$$\begin{aligned}
&\theta_{\tilde{v}} = 1, \\
&\theta_u = 1/2 \text{ for all } u \in U, \\
&\delta_{t_1} = \varepsilon_{t_2} = \phi_{t_3} = 6m+1, \\
&\delta_{t_2} = \delta_{t_3} = \varepsilon_{t_1} = \varepsilon_{t_3} = \phi_{t_1} = \phi_{t_2} = -6m-1.
\end{aligned}$$

Set all the other components of $\theta, \delta, \varepsilon, \phi$ to zero. Note that $|U| = 2+2(2m-1)+2(2m) = 8m$. The resulting objective value is

$$\delta_{t_1} + \varepsilon_{t_2} + \phi_{t_3} + \sum_{u \in W} \theta_u = 3(6m+1) + 8m(1/2) + 1 = 22m+4.$$

Hence, to complete the proof, it is sufficient to to find α, β, γ such that all the constraints in $(DSLP)$ are satisfied.

If we ignore the equality constraints for the moment, the problem reduces to three separate feasible flow problems on H'_q. One has α as flow values and θ and δ as demands, another has β as flow values and θ and ε as demands, and the last has γ as flow values and θ and ϕ as demands. In each of these problems, we seek flows such that the net out-flow at every vertex is at least the demand at the vertex.

These flow problems can be simplified further, as follows. Consider α, for example. In view of the constraint $\alpha_{uv} + \alpha_{vu} = \frac{c_{uv}}{2}$, to specify these two values, it is enough to specify their difference $\hat{\alpha}_{uv} = \alpha_{uv} - \alpha_{vu}$, the "net flow" in direction (u, v). Moreover, we can choose an orientation (u, v) or (v, u) so that this difference is non-negative. Then the requirement on these netflows is, again, that the net flow out of each vertex be at least its demand, and that, if edge uv is oriented from u to v, then its net flow be non-negative and at most $\frac{c_{uv}}{2}$. We describe values for α and β, omitting those for γ, since it is symmetrical to β. It is straightforward to check that they have the required properties.

Values for α. This solution is indicated in Figure 9, where we show the orientations and net flow values $\hat{\alpha}$. The vertices in U have demand $\frac{1}{2}$ and are circled. Any other vertex having nonzero demand has the demand adjacent to the vertex. Note that a number of edges have $c_{uv} = 0$ and are omitted from the figure.

Recall that $\tilde{v}$ is the vertex on the midpoint of the line joining t_2 and t_3. Let $\tilde{u}$ denote the neighbour of $\tilde{v}$ on the line between $\tilde{v}$ and t_2 and $\tilde{w}$ denote the neighbour of $\tilde{v}$ on the line between $\tilde{v}$ and t_3. Set $\hat{\alpha}_{\tilde{u}\tilde{w}} = 0$. For each $i \in \{2, 3\}$, set $\hat{\alpha}_{\tilde{v}t_i} = c_{\tilde{v}t_i}/2$. For any edge (u, v) of H'_q that is parallel to one of the (oriented) line segments from t_1 to t_2 or t_3, set $\hat{\alpha}_{uv} = \frac{c_{uv}}{2}$.

It remains to consider the "horizontal" edges. First, suppose uv lies on the segment between $\tilde{u}$ and t_2 or on the segment between $\tilde{w}$ and t_3. Assuming that (u, v) points toward the terminal, set $\hat{\alpha}_{uv} = i$, where i is the graph distance between v and $\tilde{v}$ in G_q. Now suppose that uv lies on the horizontal line containing two vertices $w_1, w_2 \in U$. If uv is on the segment between w_1 and w_2, then set $\hat{\alpha}_{uv} = 0$. Otherwise, if (u, v) points away from this segment, set $\hat{\alpha}_{uv} = \frac{c_{uv}}{2}$. The only remaining possibility for a horizontal edge uv occurs when the distance from t_1 to u is less than $2m$, in which case $c_{uv} = 0$, so $\hat{\alpha}_{uv} = 0$. Note that the demand constraints for α are actually satisfied with equality at all vertices except $\tilde{v}$.

Values for β. The solution we are about to describe is indicated in Figure 10, which shows the net flows $\hat{\beta}$ and the demands, as in Figure 9. Set $\hat{\beta}_{\tilde{u}\tilde{w}} = \frac{c_{\tilde{u}\tilde{w}}}{2}$. Set $\hat{\beta}_{t_2\tilde{v}} = \hat{\beta}_{\tilde{v}t_3} = m+1$.

