QoS Multimedia Multicast Routing

Ion Mandoiu[1], Alex Olshevsky[2], Alex Zelikovsky[3]

[1] Department of Computer Science, University of Connecticut

Storrs, CT 06269-2155, E-mail: ion@engr.uconn.edu

[2] Laboratory for Information and Decision Systems, Massachusetts Institute of Technology

Cambridge, MS 02139, E-mail: alex_o@mit.edu

[3] Department of Computer Science, Georgia State University

Atlanta, GA 30303, E-mail: alexz@cs.gsu.edu

September 12, 2005

1 Introduction

Recent progress in audio, video, and data storage technologies has given rise to a host of high-bandwidth real-time applications such as video conferencing. These applications require Quality of Service (QoS) guarantees from the underlying networks. Thus, multicast routing algorithms which manage network resources efficiently and satisfy the QoS requirements have come under increased scrutiny in recent years [19]. The focus on multimedia data transfer capability in networks is expected to further increase as video conferencing applications gain popularity.

It is becoming apparent that new network mechanisms will be required to provide differentiated quality guarantees to network users. Of particular importance is the problem of optimal multimedia distribution from a source to a collection of users with heterogeneous demands. Multimedia distribution is usually done via multicast trees. There are two main reasons for using trees in multicast routing: (a) the data can be transmitted concurrently to destinations along the branches of the tree, and (b) only a minimum number

of copies of the data must be transmitted since information replication is limited to the branshing points of the tree [22]. The bandwidth savings obtained from the use of multicast trees can be maximized by using optimal or nearly optimal multicast tree algorithms, and future networks are expected to integrate such algorithms into their operation [4].

Several versions of the QoS multicast problem have been studied in the literature. These versions seek routing tree cost minimization subject to (1) end-to-end delay, (2) delay variation, and/or (3) minimum bandwidth constraints (see, e.g., [4, 17, 13]). This chapter deals with the case of minimum bandwidth constraints, that is, the problem of finding an optimal multicast tree when each terminal possesses a different rate of receiving information. This problem is a generalization of the classical Steiner tree problem and therefore NP-hard [7]. Formally, given a graph $G = (V, E)$, a source s, a set of terminals S, and two functions: $length : E \to R^+$ representing the length of each edge and $rate : S \to R^+$ representing the rate of each terminal, a *multicast tree* T is a tree in G spanning s and S. The *rate* of an edge e in a multicast tree T, denoted by $rate(e, T)$, is the maximum rate of a downstream terminal, i.e., of a terminal in the connected component of $T - e$ which does not contain s. The *cost* of a multicast tree T is defined as

$$cost(T) = \sum_{e \in T} length(e) \cdot rate(e)$$

Quality of Service Multicast Tree (QoSMT) Problem: Given a network $G = (V, E, length, rate)$ with source $s \in V$ and set of terminals $S \subseteq V$, find a minimum cost multicast tree in G.

Without loss of generality, in this paper we further assume that the rates belong to a given discrete set of possible rates: $0 = r_0 < r_1 < \ldots < r_N$. The QoSMT problem is equivalent to the Grade of Service Steiner Tree problem [21], which has a slightly different formulation: in the latter the network has no source node and rates r_e must be assigned to edges so that the minimum edge rate on the tree path from a terminal with rate r_i to a terminal with rate r_j is at least $min(r_i, r_j)$, and such that the total tree cost is minimized. A more general *QoSMT with Priorities* was considered by Charikar et al. [7]. In this version of the problem the cost of an edge e is given arbitrarily instead of being equal to the length times the rate. In other words, edge costs in QoSMT with Priorities are not required to be proportional to edge rates. This generalization seems more difficult – the best known approximation ratio is logarithmic (which also holds for multiple multicast

groups) [7].

The QoSMT problem was introduced in the context of multimedia distribution over communication networks by Maxemchuk [13]. Maxemchuk suggested a low-complexity heuristic that can be used to build reliable multicast trees in many practical applications. Following Maxemchuk, Charikar et al [7] gave a useful approximation algorithm that finds a solution within $e\alpha$ of the optimal, where $\alpha < 1.550$ is the best approximation ratio of an algorithm for the Steiner tree problem. This is the first known algorithm with a constant approximation ratio for this problem. Recently, an approximation ratio of 3.802 based on accurate estimation of Steiner tree length has been achieved in [12].

We note that the problem QoSMT problem was also considered previously (under different names) in the operations research literature. A number of results for particular instances of the problem were obtained: Current et al. [9] gave an integer programming formulation for the problem and proposed a heuristic algorithm for its solution. Some results for the case of few rates were obtained by Balakrishnan et al. in [1] and [2]. Specifically, [2] (see also [21]) suggested an algorithm for the case of two non-zero rates with approximation ratio of $\frac{4}{3}\alpha < 2.066$. An improved approximation algorithm with a ratio of 1.960 was proposed in [12]. For the case of three non-zero rates, Mirchandani [14] gave an 1.522-approximation algorithm.

This chapter is organized as follows. First, we describe centralized algorithms for the QoSMT problem, spending the bulk of the time on the algorithms in [12], which have best approximation factors to date. Although these algorithms have superior quality, they cannot be easily adjusted for operation in a distributed environment. Thus, we then describe a more practical primal-dual approach to the QoSMT problem following [6]. This approach yields algorithms that have natural distributed implementations, and work well even when the multimedia source does not have exact knowledge of network topology. We conclude with an experimental comparison showing the advantage of the primal-dual approach over practical heuristics proposed in the literature.

2 Centralized Approximation Algorithms

Tables 1.1 and 1.2 summarize the approximation ratios of known centralized algorithms for the QoSMT problem, for the cases of two non-zero rates and unbounded number of non-zero rates, respectively. In this

table we present the approximation ratios achievable using various Steiner tree approximation algorithms as a subroutine. Note that along with the best approximation ratios resulting from the use of the loss-contracting Steiner tree algorithm in [18], we also give approximation ratios resulting from the use of the more practical Steiner tree algorithms from [16, 3, 20, 24]. In this section we briefly discuss Maxemchuk's [13] and Charikar et al. [7] methods, and then give give a detailed description of best-to-date β-convex approximation algorithms of Karpinski et al [12].

2.1 Summary of Approximation Factors for the QoSMT Problem

2.2 Maxemchuk's Algorithm

Maxemchuk [13] introduced the QoSMT problem and proposed the first heuristic to solve this problem. His algorithm is a modification of the MST heuristic for Steiner Trees [20] (see Figure 1.1).

The extensive experiments given in [13] demonstrate that this method works well in practice. Nevertheless, the following example shows that the method may produce arbitrarily large error (linear in the number of rates) compared with the optimal tree. Consider the natural generalization of the example in Figure 1.2 with an arbitrary number k of distinct rates. Its optimal solution has a cost of about 1, whereas Maxemchuk's method returns a solution of cost about $(k+1)/2$. As there are $2^{k-1}+1$ nodes, this cost can also be written as $1+\frac{1}{2}\log_2(n-1)$, where n is the number of nodes in the graph. We conclude that the approximation ratio of Maxemchuk's algorithm is no better than linear in the number of rates and no better than logarithmic in the number of nodes in the graph.

