Modifying Networks to Obtain Low Cost Trees

S. O. Krumke[1] H. Noltemeier[1] M. V. Marathe[2] S. S. Ravi[3] K. U. Drangmeister[1]

February 28, 1996

Abstract

We consider the problem of reducing the edge lengths of a given network so that the modified network has a spanning tree of small total length. It is assumed that each edge e of the given network has an associated function C_e that specifies the cost of shortening the edge by a given amount and that there is a budget B on the total reduction cost. The goal is to develop a reduction strategy satisfying the budget constraint so that the total length of a minimum spanning tree in the modified network is the smallest possible over all reduction strategies that obey the budget constraint.

We show that in general the problem of computing optimal reduction strategy for modifying the network as above is NP-hard and present the first polynomial time approximation algorithms for the problem, where the cost functions C_e are allowed to be taken from a broad class of functions. We also present improved approximation algorithms for the class of treewidth-bounded graphs when the cost functions are linear. Our results can be extended to obtain approximation algorithms for more general network design problems such as those considered in [GW, GG+94].

Keywords: Location Theory, Approximation Algorithms, Parametric Search, Computational Complexity, NP-hardness.

1 Introduction

The problem of computing a minimum spanning tree for a network is a well studied problem in computer science. In this paper, we consider a variant of the problem where the goal is to shorten the edge lengths of a given network so that the length of a minimum spanning tree in resulting network is as small as possible. The problem is considered in a context where there is a cost associated with shortening a link and there is a budget constraint on the total cost of shortening the edges. Such a problem models situations wherein an organization desires to take advantage of technological advances to upgrade the communications network interconnecting its branches and has allocated a fixed budget to do so. The goal is to devise a strategy that upgrades the links of the network so that the total upgrading cost is within the chosen budget, and the length of a minimum spanning tree in the upgraded network is the smallest among all strategies that obey the budget constraint. A precise definition of the problem is given in the next section. Such problems arise in diverse areas such as design of high speed communication networks [KJ83], video on demand [KPP93], teleconferencing [KPP92, Komp], VLSI design [CK+92, CR91, ZPD94], database retrieval [Ra94], etc.

Most of the network improvement problems considered in this paper are NP-hard. Given these hardness results, we aim at finding efficient approximation algorithms for these problems. Define an (α, β)-approximation algorithm as a polynomial-time algorithm that produces a solution within α times the optimal function value, violating the budget constraint by a factor of at most β.

The main contribution of this paper is to develop a framework for formulating such network improvement problems. We provide the first polynomial time (α, β)-approximation algorithms for several versions of the problem. In the next section we formally define the problem considered in this paper and summarize our results.

[1]Department of Computer Science, University of Würzburg, Germany.

[2]Los Alamos National Laboratory, Los Alamos, NM. The work is supported by the Department of Energy under Contract W-7405-ENG-36.

[3]Department of Computer Science, University at Albany, Albany, NY.

2 Definitions and Summary of Results

Let $G = (V, E)$ be an undirected graph. Associated with each edge $e \in E$, there are two nonnegative values as follows: $L(e)$ denotes the *length* or the *weight* of the edge e and $L_{\min}(e)$ denotes the *minimum length* to which the edge e can be reduced. Consequently, we assume throughout the presentation that $L_{\min}(e) \leq L(e)$. The nonnegative *cost function* C_e indicates how expensive it is to reduce the length of e by a certain amount. We assume without loss of generality that $C_e(0) = 0$ for all edges $e \in E$.[4]

Given a budget B, we define a *feasible reduction* to be a nonnegative function r defined on E with the following properties: For all edges $e \in E$, $L(e) - r(e) \geq L_{\min}(e)$ and $\sum_{e \in E} C_e(r(e)) \leq B$.

Let T be a spanning tree of G. The *total length* of T under the weight function L, denoted by $L(T)$, is defined to be the sum of the lengths of the edges that are in T. We denote the total weight of a minimum total length spanning tree with respect to the weight function L by $\mathrm{MST}(G, L)$. If r is a (feasible) reduction in G we can consider the graph G with the edge weights given by the difference function $L - r$, i.e. $(L - r)(e) := L(e) - r(e)$ for all $e \in E$. The weight of the MST in G with respect to this function is denoted by $\mathrm{MST}(G, L - r)$.

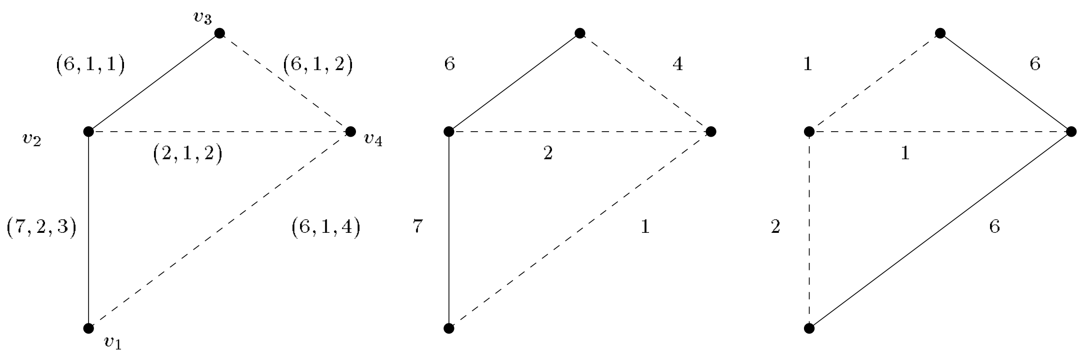
We are now ready to state the general problem studied in this paper.

Definition 2.1 The *Budget Minimum Total Cost Spanning Tree Problem* (BMST) is to find a feasible reduction r such that $\mathrm{MST}(G, L - r)$ has the least possible value.

Definition 2.2 Let $\alpha, \beta \geq 1$ be constants. We say that an algorithm is an (α, β)-*approximation algorithm* for BMST, if for each instance, the algorithm returns a reduction r of cost at most βB such that

$$\frac{\mathrm{MST}(G, L - r)}{\mathrm{MST}(G, L - r^*)} \leq \alpha, \tag{1}$$

where r^* denotes an optimal edge-reduction on G of cost at most B.



(a) The original graph G (b) Modification of G with cost 24 (c) Optimal Modification for a budget of 22

Figure 1: An example of a graph modification via edge reductions.