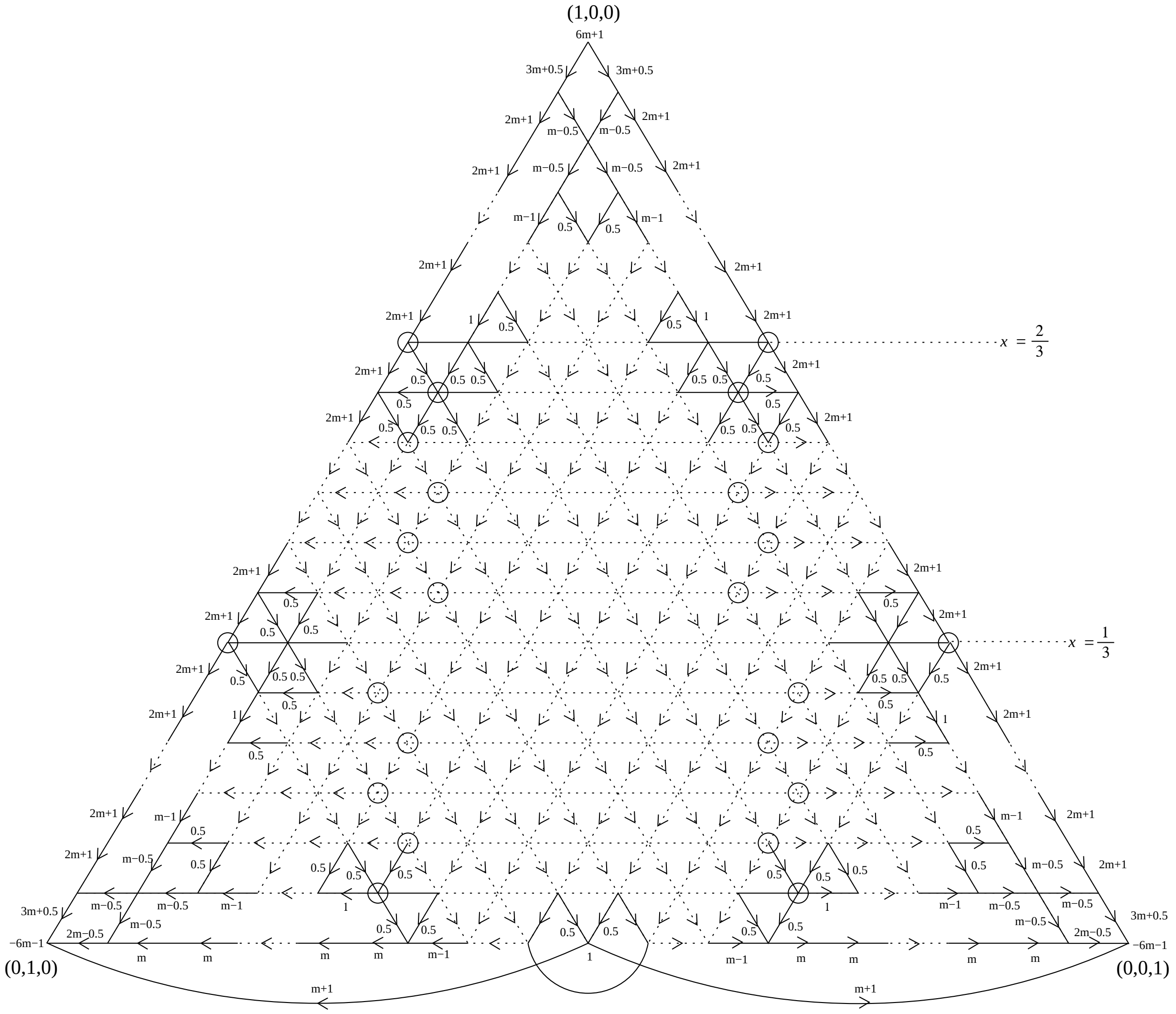

Figure 9: Net flow values for α

For each edge (u, v) that is parallel to the (oriented) line segment from t_2 to t_1 or t_3, set $\hat{\beta}_{uv} = \frac{c_{uv}}{2}$.

