

On the Integrality Gap of a Natural Formulation of the Single-Sink Buy-at-bulk Network Design Problem ^{*}

Naveen Garg¹, Rohit Khandekar¹, Goran Konjevod², R. Ravi³, F.S. Salman⁴,
and Amitabh Sinha³

¹ Department of Computer Science and Engineering,
Indian Institute of Technology, New Delhi, India.

{naveen, rohitk}@cse.iitd.ernet.in

² Department of Computer Science and Engineering,
Arizona State University, Tempe, AZ 85287, USA.

goran@asu.edu

³ GSIA, Carnegie Mellon University,
Pittsburgh PA 15213-3890, USA.

{ravi, asinha}@andrew.cmu.edu

⁴ Krannert School of Management,
Purdue University, West Lafayette, IN 47907, USA.

salmanf@mgmt.purdue.edu

Abstract. We study two versions of the single sink *buy-at-bulk* network design problem. We are given a network and a single sink, and several sources which demand a certain amount of flow to be routed to the sink. We are also given a finite set of cable types which have different cost characteristics and obey the principle of economies of scale. We wish to construct a minimum cost network to support the demands, using our given cable types. We study a natural integer program formulation of the problem, and show that its integrality gap is $O(k)$, where k is the number of cable types. As a consequence, we also provide an $O(k)$ -approximation algorithm.

1 Introduction

1.1 Motivation

We study two network design problems which often arise in practice. Consider a network consisting of a single server and several clients. Each client wishes to route a certain amount of flow to the server. The cost per unit flow along an edge is proportional to the edge length. However, we can reduce the cost per unit flow of routing by paying a certain fixed cost (again proportional to the length

^{*} This research was supported by a faculty development grant awarded to R. Ravi by the Carnegie Bosch Institute, Carnegie Mellon University, Pittsburgh PA 15213-3890.

of the edge). We call the problem of finding a minimum cost network supporting the required flow the *deep-discount* problem.

Alternatively, at each edge we might be able to pay for and install a certain capacity selected from a set of allowable discrete capacity units, and then route flow (up to the installed capacity) for free. The problem of finding a minimum cost network in this scenario is called the *buy-at-bulk* network design problem [14].

Both the above problems reflect economies of scale, in the cost of routing unit flow as the installation cost increases (the deep-discount problem), and in the cost per unit capacity as the capacity increases (the buy-at-bulk problem).

1.2 Our results

The two problems are in fact equivalent up to a small loss in the value of the solution. In this paper, we focus on the deep-discount problem. We study the structure of the optimum solution, and show that an optimal solution exists which is a tree. We provide a natural IP formulation of the problem, and show that it has an integrality gap of the order of the number of cables. We also provide a polynomial time approximation algorithm by rounding the LP relaxation.

1.3 Previous work

Mansour and Peleg [11] gave an $O(\log n)$ -approximation for the single sink buy-at-bulk problem with a single cable type (only one discrete unit of capacity allowable) for a graph on n nodes. They achieved this result by using a low-weight, low-stretch spanner construction [2].

Designing networks using facilities that provide economies of scale has attracted interest in recent years. Salman *et al* [14] gave an $O(\log D)$ approximation algorithm for the single sink buy-at-bulk problem in Euclidean metric spaces, where D is the total demand. Awerbuch and Azar [4] gave a randomized $O(\log^2 n)$ approximation algorithm for the buy-at-bulk problem with many cable types and many sources and sinks, where n is the number of nodes in the input graph. This improves to $O(\log n \log \log n)$ using the improved tree metric construction of Bartal [5]. For the single sink case with many cable types, an $O(\log n)$ approximation was obtained by Meyerson, Munagala and Plotkin [12] based on their work on the Cost-Distance two metric network design problem. Salman *et al* also gave a constant approximation in [14] for the single cable type case using a LAST construction [10] in place of the spanner construction used in [11]. The approximation ratio was further improved by Hassin, Ravi and Salman [9].

Andrews and Zhang [3] studied a special case of the single-sink buy-at-bulk problem which they call the *access network design* problem and gave an $O(k^2)$ approximation, where k is the number of cable types. As in the deep-discount problem, they use a cost structure where each cable type has a buying and a routing cost, but they assume that if a cable type is used, the routing cost is at least a constant times the buying cost.

An improved approximation to the problem we study was obtained simultaneously but independently by Guha, Meyerson and Munagala [7], who designed a constant-factor approximation algorithm. Their algorithm is combinatorial and is based on their prior work on the access network design problem [8], as opposed to our focus on the LP relaxation and its integrality gap.

1.4 Outline of the paper

In the next section, we define the deep-discount problem formally and show its relation to the k -cable buy-at-bulk problem. In Sections 3 through 6, we introduce and study our integer program formulation and show that it has low integrality gap. We conclude with time complexity issues and open questions in Section 7.