Example: Consider the graphs given in Figure 1. Figure 1(a) shows a graph G where each edge e is associated with the three values $(L(e), L_{\min}(e), C_e)$. The third parameter C_e represents the cost of reducing the length of the edge by a unit amount, i.e. the cost function on each edge in this simple example is linear and is given by $C_e(t) = C_e \cdot t$. The result of a modification of G is shown in Figure 1(b). The edges belonging to the minimum spanning tree are drawn as dashed lines. The modification corresponding to Figure 1(b) involves a cost of 24 and the weight of the resulting tree is 7. Figure 1(c) shows the graph with edge lengths resulting from a reduction that is optimal among all reductions of cost no more than 22. There, the weight of the spanning tree resulting from the reduction is 4. Thus, the reduction of Figure 1(b) is a $(7/4, 24/22)$-approximation to an optimal solution with budget 22.

[4]Any reduction will incur a minimum cost of $\sum_{e \in E} C_e(0)$ and we can subtract this sum from the budget B in advance.

Trees	Treewidth bounded graphs	General graphs
"Easy" Solvable by a Greedy algorithm for linear reduction costs	NP-hard $(1+\varepsilon, 1+\xi)$-approximable for any fixed $\varepsilon, \xi > 0$	NP-hard $(1+1/\gamma, 1+\gamma)$-approximable[5] for any fixed $\gamma > 0$

Table 1: Approximation and Hardness Results for BMST. Similar results hold for other general network design problems such as those considered in [GG+94].

The results obtained in this paper are summarized in Table 1. The approximation algorithm for BMST can be extended significantly. For example, using our ideas in conjunction with the results of Goemans et. al. [GG+94], we can obtain similar approximation results for finding budget constrained minimum-cost generalized Steiner trees, minimum-cost k-edge connected subgraphs and other network design problems specified by weakly supermodular functions introduced in that paper.

The remainder of the paper is organized as follows. In Section 3 we briefly discuss the structure of an optimal solution for linear reduction costs. Section 4 presents our approximation algorithm for general graphs, while in Section 5 we provide an improved version for the class of treewidth-bounded graphs. Section 6 contains the hardness results for the problems tackled in the paper.

As far as we know, the problems considered in this paper have not been previously studied. Recently in an independent effort Frederickson and Solis-Oba [FS96] considered the problem of increasing the weight of the minimum spanning tree in a graph subject to a budget constraint where the cost functions are assumed to be linear in the weight increase. Berman [Be92] considers the problem of shortening edges in a given tree to minimize its shortest path tree weight. In contrast to the work presented here, this problem is shown to be solvable in strongly polynomial time. Phillips [Ph93] studies the problem of finding an optimal strategy for reducing the capacity of the network so that the residual capacity in the modified network is minimized. The problems studied here and in [Ph93, Be92] can be broadly classified as types of bicriteria problems. Recently, there has been substantial work on finding efficient approximation algorithms for a variety of bicriteria problems (see [KP95, Ha92, MRS$^+$95, RR+93, Ra94, Wa92, ZPD94] and the references therein).

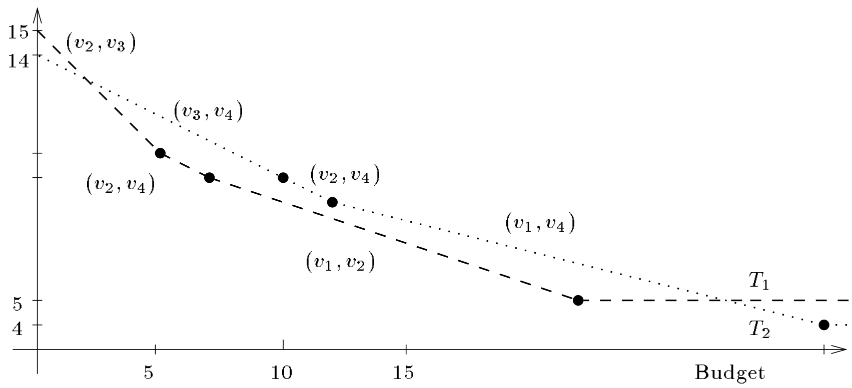


Figure 2: Remaining weight of the trees T_1 and T_2 as a function of the budget.

3 Structure of an Optimal Solution

In this section we comment on the structure of optimal solutions to the BMST-problem for linear reduction costs on the edges, i.e. $C_e(t) = C_e \cdot t$ for all $e \in E$ and constants C_e. We also look at special cases of the problem that can be solved in polynomial time.

[5] The length of the spanning tree produced is at most $(1+1/\gamma)$ times that of an optimal tree plus an additive constant of ε that can be made arbitrarily close to zero.

First, suppose that the given budget B is zero. Then BMST reduces to the well known minimum spanning tree problem (with length function $L(e)$), and is known to be optimally solvable by classical algorithms (e.g. Prim's algorithm [CLR]). Similarly, if $B = +\infty$ (i.e., there is no bound on the cost of upgrading the network), the BMST problem again reduces to the minimum spanning tree problem but this time with edge-lengths given by $L_{\min}$.

Optimal solutions to BMST also exhibit some structure in the general case (i.e., $B \notin \{0, +\infty\}$). Any (feasible) reduction r induces a tree in a natural way, namely a minimum spanning tree T_r in the modified graph $r(G)$. Observe that the quality of the solution produced via the reduction r depends solely on the weight of T_r, so all the cost incurred in upgrading edges not in T_r is wasted. Moreover, for any *fixed* tree T in G, the Greedy-strategy that successively reduces a cheapest available edge is an optimal reduction strategy. Thus, if we already knew a minimum spanning tree T_{r^*} corresponding to an optimal reduction r^*, we could solve BMST quite easily.

This observation also suggests a very simple exponential time algorithm for solving BMST: Enumerate all spanning trees in G, apply the Greedy strategy to each of them and then select the best solution. Unfortunately, a graph G with n nodes can have n^{n-2} different spanning trees.

We now discuss the *sensitivity* of optimal reduction strategies to changes in the given budget B. If we fix a spanning tree and plot the weight of that tree as a function of the money spent on it in a Greedy manner, we see that each piece corresponds to a budget range where one particular edge e is shortened. Thus it is easy to see that the piece has slope $-1/C_e$.

