The Steiner tree reoptimization problem with sharpened triangle inequality

Report

Author(s):
Böckenhauer, Hans-Joachim; Freiermuth, Karin; Hromkovič, Juraj; Mömke, Tobias; Sprock, Andreas; Steffen, Björn

The Steiner Tree Reoptimization Problem with Sharpened Triangle Inequality[*]

Hans-Joachim Böckenhauer, Karin Freiermuth, Juraj Hromkovič,
Tobias Mömke, Andreas Sprock, and Björn Steffen

Department of Computer Science, ETH Zurich, Switzerland,
{hjb,fkarin,jhromkov,tmoemke,asprock,bjoern.steffen}@inf.ethz.ch

Abstract. In this paper, we deal with several reoptimization variants of the Steiner tree problem in graphs obeying a sharpened β-triangle inequality. A reoptimization algorithm exploits the knowledge of an optimal solution to a problem instance for finding good solutions for a locally modified instance. We show that, in graphs satisfying a sharpened triangle inequality (and even in graphs where edge-costs are restricted to the values 1 and $1+\gamma$ for an arbitrary small $\gamma > 0$), Steiner tree reoptimization still is NP-hard for several different types of local modifications, and even APX-hard for some of them.

As for the upper bounds, for some local modifications, we design linear-time $(1/2 + \beta)$-approximation algorithms, and even polynomial-time approximation schemes, whereas for metric graphs ($\beta = 1$), none of these reoptimization variants is known to permit a PTAS. As a building block for some of these algorithms, we employ a 2β-approximation algorithm for the classical Steiner tree problem on such instances, which might be of independent interest since it improves over the previously best known ratio for any $\beta < 1/2 + \ln(3)/4 \approx 0.775$.

1 Introduction

The Steiner tree problem is a very prominent optimization problem with many practical applications, especially in network design, see for example [16,18]. Given a complete weighted graph $G = (V, E)$ with edge cost function c and a set $S \subseteq V$ of vertices called terminals, the Steiner tree problem consists of finding a minimum-cost connected subgraph of G containing all vertices from S. The problem is known to be APX-hard, even if the edge costs are restricted to 1 and 2 [4]. A minimum spanning tree on the terminal vertices is sufficient for achieving a 2-approximation (see, e. g., [18]), and the best currently known approximation ratio for the Steiner tree problem is $1 + \ln(3)/2 \approx 1.55$ for general edge costs and 1.28 for edge costs 1 and 2 [19].

In this paper, we analyze the hardness of even more restricted input instances. More precisely, we consider all instances where the edge costs are restricted to

---

the values 1 and $1 + \gamma$ for any $0 < \gamma \leq 1$. Also this restricted problem variant is known to be APX-hard [15]. In particular, restricting the edge costs in the described way also covers the class of Steiner tree problems where the edge-costs satisfy the sharpened β-triangle inequality, i. e., where the cost function c satisfies the condition $c(\{v_1, v_2\}) \leq \beta \cdot (c(\{v_1, v_3\}) + c(\{v_3, v_2\}))$, for some $1/2 \leq \beta < 1$ and for all vertices v_1, v_2, and v_3. The graphs satisfying a sharpened triangle inequality form a subclass of the class of all metric graphs. Intuitively speaking, for vertices that are points in the Euclidean plane, a parameter value $\beta < 1$ prevents that three vertices can be placed on the same line. For more details and motivation of the sharpened triangle inequality, see [9].

For analyzing how the transition from metric Steiner graphs ($\beta = 1$) to Steiner graphs with sharpened β-triangle inequality influences the computational hardness, we consider the question whether additional knowledge about the input is helpful for finding a good solution. More precisely, we consider the model of reoptimization algorithms which handles problems where an instance together with one of its optimal solutions is given and the problem is to find a good solution for a locally modified instance. This concept of reoptimization was mentioned for the first time in [20] in the context of postoptimality analysis for a scheduling problem. Since then, the concept of reoptimization has been investigated for several different problems like the traveling salesman problem [1,3,8,12], knapsack problems [2], covering problems [7], and the shortest common superstring problem [6]. In these papers, it was shown that, for some problems, the reoptimization variant is exactly as hard as the original problem, whereas reoptimization can help a lot for improving the approximation ratio for other problems, for an overview of some results see also [11]. These results show that the reoptimization concept gives new insight into the hardness of the underlying optimization problems and allows for a more fine-grained complexity analysis.

The Steiner tree reoptimization problem in general weighted graphs was previously investigated in [5,10,13] for various types of local modifications. We show that eight reoptimization variants (insertion and deletion of terminal or nonterminal vertices, increasing and decreasing edge costs, and changing the status of vertices from terminal to non-terminal and vice versa) are NP-hard on graphs with edge costs restricted to 1 and $1 + \gamma$. The best approximation algorithms for the four reoptimization variants considered in [5] (a terminal becomes a nonterminal or vice versa; the cost of an edge increases or decreases) achieve a constant approximation ratio in metric graphs, i. e., graphs that satisfy the triangle inequality for $\beta = 1$. Here, we show that, on β-metric graphs, all of these four cases permit, in contrast to the non-reoptimization problem, a PTAS for any $\beta < 1$. When the local modification, however, consists in removing vertices, we show that the Steiner tree reoptimization is as hard to approximate as the original problem.

The two algorithmically most interesting reoptimization variants are the addition of terminal and of nonterminal vertices. For these modifications, we prove the APX-hardness of the corresponding reoptimization variants, which solves also the analogous open problem for reoptimization in Steiner trees with arbi-

trary edge costs. Escoffier et al. [13] designed simple linear-time algorithms for metric input instances ($\beta = 1$) which achieve an approximation ratio of $3/2$. Using the same algorithms, but a much more complex and technically involved analysis, we prove a $(1/2 + \beta)$-approximation for graphs satisfying a sharpened β-triangle inequality. Note that the ratio $(1/2 + \beta)$ tends to $3/2$ for β tending to 1 and to 1 for β tending to $1/2$. These proofs employ a 2β-approximation algorithm for the classical non-reoptimization version of the Steiner tree problem in β-metric graphs which may be of independent interest since it improves over the previously best known ratio for any $\beta < 1/2 + \ln(3)/4 \approx 0.775$.

2 Preliminaries

Given a graph $G = (V, E)$ and a subset $S \subseteq V$ of vertices, called *terminals*, a *Steiner tree* for (G, S) is a subtree T of G spanning all terminals, i.e., $T = (V(T), E(T))$ is a tree such that $S \subseteq V(T) \subseteq V$ and $E(T) \subseteq E$. The vertices in $V - S$ are called *non-terminals*.

In a weighted graph $G = (V, E)$ with cost function $c : E \to \mathbb{Q}^+$, a *minimum Steiner tree* is a Steiner tree T of minimum cost, i.e., minimizing $\sum_{e \in E(T)} c(e)$ over all Steiner trees of G. In the remainder of the paper, we denote by $G = (V, E)$ a complete, undirected edge-weighted graph with a cost function c. The vertex set V of G is also denoted by $V(G)$ and the edge set E of G is also denoted by $E(G)$. Furthermore, we denote by $S \subseteq V(G)$ the set of terminals of G. The sum of the costs of all edges in a subgraph H of G is defined by $c(H) = \sum_{e \in E(H)} c(e)$. For the cost of an edge $\{x, y\}$, we use the notation $c(x, y)$ instead of $c(\{x, y\})$. We are now ready to define the underlying optimization problem for our further investigations.

The *minimum Steiner tree problem (Min-STP)* in connected edge-weighted graphs is the problem of finding a minimum Steiner tree for an input instance (G, S, c). If the cost function c satisfies the β-triangle inequality, the minimum Steiner tree problem on the the input instance (G, S, c) is called *Min-Δ_β-STP*. Similar to the Min-Δ_β-STP, we consider the problem *Min-$(1, 1 + \gamma)$-STP,* where only edges of cost 1 and $1 + \gamma$ are allowed. The relation of the Min-Δ_β-STP and the Min-$(1, 1 + \gamma)$-STP is as follows.