2 Problem definition and inter-reductions

2.1 The deep-discount problem

Let $G = (V, E)$ be a graph with edge-lengths $l : E \rightarrow \mathbb{R}^+$. Let $d(u, v)$ denote the length of the shortest path between vertices u and v . We are given sources $\{v_1, \dots, v_m\} = \mathcal{S}$ which want to transport $\{\mathbf{dem}_1, \dots, \mathbf{dem}_m\}$ units of flow respectively to a common sink $t \in V$. We also have a set of k discount types $\{\kappa_0, \kappa_1, \dots, \kappa_{k-1}\}$ available for us to purchase and install. Each cable κ_i has an associated fixed cost p_i and a variable cost r_i . If we install cable κ_i at edge e and route f_e flow through it, the contribution to our cost is $l_e(p_i + f_e r_i)$. We may therefore view the installation of cable κ_i at an edge as paying a fixed cost $p_i l_e$ in order to obtain a discounted rate r_i of routing along this edge. The problem of finding an assignment of discount types to the edges and routing all the source demands to the sink at minimum total cost is the deep discount problem with k discount types (DD for short).

Let us order the rates as $r_0 > r_1 > \dots > r_{k-1}$. The rate $r_0 = 1$ and the price $p_0 = 0$ correspond to not using any discount. (It is easy to see that we may scale our cost functions so that this is true in general.) Observe that if $p_i \geq p_{i+1}$ for some i then κ_i will never be used. Therefore, without loss of generality, we can assume $p_0 < p_1 < \dots < p_{k-1}$.

2.2 The buy-at-bulk problem with k -cable types

In an edge-weighted undirected graph, suppose we are given a set of sources $\{v_1, \dots, v_m\}$ which want to transport $\{\mathbf{dem}_1, \dots, \mathbf{dem}_m\}$ units of flow respectively to a common sink t . We have available to us k different cable types, each having capacity u_i and cost c_i . We wish to buy cables such that we have enough capacity to support the simultaneous flow requirements. We are allowed to buy multiple copies of a cable type. There is no flow cost; our only cost incurred is the purchase price of cables. The problem of finding a minimum cost feasible network is the buy-at-bulk problem with k cable types (BB for short). It is **NP**-Hard even when $k = 1$ [14].

2.3 Approximate equivalence of BB and DD

Suppose we are given a BB instance $BB = (G, c, u)$ on a graph G with k cable types having costs and capacities $(1, 1), (c_1, u_1), \dots, (c_{k-1}, u_{k-1})$. We transform it into an instance of DD by setting edge costs (fixed and per-unit) $(0, 1), (c_1, \frac{c_1}{u_1}), \dots, (c_{k-1}, \frac{c_{k-1}}{u_{k-1}})$, and call this $DD(BB)$.

Conversely, given a DD instance $DD = (G, p, r)$ on a graph G with k discount types having prices and variable costs $(0, 1), (p_1, r_1), \dots, (p_{k-1}, r_{k-1})$, we transform it into a BB instance $BB(DD)$ with cable types having costs and capacities $(1, 1), (p_1, \frac{p_1}{r_1}), \dots, (p_{k-1}, \frac{p_{k-1}}{r_{k-1}})$.

It is easy to see that $BB(DD(BB)) = BB$ and $DD(BB(DD)) = DD$; i.e., the two transformations are inverses of each other. For a problem instance X , we abuse notation to let X also denote the cost of a feasible solution to it. Let X^* denote the cost of an optimal (integer) solution to X .

Lemma 1. $BB \leq DD^*(BB)$

Proof. Consider an edge e and let the flow on e in $DD^*(BB)$ be x_e . If the solution uses discount type 0 on e , then the BB solution does not pay any more than the routing cost already paid. If it uses a discount type $i > 0$, we install $\lceil \frac{x_e}{u_i} \rceil$ copies of cable type i at this edge. Clearly this gives us a feasible solution to BB . For this edge $DD^*(BB)$ has routing cost $\frac{l_e x_e c_i}{u_i}$ and building cost $l_e c_i$, hence a total cost of $l_e c_i (1 + \frac{x_e}{u_i})$. The BB solution has cable cost $l_e c_i \lceil \frac{x_e}{u_i} \rceil$ on edge e , which is no more than the total cost incurred by edge e in $DD^*(BB)$.

Lemma 2. $DD \leq 2BB^*(DD)$

Proof. We will initially allow for multiple discount types on each edge and pay for it all. A pruned solution will satisfy all the desired properties and cost only less. Hence let x_e^i be the number of copies of cable type i used at edge e . We only consider edges with non-zero flow, that is, where $x_e^i > 0$. Note that the flow on e is at most $x_e^i p_i / r_i$. We purchase a discount type i cable on edge e and route the flow through it on this discounted cable. We pay no more than $p_i l_e + \frac{x_e^i p_i}{r_i} r_i l_e \leq (x_e^i + 1) p_i l_e$, which is no more than two times the cost $x_e^i p_i l_e$ already paid by $BB(DD)$, since $x_e^i \geq 1$.

Together, the above two Lemmas imply that $BB^*(DD) \leq BB(DD) \leq DD^* \leq DD \leq 2BB^*(DD)$, so that a ρ approximation algorithm for BB gives a 2ρ approximation algorithm for DD. Similarly, a ρ approximation to DD is a 2ρ approximation to BB.