2.3 The Charikar-Naor-Schieber Algorithms

Charikar et al [7] gave the first constant-factor approximation algorithms for the QoS Steiner tree problem. The simplest version is a *binary rounding* algorithm. In its first step, all rates are rounded to the closest power of two to produce the rounded up instance of this problem (clearly, this at most doubles the cost of an optimal solution). In its second step, Steiner trees are computed separately for each rate (within some approximation ratio α). The union of these trees is the final solution.

Consider the network obtained by replacing each edge of rate 2^i in an optimal solution by $i+1$ parallel

edges of rates $2^0, 2^1, \ldots, 2^{i-1}, 2^i$, respectively. In the new network, edges of a specific rate form a Steiner tree spanning all terminals of the respective rate. Since the optimal cost in this new network is no more than twice the cost of the rounded up instance, taking the union of all the computed Steiner trees introduces another factor of two to the approximation ratio. Thus the final approximation factor is $2 \cdot \alpha \cdot 2 = 4\alpha$.

Using a randomization technique, Charikar, Naor, and Schieber [7] reduce the approximation ratio to $e\alpha \approx 4.21$, where $e \approx 2.71$ is the Euler constant and $\alpha \approx 1.55$ is the currently best approximation ratio for the Steiner Tree problem.

2.4 β-Convex Steiner Tree Approximation Algorithms

In this section we introduce the notion of β-convex Steiner tree approximation algorithms and show tighter upper bounds on their output when applied to the QoSMT problem.

We begin by reviewing some Steiner tree definitions. A Steiner tree is a minimum-length tree connecting a subset of the graph's nodes. The nodes in a subset are usually referred to as *terminal* nodes. A Steiner tree is called *full* if every terminal is a leaf. A Steiner tree can be decomposed into components which are full by breaking the tree up at the non-leaf terminals. A Steiner tree is called *k-restricted* if every full component has at most k terminals. Let us denote the length of the optimum k-restricted Steiner tree as opt_k and the length of the optimum unrestricted Steiner tree as opt. Let the k-restricted Steiner ratio ρ_k be $\rho_k = sup\frac{opt_k}{opt}$, where the supremum is taken over all instances of the Steiner tree problem. It has been shown in [5] that $\rho_k = \frac{(r+1)2^r+s}{r2^r+s}$, where r and s are obtained from the decomposition $k = 2^r + s$, $0 \leq s < 2^r$. A slightly tighter bound on the length of the optimal k-restricted Steiner tree has been established in [12].

Theorem 1.1 *[12] For every Steiner tree T partitioned into edge-disjoint full components T^i,*

$$opt_k \leq \sum_i \left(\rho_k(l(T^i) - D(T^i)) + D(T^i)\right),$$

where $l(T^i)$ is the length of the full component T^i and $D(T^i)$ is the length of the longest path in T^i.

β-convexity of Steiner tree approximation has been introduced in [12]. A Steiner tree heuristic A is called a *β-convex α-approximation Steiner tree algorithm* if there exist an integer m and non-negative real numbers λ_i, $i = 2, \ldots, m$, with $\beta = \sum_{i=2}^{m} \lambda_i$ and $\alpha = \sum_{i=2}^{m} \lambda_i \rho_i$ such that the length of the tree computed by A, $l(A)$,

is upper bounded by

$$l(A) \leq \sum_{i=2}^{m} \lambda_i opt_i,$$

where opt_i is the length of the optimal i-restricted Steiner tree.

The MST-algorithm [20] is 1-convex 2-approximation since its output is the optimal 2-restricted Steiner tree of length opt_2. Every k-restricted approximation algorithm from [3] is 1-convex – the sum of coefficients in the approximation ratio always equals to 1, e.g., for $k = 3$, it is 1-convex 11/6-approximation algorithm since the output tree is bounded by $\frac{1}{2}opt_2 + \frac{1}{2}opt_3$. The output tree for PTAS [16] converges to the optimal 3-restricted Steiner tree and has length $(1 + \epsilon)opt_3$, therefore, it is $(1 + \epsilon)$-convex $\frac{5}{3}(1 + \epsilon)$-approximation algorithm. The currently best approximation ratio of $1 + \frac{\sqrt{3}}{2}$ is achieved by heuristic from [18] which is not known to be β-convex for any value of β.

Given a β-convex α-approximation algorithm A, it follows from Theorem 1.1 that

$$l(A) \leq \sum_i \lambda_i opt_i \leq \sum_i \lambda_i \rho_i (opt - D) + \beta D = \alpha(opt - D) + \beta D \tag{2.1}$$

Let OPT be the optimum cost QoSMT tree T, and let t_i be the length of rate r_i edges in T. Then,

$$cost(OPT) = \sum_{i=1}^{N} r_i t_i$$

Let OPT_k be the subtree of the optimal QoS Multicast tree OPT induced by edges of rate r_i, $i \geq k$. The tree OPT_k spans the source s and all nodes of rate r_k and, therefore, an optimal Steiner tree connecting s and rate-r_k nodes cannot be longer than

$$l(OPT_k) = \sum_{i=k}^{N} t_i$$

The main idea of the QoSMT algorithms in [12] is to reuse connections for the higher rate nodes when connecting lower rate nodes. When connecting nodes of rate r_k, we collapse nodes of rate strictly higher than r_k into the source s thus allowing to reuse higher rate connections for free. Let T_k be an approximate Steiner tree connecting the source s and all nodes of rate r_k after collapsing all nodes of rate strictly higher than r_k into the source s and treating all nodes of rate lower than r_k as Steiner points. If we apply an α-approximation Steiner tree algorithm for finding T_k, then the resulted length can be bounded as follows

$$l(T_k) \leq \alpha l(OPT_k) = \alpha t_k + \alpha t_{k+1} + \ldots + \alpha t_N$$

The following lemma shows that if the tree T_k is obtained using β-convex α-approximation Steiner tree algorithm, then a tighter upper bound on the length of T_k holds.

Lemma 1.1 *[12] Given an instance of the QoSMT problem, the cost of the tree T_k computed by a β-convex α-approximation Steiner tree algorithm is at most*

$$cost(T_k) \leq \alpha r_k t_k + \beta(r_k t_{k+1} + r_k t_{k+2} + \cdots + r_k t_N)$$

Proof. Let OPT_k be the subtree of the optimal QoS Multicast tree OPT induced by edges of rate r_i, $i \geq k$. By duplicating nodes and introducing zero length edges, it can be assumed that OPT_{k+1} is a complete binary tree with the set of leaves consisting of the source s and all nodes of rate at least r_{k+1}. The edges of rate r_k form subtrees attached to the tree OPT_{k+1} connecting rate r_k nodes to OPT_{k+1} (see Figure 1.3(a)).

Note that edges of any binary tree T can be partitioned into the edge-disjoint paths connecting internal nodes with leaves as follows. Each internal node v (including the degree-2 root) is split into two nodes v_1 and v_2 such that v_1 becomes a leaf incident to one of the downstream edges and v_2 becomes a degree-2 node (or a leaf if v is the root) incident to an edge connecting v to its parent (if v is not the root) and another downstream edge. Since each node is incident to a downstream edge, each resulted connected component will be a path containing exactly one leaf of T connected to an internal node of T.