Lemma 1. *For any graph $G = (V, E)$ and any $0 < \gamma \leq 1$, any cost function $c : E \to \{1, 1 + \gamma\}$ satisfies also the $(1 + \gamma)/2$-triangle inequality.*

Proof. Let x, y, and z be three different vertices in V. If all three edges $\{x, y\}$, $\{y, z\}$, and $\{z, x\}$ cost the same, then obviously our claim holds. Otherwise, either two of the edges have cost 1 or two of the edges have cost $1 + \gamma$. Since $1 + \gamma \leq \frac{1+\gamma}{2} \cdot (1 + 1)$, our claim also holds for these cases. $\square$

We now formally define the reoptimization variants of Min-Δ_β-STP that we consider.

Definition 1. *The* minimum Steiner tree reoptimization problem with the local modification *lm* (Min-STRP-*lm*) *is the following optimization problem. The goal*

is to find a minimum Steiner tree for an input instance (G', S', c'), given an optimal Steiner tree T_{Old} for the instance (G, S, c), where $(G', S', c') = lm((G, S, c))$.

We consider the following local modifications. When adding a non-terminal (AddNonTerm), $V(G') = V(G) \uplus \{v_{\text{New}}\}$, $S' = S$, and c is the restriction of c' to $V(G)$. When adding a terminal vertex (AddTerm), $V(G') = V(G) \uplus \{v_{\text{New}}\}$, $S' = S \uplus \{v_{\text{New}}\}$, and c is the restriction of c' to $V(G)$. When removing a terminal or non-terminal vertex v (RemNonTerm, RemTerm), $V(G') = V(G) \backslash v$, $S' = S \backslash v$, and c' is the restriction of c to $V(G')$. When increasing or decreasing the cost of one edge e (IncEdge, DecEdge), $G' = G$, $S' = S$, $c'(f) = c(f)$ for all $f \in E(G) \backslash \{e\}$, and $c'(e)$ is larger or smaller than $c(e)$, respectively. When changing the status of a vertex v (Term→NonTerm, NonTerm→Term), $G' = G$, $S' = S \backslash v$ or $S' = S \cup \{v\}$, respectively, and $c' = c$.

The corresponding problem variants where the edge cost function c' satisfies the sharpened β-triangle inequality for some $1/2 \leq \beta \leq 1$ and the variant with edge costs in $\{1, 1 + \gamma\}$ are Min-Δ_β-STRP-lm and Min-$(1, 1 + \gamma)$-STRP-lm, respectively.

3   Reoptimization Hardness

All reductions in this paper require a transformation of an instance of the satisfiability problem into a $(1, 1 + \gamma)$-Steiner tree instance. The following transformation extends the one used in the NP-hardness proof for the Steiner tree problem in [17].

Transformation 1. Let Φ be a SAT instance with m clauses $C_1^\Phi, C_2^\Phi, \ldots, C_m^\Phi$ and n variables $x_1^\Phi, x_2^\Phi, \ldots, x_n^\Phi$. We construct from Φ a $(1, 1 + \gamma)$-Steiner tree instance $(G = (V, E), S \subset V, c : E \to \{1, 1 + \gamma\})$. In the construction, we say that two vertices are *connected*, if and only if the edge between these vertices has cost one.

In V, there are m terminal vertices $C_1, C_2, \ldots, C_m$ representing the clauses and $2 \cdot n$ nonterminal vertices $x_1, \bar{x}_1, x_2, \bar{x}_2, \ldots, x_n, \bar{x}_n$ representing the positive and negative variables. Each C_i is connected to the k nonterminal vertices that represent the literals within the clause C_i^Φ. For connecting the variables, we introduce a terminal vertex y. For every i, x_i and $\bar{x}_i$ are connected to y (see Fig. 1). The idea of the construction is to build a Steiner tree by connecting each clause-vertex of a satisfied clause with exactly one variable-vertex. We have to ensure, however, that for every i, either x_i or $\bar{x}_i$ is in an optimal Steiner tree. To this end, we introduce auxiliary terminal vertices a_{ij} for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, \lceil 1/\gamma \rceil$. For any i and j, a_{ij} is connected to x_i and to $\bar{x}_i$. Furthermore, x_i and $\bar{x}_i$ are connected. All remaining edges have cost $1 + \gamma$. Fig. 1 shows an example of a Steiner tree for a satisfiable formula.

Normal Form. Given a $(1, 1 + \gamma)$-Steiner tree instance $(G = (V, E), S, c)$ that was constructed from a SAT formula using Transformation 1 and a Steiner tree T within that instance, we construct a new Steiner tree T' that is at most as expensive as T and satisfies some structural properties. In the following, we refer to vertices x_i and $\bar{x}_i$ as *variable-vertices* and to the vertices C_j as *clause-vertices*.
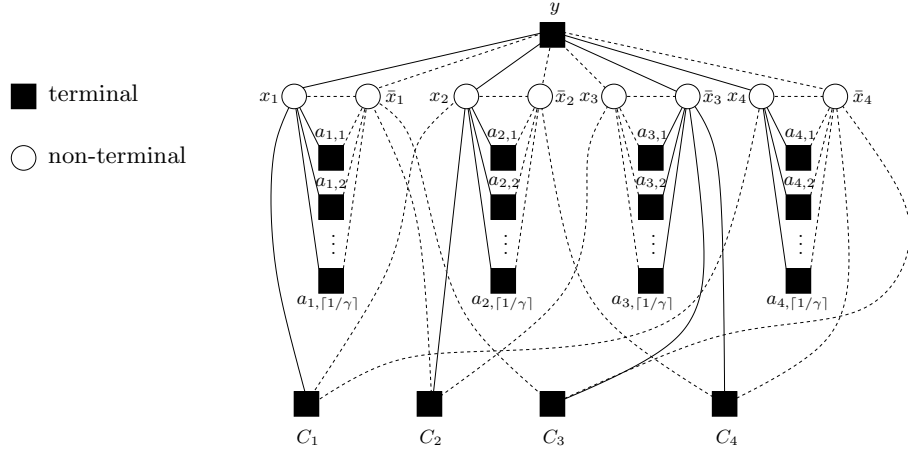
Fig. 1. Steiner tree for the formula (x_1, x_2, x_4), $(\bar{x}_1, x_2, x_3)$, $(\bar{x}_1, \bar{x}_3, \bar{x}_4)$, $(\bar{x}_2, \bar{x}_3, \bar{x}_4)$. The squares are terminal vertices and the circles are non-terminal vertices. All edges depicted in the graph have cost 1. All remaining edges have cost $1 + \gamma$. The solid edges form a minimal Steiner tree that corresponds to the assignment $x_1 = 1$, $x_2 = 1$, $x_3 = 0$, and $x_4 = 1$.

In the Steiner tree in normal form, for each i, either x_i or $\bar{x}_i$ is connected to the terminal vertex y and to all auxiliary vertices $a_{i,j}$. Furthermore, all auxiliary vertices are leaves. This way, the edges incident to the auxiliary vertices define an assignment of the variables. We define X as the set of all variable-vertices that are adjacent to auxiliary variables. Moreover, each clause-vertex C_i is a leaf that is connected to a variable-vertex x_j or $\bar{x}_j$ and there is no clause-vertex that is connected to a variable-vertex outside of X via an edge of cost $1 + \gamma$.

In the following, we describe different types of "bad" structures that may be contained in T and we give transformations to remove them. Each step has to be repeated until all occurrences of that type have been eliminated.

(1) First, we ensure that each auxiliary vertex is a leaf in T'. Let $a_{i,j}$ be an auxiliary vertex in T that has (at least) two neighbors v_1 and v_2, such that v_1 is *not* an auxiliary vertex. (Note that, since T is connected, the absence of such a vertex $a_{i,j}$ of vertices means that all auxiliary vertices are leaves.)