Given the above relations, we focus on the deep-discount formulation in this paper. One reason for choosing to work with this version is presented in Section 3 where we show there are always acyclic optimal solutions for the deep-discount problem, while this is not always the case for the buy-at-bulk problem (see, e.g., [14]). However, we continue to use the term ‘‘cable type’’ to refer to the discount type used in an edge from time to time, even though a solution to the deep-discount problem involves choices of discount types on edges rather than installation of cables.

3 Structure of an optimum solution to the deep-discount problem

Let us look at how an optimal solution allocates the discount types to edges and routes the flow to the sink. Clearly an edge will use only one type of discount. Suppose in an optimum, an edge e uses discount- i . Define a new length function $l'_e := r_i l_e$. Clearly once the fixed cost for the discount is paid, the routing cost is minimized if we route along a shortest path according to the length function l' . Therefore, there is an optimum which routes along shortest paths according to such a length function l' . As a result, we can also assume that the flow never splits. That is, if two commodities share an edge then they share all the subsequent edges on their paths to t . This is because we can choose a shortest path tree in the support graph; flow along this tree to the root will never split.

The cost of routing f units of flow on an edge e using discount- i is $l_e(p_i + r_i f)$. So the discount type corresponding to minimum cost depends only on f and is given by $\text{type}(f) := \min \arg_i \{p_i + r_i f \mid 0 \leq i < k\}$.

Lemma 3. *The function $\text{type}(f)$ defined above is non-decreasing in f .*

Proof. Consider any two discount types, say i and j with $i < j$. We know that $r_i > r_j$ and $p_i < p_j$. As the cost functions $p + r f$ are linear, there is a critical value f of flow such that, $p_i + r_i f = p_j + r_j f$. If the flow is smaller than f , discount- i is better than discount- j . And if the flow is more than f , discount- j is better than discount- i . This proves the lemma.

Suppose the optimal flow is along a path P from v_j to t . As the flow never splits, the flow along P is non-decreasing as we go from v_j to t . Hence from the above lemma, the discount type never decreases as we go from v_j to t . We call this property *path monotonicity*.

Summarizing, we have the following.

Theorem 1. *There exists an optimum solution to the deep-discount problem which satisfies the following properties.*

- *The support graph of the solution is a tree.*
- *The discount types are non-decreasing along the path from a source to the root.*

Similar results were proved independently (but differently) in [3] and [7]. Figure 1 illustrates the structure of an optimum assuming 3 discount types.

4 Linear program formulation and rounding

4.1 Overview of the algorithm

First we formulate the deep-discount problem as an integer program. We then take the linear relaxation of the IP, and solve it to optimality. Clearly an optimal solution to the LP is a lower bound on the optimal solution to the IP.

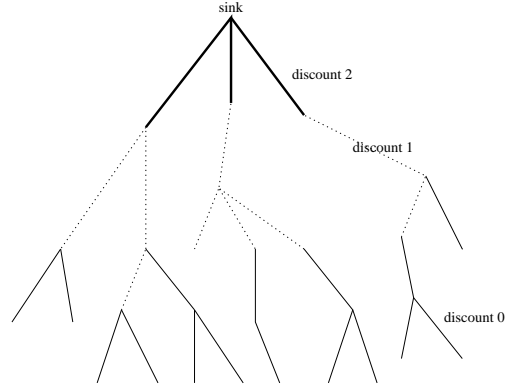


Fig. 1. Structure of an optimum solution to the deep-discount problem with three discount types.

We now use the LP solution to construct our solution. We have already seen that there is an integer optimal solution which is a layered tree. We construct such a tree in a *top-down* manner, starting from the sink. We iteratively augment the tree by adding cables of the next available lower discount type. At each stage we use an argument based on the values of the decision variables in an optimal LP solution to charge the cost of our solution to the LP cost. We thus bound the cost of building our tree. We next bound the routing costs by an argument which essentially relies on the fact that the tree is layered and that distances obey the triangle inequality.

4.2 Integer program formulation

We now present a natural IP formulation of the deep-discount problem. As is usual for flow problems, we replace each undirected edge by a pair of anti-parallel directed arcs, each having the same length as the original (undirected) edge. We introduce a variable z_e^i for each $e \in E$ and for each $0 \leq i < k$, such that, $z_e^i = 1$ if we are using discount- i on edge e and 0 otherwise. The variable $f_{e;i}^j$ is the flow of commodity j on edge e using discount- i . For a vertex set S (or a singleton vertex v), we define $\delta^+(S)$ to be the set of arcs leaving S . That is, $\delta^+(S) = \{(u, v) \in E : u \in S, v \notin S\}$. Analogously, $\delta^-(S) = \{(u, v) \in E : u \notin S, v \in S\}$. The formulation is given in Figure 2.