The binary tree OPT_{k+1} broken into edge-disjoint paths described above along with all nodes of rate r_k that attached to them is shown on Figure 1.3(b). A resulted connected component OPT_k^i consisting of a path $D_k^i = OPT_k^i \cap OPT_{k+1}$ and attached Steiner trees with edges of rate r_k is shown on Figure 1.3(c). Note that the total length of the paths D_k^i is $l(OPT_{k+1}) = t_{k+1} + t_{k+2} + \cdots + t_N$. By Theorem 1.1, decomposing the tree T_k along these full components OPT_k^i results in the following upper bound:

$$\begin{aligned} l(T_k) &\leq \sum_i \left[\alpha(l(OPT_k^i) - D_k^i) + \beta D_k^i\right] \\ &= \alpha t_k + \beta(t_{k+1} + t_{k+2} + \cdots + t_N) \end{aligned}$$

The lemma follows by multiplying the last inequality by r_k.

2.5 β-Convex Approximation for QoSMT with Two Rates

In practice, it is often the case that only few distinct rates are requested by the terminals. This is why the QoS problem with two or three rates has a long history [1, 2, 14, 21]. The previously-best results of [14] and [21] have produced algorithms with approximation factor equal to 2.667 (provided that the MST heuristic is used to compute Steiner trees).

In this section approximation factors for the QoSMT problem with two non-zero rates are derived for the balancing algorithm based on β-convex Steiner tree approximation (see Figure 1.4) [12].

Recall that an edge e has rate r_i if the largest node rate in the component of $T - \{e\}$ that does not contain the source is r_i. Let the optimal Steiner tree in G have cost $opt = r_1 t_1 + r_2 t_2$, with t_1 being the total length of the edges of rate r_1 and t_2 being the total length of the edges of rate r_2. The algorithm in Figure 1.4 uses as subroutines two Steiner tree algorithms: an algorithm A_1 with an approximation ratio of α_1, and a β-convex algorithm A_2 with an approximation ratio of α_2. It outputs the minimum cost Steiner tree between the tree $ST1$ obtained by running A_1 with a set of terminals containing the source and the nodes with both high and low non-zero rate, and the tree $ST2$ obtained by running A_1 with a set of terminals containing the source and all high rate nodes, contracting the resulting tree into the source, and running A_2 with a set of terminals containing the contracted source and the low rate nodes.

Theorem 1.2 *[12] The algorithm in Figure 1.4 has an approximation ratio of*

$$\max\left\{\alpha_2, \quad \max_r \ \alpha_1 \frac{\alpha_1 - \alpha_2 r + \beta r}{\alpha_1 - \alpha_2 r + \beta r^2}\right\}$$

Proof. The cost of ST1 is bounded by $cost(ST1) \leq \alpha_1 r_2 (t_1 + t_2)$. To obtain a bound on the cost of ST2 note that $cost(T_2) \leq \alpha_1 r_2 t_2$, and that, by Lemma 1.1, $cost(T_1) \leq \alpha_2 r_1 t_1 + \beta r_1 t_2$.

Thus, the following two bounds for the costs of $ST1$ and $ST2$ follow:

$$\begin{aligned} cost(ST1) &\leq \alpha_1 r_2 t_1 + \alpha_1 r_2 t_2 \\ cost(ST2) &\leq \alpha_1 r_2 t_2 + \alpha_2 r_1 t_1 + \beta r_1 t_2 \end{aligned}$$

Let us distinguish the following two cases:

Case 1: Let $\beta r_1 \leq (\alpha_2 - \alpha_1) r_2$. Then

$$cost(ST2) \leq \alpha_1 r_2 t_2 + \alpha_2 r_1 t_1 + \beta r_1 t_2$$

$$\begin{aligned} &\leq \alpha_1 r_2 t_2 + \alpha_2 r_1 t_1 + (\alpha_2 - \alpha_1) r_2 t_2 \\ &\leq \alpha_2 (r_2 t_2 + r_1 t_1) \\ &= \alpha_2 opt \end{aligned}$$

Case 2: Let $\beta r_1 > (\alpha_2 - \alpha_1) r_2$. Then the following two values are positive

$$\begin{aligned} x_1 &= \frac{r_1}{\alpha_1 r_2}(\beta r_1 - (\alpha_2 - \alpha_1) r_2) \\ x_2 &= r_2 - r_1 \end{aligned}$$

The following linear combination will be bounded

$$\begin{aligned} x_1 cost(ST1) + x_2 cost(ST2) &= \frac{r_1(\beta r_1 - (\alpha_2 - \alpha_1) r_2)}{\alpha_1 r_2} cost(ST1) + (r_2 - r_1) cost(ST2) \\ &\leq r_1(\beta r_1 - (\alpha_2 - \alpha_1) r_2)(t_1 + t_2) \\ &\quad + (r_2 - r_1)(\alpha_1 r_2 t_2 + \alpha_2 r_1 t_1 + \beta r_1 t_2) \\ &= ((\beta - \alpha_2) r_1^2 + r_1 r_2 \alpha_1) t_1 + ((\beta - \alpha_2) r_1 r_2 + r_2^2 \alpha_1) t_2 \\ &= ((\beta - \alpha_2) r_1 + r_2 \alpha_1)(r_1 t_1 + r_2 t_2) \\ &\leq (\beta r_1 + \alpha_1 r_2 - \alpha_2 r_1) opt \end{aligned} \tag{2.2}$$

Let *Approx* be the cost of the tree produced by the approximation algorithm. The inequality (2.2) implies that

$$\begin{aligned} Approx &= \min\{cost(ST1), cost(ST2)\} \\ &= \frac{x_1 \min\{cost(ST1), cost(ST2)\} + x_2 \min\{cost(ST1), cost(ST2)\}}{x_1 + x_2} \\ &\leq \frac{x_1 cost(ST1) + x_2 cost(ST2)}{x_1 + x_2} \\ &\leq \frac{\beta r_1 + \alpha_1 r_2 - \alpha_2 r_1}{\frac{r_1}{\alpha_1 r_2}(\beta r_1 - (\alpha_2 - \alpha_1) r_2) + r_2 - r_1} opt \\ &\leq \alpha_1 \frac{\beta r_1 r_2 + \alpha_1 r_2^2 - \alpha_2 r_1 r_2}{\beta r_1^2 - (\alpha_2 - \alpha_1) r_2 r_1 + \alpha_1 r_2^2 - \alpha_1 r_1 r_2} opt \\ &\leq \alpha_1 \frac{\alpha_1 - \alpha_2 r + \beta r}{\alpha_1 - \alpha_2 r + \beta r^2} opt \end{aligned}$$

where $r = \frac{r_1}{r_2}$.

Summarizing the two cases we obtain that *Approx* is at most the maximum of two values – $\alpha_2 opt$ and $\alpha_1 \frac{\alpha_1 - \alpha_2 r + \beta r}{\alpha_1 - \alpha_2 r + \beta r^2} opt$ – which proves the theorem.