We distinguish two cases. First, let us assume that $c(a_{i,j}, v_2) = 1 + \gamma$. Then we replace the edge $\{a_{i,j}, v_2\}$ by $\{v_1, v_2\}$. Otherwise, if $c(a_{i,j}, v_2) = 1$, we replace $\{a_{i,j}, v_1\}$ by $\{v_1, v_2\}$. Since $v_2 \in \{x_i, \bar{x}_i\}$ in this case and $c(x_i, \bar{x}_i) = 1$ for any i, this transformation does not increase the cost of T'. Eventually, after sufficiently many repetitions of this step, each auxiliary vertex is connected to exactly one vertex, and there are no edges connecting two auxiliary vertices.

(2) Since $\lceil 1/\gamma \rceil \cdot \gamma \geq 1$, we can eliminate all edges of cost $1 + \gamma$ that are incident to auxiliary vertices. If, for some i, all auxiliary vertices $a_{i,j}$ are connected with cost $1 + \gamma$, then we can remove all of those edges, connect each of the vertices $a_{i,j}$

to x_i, and connect x_i to y, which does not increase the overall cost. Otherwise, at least one of the auxiliary variables is connected to either x_i or $\bar{x}_i$. Then, connecting each vertex $a_{i,j}$ to the same vertex (and removing the old incident edge) results in a new Steiner tree that has at most the same cost as the old one. If there is an i such that an auxiliary variable $a_{i,j}$ is connected to x_i and another auxiliary vertex $a_{i,j'}$ is connected to $\bar{x}_i$, we connect all auxiliary variables to x_i and remove all edges to $\bar{x}_i$.

(3) Now, for each i, all auxiliary vertices a_{ij} in T' are connected to one vertex $v_i \in \{x_i, \bar{x}_i\}$. Suppose that, for some i, v_i is *not* connected to y. Then, since T' is a connected graph, there is path from v_i to y that starts with an edge e. We replace e by $\{v_i, y\}$.

(4) Next, we ensure that the clause-vertices are only connected to variable-vertices. Suppose that e is an edge in T' that connects two clause-vertices C_i and C_j. From one of the two vertices, say from C_i, there is a path to y that does not contain e. In this case, we remove e and connect C_j to a vertex-variable from X. Similarly, we replace each edge between a clause-vertex and y by an edges between that clause vertex and a variable-vertex from X. Therefore, we have removed all edges between clause-vertices and non-variable-vertices. (Remember that all auxiliary vertices are leaves.)

(5) To ensure that the clause-vertices are leaves, suppose that there is a variable vertex v_i (either x_i or $\bar{x}_i$) that is not connected to y but it is connected to several clause vertices. Then, since T' is a tree, there is exactly one clause vertex C_j adjacent to v_i that is connected to y by a path not containing v_i. We remove the first edge of the path and insert the edge $\{v_i, y\}$.

Note that, if all variable-vertices connected to clause-vertices are also connected to y, then any clause vertex that is not a leaf would have two disjoint paths to y. But since T' is a tree, this cannot happen.

(6) Now we refine T' as follows. If there is any clause-vertex C_i that is connected to a variable-vertex $v_j \notin X$ with cost $1 + \gamma$, then we replace the edge $\{C_i, v_j\}$ by an edge between C_i and a vertex from X. If due to the transformation a variable-vertex becomes a leaf, we remove its incident edge.

(7) Finally, we modify X and some of its incident edges. The purpose of this step is to ensure that for each i, the majority of edges from clause vertices to the variable-vertices x_i and $\bar{x}_i$ is incident to X. If there is a variable-vertex $x_i \in X$ such that more clause-vertices are connected to $\bar{x}_i$ via edges of cost 1 than to x_i, we connect all adjacent auxiliary vertices and all clause-vertices that are connected to x_i with cost $1 + \gamma$ to $\bar{x}_i$ instead and we update X by adding $\bar{x}_i$ and removing x_i. For $\bar{x}_i \in X$, the transformation is analogous.

Lemma 2. *Let Φ be a Boolean formula with m clauses and n variables. Then the Steiner tree instance (G, S, c) obtained by applying Transformation 1 has an optimal solution of cost $(1 + \lceil 1/\gamma \rceil)n + m$ if and only if Φ is satisfiable and, if Φ is not satisfiable, the cost of any solution is higher than that value. Furthermore,*

from a satisfying assignment for Φ one can efficiently compute an optimal Steiner tree in (G, S, c).

Proof. For distinguishing the clauses and variables of Φ from the corresponding vertices in the Steiner tree instance, we denote the clauses by C_i^Φ and the variables by x_i^Φ and $\bar{x}_i^\Phi$.

First, let us assume that Φ is satisfiable. Similar as in the proof of Lemma 3, we construct the following Steiner tree T. Let $\varphi = (\varphi_1, \varphi_2, \ldots, \varphi_n) \in \{0, 1\}^n$ be a satisfying assignment of Φ. For each variable x_i^Φ, if $\varphi_i = 1$ then we connect all auxiliary variables $a_{i,j}$ as well as y to the vertex x_i. Otherwise, if $\varphi_i = 0$, then we connect these vertices to $\bar{x}_i$. This way, the variable-vertices in the constructed tree correspond to the assignment φ. Let X be the set of variable-vertices that are now part of the constructed tree. For any clause C_i^Φ that is satisfied by φ, we use one edge of cost 1 to connect the clause-vertex C_i to a vertex from X. Then the overall cost of the solution is $((1 + \lceil 1/\gamma \rceil)n + m$. Since T is in normal form and all clause-vertices are connected with edges of cost 1, T is minimal.

Now, let us assume that Φ is *not* satisfiable. Let T be a minimal Steiner tree in normal form for the given instance, where the set X contains all variable-vertices connected to auxiliary vertices. Then there is a subtree of T containing all auxiliary-vertices, X, and m that has a cost of $(1 + \lceil 1/\gamma \rceil)n$. Since Φ is not satisfiable, there must be at least one clause-vertex that has no edge of cost 1 to X. Thus, the cost of all edges for connecting the clause-vertices sums up to at least $m + \gamma$. Therefore, the overall cost of the Steiner tree is at least $((1 + \lceil 1/\gamma \rceil)n + m + \gamma > ((1 + \lceil 1/\gamma \rceil)n + m$. $\qquad\square$

For showing the hardness of approximation of the $(1, 1+\gamma)$-Steiner tree reoptimization problem with adding vertices, we use a gap-preserving reduction from Es-OCC-MaxEkSAT i. e., maximum satisfiability, where each variable occurs in exactly s clauses and each clause consists of exactly k literals. We use formalisms similar to [21].

Lemma 3. *There is a gap-preserving reduction from Es-OCC-MaxEkSAT to the $(1, 1 + \gamma)$-Steiner tree reoptimization problem with adding a single vertex by transforming a Boolean formula Φ into a reoptimization instance consisting of an old instance (G_O, S_O, c_O), an optimal Steiner tree T_O, and a modified instance (G_N, S_N, c_N), such that*

- *if an optimal assignment for Φ satisfies at least $p \cdot m$ clauses, then there is a Steiner tree in G of cost at most $m \cdot ((1 + \lceil 1/\gamma \rceil)k/s + 1 + (1 - p) \cdot \gamma) + m \cdot \lceil 1/\gamma \rceil + 1$, and*
- *if an optimal assignment for Φ satisfies less than $(p - \varepsilon) \cdot m$ clauses, then any Steiner tree in G costs at least $m \cdot ((1 + \lceil 1/\gamma \rceil)k/s + 1 + (1 - (p - \varepsilon)) \min\{\gamma, 2/s\}) + m \cdot \lceil 1/\gamma \rceil + 1$,*

where m is the number of clauses and $\varepsilon > 0$ is a constant.