The first term in the objective function is the cost of purchasing the various discount types at each edge; we call this the *building* cost. The second term is the total cost (over all vertices v_j) of sending \mathbf{dem}_j amount of flow from vertex v_j to the sink; we call this the *routing* cost of the solution. These two components of the cost of an optimal solution are referred to as OPT_{build} and OPT_{route} respectively.

$$\min \sum_{e \in E} \sum_{i=0}^{k-1} p_i z_e^i l_e + \sum_{v_j \in \mathcal{S}} \sum_{e \in E} \sum_{i=0}^{k-1} \text{dem}_j^i f_{e,i}^j r_i l_e$$

subject to:

$$\begin{aligned} (i) \quad & \sum_{e \in \delta^+(v_j)} \sum_{i=0}^{k-1} f_{e,i}^j \geq 1 && \forall v_j \in \mathcal{S} \\ (ii) \quad & \sum_{e \in \delta^-(v)} \sum_{i=0}^{k-1} f_{e,i}^j = \sum_{e \in \delta^+(v)} \sum_{i=0}^{k-1} f_{e,i}^j && \forall v \in V \setminus \{v_j, t\}, 1 \leq j \leq m \\ (iii) \quad & \sum_{e \in \delta^-(v)} \sum_{i=q}^{k-1} f_{e,i}^j \leq \sum_{e \in \delta^+(v)} \sum_{i=q}^{k-1} f_{e,i}^j && 0 \leq q < k, \forall v \in V \setminus \{v_j, t\}, \\ & && 1 \leq j \leq m \\ (iv) \quad & f_{e,i}^j \leq z_e^i && \forall e \in E, 0 \leq i < k \\ (v) \quad & \sum_{i=0}^{k-1} z_e^i \geq 1 && \forall e \in E \\ (vi) \quad & z, f && \text{non-negative integers} \end{aligned}$$

Fig. 2. Integer program formulation of the deep-discount problem.

The first set of constraints ensures that every source has an outflow of one unit which is routed to the sink. The second is the standard flow conservation constraints, treating each commodity separately. The third set of constraints enforces the path monotonicity discussed in the preceding Section, and is therefore valid for the formulation. The fourth simply builds enough capacity, and the fifth ensures that we install at least one cable type on each arc. Note that this is valid and does not add to our cost since we have the default cable available for installation at zero fixed cost.

Relaxing the integrality constraints (vi) to allow the variables to take real non-negative values, we obtain the LP relaxation. This LP has a polynomial number of variables and constraints, and can be therefore solved in polynomial time. The LP relaxation gives us a lower bound which we use in our approximation algorithm.

5 The rounding algorithm

5.1 Pruning the set of available cables

We begin by pruning our set of available cables, and we show that this does not increase the cost by more than a constant factor. This pruning is useful in the analysis.

The following lemma shows that the cost of solution does not increase by a large factor if we restrict ourselves to rates that are sufficiently different, that is, they decrease by a constant factor.

Let OPT be the optimum value with rates r_0, r_1, \dots, r_{k-1} and corresponding prices p_0, p_1, \dots, p_{k-1} . Let $\epsilon \in (0, 1)$ be a real number. Assume that $\epsilon^{l-1} \geq r_{k-1} > \epsilon^l$. Now, let us create a new instance as follows. Let the new rates be $1, \epsilon, \dots, \epsilon^{l-1}$. For each i , let the price corresponding to ϵ^i be p_j , where r_j is the largest rate not bigger than ϵ^i . Let OPT' be the optimum value of this new problem.

Lemma 4. $OPT'_{route} \leq \frac{1}{\epsilon} OPT_{route}$

Proof. Consider an edge e which uses discount- j in OPT . In the solution of the new problem, change its discount type to ϵ^i such that $\epsilon^i \geq r_j > \epsilon^{i+1}$. Thus, for this edge, the price does not increase and the routing cost increases by a factor at most $1/\epsilon$.

Since $OPT'_{build} \leq OPT_{build}$, we have as a consequence that $OPT' \leq \frac{1}{\epsilon} OPT$. Hereafter we assume that the rates r_0, r_1, \dots, r_{k-1} decrease by a factor at least ϵ for some $0 < \epsilon < 1$, thereby incurring an increase in cost by a factor of at most $1/\epsilon$.

5.2 Building the solution: Overview

Recall that G is our input graph, and k is the number of cable types. We also have a set of parameters $\{\alpha, \beta, \gamma, \delta\}$, all of which are fixed constants and whose effect will be studied in the analysis in Section 6.

We build our tree in a top-down manner. We begin by defining T_k to be the singleton vertex $\{t\}$, the sink. We then successively augment this tree by adding cables of discount type i to obtain T_i , for i going down $k-1, k-2, \dots, 1, 0$. Our final tree T_0 is the solution we output. Routing is then trivial – simply route along the unique path from each source to the sink in the tree.