Figure 2 on the page before shows the plots corresponding to the tree T_1 consisting of the edges (v_2, v_3), (v_2, v_4), (v_1, v_2) and the tree T_2 consisting of the edges (v_3, v_4), (v_2, v_4) and (v_1, v_4) taken from the example graph of Figure 1 on page 2. As can be seen from Figure 2, the plots for different trees can cross each other multiple times. If we plot the weights of all spanning trees on the same set of axes, the lower envelope gives the optimal remaining weight per budget. It is easy to see that the lower envelope can have an exponential number of linear pieces.

4 Approximation Algorithms for BMST on General Graphs

In this section, we present our approximation algorithm for the BMST problem. As mentioned earlier, the approximation algorithm extends easily to a broad class of network improvement problems where the objective to be minimized is the total cost of a connected subnetwork (e.g. budget constrained minimum Steiner tree problem).

We first give an informal description of how the algorithm works. The main procedure uses a parametric search. In this search, the algorithm tries to find a good compromise between weighing the total length and the corresponding reduction cost of a tree in general. To this end, the algorithm performs a binary search with parameter K on the interval $\mathcal{I} := [\frac{1}{\gamma}(n-1)\min_{e\in E} L_{\min}(e), \frac{1}{\gamma}(n-1)\max_{e\in E} L(e)]$. Note that if $\mathrm{MST}(G, L - r^*)$ denotes the total weight of a minimum spanning tree after an optimal reduction r^* then $\frac{1}{\gamma}\,\mathrm{MST}(G, L - r^*) \in \mathcal{I}$.

For each $K \in \mathcal{I}$, which is probed with the help of a test procedure during the search, the algorithm first calculates a coarse heuristic measure that indicates how important it is to shorten an edge. Then, for each edge e in the graph the blend of its length and the reduction cost is refined by using the information in the cost function C_e. After calculating such *compound costs* for the edges, we compute a minimum spanning tree with respect to these costs. The algorithm stops when a good blend has been found, meaning in this context that there exists a tree of total compound cost that is small compared to the current parameter K.

For large values of K the reduction costs on the edges are weighted more than their lengths and the algorithm will tend to reduce the edge lengths only by a small amount, resulting in low overall reduction costs and more or less heavy trees. Also, since K is large, the test on the compound cost of the minimum spanning tree computed will succeed. The algorithm now tries to reduce K as much as possible and find a minimum $K \in \mathcal{I}$ such that it can successfully compute a light compound cost spanning tree.

Our approximation algorithm for BMST is shown in Figure 3 on the next page. This algorithm uses the test procedure given in Figure 4.

Procedure Heuristic-BMST(γ, ε)

1 Perform a binary search on the interval $\mathcal{I} \in [\frac{(n-1)\min_{e\in E} L_{\min}(e)}{\gamma}, \frac{(n-1)\max_{e\in E} L(e)}{\gamma}]$ with a spacing of ε to find the minimum value K' such that Test-Blend(K') returns "Yes".

2 Let T' be the tree generated by Test-Blend(K') and let t_e $(e \in T')$ be the corresponding "fine tuned" blend parameters.

Define the reduction r by $r(e) := 0$ if e is not included in T and by $r(e) := t_e$ otherwise.

4 **return** r and T.

Figure 3: Main Procedure for the approximation of BMST.

Procedure Test-Blend(K)

1 **Comment:** This procedure tries to estimate whether in the current blend of lengths and reduction costs, the costs are weighted strongly enough (i.e. K large enough) resulting in a low cost reduction. For this purpose, it uses the heuristic measure computed in Step 2.

2 **for** each edge e let

$h_K(e) = \min_{t\in[0,L(e)-L_{\min}(e)]} \left(L(e) - t + \frac{K}{B}C_e(t)\right)$.

Also, let t_e be the value of t which achieves the value $h_K(e)$.

3 Compute a minimum spanning tree T in G using the weight $h_K(e)$ for each $e \in E$.

Let $h_K(T)$ denote the cost of this spanning tree.

4 **if** $h_K(T) \leq (1+\gamma)K$ **then return** "Yes" **else return** "No".

Figure 4: Test procedure used for the approximation of BMST.

4.1 Correctness and Performance Guarantee.

The performance guarantee provided by the algorithm Heuristic-BMST is summarized in the following theorem.

Theorem 4.1 For any fixed $\gamma, \varepsilon > 0$, Heuristic-BMST is an approximation algorithm for BMST that finds a solution whose length is at most $(1 + \frac{1}{\gamma})$ times that of a minimum length spanning tree plus an additive constant of at most ε, and the total cost of the improvement is at most $(1+\gamma)$ times the budget B.

The proof of Theorem 4.1 relies mainly on the following lemma, which ensures that the binary search in the main procedure works correctly. In stating this lemma, we use the notation introduced in the two procedures (Heuristic-BMST and Test-Blend) described above.

Lemma 4.2 Define F on $\mathbb{R}_{>0}$ by $F(K) := \frac{\mathrm{MST}(G,h_K)}{K}$. Then F is monotonically nonincreasing on $\mathbb{R}_{>0}$.

Proof: Let $K^{(1)}$ and $K^{(2)}$ be two positive numbers such that $K^{(1)} < K^{(2)}$. For $i = 1, 2$ let $T^{(i)}$ be a minimum spanning tree in G under the cost function $h_{K^{(i)}}$. Then

$$h_{K^{(i)}}(T^{(i)}) = \sum_{e\in T^{(i)}} h_{K^{(i)}}(e) = \underbrace{\sum_{e\in T^{(i)}} (L(e) - t_e^{(i)})}_{=:\, L^{(i)}} + \frac{K^{(i)}}{B} \underbrace{\sum_{e\in T^{(i)}} C_e(t_e^{(i)})}_{=:\, C^{(i)}} =: L^{(i)} + \frac{K^{(i)}}{B} C^{(i)}. \tag{2}$$

Here $t_e^{(i)}$ are the values chosen in Step 2 of Test-Blend which minimize $L(e) - t + \frac{K^{(i)}}{B}C_e(t)$ on the interval $[0, L(e) - L_{\min}(e)]$. By dividing the last equation by $K^{(i)}$ we obtain that

$$F(K^{(i)}) = \frac{L^{(i)}}{K^{(i)}} + \frac{C^{(i)}}{B} \text{ for } i \in \{1,2\}. \tag{3}$$