Proof. Our proof is structured as follows. For any input instance Φ of Es-MaxEkSAT, we consider the input instance (G_t, S_t, c_t) created by applying
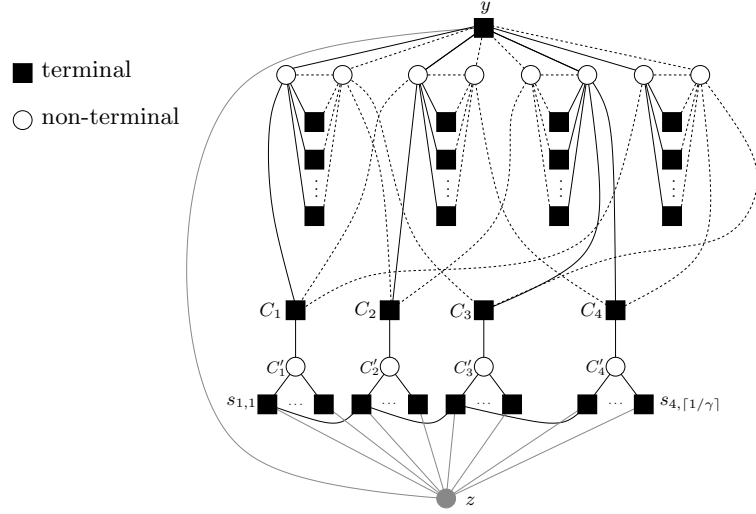
Fig. 2. Example of a Steiner tree as used in Lemma 3.

Transformation 1. In (G_t, S_t, c_t), we replace the clause-vertices by gadgets, which leads to a new Steiner tree instance (G, S, c) (see Figure 2). Within this instance, we can efficiently compute an optimal Steiner tree. By adding a terminal or non-terminal vertex z, however, we obtain a new input instance (G_N, S_N, c_N). For this instance, we give an upper and a lower bound on the cost of an optimal Steiner tree depending on the number of clauses that are satisfiable within Φ. With these bounds, the gap between p and $p-\varepsilon$ translates into a new gap by applying the computed lower bound for p and the upper bound for $p-\varepsilon$. Note that all intermediate instances and solutions can be computed from Φ in polynomial time and thus they do not significantly help to solve (G_N, S_N, c_N).

Let n be the number of different variables in Φ. For distinguishing the clauses and literals of Φ from the corresponding vertices in the Steiner tree instance, we denote the clauses by C_i^{Φ} and the literals by x_i^{Φ} and $\bar{x}_i^{\Phi}$. Furthermore, let σ be the value such that $\sigma \cdot m$ clauses of Φ are satisfied by an optimal assignment $\varphi \in \{0, 1\}^n$.

For each clause-vertex C_i, we introduce a nonterminal vertex C_i' and $\lceil 1/\gamma \rceil$ terminal vertices $s_{i,1}, s_{i,2}, \ldots, s_{i,\lceil 1/\gamma \rceil}$. In the following, we refer to these terminal vertices as *split-vertices*. Each vertex C_i' is connected to all vertices belonging to the gadget of its clause. Furthermore, for $1 \le i < m$, the vertex $s_{i,1}$ is connected to $s_{i+1,1}$. As before, all remaining edges have a cost of $1 + \gamma$.

Using a similar argumentation as for the auxiliary vertices, we conclude that there is an optimal Steiner tree that, for each i, contains all edges between C_i' and its split-vertices. Furthermore, each such component has to be connected to the remaining graph with a cost of at least 1. Thus each of these gadgets adds a cost of at least $1 + \lceil 1/\gamma \rceil$ to any solution, which sums up to at least

$m \cdot (1 + \lceil 1/\gamma \rceil)$ additional cost to an optimal solution for (G, S, c). Therefore, applying Lemma 2, the cost of an optimal solution for (G, S, c) is at least $(1 + \lceil 1/\gamma \rceil)n + m + m \cdot (1 + \lceil 1/\gamma \rceil)$.

Now, we construct an optimal Steiner tree T_O for (G, S, c). Without loss of generality, let us assume that the clauses and variables are labelled in such a way that C_1^Φ contains x_1^Φ. Then T_O contains, for $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, \lceil 1/\gamma \rceil$, and $k = 1, 2, \ldots, m$, all edges $\{y, x_i\}$, $\{a_{i,j}, x_i\}$, $\{x_1, C_1\}$, $\{C_1, C_1'\}$, $\{C_k', s_{k,j}\}$, and, for $k = 1, 2, \ldots, m - 1$, $\{s_{k,1}, s_{k+1,1}\}$. This way, T_O is obviously a valid Steiner tree and, since each of its edges is of cost 1, the cost of T_O is $(1 + \lceil 1/\gamma \rceil)n + 1 + (m - 1) + m \cdot (1 + \lceil 1/\gamma \rceil)$, which we have already shown to be optimal.

Now, the local modification is to add the vertex z in such a way that z is connected to all vertices $s_{k,j}$ and to y. This way, we obtain the new, locally modified instance (G_N, S_N, c_N) with one added vertex.

We begin with the upper bound. To this end, we construct the following Steiner tree. For each variable x_i^Φ, if $\varphi_i = 1$, then we connect all auxiliary variables $a_{i,j}$ as well as y to the vertex x_i. Otherwise, if $\varphi_i = 0$, then we connect these vertices to $\bar{x}_i$. This way, the variable-vertices in the constructed tree correspond to the assignment φ. Let X be the set of variable-vertices that are now part of the constructed tree. For any clause C_i^Φ that is satisfied by φ, we use one edge of cost 1 to connect the clause-vertex C_i to a vertex from X. We connect all clause-vertices of non-satisfied clauses to an arbitrary vertex from X with cost $1 + \gamma$. Furthermore, we use all $m \cdot \lceil 1/\gamma \rceil + 1$ edges of cost 1 from z. The overall cost of that Steiner tree is $(1 + \lceil 1/\gamma \rceil)n + m + (1 - \sigma)m \cdot \gamma + m \cdot \lceil 1/\gamma \rceil + 1$. Note that the upper bound decreases with growing σ.

For the lower bound, note that, since we assume $\gamma \leq 1$, there is an optimal solution in normal form for (G_N, S_N, c_N) that contains z and all its edges of cost 1. Suppose that, in some optimal solution in normal form, there is a vertex $s_{k,j}$ that is not connected to z. Then, according to our argumentation above, we can assume this vertex to be connected to C_k'. But then, connecting all splitting-vertices of C_k' to z instead of to C_k' does not increase the cost of the solution. We still have to ensure that the modified solution is connected. If the solution already contains the edge $\{y, z\}$, then either C_k is connected to the remaining tree and we are done or C_k is only connected to C_k'. In the latter case, there must be an edge e in the given optimal Steiner tree that connects the gadget of clause k to the remaining graph. By removing the two edges e and $\{C_k, C_k'\}$ and adding an edge between C_k and a vertex from X, however, we obtain a connected solution of at most the same cost. For amortizing the cost of $\{y, z\}$, we only need one clause gadget where its clause-vertex can be connected to X with cost 1, which always exists in an optimal solution in normal form. With this discussion, it is not hard to verify that an optimal solution does not need the vertices C_i'.

Let T be an optimal Steiner tree for the given instance in normal form that contains z and all of its incident edges of cost 1 and none of the vertices C_i'. The subtree T' induced by the auxiliary variables, the vertex y, and the variable-

vertices that are connected to auxiliary variables has a fixed cost of $(1+\lceil 1/\gamma\rceil)n$. Additionally, all clause-vertices have to be in T. For each of them, there is an edge of cost at least 1. There are at most σm clause-vertices connected to T' with cost 1, since more such clause-vertices would imply that there is an assignment φ' that satisfies more than σm clauses in Φ. The remaining clause-vertices are either connected with cost $1+\gamma$ to T', or they are connected to variable vertices outside of T' with cost 1. In the latter case, however, the number of clause-vertices connected to the same variable-vertex v is limited, because there are exactly s clauses in Φ with the same variable. Furthermore, there cannot be more than $s/2$ different clauses connected to v, since otherwise changing φ such that v is in T' would satisfy additional clauses, contradicting the optimality of φ. For connecting v to T', an edge of cost 1 is necessary. Therefore, the average cost for connecting a clause-vertex to a variable-vertex outside of T' is at least $(s/2+1)/(s/2) = 1+2/s$. Altogether, a lower bound on the cost of the optimal Steiner tree is $(1 + \lceil 1/\gamma\rceil)n + m + (1-\sigma)m\min\{\gamma, 2/s\} + m \cdot \lceil 1/\gamma\rceil + 1$.