Our basic strategy for constructing the tree T_i from T_{i-1} is to first identify a subset of demand sources that are not yet included in T_{i-1} by using information from their contributions to the routing cost portion of the LP relaxation. In particular, we order these candidate nodes in non-decreasing order of the radius of a ball that is defined based on the routing cost contribution of the center of the ball. We then choose a maximal set of non-overlapping balls going forward in this order. This intuitively ensures that any ball that was not chosen can be charged for their routing via the smaller radius ball that overlapped with it that is included in the current level of the tree. After choosing such a subset of as yet unconnected nodes, we build an approximately minimum building cost Steiner tree with these nodes as terminals and the (contracted) tree T_{i-1} as the root. The balls used to identify this subset now also serve a second purpose of relating the building cost of the Steiner tree to the fractional optimum. Finally, in a third step, we convert the approximate Steiner tree rooted at the contracted T_{i-1} to

a LAST (light approximate shortest-path tree [10]) which intuitively ensures that all nodes in the tree are within a constant factor of their distance from the root T_{i-1} in this LAST without increasing the total length (and hence the building cost) of the tree by more than a constant factor. This step is essential to guarantee that the routing cost via this level of the tree does not involve long paths and thus can be charged to within a constant factor of the appropriate LP routing cost bound.

5.3 Building the solution: Details

The details of the algorithm are presented in Figure 3. Let C_j^i denote the fraction of the routing cost for routing unit flow from v_j to t corresponding to discount- i , that is, $C_j^i = \sum_e f_{e,i}^j r_i l_e$. Hence $\sum_{0 \leq i < k} C_j^i$ is the total routing cost for vertex v_j . For a vertex v and a positive number R , let $B(v, R) = \{u \in V : d(u, v) \leq R\}$ denote the ball of radius R centered at vertex v .

Selecting vertices for inclusion in the current level. The bulk of the work is done in Step 4. We first choose a certain set of vertices (S_i at level i), and then build a Steiner tree connecting the chosen vertices to the root component (T_{i+1}). We note that this step is somewhat similar to the “tour ball” construction in [13].

Building the Steiner tree. We build balls $B(v_j, \delta R_j^i)$ around each selected vertex v_j . We note that we will choose $\delta < \gamma$, where γ is the dilation parameter for the radius of the balls used in the vertex selection step. We then build an approximately minimum Steiner tree which connects these selected balls to T_{i+1} . More formally, we contract each ball and introduce a new node for it. We also contract T_{i+1} and introduce a node for it. Then we run an approximation algorithm to find a Steiner tree connecting all the selected nodes that has cost at most twice the value of a fractional Steiner tree, i.e., within twice the cost of an LP relaxation for the Steiner tree problem on these nodes (See, e.g., [1]). Then we un-contract the balls and extend the edges of the resulting forest incident on the boundary of $B(v_j, \delta R_j^i)$ with direct edges to the center v_j . Thus we have a tree connecting all the selected vertices v_j to T_{i+1} .

Converting the Steiner tree to a LAST. The Steiner tree constructed so far may have a very large diameter, since we have not taken the routing into consideration so far. Hence it may lead to very high routing costs in the solution. To get around this, we use a construction due to Khuller, Raghavachari and Young [10] which achieves short paths from a root node while being light.

Definition 1 (Light approximate shortest-path tree). Let $G = (V, E)$ be a graph with a length function $l : E \rightarrow \mathbb{R}^+$ and let $t \in V$ be a root vertex. Let $\alpha, \beta > 1$ be real numbers. An (α, β) -LAST rooted at t is a tree T in G such that

Algorithm Deep-discount($G, K, \alpha, \beta, \gamma, \delta$)

G : input graph
 K : set of cables
 $\alpha, \beta, \gamma, \delta$: parameters (fixed constants)

1. Prune the set of available cables as described in 5.1.
2. Solve the LP relaxation of the IP described in 4.2.
3. $T_k = \{t\}$.
4. For $i = k - 1, k - 2, \dots, 1$:
 - Define $S_i := \emptyset$
 - $\forall v_j \notin T_{i+1}$:
 - $R_j^i = \frac{C_j^0 + \dots + C_j^{i-1}}{r_{i-1}}$.
 - If $T_{i+1} \cap B(v_j, \gamma R_j^i) \neq \emptyset$,
 - proxy** $_i(v_j) :=$ any (arbitrary) vertex in $T_{i+1} \cap B(v_j, \gamma R_j^i)$
 - $S_i := S_i \cup \{v_j\}$.
 - Order the remaining vertices $L_i = V \setminus (T_{i+1} \cup S_i)$ in nondecreasing order of their corresponding ball radii.
 - While $L_i \neq \emptyset$:
 - Let $B(v_j, \gamma R_j^i)$ be the smallest radius ball in L .
 - $\forall u \in L \cap B(v_j, \gamma R_j^i)$:
 - proxy** $_i(u) := v_j$
 - $L := L \setminus \{u\}$
 - $L := L \setminus \{v_j\}$
 - $S_i := S_i \cup \{v_j\}$
 - Comment:** S_i is the set of sources chosen to be connected at this level.
 - Contract T_{i+1} to a singleton node t_{i+1} .
 - Build a Steiner tree ST_i with $S_i \cup \{t_{i+1}\}$ as terminals (Elaborated in the text below – the parameter δ is used here).
 - Use discount type i for these edges.
 - Convert ST_i into an (α, β) -LAST rooted at t_{i+1} , denoted $LAST_i$.
 - Define $T_i := T_{i+1} \cup LAST_i$.
5. For every source vertex $v_j \notin T_1$:
 - Compute a shortest path P from v_j to any node in T_1 .
 - Augment T_1 by including the edges in P .
 - Use cable type 0 on the edges in P .
 - $T_0 := T_1$.
6. Route along shortest paths in T_0 . This is the solution we output.