In the next step we find an upper bound for $F(K^{(2)})$. To this end, we estimate the weight of each edge in $T^{(1)}$ under the cost function $h_{K^{(2)}}$. Let $e \in T^{(1)}$ be an arbitrary edge. Then by the choice of $t_e^{(2)}$ in Step 2 of

Test-Blend we have that

$$\begin{aligned} h_{K^{(2)}}(e) &= L(e) - t_e^{(2)} + \frac{K^{(2)}}{B} C_e(t_e^{(2)}) \\ &= \min_{t \in [0, L(e) - L_{\min}(e)]} \left(L(e) - t + \frac{K^{(2)}}{B} C_e(t) \right) \\ &\leq L(e) - t_e^{(1)} + \frac{K^{(2)}}{B} C_e(t_e^{(1)}). \end{aligned} \tag{4}$$

Summing up the inequalities in (4) over all $e \in T^{(1)}$, we obtain:

$$h_{K^{(2)}}(T^{(1)}) \leq L^{(1)} + \frac{K^{(2)}}{B} C^{(1)}. \tag{5}$$

Dividing (5) by $K^{(2)}$ and using the fact that $h_{K^{(2)}}(T^{(2)}) \leq h_{K^{(2)}}(T^{(1)})$ this results in

$$F(K^{(2)}) \leq \frac{L^{(1)}}{K^{(2)}} + \frac{C^{(1)}}{B} < \frac{L^{(1)}}{K^{(1)}} + \frac{C^{(1)}}{B} = F(K^{(1)}). \tag{6}$$

The strict inequality in the chain above stems from the fact that $K^{(1)} < K^{(2)}$. This completes the proof of the lemma. □

Corollary 4.3 If the procedure Test-Blend returns "Yes" for some $K' > 0$ then it also returns "Yes" for all $K > K'$. Thus, the binary search in Heuristic-BMST works correctly.

Proof: Let T' be a minimum spanning tree with respect to $h_{K'}$. Then, since Test-Blend(K') returns "Yes" we have that $h_{K'}(T') \leq (1+\gamma)K'$; i.e. $F(K') \leq (1+\gamma)$. Thus it follows by Lemma 4.2 on the preceding page that $F(K) \leq (1+\gamma)$ for all $K > K'$. Since $F(K) = \frac{\mathrm{MST}(G, h_K)}{K}$, this is equivalent to saying that $\mathrm{MST}(G, h_K) \leq (1+\gamma)K$ for all $K > K'$. □

Proof of Theorem 4.1: Let r^* be an optimal feasible reduction and let T^* be a minimum spanning tree in G with respect to the weight function $L - r^*$. For the sake of shorter notation let $L^* := (L - r^*)(T^*)$ be its total weight in the graph with the edge lengths resulting from the optimal reduction r^*.

We now show Test-Blend would return "Yes" if called with the value $\tilde{K}$ which is the smallest value in the ε-spacing of $\mathcal{I} = [\frac{1}{\gamma}(n-1)\min_{e \in E} L_{\min}(e), \frac{1}{\gamma}(n-1)\max_{e \in E} L(e)]$ satisfying $\tilde{K} \geq L^*/\gamma$. Thus, $\tilde{K}$ is some rational number satisfying

$$\tilde{K} = L^*/\gamma + \varepsilon', \text{ where } 0 \leq \varepsilon' < \varepsilon. \tag{7}$$

For each edge $e \in T^*$ we can estimate the weight $h_{K^*}(e)$ similar to inequality (4) in the proof of Lemma 4.2. This way, we see that the weight of T^* under $h_{\tilde{K}}$ is no more than $L^* + \frac{\tilde{K}}{B} B$. Consequently, the minimum spanning tree with respect to $h_{\tilde{K}}$ that would be found by the procedure during the call has $h_{\tilde{K}}$-weight at most

$$L^* + \tilde{K} \stackrel{(7)}{=} \gamma(\tilde{K} - \varepsilon') + \tilde{K} \leq (1+\gamma)\tilde{K}.$$

Hence, the test in Step 4 of Test-Blend would be successful and the procedure would return "Yes". Since we know by Corollary 4.3 that the binary search correctly locates a minimum value K', this now implies that the minimum value K' must satisfy $K' \leq \tilde{K} = L^*/\gamma + \varepsilon'$. Let T' be the minimum spanning tree found by Test-Blend(K'). Since $K', B \geq 0$ and $C_e(t) \geq 0$ for all t, we have:

$$h_{K'}(T') = \sum_{e \in T'} (L(e) - t'_e) + \frac{K'}{B} \sum_{e \in T'} C_e(t'_e) \geq \sum_{e \in T'} (L(e) - t'_e). \tag{8}$$

Here again the numbers t'_e are the values of t chosen in Step 2 of the test procedure. For the reduction r which is calculated in Step 3 of Heuristic-BMST it now follows from (8) that

$$\mathrm{MST}(G, L - r) \leq (L - r)(T') \leq h_{K'}(T'). \tag{9}$$

Moreover,

$$h_{K'}(T_{K'}) \le h_{K'}(T^*) \le L^* + \frac{K'}{B}B \le L^* + \tilde{K} \overset{(7)}{=} L^* + \frac{L^*}{\gamma} + \varepsilon' \le (1+\frac{1}{\gamma})L^* + \varepsilon = (1+\frac{1}{\gamma})\,\mathrm{MST}(G, L - r^*) + \varepsilon.$$

Using this result in (9), we get $\mathrm{MST}(G, L-r) \le (1+\frac{1}{\gamma})\,\mathrm{MST}(G, L-r^*) + \varepsilon$, which proves the claimed performance of the algorithm with respect to the weight of an MST in the graph after applying the reduction r.