Note that there are $k \cdot m$ literals in Φ, and each variable occurs exactly s times, and therefore $n = km/s$. Therefore, the upper bound is $m \cdot ((1 + \lceil 1/\gamma\rceil)k/s + 1 + (1-(p-\varepsilon))\cdot\gamma) + m \cdot \lceil 1/\gamma\rceil + 1$ and the lower bound is $m \cdot ((1 + \lceil 1/\gamma\rceil)k/s + 1 + (1-p)\min\{\gamma, 2/s\}) + m \cdot \lceil 1/\gamma\rceil + 1$. $\qquad\square$

With this preparation, showing the APX-hardness for adding vertices can be done by using the APX-hardness of E5-OCC-MaxE3SAT [14]. For removing a terminal or a non-terminal, a simple argumentation shows the corresponding reoptimization problem to be as hard as the original problem.

Theorem 1. *The problem Min-$(1, 1+\gamma)$-STRP-lm for $\gamma > 0$ as well as the problem Min-Δ_β-STRP-lm for $\beta > 1/2$ and lm $\in \{AddTerm, AddNonTerm, RemTerm, RemNonTerm\}$ are APX-hard.*

Proof. For the Min-$(1, 1+\gamma)$-STP, consider the problem E5-OCC-MaxE3SAT. There is a ε such that it is NP-hard to decide whether all m clauses or at most $m\varepsilon$ clauses are satisfiable [14]. Thus, due to Lemma 3, it is NP-hard to decide whether an optimal Steiner tree in a $(1, 1+\gamma)$-Steiner tree instance with $1 + (2 + \lceil 1/\gamma\rceil) \cdot (3m/5) + m + 1$ vertices costs at most $m \cdot ((1 + \lceil 1/\gamma\rceil)3/5 + 1 + 0\cdot\gamma) + m \cdot \lceil 1/\gamma\rceil + 1$ or at least $m\cdot((1+\lceil 1/\gamma\rceil)3/5+1+(1-(1-\varepsilon))\min\{\gamma, 2/5\})+m\cdot\lceil 1/\gamma\rceil+1$. This implies that there is no

$$\frac{(1 + \lceil 1/\gamma\rceil)3/5 + 1 + \varepsilon\min\{\gamma, 2/5\} + \lceil 1/\gamma\rceil + 1/m}{(1 + \lceil 1/\gamma\rceil)3/5 + 1 + \lceil 1/\gamma\rceil + 1/m}$$
$$= 1 + \frac{\varepsilon\min\{\gamma, 2/5\}}{(1 + \lceil 1/\gamma\rceil)3/5 + 1 + \lceil 1/\gamma\rceil + 1/m}$$

approximation algorithm for the Min-$(1, 1+\gamma)$-STP. For any $\beta > 1/2$, due to Lemma 1, the Min-Δ_β-STP contains all instances of the Min-$(1, 2\beta)$-STP.

For the local modifications, where vertices are removed, we show that the problem stays as hard as the original problem without reoptimization. More precisely, we show that an α-approximation for any of the two reoptimization

problems implies an α-approximation for the Min-$(1, 1 + \gamma)$-STP. Therefore, the APX-hardness of the reoptimization problems follows. Let $(G = (V, E), S, c)$ be a Min-$(1, 1 + \gamma)$-STP instance. We assume without loss of generality that all optimal Steiner trees for that instance contain at least one non-terminal. Otherwise, a minimum spanning tree on the vertices of S is an optimal solution. Then we construct the instance $(G' = (V', E'), S', c')$ from (G, S, c) by adding a new vertex v to V (either terminal or non-terminal) such that $c'(\{v, w\}) = 1$ for all $w \in S$ and $c'(\{v, w'\}) = 1 + \gamma$ for all $w' \in V \setminus S$. Since any optimal Steiner tree in G has (due to the contained non-terminal) at least $|S|$ edges, connecting v to all edges of S yields an optimal Steiner tree in (G', S', c'). Now suppose that the reoptimization problem, where (G', S', c') and the discussed optimal solution are given and (G, S, c) is the new, modified input instance, is α-approximable. Then also the original problem is α-approximable since we can efficiently construct the reoptimization problem and use its computed solution. $\qquad\square$

Note that Theorem 1 implicitly provides an alternative proof for the APX-hardness of the problems Min-$(1, 1 + \gamma)$-STP and Min-Δ_β-STP.

We now show that also the remaining reoptimization problems from Definition 1 are NP-hard.

Theorem 2. *The Steiner tree reoptimization problem Min-$(1, 1 + \gamma)$-STRP-lm is NP-hard for all local modifications from Definition 1.*

We restrict ourselves to show only the hardness for the local modifications NonTerm→Term and Term→NonTerm, since these two cases already contain most of the required ideas. The remaining local modifications only require minor changes.

Proof. We reduce from SAT, let Φ be a CNF formula with m clauses and n variables.

For Min-$(1, 1 + \gamma)$-STRP-NonTerm→Term, we construct a formula Φ' and a formula Φ'' such that the transformation of Φ'' is a locally modified instance of the transformation of Φ'. Consider the formula Φ', where a new variable x that is added to each clause of Φ. It is clear that assigning 1 to each of the variables satisfies Φ'. Let I' be the Steiner tree instance obtained by transforming Φ' and adding a clause-vertex $\bar{x}$ as non-terminal vertex. Due to Lemma 2, we can compute a minimal Steiner tree T in I_1 such that T has a cost of $(1 + \lceil 1/\gamma \rceil)(n + 1) + m$. For constructing a modified instance I_2, let us consider the formula Φ'' obtained by adding the clause $\{\bar{x}\}$ to Φ'. Then the instance I_2 is the same instance as I_1, except that the non-terminal clause-vertex $\bar{x}$ now is a terminal vertex. For satisfying Φ'', the assignment of x has to be 0. Therefore, Φ'' is satisfiable if and only if Φ is satisfiable. Thus, due to Lemma 2, Φ is satisfiable if and only if I_2 has a minimal Steiner tree of cost $(1 + \lceil 1/\gamma \rceil)(n + 1) + m + 1$.

The reduction for Min-$(1, 1 + \gamma)$-STRP-Term→NonTerm is similar to the previous one, but I_1 has two additional clause-vertices $\{x\}$ and $\{\bar{x}\}$. The clause-vertex $\{x\}$ is then removed in I_2. The formula Φ' that corresponds to I_1 is not satisfiable (since no assignment satisfies both clauses $\{x^\Phi\}$ and $\{\bar{x}^\Phi\}$), according

to Lemma 2 no solution for I_1 has a cost of less than $(1 + \lceil 1/\gamma \rceil)n + m + \gamma$. But taking a solution as for NonTerm→Term and connecting $\{\bar{x}\}$ directly to the variable-vertex x has a cost of exactly $(1 + \lceil 1/\gamma \rceil)n + m + \gamma$. Thus, a solution for I_1 can efficiently be computed, but as for NonTerm→Term, solving I_2 optimally is as hard as deciding the satisfiability of Φ. $\qquad \square$

4 Approximation Algorithms

We start this section by presenting an approximation algorithm for the problem Min-Δ_β-STP. This algorithm, besides being interesting by itself, will be useful for some of the subsequent approximation algorithms for reoptimization.

In any instance (G, S, c) consisting of terminals only, i.e., if $S = V(G)$, the minimum spanning tree is an optimal solution to the minimum Steiner tree problem. Intuitively speaking, the minimum Steiner tree problem can be viewed as the problem of finding a subset Y of non-terminals which minimizes the cost of a minimum spanning tree on $S \cup Y$ over all possible choices of Y. But also in those graphs in which the optimal solutions contain non-terminal vertices, the minimum spanning tree on the set of terminals gives a useful approximation of the minimum Steiner tree. To estimate the quality of this approximation, we need the following technical lemma for the subsequent approximability result.