Fig. 3. The algorithm

the total length of T is at most α times the length of an MST of G , and for any vertex $v \in V$, the length of the (v, t) path along T is at most β times the length of a shortest (v, t) path in G .

The Steiner tree constructed can now be transformed into an (α, β) -LAST rooted at t_{i+1} (the contracted version of T_{i+1}) for some constants $\alpha, \beta > 1$ using the algorithm of [10]. The edges in the LAST will use discount- i . We then uncontract the root component. This breaks up the LAST into a forest where each subtree is rooted at some vertex in the un-contracted tree T_{i+1} . Define T_i to be the union of T_{i+1} and this forest.

In the last stage, we connect each source v_j not in T_1 to T_1 by a shortest path, using discount-0, thereby extending T_1 to include all remaining source vertices in T_0 .

6 Analysis

We use the LP optimum OPT as a lower bound on the integer optimum. Let $OPT_{build} = \sum_e \sum_i p_i z_e^i l_e$ denote the total price paid for purchasing cables of all discount types in an LP optimum. Similarly, let $OPT_{route} = \sum_{v_j} \sum_e \sum_i \text{dem}_j f_{e;i}^j r_i l_e$ be the total routing cost paid in that optimum. Thus $OPT = OPT_{build} + OPT_{route}$ is a lower bound on the total cost. In this section, we prove that, for our algorithm, the total building cost is $O(k \cdot OPT_{build})$ and the total routing cost is $O(k \cdot OPT_{route})$. Thus we establish that the integrality gap of the formulation is no more than $O(k)$.

6.1 Building cost

We analyze the total price paid for installing discount- i cables when we augment the tree T_{i+1} to T_i .

Note that in building an (α, β) -LAST from the tree, we incur a factor of at most α in the building cost. We argue that the cost of building the tree at the current stage is $O(OPT_{build})$. Then, summing over all k stages, we get that the total building cost is $O(k \cdot OPT_{build})$.

For any source vertex v , the following Lemma proves that there is sufficient fractional z -value crossing a ball around v to allow us to pay for an edge crossing the ball. Since the LP optimum pays for this z , we can charge the cost of our edge to this fractional z and hence obtain our approximation guarantee.

Lemma 5. *Let $S \subset V$ be a set of vertices such that $t \notin S$ and $B(v_j, \delta R_j^i) \subset S$. Then,*

$$\sum_{q=i}^{k-1} \sum_{e \in \delta^+(S)} z_e^q \geq 1 - \frac{1}{\delta}.$$

Proof. Assume for the sake of contradiction that the sum is less than $1 - 1/\delta$. So the total flow starting from the source v_j which crosses S using discount types smaller than i is more than $1/\delta$. As it pays at least r_{i-1} per unit distance per unit flow, the total routing cost is more than

$$\frac{\delta R_j^i r_{i-1}}{\delta} = \frac{C_j^0 + \dots + C_j^{i-1}}{r_{i-1}} r_{i-1} = C_j^0 + \dots + C_j^{i-1}$$

This is a contradiction, as the total cost spent in discount types smaller than i is exactly $C_j^0 + \dots + C_j^{i-1}$.

We built a LAST which used discount- i . So the building cost of the LAST is p_i times the length of the LAST. The following Lemma gives a bound on this cost.

Lemma 6. *The cost of the LAST built at any stage is $O(OPT_{build})$.*

Proof. If we scale up the z -values in the optimum by a factor $\delta/(\delta - 1)$, Lemma 5 indicates that we have sufficient z -value of types i or higher to build a Steiner tree connecting the balls $B(v_j, \delta R_j^i)$ to T_{i+1} . If we use the primal dual method [1], we incur an additional factor of 2 in the cost of the Steiner tree as against the LP solution z -values. Thus, its cost will be at most

$$2 \frac{\delta}{\delta - 1} p_i \sum_{q=i}^{k-1} \sum_e z_e^q \leq 2 \frac{\delta}{\delta - 1} OPT_{build}.$$

After un-contracting the balls, we extended the forest to centers v_l by direct edges between the boundaries of forest edges in $B(v_l, \delta R_l^i)$ and v_l . We can account for this extension by using the following observation. For a center v_l , the cost of extension is at most $\frac{\delta}{\gamma - \delta}$ times the cost of the forest inside $B(v_l, \gamma R_l^i)$. Furthermore, during the selection of the vertices, we ensured that for any two selected vertices v_l and v_j , the balls $B(v_l, \gamma R_l^i)$ and $B(v_j, \gamma R_j^i)$ are disjoint. Thus the total cost of the extended tree is at most $1 + \frac{\delta}{\gamma - \delta}$ times the cost of the previous forest. Hence cost of the Steiner tree built is at most $2 \frac{\gamma}{\gamma - \delta} \frac{\delta}{\delta - 1} OPT_{build}$. Subsequently, the cost of the LAST built from this tree is at most $2\alpha \frac{\gamma}{\gamma - \delta} \frac{\delta}{\delta - 1} OPT_{build}$ [10]. For fixed constants α, δ, γ with $\gamma > \delta$, this is $O(OPT_{build})$ and completes the proof.