Theorem 1.2 implies numerical bounds on the approximation ratios. Using that $\alpha_1 = 1 + \ln 3/2 + \epsilon$ for the algorithm from [18], $\alpha_2 = 5/3 + \epsilon$ for the algorithm from [16], $\alpha_1 = \alpha_2 = 11/6$ for the algorithm from [3], and $\alpha_1 = \alpha_2 = 2$ for the MST heuristic, and $\beta \to 1$ for all of the above algorithms (except for the algorithm from [18]), we maximize the expression in Theorem 1.2 to obtain the following theorem.

Theorem 1.3 *[12] If the algorithm from [18] is used as A_1 and the algorithm from [16] is used as A_2, then the approximation ratio of the QoSMT algorithm in Figure 3 is $1.960 + \epsilon$. If the algorithm from [16] is used in place of both A_1 and A_2, then the approximation ratio is $2.059 + \epsilon$. If the algorithm from [3] is used in place of both A_1 and A_2, then the ratio is 2.237. If the MST heuristic is used in place of both A_1 and A_2, then the ratio is 2.414.*

2.6 β-Convex Approximation for QoSMT with Unbounded Number of Rates

In this section, we describe and prove the performance ratios of β-convex approximation algorithms for the case of the QoSMT problem with arbitrarily many non-zero rates $r_1 < r_2 < \cdots < r_N$ [12]. The algorithm (see Figure 1.5) is a modification of the algorithm in [7]. As in [7], node rates are rounded up to the closest power of some number a starting with a^y, where y is picked uniformly at random between 0 and 1. In other words, the given rates are replaced with numbers from the set $\{a^y, a^{y+1}, a^{y+2}, \ldots\}$. The major difference is that each approximate Steiner tree, T_k, constructed over nodes of rounded rate a^{y+k} is contracted in increasing order of k instead of simply taking union of T_k's according to [7]. This allows contracted edges to be reused at zero cost by Steiner trees connecting lower rate nodes. The following analysis from [12] of this improvement shows that it decreases the approximation ratio from 4.211 to 3.802.

Let T_{opt} be the optimal QoS Multicast tree, and let t_i be the total length of the edges of T_{opt} with rates rounded to a^{y+i}. First, we prove the following "randomized doubling" lemma corresponding to Lemma 4 from [7].

Lemma 1.2 *[12] Let S be the cost of T_{opt} after rounding node rates as in Figure 1.5, i.e., $S = \sum_{i=0}^{n} t_i a^{y+i}$.*

Then,

$$S \leq \frac{a-1}{\ln(a)} cost(T_{opt})$$

Proof. First, note that an edge e used at rate r in T_{opt} will be used at the rate a^{y+m}, where m is the smallest integer i such that a^{y+i} is no less than r. Indeed, e is used at rate r in T_{opt} if and only if the maximum rate of a node connecting to the source via e is r, and every such node will be rounded to a^{y+m}. Next, let $r = a^{x+m}$. If $x \leq y$, then the rounded up cost is a^{y-x} times the original cost; otherwise, if $x > y$, is a^{y+1-x} times the original cost. Hence, the expected factor by which the cost of each edge increases is

$$\int_0^x a^{y+x-1} dy + \int_x^1 a^{y-x} dy = \frac{a-1}{\ln a}$$

By linearity of expectation, the expected cost after rounding of T_{opt} is

$$S \leq \frac{a-1}{\ln a} cost(T_{opt})$$

Theorem 1.4 *[12] The algorithm given in Figure 1.5 has an approximation ratio of*

$$\min_a \left(\alpha \frac{a}{\ln a} - (\alpha - \beta) \frac{1}{\ln a} \right)$$

Proof. (Sketch) Let *Approx* be the cost of the tree returned by the algorithm in Figure 1.5, and $Approx_k$ be the cost of the tree T_k constructed by the algorithm when considering rate r_k. Then, by Lemma 1.1,

$$Approx_k \leq \alpha a^{y+k} t_k + \beta a^{y+k+1} t_{k+1} + \beta a^{y+k+2} t_{k+2} + \cdots + \beta a^{y+n} t_n$$

Summing up all the $Approx_k$'s (we omit the details), we get an upper bound of

$$\begin{aligned} (\alpha - \beta) S + \beta S \left(1 + \frac{1}{a} + \frac{1}{a^2} + \cdots \right) &\leq (\alpha - \beta) \frac{a-1}{\ln a} cost(T_{opt}) + \beta \frac{a}{\ln a} cost(T_{opt}) \\ &= \left(\alpha \frac{a}{\ln a} - (\alpha - \beta) \frac{1}{\ln a} \right) cost(T_{opt}) \end{aligned}$$

where the last inequality follows from Lemma 1.2.

Note that the corresponding approximation ratio in [7] is larger and equals $\alpha \frac{a}{\ln a}$ attaining minimum for $a = e$. The minimum of the approximation ratio in Theorem 1.4 can be obtained numerically – it is equal to 3.802, 4.059, respectively 4.311, when the β-convex α-approximation Steiner tree algorithm used in Figure 4 is the algorithm in [16], [3], respectively the MST heuristic. Finally, the algorithm in Figure 1.5 can be derandomized using the same techniques as in [7].

3 Primal-Dual Approach to the QoSMT Problem

In this section we discuss several primal-dual heuristics for the QoSMT problem due to Calinescu et al. [6]. A simpler integer linear program and two primal-dual algorithms based on it are discussed in Sections 3.1 and 3.2. A tighter integer linear program and an associated 4.311-approximation primal-dual algorithm are then described in Section 3.3.

3.1 A Simpler ILP Formulation

The QoSMT problem can be formulated as an integer program as follows. Consider a network $G = (V, E, length, rate)$ with a source node s and a set of terminal nodes. Let $r_1 < r_2 < \cdots < r_N$ be all rate values assigned to the terminals. It simplifies notation to assume that every node has a rate by considering an extra rate $r_0 = 0$ (assign rate r_0 to each non-terminal node). As above the source s has the highest rate. Construct a new network $G' = (V, E', cost, rate)$ by replacing each edge e of G with k edges $(e, r_1), (e, r_2), \ldots, (e, r_k)$ and setting $cost((e, r_i)) = r_i \cdot length(e)$.

Let $x_{(e,r)}$ be a boolean variable denoting whether edge e is used at rate r in an optimum tree. The QoS Steiner tree problem can be formulated as

$$\min \quad \sum_{(e,r)\in E'} x_{(e,r)} \cdot r \cdot length(e) \tag{3.3}$$

$$\text{s.t.} \quad \sum_{\substack{(e,r)\in\delta(C)\\ r\geq r_C}} x_{(e,r)} \;\geq\; 1, \quad \forall C \subseteq V \setminus \{s\} \tag{3.4}$$

$$x_{(e,r)} \;\in\; \{0, 1\} \tag{3.5}$$

where $\delta(C)$ denotes the set of edges with exactly one endpoint in C and r_C denotes the maximum rate of a node in C. Note that (3.3) gives the cost of an optimal solution, while (3.4) guarantees that each terminal is connected to the source through a collection of edges of rate no less than its rate.