Lemma 4. *Given an input instance (G, S, c) for Min-Δ_β-STP, for some $1/2 \leq \beta \leq 1$, and a minimum Steiner tree T_{Opt} for this instance, let $T_1, \dots, T_k$ be the maximal subtrees of T_{Opt} consisting of non-terminals only. For any T_i, let $N(T_i)$ be the set of neighbors of T_i in T_{Opt}. Then there exists a connected subgraph H of G with the following properties.*

1. $V(H) = S$,
2. H *contains a cycle on the vertices of $N(T_i)$ for all $i \in \{1, \dots, k\}$, and*
3. $c(H) \leq 2\beta \cdot c(T_{\mathrm{Opt}})$.

Proof. If there exists an optimal solution T_{Opt} without non-terminals, then $H = T_{\mathrm{Opt}}$ is a minimum spanning tree and thus $c(H) = c(T_{\mathrm{Opt}}) \leq 2\beta \cdot c(T_{\mathrm{Opt}})$.

Let $T_1, \dots, T_k$ be the maximal subtrees of T_{Opt} consisting of non-terminals only. For all $i \in \{1, \dots, k\}$, let T_i' be the subtree of T_{Opt} containing exactly the vertices $V(T_i) \cup N(T_i)$. Let C_i be the cycle on the vertices of $N(T_i)$ as ordered by a depth-first traversal of T_i' (see Fig. 3). In T_i', there are no neighboring terminals, which means that there do not exist any two terminals $v_1, v_2 \in V(T_i')$ such that $\{v_1, v_2\} \in E(T_i')$. This implies that every edge in C_i is a shortcut of at least two edges from T_{Opt}. From this, and since the depth-first traversal visits every edge exactly two times, the cost of C_i can be estimated by $c(C_i) \leq 2\beta \cdot c(T_i')$.

Let $E_{\mathrm{circus}} = \bigcup_{1 \leq i \leq k} E(C_i)$ be the set of all cycle edges and let $E_{\mathrm{arbor}} = E_{Opt} - \bigcup_{1 \leq i \leq k} E(T_i')$ be the set of all edges between terminals in T_{Opt}. The graph H is the union of the cycles C_i, for all $i \in \{1, \dots, k\}$, together with the subtrees of T_{Opt} which contain terminals only, i.e., $V(H) = S$ and $E(H) = E_{\mathrm{circus}} \cup E_{\mathrm{arbor}}$. Thus, H satisfies the constraints 1 and 2. The graph H is connected since the
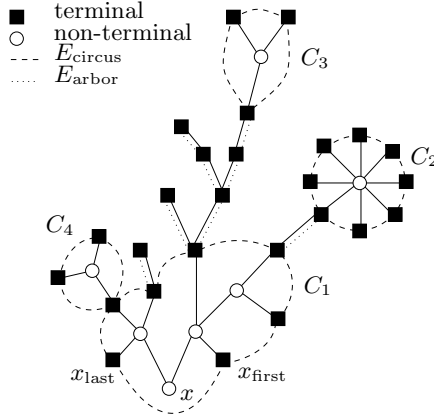
Fig. 3. The construction in the proof of Lemma 4

terminals inside the subtrees T'_i are connected by the cycle C_i. Moreover, the subtrees T'_i are connected either by a common vertex or by edges from E_{arbor}.

We know that $c(T_{Opt}) = \sum_{i=1}^{k} c(T'_i) + c(E_{arbor})$. By definition of T'_i, any two trees T'_j, T'_l are edge-disjoint, therefore $c(E_{circus}) \leq 2\beta \cdot \sum_{i=1}^{k} c(T'_i)$. This implies $c(H) = c(E_{circus}) + c(E_{arbor}) \leq 2\beta \cdot \sum_{i=1}^{k} c(T'_i) + c(E_{arbor}) \leq 2\beta \cdot c(T_{Opt})$, which proves that H also satifies constraint 3. $\qquad\square$

Theorem 3. *Let G be graph with cost function c satisfying the β-triangle inequality for $1/2 \leq \beta \leq 1$. Let S be a set of terminals. Then the minimum spanning tree on S is a 2β-approximation of the minimum Steiner tree for the instance (G, S, c).*

Proof. This follows directly from the constraints 1 and 2 of Lemma 4. $\qquad\square$

Polynomial-Time Approximation Schemes. Here, we present polynomial-time approximation schemes for some of the reoptimization variants. The concept in all cases relies on the properties of graphs with sharpened triangle inequality.

Theorem 4. *Let (G, S, c) be a Steiner tree instance where c satisfies the sharpened β-triangle inequality for some $1/2 < \beta < 1$. Then there is a PTAS for the reoptimization variants of the Min-Δ_β-STP when the edge-costs are increased or decreased and when the status of vertices is changed, i.e., when a terminal becomes a non-terminal or vice versa.*

Proof. Let $e = \{u, v\}$ be a cheapest edge in G and let $f = \{x, y\}$ be a most expensive edge in G. Without loss of generality let us assume that $c(e) = 1$. Then the cost of f can be bounded from above by $c(f) \leq 2\beta^2/(1-\beta)$, see [9] for a proof. The existence of a PTAS for Steiner tree reoptimization in graphs with edge-costs bounded by a constant was shown in [10] for changing the status of

---

Algorithm 1 Approximation algorithm for Min-Δ_β-STRP-AddNonTerm

---

Input: $(G = (V, E), S, c)$, T_{Old}, $(G_{\text{New}} = (V \cup v_{\text{New}}, E \cup \{\{v_{\text{New}}, x\} \mid x \in V\}), S, c)$
 1: Compute the minimum spanning tree T_{MST} on the vertex set $S \cup \{v_{\text{New}}\}$.
 2: Compute the best solution T_{Alg} between T_{Old} and T_{MST}.
Output: The Steiner tree T_{Alg}.

---

a vertex. With the same argument, we can also conclude that there is a PTAS when changing the the edge costs. □

Adding a Non-Terminal. Now, we consider the Steiner tree reoptimization problem with the local modification of adding a non-terminal to the graph. We assume that the new instance satisfies the β-triangle inequality for the same β as the old instance. We design an algorithm for Min-Δ_β-STRP-AddNonTerm that outputs the better of the following two feasible solutions: The first feasible solution is simply the given optimal solution to the old instance, the second is obtained by computing a minimum spanning tree on the terminals together with the newly inserted vertex. This procedure is shown in Algorithm 1.

For analyzing the cost of T_{MST} as computed in Algorithm 1, we want to compare it to an optimal solution T_{Opt} for the new instance. For this comparison, we deal with every subtree of T_{Opt} rooted in a neighbor of v_{New}, together with its connection to v_{New}, separately. For our estimations, we need the following technical lemma which is a generalization of Lemma 4.

Lemma 5. *Let G be a graph and let $S \subseteq V$ be a set of terminals in G. Let T be a subtree of G rooted in some non-terminal vertex x of degree ≥ 2, and let $S_T = S \cap V(T)$ be the set of terminals in T. Let $v_{\text{New}} \in V - V(T)$ be one additional vertex. By T' we denote the tree $(V(T) \cup \{v_{\text{New}}\}, E(T) \cup \{\{v_{\text{New}}, x\}\})$. Let x_{first} be the first and x_{last} be the last terminal in T as found by a depth-first search starting from x. Then there exists a connected subgraph H with $V(H) = \{v_{\text{New}}\} \cup S_T$ with costs $c(H) \leq \beta \cdot c(v_{\text{New}}, x) + 2\beta \cdot c(T) - \beta \cdot c(x, x_{\text{last}})$.*

Proof. According to Lemma 4, there exists a connected subgraph H' of G on the vertices of S_T with cost $c(H') \leq 2\beta \cdot c(T)$.

There exists a maximal subtree T_x of T consisting of non-terminals only which contains x as described in the proof of Lemma 4. Moreover, there exists a cycle C_x on the neighbors of T_x such that the edge $\{x_{\text{first}}, x_{\text{last}}\}$ is contained in C_x.