The total building cost is the sum of building costs at each stage, and we have k such stages. Thus, we have the following.

Lemma 7. *The total building cost is $O(k \cdot OPT_{build})$.*

6.2 Routing cost

After constructing the tree, for each source vertex v_j , we route the corresponding commodity along the unique (v_j, t) path on the tree. Let $OPT_j = \sum_i C_j^i$ denote

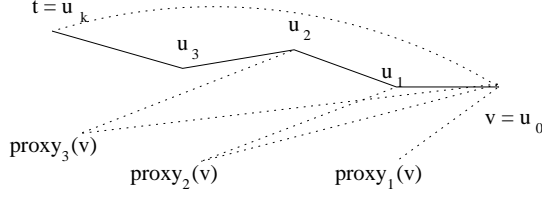


Fig. 4. Analysis for routing cost.

the routing cost per unit flow for v_j in the optimum. We prove that the routing cost for a source v_j is $O(k)$ times OPT_j . Thus the total routing cost is $O(k \cdot OPT_{route})$.

Refer to Figure 4 for the following analysis.

Lemma 8. *For any source vertex v_j , the cost of routing unit amount of its corresponding commodity is $O(k \cdot OPT_j)$.*

Proof. Let the (v_j, t) path along T_0 be $v_j = u_0, u_1, \dots, u_k = t$ such that the sub-path (u_i, u_{i+1}) uses discount- i for $0 \leq i < k$. Note that if discount- i is not used, then $u_i = u_{i+1}$. Let $d_T(u_i, u_{i+1})$ be the distance between u_i and u_{i+1} in the tree T_0 . Then, for v_j , the routing cost per unit flow is $\sum_i r_i d_T(u_i, u_{i+1})$.

For $1 \leq i < k$, let $\text{proxy}_i(v_j)$ denote the proxy of v_j in stage $k - i$. Moreover, for all j , define $\text{proxy}_k(v_j) = t$. We have $d(v_j, \text{proxy}_{i+1}(v_j)) \leq 2\gamma \frac{C_j^0 + \dots + C_j^i}{r_i} \leq 2\gamma \frac{OPT_i}{r_i}$. We also know that $r_i d_T(u_i, u_{i+1}) \leq \beta \cdot r_i d(u_i, \text{proxy}_{i+1}(v_j))$ because when we constructed the LAST in stage $k - i$, $d(u_i, u_{i+1})$ was at most β times the shortest path connecting u_i to T_{i+1} . Also this shortest path is shorter than $d(u_i, \text{proxy}_{i+1}(v_j))$, as $\text{proxy}_{i+1}(v_j)$ was in T_{i+1} .

By induction on i we prove that, $r_i d_T(u_i, u_{i+1}) \leq M \cdot OPT_j$ for some constant M .

For the base case when $i = 0$, v_j was connected by a shortest path to T_1 . Hence $r_0 d_T(u_0, u_1) \leq r_0 d(v_j, \text{proxy}_1(v_j)) \leq r_0 \cdot 2\gamma \frac{C_j^0}{r_0} \leq 2\gamma OPT_j \leq M \cdot OPT_j$ for sufficiently large M .

Now assume $r_l d_T(u_l, u_{l+1}) \leq M \cdot OPT_j$ for all $l < i$. Using triangle inequality and the induction hypothesis, we get

$$\begin{aligned}
r_i \cdot d_T(u_i, u_{i+1}) &\leq \beta \cdot r_i \cdot d(u_i, \text{proxy}_{i+1}(v_j)) \\
&\leq \beta \cdot r_i \sum_{q=0}^{i-1} d(u_q, u_{q+1}) + \beta \cdot r_i \cdot d(u_0, \text{proxy}_{i+1}(v_j)) \\
&= \beta \sum_{q=0}^{i-1} \frac{r_i}{r_q} \cdot r_q \cdot d(u_q, u_{q+1}) + \beta \cdot r_i \cdot d(v_j, \text{proxy}_{i+1}(v_j))
\end{aligned}$$

$$\begin{aligned}
&\leq \beta \sum_{q=0}^{i-1} \epsilon^{i-q} \cdot M \cdot OPT_j + \beta \cdot 2\gamma OPT_j \\
&\leq \left(\frac{\beta\epsilon}{1-\epsilon}\right) M + 2\beta\gamma OPT_j \\
&\leq M \cdot OPT_j
\end{aligned}$$

for $M \geq \frac{2\beta\gamma(1-\epsilon)}{1-\epsilon(1+\beta)}$. This completes the induction. Summing over all edges in the path from v_j to t , we get the statement of the lemma.