We now estimate the cost of the reduction r found by our heuristic. Note that the cost of r is exactly $\sum_{e \in T'} C(t_e)$. We have

$$\frac{K'}{B}\sum_{e\in T'} C(t'_e) \le \sum_{e \in T'} (L(e) - t'_e + \frac{K'}{B}C(t'_e)) = h_{K'}(T') \le (1+\gamma)K'.$$

Dividing the last chain of inequalities by $\frac{K'}{B}$ yields that the budget B is violated by a factor of at most $(1+\gamma)$ as claimed in the theorem. □

4.2 Running Time

We now show that the algorithm can be implemented to run in polynomial time for a broad class of reduction cost functions C_e on the edges of the graph. Let $L_{\max} = \max_{e\in E} L(e)$. Then the total number of calls to Procedure Test-Blend is in $\mathcal{O}(\log(\frac{nL_{\max}}{\gamma\varepsilon}))$. Since γ and ε are fixed, the test procedure is called only a polynomial number of times. Thus, to prove that the overall running time of the algorithm is polynomial, it suffices to show that each execution of Test-Blend can be completed in polynomial time. Here, the only fact to show is that we can minimize the function $f_e(t) := L(e) - t + \frac{K}{B}C_e(t)$ on the compact interval $\mathcal{I}' := [0, L(e) - L_{\min}(e)]$ in Step 2 of the procedure in polynomial time. The rest of the procedure consists of computing a minimum spanning tree which can be done in $O(n + m\log\beta(m,n))$ time using the algorithm of Gabow et. al. [GGS86], where $\beta(m,n) = \min\{i \mid \log^{(i)} n \le m/n\}$.

Consider the execution of Test-Blend for a given value of K. Observe that in Step 2 the number $L(e)$ is an additive constant and $\frac{K}{B}$ is a constant factor. Thus, the constrained minimization of f_e can be done easily for for the following sample classes of functions C_e:

1. Linear functions, i.e. $C_e(t) = C_e \cdot t$ for a constant C_e: Then f_e is a linear function in t and the minimum is attained at one of the endpoints of $\mathcal{I}'$. Minimizing f_e can be done in constant time. Thus, the total running time of the heuristic is $\mathcal{O}(\log(\frac{nL_{\max}}{\gamma\varepsilon})(n + m\log\beta(m,n)))$.

2. Concave functions: Let $\Delta(e) := L(e) - L_{\min}(e)$. Then, for any $0 < \lambda < 1$ we have by the concavity of C_e (which implies the concavity of f_e):

$$f_e(\lambda \cdot 0 + (1-\lambda)\Delta(e)) \ge \lambda f_e(0) + (1-\lambda) f_e(\Delta(e)) \ge \min\{f_e(0), f_e(\Delta(e))\}.$$

   Thus, the minimum of f_e is again either at 0 or at $L(e) - L_{\min}(e)$.

3. Differentiable convex functions where we can find a root of the equation $C'_e(t) = \frac{B}{K}$ explicitly.

4. Functions that are piecewise of one of the types described above. Observe that the number of pieces is polynomial in the input size.

For the first three classes of functions that are mentioned above the total effort for our algorithm consists essentially of $\mathcal{O}(\log(\frac{nL_{\max}}{\gamma\varepsilon}))$ minimum spanning tree computations, which does not only result in an overall polynomial time but also in a complexity that is feasible in practice.

4.3 Notes on the Algorithm

It should be noted here that our Algorithm Heuristic-BMST can be modified easily to handle the case when the reduction is required to be either integer valued or to satisfy $r(e) \in \{0, L(e) - L_{\min}(e)\}$ for all $e \in E$. In this case, Step 2 of Test-Blend is modified in such a way that the minimization is carried out only over the integers in $[0, L(e) - L_{\min}(e)]$ or on the two element set $\{0, L(e) - L_{\min}(e)\}$ respectively. Due to lack of space we omit the details.

Integer valued reductions are helpful to model discrete steps of improvement, e.g. the addition of a number of communication links parallel to already existing ones in the network. Reductions that take values only from $\{0, L(e) - L_{\min}(e)\}$ can be used to model the insertion of alternative edges to the graph G, with the reduction of the edge e corresponding to the construction of a new edge e' parallel to e with length $L_{\min}(e)$.

So far, we have assumed that the function $f_e(t) = L(e) - t + \frac{K}{B} C_e(t)$ can be minimized *exactly*. This indeed is not necessary to obtain an approximation algorithm with a constant factor approximation for BMST. In fact, one can show that if in Step 2 of procedure Test-Blend we find a value t' satisfying

$$f_e(t') \le \alpha \cdot \min_{t \in [0, L(e) - L_{\min}(e)]} f_e(t)$$

for some $\alpha \ge 1$ and modify Step 4 to check whether the compound weight of the tree is at most $\alpha^2(1+\gamma)K$, this will lead to a polynomial time algorithm which produces a reduction of cost at most $\alpha^2(1+\gamma)B$ and a corresponding MST of total length at most $\alpha(1 + 1/\gamma)$ times that of an optimal tree plus an additive constant of ε.

5 Improved Approximation Ratios for Treewidth Bounded Graphs and Linear Reduction Costs

In this section we will show how to obtain an improved algorithm for the class of treewidth bounded graphs under the additional assumption that the reduction costs on the edges are *linear*. The basic idea behind the algorithm in this section is to reduce the problem of improving the tree to some appropriately chosen bicriteria problem. To this end we recall the following result from [MRS+95]:

Theorem 5.1 1. There is a polynomial-time algorithm that, given an undirected graph G on n nodes with two nonnegative integral costs E and F on its edges, a bound $\mathcal{E}$, and a fixed $\gamma > 0$, constructs a spanning tree of G of total E-cost at most $(1+\gamma)\mathcal{E}$ and of total F-cost at most $(1 + 1/\gamma)$ times that of the minimum-F-cost of any spanning tree with total E-cost at most $\mathcal{E}$.

2. For the class of treewidth-bounded graphs, there is a polynomial time algorithm that returns a spanning tree of total E-cost at most $\mathcal{E}$ and and of total F-cost at most $(1+\varepsilon)$ times that of any spanning tree with total E-cost at most $\mathcal{E}$. □

We will use the second part of the theorem to obtain an improved approximation. We note here that using first part of Theorem 5.1 instead, we could also construct a $((1+\frac{1}{\gamma}), (1+\xi)(1+\gamma))$ approximation algorithm for BMST on general graphs for any fixed $\xi > 0$, if we restrict ourselves to linear reduction costs. Since our algorithm from the last section already gives us a $(1+\frac{1}{\gamma}, 1+\gamma)$ performance for far more general classes of cost functions this is not as interesting. Also, our approximation algorithm from Section 4 does not need any additional space, while the construction presented below transforms the original graph into a graph having more nodes and edges.