From C_x, we construct a path P_x containing $N(T_x)$ by deleting the edge $\{x_{\text{first}}, x_{\text{last}}\}$. The desired graph H is now obtained from H' by substituting P_x for C_x and adding the edge $\{v_{\text{New}}, x_{\text{first}}\}$ (see Fig.4). It remains to show that the cost of H satisfies the constraint of the lemma. By T'_x we denote the subtree of T on the vertices from $V(T_x) \cup N(T_x)$.

As already shown in the proof of Lemma 4, analyzing a depth-first traversal of T'_x leads to $c(C_x) \leq 2\beta \cdot c(T'_x)$. Without loss of generality, such a depth-first
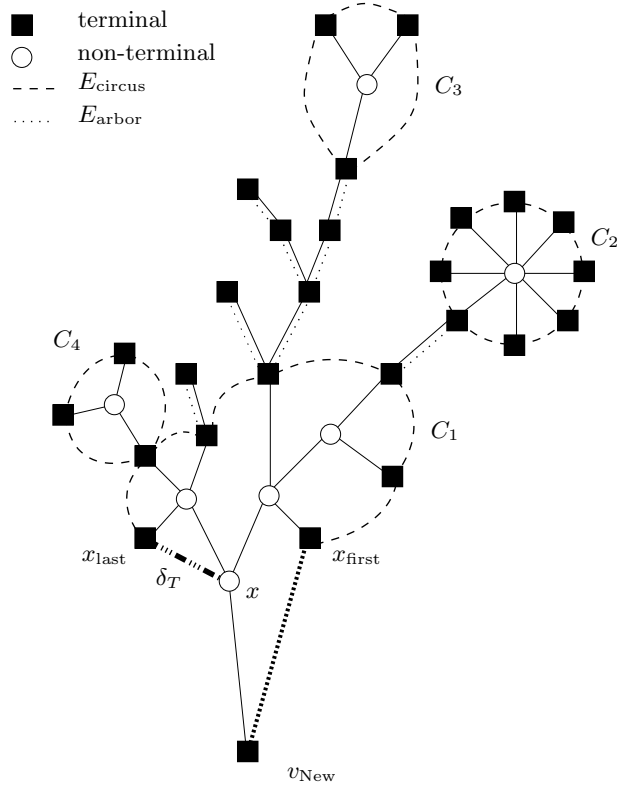
Fig. 4. The construction in the proof of Lemma 5

traversal may start from vertex x. The vertices x_{first} and x_{last} are the first and the last terminals on the traversal path. This traversal path starts with a simple path $P_{x,x_{\text{first}}}$ from x to x_{first} and ends with a simple path $P_{x_{\text{last}},x}$ from x_{last} to x.

In the proof of Lemma 4, we have seen that shortening the subpath of the traversal path between two consecutive terminals on the traversal path generates an edge whose cost can be bounded from above by β times the cost of the corresponding subpath.

Summing over all these shortened edges on the path P_x from x_{first} to x_{last} yields a cost of at most $2\beta \cdot c(T'_x) - \beta \cdot c(P_{x,x_{\text{first}}}) - \beta \cdot c(P_{x_{\text{last}},x})$, since every edge of T'_x is part of exactly two of the corresponding subpaths for the complete cycle C_x, and only the edges on the paths $P_{x,x_{\text{first}}}$ and $P_{x_{\text{last}},x}$ are used exactly once, since the cost of the edge from x_{last} to x_{first} is not included in the above sum.

As already mentioned above, H is obtained by adding the edge $\{v_{\text{New}}, x_{\text{first}}\}$ to P_x. Thus, $c(H) = c(P_T) + c(v_{\text{New}}, x_{\text{first}})$, moreover

$$c(v_{\text{New}}, x_{\text{first}}) \leq \beta \cdot (c(v_{\text{New}}, x) + c(P_{x_{\text{first}},x})).$$

From this we have

$$c(H) \leq 2\beta \cdot c(T'_x) - \beta \cdot c(P_{x,x_{\text{first}}}) - \beta \cdot c(P_{x_{\text{last}},x}) + \beta \cdot (c(v_{\text{New}}, x) + c(P_{x_{\text{first}},x}))$$
$$\leq 2\beta \cdot c(T'_x) + \beta \cdot c(v_{\text{New}}, x) - \beta \cdot c(x, x_{\text{last}})$$
$$\leq 2\beta \cdot c(T'_x) + \beta \cdot c(v_{\text{New}}, x) - \beta \cdot \delta_T.$$

□

Theorem 5. *For any* $1/2 \leq \beta \leq 1$, *Algorithm 1 is a linear-time* $\left(\frac{1}{2} + \beta\right)$-*approximation algorithm for Min-Δ_β-STRP-AddNonTerm.*

Proof. Constructing a spanning tree T_{MST} can be done in linear time, where the size of the input is measured in the number of edges. Let T_{Opt} be an optimal solution for the new instance (G_{New}, S, c), and let T_{Alg} be the Steiner tree computed by Algorithm 1. If v_{New} does not occur in T_{Opt}, then T_{Old} and thus also T_{Alg} is an optimal solution.

Thus, we assume that $v_{\text{New}} \in V(T_{\text{Opt}})$. Let $\{x_1, \ldots x_k\}$ be the set of neighbors of v_{New} in T_{Opt}. By removing v_{New} from T_{Opt}, we get a forest of k trees $T_1, \ldots, T_k$, where T_i is rooted in x_i. Let γ_1 denote the cost of all edges adjacent to v_{New} in T_{Opt}, i. e., $\gamma_1 = \sum_{i=1}^k c(v_{\text{New}}, x_i)$, and let $\gamma_2 = \sum_{i=1}^k c(T_i)$ denote the sum of costs of all trees T_i. Thus, the cost of the optimal solution satisfies $c(T_{\text{Opt}}) = \gamma_1 + \gamma_2$. We get a solution for the old instance by connecting the vertices x_1 to x_k by a path $P = (x_1, \ldots, x_k)$. The cost of this path can be estimated as $c(P) \leq 2\beta \cdot \gamma_1$. The path P together with the trees T_i constitute a solution of cost greater than or equal to $c(T_{\text{Old}})$, this implies

$$c(T_{\text{Old}}) \leq c(P) + \sum_{i=1}^k c(T_i) \leq 2\beta \cdot \gamma_1 + \gamma_2. \tag{1}$$

For each tree T_i, we can construct a graph H_i on the terminals of T_i together with v_{New} as described in Lemma 5. If the vertex x_i of T_i is a non-terminal,

$$c(H_i) \leq \beta \cdot c(v_{\text{New}}, x_i) + 2\beta \cdot c(T_i) \tag{2}$$

follows from Lemma 5, for all $\frac{1}{2} \leq \beta \leq 1$. If the root vertex x_i is a terminal, adding the edge $\{v_{\text{New}}, x_i\}$ to the graph constructed according to Lemma 4 yields

$$c(H_i) \leq c(v_{\text{New}}, x_i) + 2\beta \cdot c(T_i) \tag{3}$$

for all $\frac{1}{2} \leq \beta \leq 1$. Note that (2) implies (3) also for the trees rooted in a non-terminal.