After relaxing the integrality constraints (3.5) the dual linear program can be written as follows. For each (e, r), $C^*(e, r)$ is defined as $\{C \in V \setminus \{s\} : (e, r) \in \delta(C), r \geq r_C\}$. In words, $C^*(e, r)$ is the set of subsets C of $V \setminus \{s\}$ such that (e, r) has at least one endpoint in C and r is at least as large as r_C. Using

this definition, the dual is as follows:

$$\begin{aligned} \max \quad & \sum_{C} y_C \\ \text{s.t.} \quad & \sum_{C \in C^*(e,r)} y_C \;\leq\; r \cdot length(e), \quad \forall (e,r) \\ & y_C \;\geq\; 0 \end{aligned}$$

3.2 Two Primal-Dual Methods for the Simple ILP Formulation

The primal-dual framework applied to network design problems usually grows uniformly the dual variables associated to the "active" components of the current forest [11]. This approach fails to take into account the different rates of different nodes in the QoSMT problem. The *Naive Primal-Dual algorithm* [6] (see Figure 1.6) takes in account different rates by varying the speed at which each component grows. While the simulations in the ensuing sections show that this is a good method in practice, the solution it produces on some graphs may be very large compared to the optimal solution, as shown by the following example with two rates.

Consider two nodes of rate 1 connected by an edge of length 1 (see Figure 1.7). There is an arc between these two nodes, and on this arc there is a chain of nodes of rate ϵ. Each two consecutive nodes in the chain are at a distance δ from each other, where $\delta < 1$. Each extreme node in the chain is at a distance $\delta/2$ of its neighboring rate-1 node.

The Naive Primal-Dual applied to this graph connects the rate-ϵ nodes first, since $\frac{\delta}{2} < \frac{1}{2}$. So, the algorithm connects the rate-1 nodes via the rate-ϵ nodes, and not via the direct edge connecting them. Thus, the Naive Primal-Dual can make arbitrarily large errors (just take an arbitrarily long chain).

An improved *Restarting Primal-Dual algorithm* [6] is given in Figure 1.8. One can easily see that this is a primal-dual algorithm. Indeed, each addition of an edge to the current solution is the result of growing dual variables. Moreover, since the feasibility requirement for edge a is $\Sigma_{a \in \delta(C)} y_C \leq r \cdot length(a)$, this addition preserves the feasibility of the dual solution. The algorithm maintains forests F^{r_i} given by the edges picked at rate r_i, and the connected components of F^{r_i}, seen as sets of vertices, are denoted in the algorithm by C_{r_i}. Such a component is *active* if $r_{C_{r_i}} = r_i$ and C_{r_i} is disjoint from components of higher rate.

The Restarting Primal-Dual algorithm avoids the mistake made by the Naive Primal-Dual algorithm on

the frame example in Figure 1.7(a). Then, at time $\frac{\delta}{2}$ the rate-ϵ nodes become connected. This means that $\delta(1-\epsilon)$ of each rate-1 edge between the ϵ-rate nodes is not covered. Meanwhile, the rate-1 nodes are growing on the respective edges as shown in Figure 1.7(b).

Let us assume that the Restarting Primal-Dual algorithm uses the chain of rate-ϵ nodes to connect the two rate-1 nodes instead of the direct edge. This would imply that it takes less time to cover the chain, i.e., $\frac{1}{2}\delta(1-\epsilon)n \leq \frac{1}{2} - \frac{\delta}{2}$, where n is the number of rate-ϵ nodes. When ϵ is small, then $n\delta \leq 1$, so if the Restarting Primal-Dual algorithm uses the chain then it is correct to do so.

3.3 Primal-Dual 4.311-Approximation Algorithm

A constant-factor primal-dual approximation algorithm is obtained in [6] based on an enhanced integer linear programming formulation of the QoSMT problem. The enhanced formulation takes into account the fact that if a set $C \subset V \setminus \{s\}$ is connected to the source with edges of rate $r' > r_C$, then there should be at least **two** edges of rate r' with exactly one endpoint in C. The integer program is

$$\begin{aligned} \min \quad & \sum_{(e,r)\in E'} x_{(e,r)} \cdot r \cdot length(e) \\ \text{s.t.} \quad & \sum_{\substack{e\in\delta(C)\\ r=r_C}} x_{(e,r)} + \frac{1}{2} \sum_{\substack{e\in\delta(C)\\ r>r_C}} x_{(e,r)} \;\geq\; 1, \quad \forall C \subseteq V \setminus \{s\} \\ & x_{(e,r)} \in \{0,1\} \end{aligned}$$

The corresponding dual of the LP relaxation is

$$\begin{aligned} \max \quad & \sum_{C\subseteq V\setminus\{s\}} y_C \\ \text{s.t.} \quad & \sum_{\substack{C\,:e\in\delta(C)\\ r_C=r}} y_C + \frac{1}{2} \sum_{\substack{C\,:e\in\delta(C)\\ r_C<r}} y_C \;\leq\; r \cdot length(e) \\ & y_C \;\geq\; 0 \end{aligned} \tag{3.6}$$

The core algorithm presented in Figure 1.9 is preprocessed with random bucketing of rates as in Section 2.6 (see also Step 1 in Figure 1.5). Let a be a real (to be picked later) and y be a real picked uniformly at random from the interval $[0\,..\,1]$. Every node of rate r is replaced by a node of rate $a^{\gamma+j}$, where j is the integer satisfying $a^{\gamma+j-1} < r \leq a^{\gamma+j}$.

The primal-dual part follows the classical framework [11], and works in stages starting from the lower rate to the highest. During the execution of the algorithm, edges are picked at a certain rate (in other words, $x_{(e,r)}$ is set to 1) one by one. Before executing step 3 at rate r for the ith time, the set of edges picked at rate r by the algorithm forms a forest F_i^r. (An edge can be picked at several rates, but it is kept in at most one such rate in the final solution because of the reverse delete step.) A component C of F_i^r is called an *r-component* if $r_C = r$.

Using Constraint (3.6), it follows by induction on j that, for an edge e and a rate $a^{\gamma+j}$, we have

$$\begin{aligned}\sum_{\substack{C\,:e\in\delta(C)\\ r_C\le a^{\gamma+j}}} y_C &\le length(e)a^{\gamma+j}\sum_{i=0}^{j}\left(\frac{1}{2a}\right)^i \\ &\le length(e)a^{\gamma+j}\frac{2a}{2a-1}.\end{aligned}$$

For an edge picked by the algorithm at rate r, Constraint (3.6) is tight and therefore

$$\sum_{\substack{C\,:e\in\delta(C)\\ r_C\le a^{\gamma+j}}} y_C \ge length(e)\frac{2a-2}{2a-1}a^{\gamma+j}. \tag{3.7}$$

Exactly as in [11], the number of edges of rate r in the final solution which cross the active r-components at some moment (an edge being counted twice if it crosses two r-components) is at most twice the number of active r-components. Equation (3.7) and exactly the same argument as in Theorem 4.2 of [11] imply that the cost of the solution of the algorithm is bounded by $(2(2a-1)/(2a-2))\sum y_C \le ((2a-1)/(a-1))\,opt$, as any feasible solution for the dual linear program has value at most the value of any feasible solution of the primal.