We now describe our improved approximation algorithm for linear reduction costs on treewidth-bounded graphs. Due to lack of space we only sketch the details. First we transform the original graph to another graph that can be fed into the algorithm from Theorem 5.1. To this end, we replace each edge $e = (u, v)$ of the original graph by a certain subgraph in such a way that the treewidth does not increase. The transformation procedure is shown in Figure 5 on the next page and an example of a transformation is displayed in Figure 6.

Let G be the original graph and G' be the graph obtained as a result of the transformation. Also, let $\mathrm{tw}(G)$ and $\mathrm{tw}(G')$ denote the treewidths of G and G' respectively. We have the following observation.

Observation 5.2 Whenever $\mathrm{tw}(G) \ge 3$, we have that $\mathrm{tw}(G) = \mathrm{tw}(G')$. □

5.1 Correctness and Performance Guarantee

Let r^* denote the optimal reduction involving a cost of at most B and let T^* be a minimum spanning tree in $r^*(G)$ and let $L^* := \mathrm{MST}(G, L - r^*)$ be its weight in the modified graph. Also, let T' be a tree in G' with minimum total F-cost $F' := F(T')$ among all trees in G' that have E-cost at most ω. The performance guarantee provided by the algorithm Heuristic-TW-BMST shown in Figure 7 on the next page is summarized in the following theorem:

Procedure Transform(ξ)

1. **for** each edge $e = (u, v)$ in the graph let b_e be chosen so that $(1+\xi)^{b_e} \leq L(e) - L_{\min}(e) \leq (1+\xi)^{b_e+1}$.
2. Add $b_e + 2$ new vertices r_k, $k = -1, 0, \ldots, b_e$, which are joined together in a simple cycle.
3. **for all** $k, -1 \leq k \leq b_e$, join r_k to both u and v.
4. For $k \geq 0$, the edge (u, r_k) has E-cost $E(u, r_k) := L(e) - (1+\xi)^k$ and F-cost $(1+\xi)^k C_e$, while the edge (u, r_{-1}) has E-cost $L(e)$ and F-cost 0. All the edges (r_k, v) and (r_k, r_{k+1}) have their E-cost and F-cost set to zero.
   (An example of the above transformation for $b_e = 2$ can be seen in Figure 6).

Figure 5: Procedure used to transform G to G' in the approximation of BMST on treewidth bounded graphs.

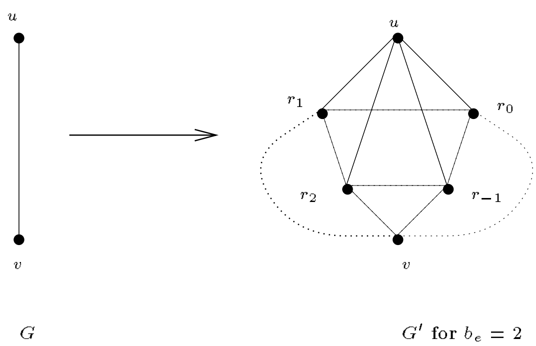


Figure 6: Example for the transformation on treewidth bounded graphs.

Theorem 5.3 For the class of treewidth bounded graphs and linear reduction costs the following statement holds: For all fixed $\varepsilon, \xi > 0$, Heuristic-TW-BMST is a polynomial time $((1+\varepsilon), (1+\xi))$-approximation algorithm for BMST.

Proof: Let us first understand the relationship between B' and B and that between F' and L^*. Consider the tree T^* in G. We can define a tree T'' in G' in the following way: For an edge $e = (u, v) \in T^*$ that is reduced by $r^*(e)$ we select an edge (u, r_j) in G' of E-cost $L(e) - b(e)$, where $b(e)$ is selected in such a way that $\frac{b(e)}{(1+\xi)} \leq r^*(e) \leq b(e)$. We also select the edge (r_j, v) to belong to T''. Observe that the edge (u, r_j) selected in the above fashion has its length reduced by at most $(1+\xi) \cdot r^*(e)$ and at least by $r^*(e)$. Using this fact the following claim can be proven.
Claim: The F-cost of the tree T'' is at most $(1+\xi)B$. The total E-cost of the tree T'' in G' is at most L^*. □

Hence we have demonstrated a witness tree T'' such that if the bound on the E-length is L^*, then the F-cost of this tree is bounded from above by $B' := (1+\xi)B$. Consequently, the *minimum* F-cost tree T' in G' (under the constraint that the E-cost does not exceed L^*) will have cost at most B'. Thus the binary search will terminate with a value $L' \leq L^*$.

Procedure Heuristic-TW-BMST(ξ, ε)

1. Call Transform(ξ) to obtain a new graph G'.
2. Let $B' := (1+\xi) \cdot B$
3. Use binary search to find the smallest integer $L' \in [(n-1)(\min_{e \in E} L_{\min}(e)), (n-1)(\max_{e \in E} L(e))$ such that the algorithm referred to in Part 2 of Theorem 5.1 called with the parameters L' for the E-cost bound $\mathcal{E}$ and $\varepsilon > 0$ returns a tree of F-cost at most B'.
4. Let T' be the tree generated by the algorithm from Theorem 5.1.
5. For each edge $e = (u, v)$, define the reduction r on e by $r(e) := 0$ if (u, r_{-1}) is included in T' and otherwise by $r(e) := (1+\xi)^k$, where $0 \leq k \leq b_e$ is the minimum value such that (u, r_k) is included in T'.
6. **return** r.

Figure 7: Main Procedure for the approximation of BMST on treewidth bounded graphs.

Specifically, for our algorithm sketched above, the total weight $\mathrm{MST}(G, L - r)$, where r is the reduction returned by the heuristic, is then bounded from above by $(1+\varepsilon)L' \leq (1+\varepsilon)L^*$. Moreover, the cost of the reduction r which is defined in Step 5 of Heuristic-TW-BMST is no more than the F-cost of the tree T', which is found with the help of the algorithm from Theorem 5.1.