Summing the routing cost bound over all source vertices v_j , we obtain that the total routing cost is no more than $O(k \cdot OPT_{route})$.

7 Conclusion

The exact approximation factor of our algorithm depends on the parameters. If we set $(\alpha, \beta, \gamma, \delta, \epsilon)$ to be $(7, \frac{4}{3}, 3, 2, \frac{1}{5})$ respectively, we obtain an approximation factor of $60k$ for both components of the cost function. The running time of our algorithm is dominated by the time to solve an LP with $O(mnk)$ constraints and variables.

7.1 Recent work

The work of Guha *et al* [7] is combinatorial, and they build their tree in a bottom up manner. Their approach is to gather demand from nodes by means of Steiner trees until it is more profitable to use the next higher type of cable available. They then connect such trees using shortest path trees that gather sufficient demand to use the next cable type. They iteratively do this until all nodes are connected to the sink. Their algorithm being purely combinatorial has a much better running time. However, their approximation ratio is a large constant, roughly 2000. We can contrast this with our approximation factor, which is $60k$ with k being the number of cables after pruning.

After learning about their work, we have been able to tighten the ratio of the building cost component of our solution to the analogous component in the LP relaxation (OPT_{build}) to a constant. We show how to do this in the extended version of our paper [6]. Essentially, we prune the set of available cables so as to get a sufficient (geometric) increase in the fixed cost of higher index cables. Subsequently, if our LP has purchased a certain amount of a certain cable, we allow ourselves to purchase the same amount of all cables of lower index. Given the geometric costs, this only results in a constant factor dilation of the LP lower bound. We show that this solution is near-optimal, and we compare ourselves against it. Our algorithm can then charge the building cost of cable type i to what the augmented LP paid for cable type i only, instead of the entire building cost of the LP. This enables us to prove that the integrality gap of the building cost component is low. However, we do not see yet how to improve the routing cost component of our solution.

7.2 Open questions

The main open question from our work is the exact integrality gap of this problem, whether it is a constant, $O(k)$, or something in between. The question of getting even better approximation ratios for this problem remains open. The problem can be generalized to allow different source-sink pairs; for this problem the current state of the art is a polylogarithmic approximation [4].

References

1. Agrawal, A., Klein, P., Ravi, R.: When trees collide: An approximation algorithm for the generalized Steiner problem on networks. *SIAM Journal of Computing*, 24(3):440-456, 1995.
2. Althöfer, I., Das, G., Dobkin, D., Joseph, D., Soares, J.: On sparse spanners of weighted graphs. *Discrete and Computational Geometry*, 9:81-100, 1993.
3. Andrews, M., Zhang, L.: The access network design problem. Proc. of the 39th Ann. IEEE Symp. on Foundations of Computer Science, 42-49, October 1998.
4. Awerbuch, B., Azar, Y.: Buy at bulk network design. Proc. 38th Ann. IEEE Symposium on Foundations of Computer Science, 542-547, 1997.
5. Bartal, Y.: On approximating arbitrary metrics by tree metrics. Proc. 30th Ann. ACM Symposium on Theory of Computing, 1998.
6. Garg, N., Khandekar, R., Konjevod, G., Ravi, R., Salman, F.S., Sinha, A.: A mathematical formulation of a transportation problem with economies of scale. Carnegie Bosch Institute Working Paper 01-1, 2001.
7. Guha, S., Meyerson, A., Munagala, K.: Improved combinatorial algorithms for single sink edge installation problems. To appear in Proc. 33rd Ann. ACM Symposium on Theory of Computing, 2001.
8. Guha, S., Meyerson, A., Munagala, K.: Hierarchical placement and network design problems. Proc. 41st Ann. IEEE Symposium on Foundations of Computer Science, 2000.
9. Hassin, R., Ravi, R., Salman, F.S.: Approximation algorithms for a capacitated network design problem. Proc. of the APPROX 2000, 167-176, 2000.
10. Khuller, S., Raghavachari, B., Young, N.E.: Balancing minimum spanning and shortest path trees. *Algorithmica*, 14, 305-322, 1993.
11. Mansour, Y., Peleg, D.: An approximation algorithm for minimum-cost network design. The Weizman Institute of Science, Rehovot, 76100 Israel, Tech. Report CS94-22, 1994; Also presented at the DIMACS workshop on Robust Communication Networks, 1998.
12. Meyerson, A., Munagala, K., Plotkin, S.: Cost-distance: Two metric network design. Proc. 41st Ann. IEEE Symposium on Foundations of Computer Science, 2000.
13. Ravi, R., Salman, F.S.: Approximation algorithms for the traveling purchaser problem and its variants in network design. Proc. of the European Symposium on Algorithms, 29-40, 1999.
14. Salman, F.S., Cheriyan, J., Ravi, R., Subramanian, S.: Approximating the single-sink link-installation problem in network design. *SIAM Journal of Optimization* 11(3):595-610, 2000.