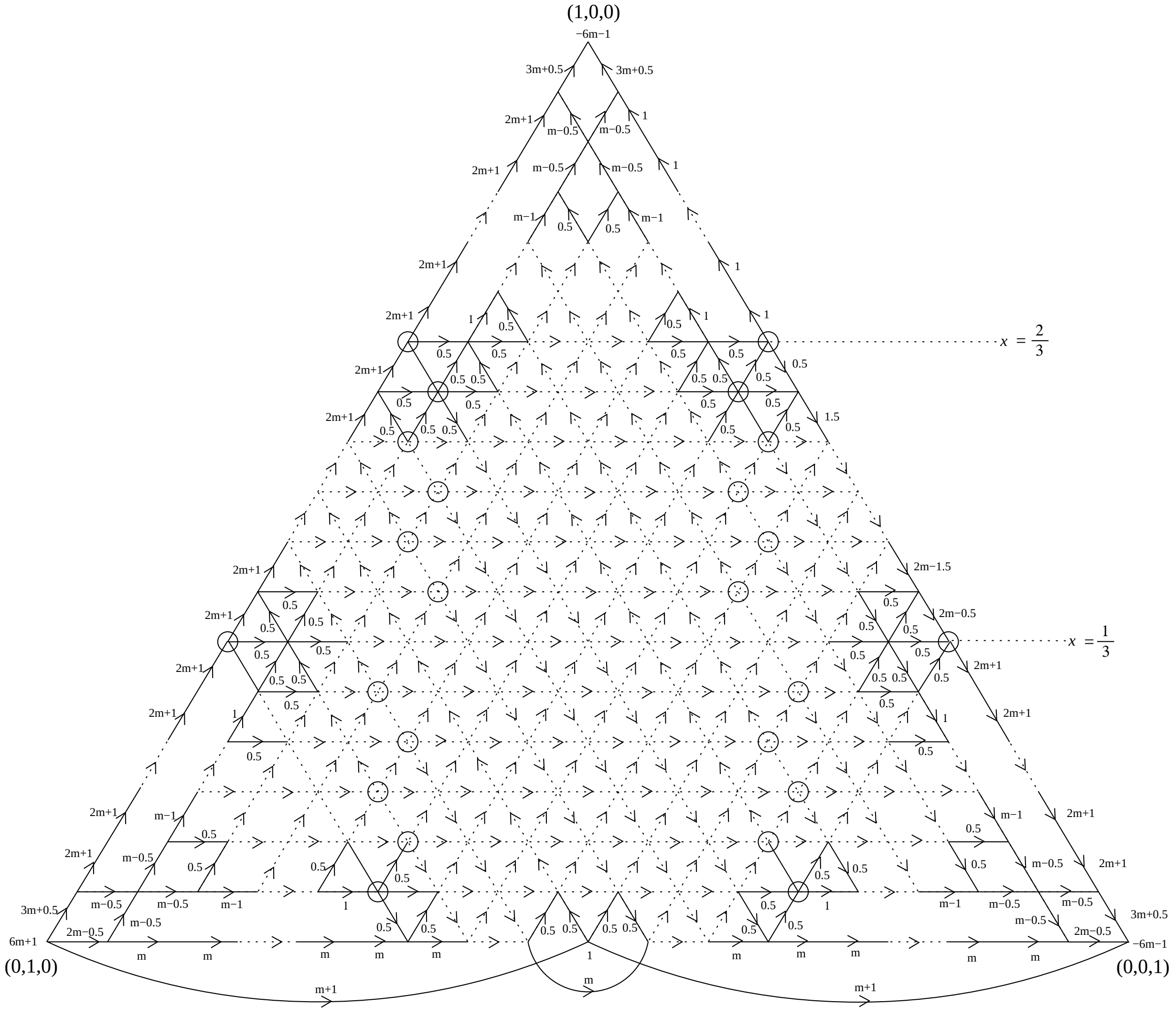

Figure 10: Net flow values for β

Consider an edge uv parallel to the line joining t_1 and t_3. First, suppose uv is on the line joining t_1 and t_3. If uv is incident with t_1 or t_3, then, assuming that v is a terminal, set $\hat{\beta}_{uv} = \frac{c_{uv}}{2} = 3m + \frac{1}{2}$. If uv is in the corner triangle containing t_1, then, assuming that (u, v) points toward t_1, set $\hat{\beta}_{uv} = 1$. If uv is in the middle hexagon, then, assuming that (u, v) points toward t_3, set $\hat{\beta}_{uv} = i + \frac{1}{2}$, where $i + 2m$ is the graph distance between u and t_1. If uv is in the corner triangle containing t_3, then, assuming that (u, v) points toward t_3, set $\hat{\beta}_{uv} = \frac{c_{uv}}{2} = 2m + 1$.