The same argument as in Section 2.6 shows that the approximation ratio of the algorithm above is $(2a-1)/\ln a$. Numerical picking the same best value for a as in Section 2.6 implies

Theorem 1.5 *[6] The output cost of the algorithm on Figure 1.9 is at most* 4.311 *times the optimum cost.*

4 Experimental Study

In this section we report experimental results with several QoSMT heuristics: Maxemchuk's [13], binary rounding [7], naive primal-dual, and restarting primal-dual algorithms. The heuristics were implemented

in C++ and compiled using `gpp` with `-O2` optimization, and run on a Sun workstation Ultra-60. The experiments were run on random testcases generated using GT-ITM generator [10] which is used for modelling internet networks [23]. Table 1.3 gives a comparison of the performance of of the aforementioned algorithms. The experiments were conducted in the presence of no Steiner nodes, respectively 50% Steiner nodes. Moreover, both arithmetic and geometric distributions of rates were tested.

Table 1.3 gives the results for instances generated using several sets of parameters. The relative solution quality of various heuristics is fairly independent on the class of instances. We note that the Naive Primal-Dual and the Charikar-Naor-Schieber algorithms most often produce comparable results which are slight improvements over the results produced by Maxemchuk's algorithm. The Restarting Primal-Dual typically produces solutions of best quality, typically $0.25 - 6\%$ better than solutions produced by Maxemchuk's algorithm; this, however, occurs at the expense of greater CPU time. We also note that the difference between algorithms increases as the number of rates increases. Figures 1.10 and 1.11 illustrate this observation in a graphical form.

References

[1] A. Balakrishnan, T.L. Magnanti, P. Mirchandani, Modeling and Heuristic Worst-Case Performance Analysis of the Two-Level Network Design Problem, *Management Science*, **40**:846–867, 1994.

[2] A. Balakrishnan, T.L. Magnanti, P. Mirchandani, Heuristics, LPs, and Trees on Trees: Network Design Analyses, *Operations Research*, **44**:478–496, 1996.

[3] P. Berman and V. Ramaiyer, Improved Approximations for the Steiner Tree Problem, *Journal of Algorithms*, **17**:381–408 1994.

[4] R. Bajaj, C.P. Ravikumar, and S. Chandra, Distributed Delay Constrained Multicast Path Setup High Speed Networks, *Proceedings of the Fourth International Conference on High Performance Computing*, pp. 438–442, 1997.

[5] A. Borchers and D.Z. Du, The k-Steiner Ratio in Graphs, *SIAM Journal on Computing*, **26**:857–869, 1997.

[6] G. Calinescu, C. Fernandes, I. Mandoiu, A. Olshevsky, K. Yang and A. Zelikovsky, Primal-Dual Algorithms for QoS Multimedia Multicast, *Proceedings of IEEE GLOBECOM*, pp. 3631–3635, 2003.

[7] M. Charikar, J. Naor, and B. Schieber, Resource Optimization in QoS Multicast Routing of Real-Time Multimedia, *IEEE/ACM Transactions on Networking*, **12**:340–348, 2004.

[8] C.J. Colbourn and G.L. Xue, Grade of service Steiner trees in series-parallel networks, in Ding-Zhu Du, J.M. Smith, and J.H. Rubinstein, eds., *Advances in Steiner Trees*, Kluwer Academic Publishers, pp. 163–174, 2000.

[9] J.R. Current, C.S. Revelle, and J.L.Cohon, The Hierarchical Network Design Problem, *European Journal of Operations Research*, **27**:57–66, 1986.

[10] http://www.cc.gatech.edu/fac/Ellen.Zegura/gt-itm/gt-itm.tar.gz

[11] M. Goemans and D. Williamson, The Primal-Dual Method for Approximation Algorithms and its Application to Network Design Problems, in D. Hochbaum, ed., *Approximation Algorithms*, PWS Publishing Company, pp. 144–191, 1997.

[12] M. Karpinski, I. Măndoiu, A. Olshevsky, and A. Zelikovsky, Improved Approximation Algorithms for the Quality of Service Steiner Tree Problem, *Algorithmica*, **42**:109–120, 2005.

[13] N. Maxemchuk, Video Distribution on Multicast Networks, *IEEE Journal on Selected Areas in Communications*, **15**:357–372, 1997.

[14] P. Mirchandani, The Multi-Tier Tree Problem, *INFORMS Journal on Computing*, **8**:202–218, 1996.

[15] K. Mehlhorn, A faster approximation algorithm for the Steiner problem in graphs, *Information Processing Letters*, **27**:125–128, 1988.

[16] H. Promel and A. Steger, A New Approximation Algorithm for the Steiner Tree Problem with Performance Ratio $\frac{5}{3}$, *Journal of Algorithms*, **36**:89-101, 2000.

[17] G.N. Rouskas and I. Baldine, Multicast routing with end-to-end delay and delay variation constraints, *IEEE J. on Selected Areas in Communications*, **15**:346–356, 1997.

[18] G. Robins and A. Zelikovsky, Tighter Bounds for Graph Steiner Tree Approximation, *SIAM Journal on Discrete Mathematics*, **19**:122–134, 2005.

[19] H.F. Salama, D.S. Reeves, and Y. Viniotis, Evaluation of Multicast Routing Algorithms for Real-Time Communication on High-Speed Networks, *IEEE Journal on Selected Areas in Communications*, **3**:332–345, 1997.

[20] H. Takahashi and A. Matsuyama, An Approximate Solution for the Steiner Problem in Graphs, *Math. Japonica*, **6:**573–577, 1980.

[21] G. Xue, G.-H. Lin, D.-Z. Du, Grade of Service Steiner Minimum Trees in the Euclidean Plane, *Algorithmica*, **31**:479–500, 2001.

[22] H. Ural and K. Zhu, An Efficient Distributed QoS Based Multicast Routing Algorithm, *Proceedings of the 21st IEEE International Performance, Computing, and Communication Conference*, pp. 27–36, 2002.

[23] E.W. Zegura, K. Calvert, and S. Bhattacharjee. How to Model an Internetwork, *Proceedings of IEEE INFOCOM*, pp. 594–602, 1996.

[24] A. Zelikovsky, An 11/6-approximation algorithm for the network Steiner problem, *Algorithmica*, **9**:463–470, 1993.

[25] A. Zelikovsky, A faster approximation algorithm for the Steiner tree problem in graphs, *Information Processing Letters* **46**:79–83, 1993.

Table 1.1: QoSMT problem with 2 rates. Runtime and approximation ratios of previously known algorithms and of the algorithms given in this paper. In the runtime, n and m denote the number of nodes and edges in the original graph $G = (V, E)$, respectively. Approximation ratios associated with polynomial-time approximation schemes are accompanied by a $+\epsilon$ to indicate that they approach the quoted value from above and do not reach this value in polynomial time.