Since we know that the cost of this tree is bounded above by $B' = (1+\xi) \cdot B$ by the fact that the binary search has indeed terminated with some L', the claimed performance guarantee with respect to the budget follows. □

5.2 Running Time.

We now show that the algorithm can be implemented to run in polynomial time. For this, observe that for a fixed value of $\xi > 0$, the number of edges added (i.e., the value of b_e) is polynomial in the size of the input. This proves that the procedure Transform runs in time $\mathcal{O}(m \log M)$ where $M = \sum_{e \in E} L(e)$. Next, observe that the binary search in the main procedure can be done in polynomial time. Thus the algorithm can be executed in polynomial time.

6 Hardness Results

In this section we will prove the BMST problem to be hard even for very restricted class of graphs and the most simple reduction cost functions.

Theorem 6.1 BMST is NP-hard, even when restricted to series-parallel graphs and even when all the reduction cost functions C_e are linear, i.e. $C_e(t) = C_e \cdot t$ for all $e \in E$.

Proof: We use a reduction from Continuous Multiple Choice Knapsack which is known to be a NP-complete problem (c.f. [GJ79], MP11). An instance of CMC-Knapsack is given by a finite set U of n items, a size $s(u)$ and value $v(u)$ for each item, a partition $U_1 \cup \cdots \cup U_k$ of U into disjoint sets and two integers S and K. The question is, whether there is a choice of a unique element $u_i \in U_i$, for each $1 \leq i \leq k$, and an assignment of rational numbers $r_i, 0 \leq r_i \leq 1$ to these elements such that $\sum_{i=1}^{k} r_i s(u_i) \leq S$ and $\sum_{i=1}^{k} r_i v(u_i) \geq K$.

Given an instance of CMC-Knapsack we construct a graph $G = (V, E)$ in the following way: We let $V = U \cup \{X, T, T_1, \ldots, T_k\}$, $E := E_1 \cup E_2 \cup E_3$ with $E_1 := \{(X, u) : u \in U\}$, $E_2 := \{(u, T_i) : u \in U_i, i = 1, \ldots, k\}$ and $E_3 := \{(T_i, T) : i = 1, \ldots, k\}$. The graph constructed this way is obviously series-parallel with terminals X and T.

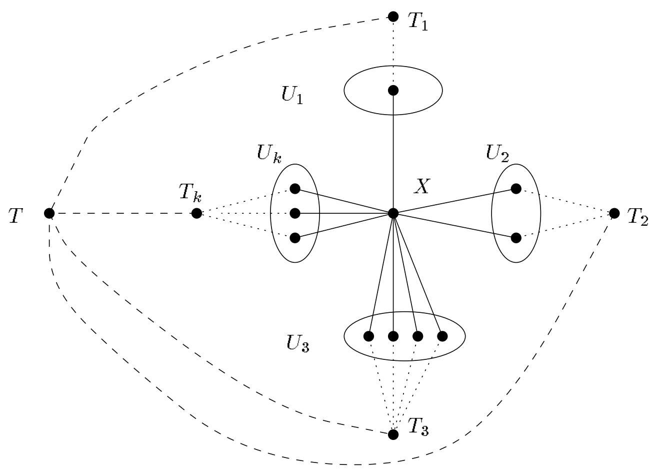


Figure 8: Graph used in the reduction from Continuous Multiple Choice Knapsack.

Define $D := \max\{v(u) : u \in U\}$. For each edge $(x, u) \in E_1$, let $L(x, u) := D, L_{\min}(x, u) := D - v(u), C(x, u) := s(u)/v(u)$. For all edges $e \in E_2$ we let $L(e) := L_{\min}(e) := C(e) := 0$, and for all edges $e \in E_3$ we define $L(e) := L_{\min}(e) := 3D$ and $C(e) := 0$. Set the bound B on the total cost to be S.

The graph is shown in Figure 8 on the preceding page. The dotted edges are of weight 0 while the dashed ones have weight $3D$. Any MST in G has weight $kD + 3D$.

By the construction, any feasible reduction can only reduce the length of the edges in E_1. Assume that r is a feasible reduction. Observe that the MST in G with edge lengths given by $(L - r)$ will always include *all* edges from E_2 (which are of weight 0) and *exactly one* edge from E_3, regardless of which edges from E_1 are affected by the reduction. Observe also that for any fixed $i \in \{1, \dots, k\}$, any MST in the modified graph will contain *exactly one* of the edges of the form (X, u'), where $u' \in U_i$. Consequently, reducing the length of *more than one* edge (X, u') with $u' \in U_i$ will *not* improve the quality of the solution, but cost money from the budget B. We thus have:

Observation: If r is a feasible reduction for the instance of BMST defined above and the weight of an MST in the modified graph is Y, then there is always a feasible reduction r', which for each $i \in \{1, \dots, k\}$ reduces at most one of the edges (X, u), $u \in U_i$ and the weight of an MST with respect to $(L - r')$ is also equal to Y. □

Let r be any reduction as defined in the above observation and for $i = 1, \dots, k$ let $e_i = (X, u_i)$ be the unique edge from x to U_i affected by the reduction. The weight of an MST in with respect to $(L - r)$ is then given by

$$3D + \sum_{i=1}^{k} (L(e_i) - r(e_i)) = 3D + k \cdot D - \sum_{i=1}^{k} r(e_i). \tag{10}$$

The cost of reduction r is given by

$$\sum_{i=1}^{k} r(e_i) C(e_i) = \sum_{i=1}^{k} r(e_i) \cdot \frac{s(u_i)}{v(u_i)} = \sum_{i=1}^{k} \frac{r(e_i)}{v(u_i)} \cdot s(u_i) \le B. \tag{11}$$

We now prove the following: There is a feasible reduction r such that $\mathrm{MST}(G, L - r) \le (3 + k)D - K$, if and only if there exists a choice of a unique element $u_i \in U_i$, $1 \le i \le k$ and an assignment of rational numbers $r_i, 0 \le r_i \le 1$ to these elements such that $\sum_{i=1}^{k} r_i s(u_i) \le B$ and $\sum_{i=1}^{k} r_i v(u_i) \ge K$.