Now, suppose uv is not on the line joining t_1 and t_3. If the second coordinates of u and v equal $\frac{1}{2}$ and (u, v) points toward t_1, set $\hat{\beta}_{uv} = \frac{c_{uv}}{2}$. If $i \in \{1, ..., 3m - 1, 3m + 1, ..., 4m\}$, there are exactly two vertices $w_1, w_2 \in U$ having second coordinate equal to i/q. Suppose uv lies on the line through w_1, w_2. If uv lies on the segment between w_1 and w_2, set $\hat{\beta}_{uv} = 0$; otherwise, assuming that (u, v) points away from the segment between w_1 and w_2, set $\hat{\beta}_{uv} = \frac{c_{uv}}{2}$.

Note that, except at the neighbour of t_1 having zero second coordinate, the demand constraints are satisfied with equality. This concludes the proof of Theorem 14.

7 Concluding Remarks

All of the results of Călinescu *et al.* [1, 2] quoted above for $k = 3$ are special cases of their results for general k. They give a linear-programming relaxation that generalizes $(LP3)$, and a corresponding generalization of the notion of triangle-embedding, an embedding into a $(k-1)$-dimensional simplex in which the terminals are mapped to the extreme points. They show that the optimal value of a k-cut is at most $\frac{3k-2}{2k}$ times the optimal value of this linear-programming problem. As a result, they obtain an approximation algorithm for the optimal k-cut problem having performance guarantee $\frac{3k-2}{2k}$. The recent paper by Karger *et al.* [11], which has some of our results for $k = 3$, also has results for $k > 3$, improving the bounds given by [1, 2]. For example, [11] gives bounds of 1.1539 for $k = 4$, 1.2161 for $k = 5$, and $1.3438 - \epsilon_k$ for all $k > 6$ where $\epsilon_k > 0$ is evaluated computationally for any fixed k. Freund and Karloff [9] gave a lower bound of $\frac{8}{7+\frac{1}{k-1}}$ on the integrality ratio for general k. However, the problem of giving a tight analysis for $k > 3$, as we now have for $k = 3$, remains open. Why is $k = 3$ apparently easier to deal with than higher values of k? One important difference is this: For $k > 3$ the analogue of (D_q) need not have an optimal solution whose positive variables correspond to flat k-cuts. This can be demonstrated with an example with $k = 4$ and $q = 4$.

Acknowledgment. We are grateful to Gruia Călinescu, Joseph Cheriyan, and Levent Tunçel for conversations about this work, and to a referee for a careful reading of the paper.

References

[1] G. Călinescu, H. Karloff, and Y. Rabani: An improved approximation algorithm for MULTIWAY CUT, *Proc. ACM Symp. on the Theory of Computing*, pp. 48–52, 1998.

[2] G. Călinescu, H. Karloff, and Y. Rabani: An improved approximation algorithm for MULTIWAY CUT. *J. Comput. Syst. Sci.*, 60 (2000), 564–574.

[3] K.K.H. Cheung, W.H. Cunningham and L. Tang, Optimal 3-terminal cuts and linear programming, Technical Report CORR 2002-24, Dept. of Combinatorics and Optimization, University of Waterloo, Waterloo, Ontario, August 2002.

[4] S. Chopra and M.R. Rao, On the multiway cut polyhedron, *Networks* 21 (1991), 51–89.

[5] W.H. Cunningham, The optimal multiterminal cut problem, in: C. Monma and F. Hwang (eds.), *Reliability of Computer and Communications Networks*, American Math. Soc., pp. 105–120., 1991.

[6] W.H. Cunningham and L. Tang, Optimal 3-terminal cuts and linear programming, Integer programming and combinatorial optimization, *Lecture Notes in Comput. Sci.,* 1610, pp. 114–125, 1999.

[7] E. Dahlhaus, D. Johnson, C. Papadimitriou, P. Seymour, and M. Yannakakis, The Complexity of multiway cuts, extended abstract, 1983.

[8] E. Dahlhaus, D. Johnson, C. Papadimitriou, P. Seymour, and M. Yannakakis, The Complexity of multiterminal cuts, *SIAM J. Comput.* 23 (1994), 864–894.

[9] A. Freund, H. Karloff, A lower bound of $8/(7+\frac{1}{k-1})$ on the integrality ratio of the Călinescu-Karloff-Rabani relaxation for multiway cut, *Inf. Process. Lett.* 75 (2000), 43–50.

[10] D. Karger, P. Klein, C. Stein, M. Thorup, and N. Young, Rounding algorithms for a geometric embedding of minimum multiway cut, *Proc. ACM Symp. on the Theory of Computing*, pp. 668-678, 1999.

[11] D. Karger, P. Klein, C. Stein, M. Thorup, and N. Young, Rounding algorithms for a geometric embedding of minimum multiway cut, *Math. Oper. Res.* 29 (2004), 436–461.