Steiner Tree Algorithm	LCA [18]	LCA +RNS[16]	BR [24, 3]	MST [20]
Runtime	polynomial	polynomial	$O(n^3)$ [25]	$O(n \log n + m)$ [15]
Approximation ratio in [2, 21]	$\frac{4}{3}\frac{1+\ln 3}{2} + \epsilon$ $< 2.066 + \epsilon$	$\frac{20}{9} + \epsilon$ $< 2.223 + \epsilon$	$\frac{22}{9}$ < 2.445	$\frac{8}{3}$ < 2.667
Improved ratio [12]	–	$1.960+\epsilon$	2.237	2.414

Table 1.2: Approximation ratios for QoSMT problem with an arbitrary number of rates.

Steiner Tree Algorithm	LCA [18]	RNS[16]	BR [24, 3]	MST [20]
Approximation ratio in [16]	$e\frac{1+\ln 3}{2} + \epsilon$ $< 4.212 + \epsilon$	$e\frac{5}{3} + \epsilon$ $< 4.531 + \epsilon$	$e\frac{11}{6}$ < 4.984	$2e$ < 5.44
Improved ratio [12]	–	$3.802 + \epsilon$	4.059	4.311

Table 1.3: Cost improvement over Maxemchuck's algorithm (%) and CPU seconds for binary rounding and two primal-dual algorithms (averages over 10 testcases).

50% steiner nodes, geometric progression rates								
R	N	Maxemchuk's	binary rounding		Naive-PD		Restart-PD	
		CPU	%G	CPU	%G	CPU	%G	CPU
1	200	0.017	0.00	0.017	-0.01	0.544	-0.01	0.325
1	300	0.050	0.00	0.052	0.04	1.372	0.04	0.946
2	200	0.027	0.00	0.026	0.43	1.271	1.03	1.125
2	300	0.070	0.00	0.072	0.93	4.573	2.17	3.747
5	200	0.044	0.00	0.044	-2.13	1.490	1.30	5.321
5	300	0.123	0.00	0.120	-0.91	5.221	1.10	16.798
10	200	0.065	0.00	0.068	-2.53	1.636	0.66	17.848
10	300	0.180	0.00	0.176	-2.61	6.582	0.24	107.125
50% steiner nodes, arithmetic progression rates								
1	200	0.016	0.00	0.017	-0.01	0.541	-0.01	0.327
1	300	0.052	0.00	0.051	0.04	1.370	0.04	0.946
2	200	0.027	0.00	0.023	-0.69	1.373	-0.00	1.136
2	300	0.071	0.00	0.070	-0.32	4.491	0.24	3.773
5	200	0.043	-0.01	0.040	1.70	1.564	2.66	5.256
5	300	0.123	-0.10	0.107	1.92	5.392	4.19	17.271
10	200	0.067	1.79	0.043	4.25	1.556	6.11	16.856
10	300	0.181	2.36	0.126	3.38	5.444	5.73	92.575
0% steiner nodes, geometric progression rates								
1	100	0.002	0.00	0.002	0.00	0.052	0.00	0.077
1	200	0.028	0.00	0.028	0.00	0.251	0.00	0.465
2	100	0.007	0.00	0.007	1.21	0.088	1.69	0.185
2	200	0.038	0.00	0.033	2.14	0.698	2.31	1.517
5	100	0.012	0.00	0.013	1.24	0.120	2.82	0.665
5	200	0.059	0.00	0.056	-0.25	1.296	1.70	6.314
10	100	0.019	0.00	0.018	-0.68	0.133	1.63	1.953
10	200	0.090	0.00	0.091	-1.97	1.466	0.73	20.525
0% steiner nodes, arithmetic progression rates								
1	100	0.005	0.00	0.005	0.00	0.054	0.00	0.078
1	200	0.026	0.00	0.026	0.00	0.247	0.00	0.457
2	100	0.005	0.00	0.006	-0.11	0.111	-0.04	0.187
2	200	0.036	0.00	0.034	-0.02	1.078	0.30	1.570
5	100	0.011	-0.17	0.011	3.70	0.114	4.60	0.656
5	200	0.059	-0.15	0.052	3.13	1.235	3.85	5.952
10	100	0.019	2.62	0.012	6.65	0.113	7.12	1.922
10	200	0.091	2.67	0.058	5.83	1.203	6.38	17.689

Input: A graph $G = (V, E, length, rate)$ with a source s in V and a collection of terminals $S \subseteq V$.

Output: A QoSMT spanning the source and the terminals.

---

1. Initialize the current tree to $\{s\}$.
2. Find a non-reached terminal t of highest rate with the shortest distance to the current tree.
3. Add t to the current tree along with a shortest path connecting it to the current tree.
4. Repeat until all terminals are spanned.

Figure 1.1: Maxemchuk's Algorithm for the QoSMT problem.

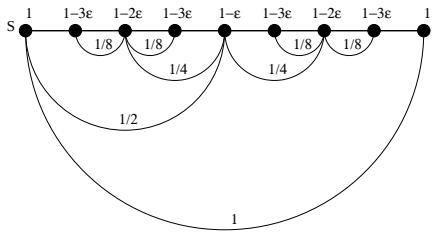

Figure 1.2: A bad example for Maxemchuk's algorithm, with $k = 4$ rates. In the figure, $\varepsilon = 1/2^{2k-1}$. The rate of each node is given above the node. The edge lengths are given on the thin curved arcs, while on the solid horizontal line each segment has length $1/2^{k-1} + \varepsilon$. The optimum, of total cost $1 + 2^{k-1}\varepsilon = 1 + 2^{k-1}(1/2^{2k-1}) = 1 + 1/2^k$, uses the solid horizontal line at rate 1. Maxemchuk's algorithm picks the thin curved arcs at a cost of $1 + (1/2)(1-\varepsilon) + 2(1/4)(1-2\varepsilon) + 4(1/8)(1-3\varepsilon) \geq ((k+1)/2)(1-1/2^k)$.

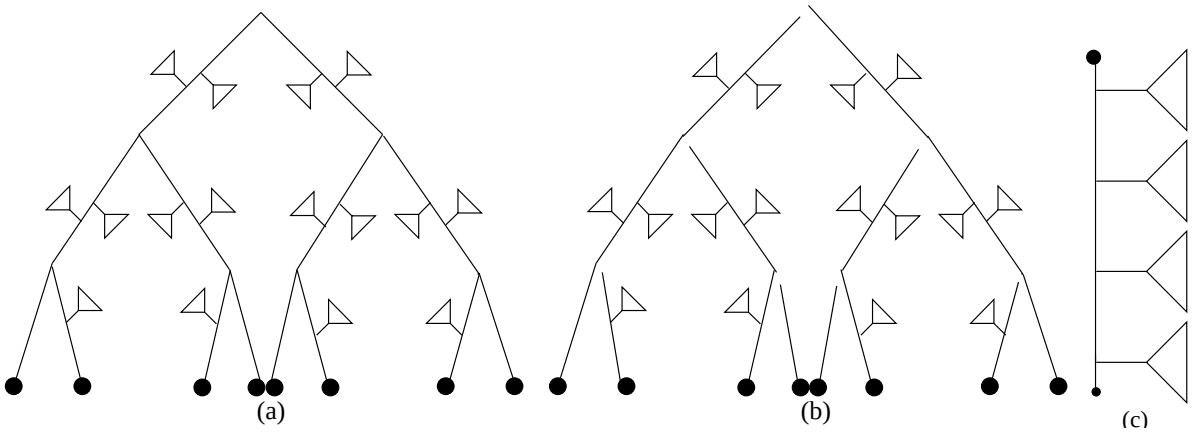

Figure 1.3: (a) The subtree OPT_k of the optimal QoS Multicast tree OPT induced by edges of rate r_i, $i \geq k$. Edges of rate greater than r_k (shown as solid lines) form a Steiner tree for $s \cup S_{k+1} \cup \ldots S_N$ (filled circles); attached triangles represent edges of rate r_k. (b) Partition of OPT_k into edge-disjoint connected components OPT_k^i each containing a single terminal of rate r_i, $i > k$. (c) A connected component OPT_k^i which consists of a path D_k^i containing all edges of rate r_i, $i > k$, and attached Steiner trees containing edges of rate r_k.