First, assume that there is a feasible reduction r such that $\mathrm{MST}(G, L - r) \le (3 + k)D - K$. Without loss of generality, we can assume that r has the properties as stated in the above observation. Then for $i = 1, \dots, k$ there is at most one edge $e_i = (X, u)$ with $u \in U_i$ such that $r(e_i) > 0$. If there is such an edge e_i, we define

$$r_i := \frac{r(e_i)}{v(u_i)} = \frac{r(e_i)}{L(e_i) - L_{\min}(e_i)} \tag{12}$$

and let $u_i := u$. If for all edges (X, u) with $u \in U_i$ we have $r(e_i) = 0$, we simply let $r_i := 0$ and choose $u_i \in U$ arbitrarily. It follows readily from the definition and the feasibility of the reduction r that $r_i \in [0, 1]$. Moreover, using Equation (11) we see that $\sum_{i=1}^{n} r_i s(u_i) \le B \le B_1$. Using equation (10) and the fact that the weight $\mathrm{MST}(G, L - r)$ is no more than $(3 + k)D - K$ we obtain that

$$\sum_{i=1}^{n} r_i v(u_i) = \sum_{i=1}^{n} \frac{r(e_i)}{v(u_i)} \cdot v(u_i) = \sum_{i=1}^{n} r(e_i) \ge K.$$

Conversely, if we can pick unique elements u_i from the sets U_i and find rational numbers $r_i \in [0, 1]$ such that $\sum_{i=1}^{n} r_i s(u_i) \le B$ and $\sum_{i=1}^{n} r_i v(u_i) \ge K$. We can define a reduction r by $r(X, u_i) := r_i v(u_i) = r_i(L(x, u_i) - L_{\min}(x, u_i))$ for $i = 1, \dots, k$ and $r(e) := 0$ for all other edges. It follows that r is indeed feasible, and using equation (10) we see that the MST in the modified graph is no heavier than $(3 + k)D - K$. □

References

[Be92] O. Berman, "Improving The Location of Minisum Facilities Through Network Modification," *Annals of Operations Research*, 40(1992), pp. 1–16.

[CK+92] J. Cong, A. B. Kahng, G. Robins, M. Sarafzadeh and C. K. Wong, "Provably Good Performance Driven Global Routing," *IEEE Transactions on Computer Aided Design*, 11(6), 1992, pp. 739-752.

[CR91] J. P. Cohoon and L. J. Randall, "Critical Net Routing," *IEEE Intern. Conf. on Computer Design,* 1991, pp. 174-177.

[CLR] T. H. Cormen, C. E. Leiserson and R. L. Rivest, *Introduction to Algorithms*, McGraw-Hill Book Co., Cambridge, MA, 1990.

[Cu85] W. Cunningham, "Optimal Attack and Reinforcement of a Network," *J. ACM*, 32(3), 1985, pp. 549-561.

[FS96] G.N. Frederickson and R. Solis-Oba, "Increasing the Weight of Minimum Spanning Trees", *Proceedings of the Sixth Annual ACM-SIAM SODA'96*, March 1996.

[GGS86] H. N. Gabow, Z. Galil, T. H. Spencer and R. E. Tarjan, "Efficient Algorithms for Finding Minimum Spanning Trees in Undirected and Directed Graphs," *Combinatorica*, 6 (1986), pp. 109–122.

[GJ79] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman and Co., San Francisco, CA, 1979.

[GW] M. X. Goemans, and D. P. Williamson, "A General Approximation Technique for Constrained Forest Problems", *Proceedings of the Third Annual ACM-SIAM SODA'92*, Jan. 1992, pp. 307-316.

[GG+94] M. X. Goemans, A. V. Goldberg, S. Plotkin, D. B. Shmoys, E. Tardos and D. P. Williamson, "Improved Approximation Algorithms for Network Design Problems," *Proceedings of the Fifth Annual ACM-SIAM SODA'94*, Jan. 1994, pp. 223-232.

[Ha92] R. Hassin, "Approximation Schemes for the Restricted Shortest Path Problem," *Math. of OR*, 17(1), 1992, pp. 36-42.

[KP95] D. Karger and S. Plotkin, "Adding Multiple Cost Constraints to Combinatorial Optimization Problems, with Applications to Multicommodity Flows," *Proc. 27th Annual ACM Symp. on Theory of Computing* (STOC'95), May 1995, pp. 18–25.

[KJ83] B. Kadaba and J. Jaffe, "Routing to Multiple Destinations in Computer Networks," *IEEE Trans. on Communication*, Vol. COM-31, Mar. 1983, pp. 343-351.

[KPP92] V. P. Kompella, J. C. Pasquale and G. C. Polyzos, "Multicasting for Multimedia Applications," *Proc. of IEEE INFOCOM '92*, May 1992.

[Komp] V. P. Kompella, J. C. Pasquale and G. C. Polyzos, "Two Distributed Algorithms for the Constrained Steiner Tree Problem," Technical Report CAL-1005-92, Computer Systems Laboratory, University of California, San Diego, Oct. 1992.

[KPP93] V. P. Kompella, J. C. Pasquale and G. C. Polyzos, "Multicast Routing for Multimedia Communication," *IEEE/ACM Transactions on Networking*, 1993, pp. 286–292.

[LY93] C. Lund and M. Yannakakis, "On the Hardness of Approximating Minimization Problems," *Proc., 25th Annual ACM Symp. on Theory of Computing* (STOC'93), May 1993, pp. 288–293.

[MRS+95] M. V. Marathe, R. Ravi, S. Sundaram, S. S. Ravi, D. J. Rosenkrantz, and H. B. Hunt III, "Bicriteria Network Design Problems", *Proc. ICALP'95*, July 1995.

[Ph93] C. Phillips, "The Network Inhibition Problem," *Proc. 25th Annual ACM STOC'93*, May 1993, pp. 288–293.

[RR+93] R. Ravi, M. V. Marathe, S. S. Ravi, D. J. Rosenkrantz and H. B. Hunt III, "Many Birds with one Stone: Multi–objective Approximation Algorithms," *Proc. 25th Annual ACM STOC'93*, May 1993, pp. 438–447.

[Ra94] R. Ravi, "Rapid Rumor Ramification: Approximating the Minimum Broadcast Time," *Proceedings of the 35th Annual FOCS'94*, Nov. 1994, pp. 202–213.

[Wa92] A. Warburton, "Approximation of Pareto optima in Multiple–Objective, Shortest Path Problems," *Oper. Res.*, Vol. 35, 1987, pp. 70–79.

[ZPD94] Q. Zhu, M. Parsa and W. Dai, "An Iterative Approach for Delay Bounded Minimum Steiner Tree Construction," Technical Report UCSC-CRL-94-39, University of California, Santa Cruz, Oct 1994.