Input: Graph $G = (V, E, l)$ with two nonzero rates $r_1 < r_2$, source s, terminal sets S_1 of rate r_1 and S_2 of rate r_2, Steiner tree α_1-approximation algorithm A_1 and a β-convex α_2-approximation algorithm A_2

Output: Low cost QoSMT spanning all terminals

---

1. Compute an approximate Steiner tree ST1 for $s \bigcup S_1 \bigcup S_2$ using algorithm A_1
2. Compute an approximate Steiner tree T_2 for $s \bigcup S_2$ (treating all other points as Steiner points) using algorithm A_1. Next, contract T_2 into the source s and compute the approximate Steiner tree T_1 for s and remaining rate r_1 points using algorithm A_2. Let ST2 be $T_1 \bigcup T_2$
3. Output the minimum cost tree among ST1 and ST2

Figure 1.4: QoSMT approximation algorithm for two non-zero rates

Input: Graph $G = (V, E, l)$, source s, sets S_i of terminals with rate r_i, positive number a, and α-approximation β-convex Steiner tree algorithm

Output: Low cost QoSMT spanning all terminals

---

1. Pick y uniformly at random between 0 and 1. Round up each rate to the closest power of some number a starting with a^y, i.e., round up to numbers in the set $\{a^y, a^{y+1}, a^{y+2}, \ldots\}$. Form new terminal sets S_i' which are unions of terminal sets with rates rounded to the same number r_i'
2. $T \leftarrow \emptyset$
3. For each non-zero rounded rate r_i', in decreasing order, do:

   Find an α-approximate Steiner tree T_i spanning $s \bigcup S_i'$

   $T \leftarrow T \cup T_i$

   Contract T_i into source s
4. Output T

Figure 1.5: Approximation algorithm for multirate QoSMT

Input: A graph $G = (V, E, length, rate)$ with a source s in V and a collection of terminals $S \subseteq V$.

Output: A QoSMT spanning the source and the terminals.

---

1. Start from the spanning forest of G with no edges.
2. Grow y_C with speed r_C for each "active" component C of the current forest. (A component C is *inactive* if it contains s and all vertices of rate r_C.)
3. Stop growing once the dual inequality for a pair (e, r) becomes tight, with e connecting two distinct components of the forest.
4. Add e to the forest, collapsing the two components.
5. Terminate when there is no active component left.
6. Keep an edge of the resulting tree at the minimum needed rate.

Figure 1.6: The Naive Primal-Dual algorithm for the QoSMT problem.

Figure 1.7: The Restarting Primal-Dual avoids the mistake of the Naive Primal-Dual. Part (a) shows duplication of the edges. Part (b) shows the components growing along the respective edges.

Input: A Graph $G' = (V, E, cost, rate)$ with source s, and a collection of terminals S.

Output: A QoSMT spanning the source and the terminal.

---

1. Grow each active C_{r_i} with speed r_i along incident edges $(e, r_j),\ j \leq i$, picking edges which become tight.
2. Continue this process until there is no active component of rate r_k.
3. Remove all edges which are not necessary for maintaining connectivity of nodes of rate r_k.
4. Accept (keep in the solution) and contract all edges of C_{r_k} (i.e., set their length/cost to 0)
5. Restart the algorithm with the new graph

Figure 1.8: The Restarting Primal-Dual algorithm for the QoSMT problem.

Input: A graph $G = (V, E, length, rate)$ with source s in V and a collection of terminals $S \subseteq V$.

Output: A QoSMT spanning the source and the terminal.

---

1. For each $r = r_1, r_2, \ldots, r_k$, execute steps 2-6.
2. Start from the spanning forest F^r of G with no edges.
3. Grow y_C uniformly for each r-component C of the current forest F^r.
4. Stop growing once the dual inequality for a pair (e, r) becomes tight, with e connecting two distinct components of F^r.
5. Add (e, r) to F^r, collapsing two of its components.
6. Terminate when there is no r-component of F^r left.
7. Traversing the list of picked edges in reverse order, remove an edge (e, r) from F^r if after (e, r)'s removal the set of edges picked form a feasible tree.

Figure 1.9: The 4.311-approximation algorithm for QoSMT problem.

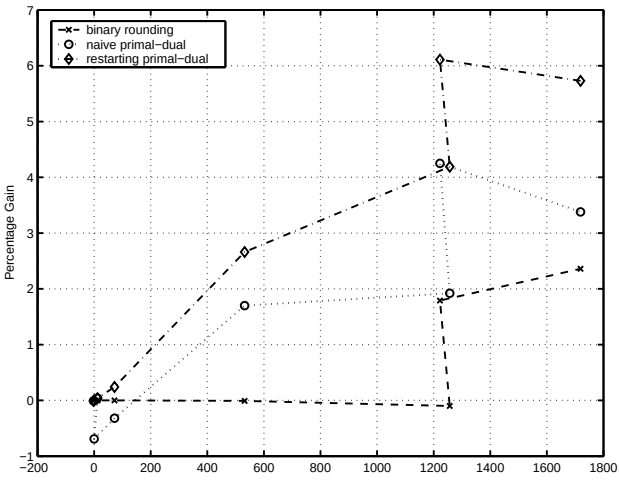

Figure 1.10: The gain of several algorithms versus Maxemchuk's algorithm, 50% Steiner nodes.

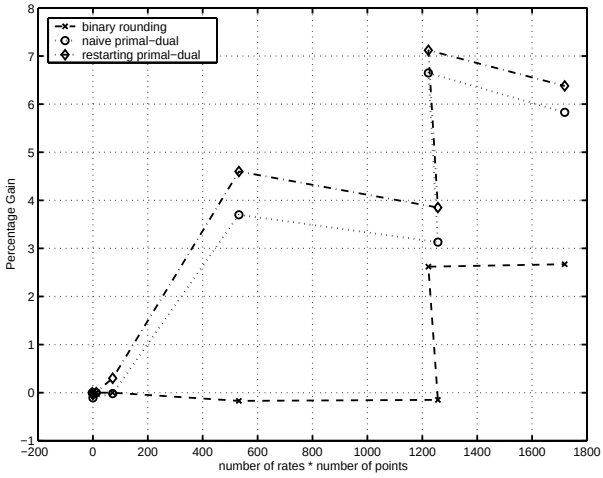

Figure 1.11: The gain of several algorithms versus Maxemchuk's algorithm, 0% Steiner nodes.