Let H be the union of the graphs $H_1, \ldots, H_k$ as a subgraph of G, i. e., $V(H) = S \cup \{v_{\text{New}}\}$ and $E(H) = \bigcup_{i=1}^k E(H_i)$. Then

$$c(T_{\text{MST}}) \leq c(H) = \sum_{i=1}^k c(H_i) \leq \sum_{i=1}^k (c(v_{\text{New}}, x_i) + 2\beta \cdot c(T_i))$$
$$\leq \sum_{i=1}^k c(v_{\text{New}}, x_i) + 2\beta \cdot \sum_{i=1}^k c(T_i)$$
$$\leq \gamma_1 + 2\beta \cdot \gamma_2. \tag{4}$$

Summing Equations (1) and (4) yields

$$2 \cdot c(T_{\text{Alg}}) \le c(T_{\text{Old}}) + c(T_{\text{MST}}) \le 2\beta \cdot \gamma_1 + \gamma_2 + \gamma_1 + 2\beta \cdot \gamma_2$$
$$\le (1 + 2\beta) \cdot (\gamma_1 + \gamma_2) = (1 + 2\beta) \cdot c(T_{\text{New}})$$

and thus $c(T_{\text{Alg}}) \le \frac{1+2\beta}{2} \cdot c(T_{\text{New}}) = \left(\frac{1}{2} + \beta\right) \cdot c(T_{\text{New}})$. $\qquad\square$

Adding a Terminal. In this section, we consider the case where the inserted vertex is a terminal. Here, the old optimal solution is not feasible for the new instance. Therefore, we analyze two different candidates for a good feasible solution. The first candidate is a minimum spanning tree for the new instance on all terminals including the newly added one. The second candidate is obtained by connecting the inserted terminal to a vertex in the old optimal solution by the cheapest edge possible. This procedure is shown in Algorithm 2.

Algorithm 2 Approximation algorithm for Min-Δ_β-STRP-AddTerm

Input: $(G = (V, E), S, c), T_{\text{Old}}, (G_{\text{New}} = (V \cup v_{\text{New}}, E \cup \{\{v_{\text{New}}, x\} \mid x \in V\}), S \cup \{v_{\text{New}}\}, c)$
 1: Compute the minimum spanning tree T_{MST} over the set of vertices $S \cup \{v_{\text{New}}\}$.
 2: Compute $T_{\text{Old}+}$ from T_{Old} by choosing the cheapest edge connecting v_{New} with S.
 3: Compute the best solution T_{Alg} between T_{Old} and T_{MST}.
Output: The Steiner tree T_{Alg}.

For analyzing the approximation ratio of Algorithm 2, we compare the costs of the computed solutions T_{MST} and $T_{\text{Old}+}$ to the costs of an optimal solution for the new instance. We distinguish two cases for the proof according to whether there exists a non-terminal among the neighbors of v_{New} in an optimal solution T_{Opt} for the new instance or not. The proof for the case where there is a neighboring non-terminal will again make use of Lemma 5.

Theorem 6. *For any $\frac{1}{2} \le \beta \le 1$, Algorithm 2 is a linear-time $\left(\frac{1}{2} + \beta\right)$-approximation algorithm for Min-Δ_β-STRP-AddTerm.*

Proof. The construction of both T_{MST} and $T_{\text{Old}+}$ is obviously possible in linear time. Let T_{Opt} be an optimal solution of the new minimum Steiner tree instance. As in the proof of Theorem 5, let T_{Old} be the optimal solution of the old instance and let T_{MST} and $T_{\text{Old}+}$ be the outputs of step 2 and step 3 of Algorithm 2, respectively. Let $\{x_1, \ldots, x_k\}$ be the neighbors of v_{New} in T_{Opt}. By deleting v_{New} from T_{Opt}, we get a set of trees $T_1, \ldots, T_k$. In the following, we denote the set of terminal neighbors of v_{New} in T_{Opt} by Γ_{T} and the set of non-terminal neighbors by Γ_{N}. We denote by γ_1 the sum of the costs of edges connecting v_{New} with a terminal in T_{Opt}, i.e., $\gamma_1 = \sum_{x_i \in \Gamma_{\text{T}}} c(v_{\text{New}}, x_i)$, and by γ_2 the sum of costs of edges connecting v_{New} with a non-terminal in T_{Opt}, i.e., $\gamma_2 =$
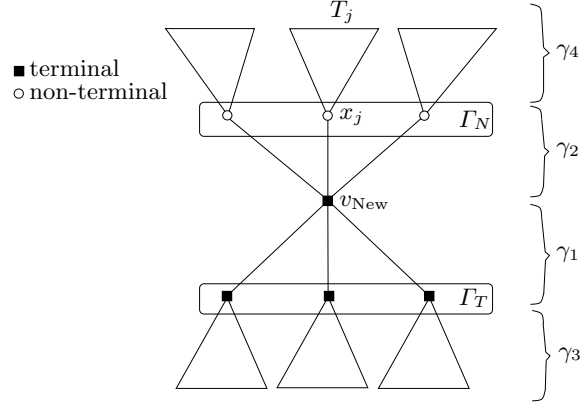
Fig. 5. The structure of the new optimal solution in the proof of Theorem 6

$\sum_{x_i \in \Gamma_N} c(v_{\text{New}}, x_i)$. Let γ_3 be the sum of the costs of all trees T_i where the root $x_i \in \Gamma_T$, and let γ_4 be the sum of the costs of all trees T_j where the root $x_j \in \Gamma_N$ (see Fig. 5). Hence, $c(T_{\text{Opt}}) = \gamma_1 + \gamma_2 + \gamma_3 + \gamma_4$.

We distinguish two cases according to the number of non-terminals adjacent to v_{New}.

Case 1: $|\Gamma_N| = 0$, ($\gamma_2 = \gamma_4 = 0$). First we look at the case where all vertices x_i in the neighbourhood of v_{New} are terminals. This implies $c(T_{\text{New}}) = \gamma_1 + \gamma_3$. In the case, where there is only one terminal x in the neighborhood of v_{New} in T_{Opt}, Algorithm 2 connects v_{New} by the edge $\{v_{\text{New}}, x\}$ in step 3 and finds the optimal solution. Thus, we may assume that $|\Gamma_T| \geq 2$. In order to estimate $c(T_{\text{MST}})$, for each tree T_i, we first construct a graph H_i containing the terminals of T_i. By connecting v_{New} to H_i by the edge $\{v_{\text{New}}, x_i\}$, we get a graph H which consists of all terminals in G_{New}, i.e., $V(H) = S \cup \{v_{\text{New}}\}$. It is obvious that the cost of T_{MST} is smaller than $c(H)$. Now we can estimate $c(T_{\text{MST}})$ as follows:

$$c(T_{\text{MST}}) \leq \sum_{i=1}^{k} c(v_{\text{New}}, x_i) + \sum_{i=1}^{k} c(H_i). \tag{5}$$

According to Lemma 4, the cost of H_i satisfies $c(H_i) \leq 2\beta \cdot c(T_i)$, and thus $\sum_{i=1}^{k} c(H_i) \leq 2\beta \cdot \gamma_3$. Together with Equation (5), we get

$$c(T_{\text{MST}}) \leq \gamma_1 + 2\beta \cdot \gamma_3. \tag{6}$$

For the cost of $T_{\text{Old+}}$, we get

$$c(T_{\text{Old+}}) = c(T_{\text{Old}}) + c(v_{\text{New}}, x_{\text{first}}),$$

where $\{v_{\text{New}}, x_{\text{first}}\}$ is the cheapest edge from v_{New} to a terminal in G. To estimate $T_{\text{Old+}}$, we connect the terminals $x_1, \ldots, x_k$ by a path P. The union of

P with the trees T_i constitutes a feasible solution for the old instance. So we know that the cost of the old optimal solution is at most the cost of the union of P and all T_i, i.e., $c(T_{\text{Old}}) \leq c(P) + \sum_{i=1}^{k} c(T_i)$. For the cost of P, this implies $c(P) \leq 2\beta \cdot \sum_{i=1}^{k} c(v_{\text{New}}, x_i) - \beta \cdot c(v_{\text{New}}, x_l) - \beta \cdot c(v_{\text{New}}, x_m)$, where $x_l \neq x_m$, and thus $c(T_{\text{Old}}) \leq 2\beta \cdot \gamma_1 + \gamma_3 - \beta \cdot c(v_{\text{New}}, x_l) - \beta \cdot c(v_{\text{New}}, x_m)$. Without loss of generality, let $c(v_{\text{New}}, x_l) < c(v_{\text{New}}, x_m)$. Then we can estimate $c(T_{\text{Old+}})$ as

$$c(T_{\text{Old+}}) \leq 2\beta \cdot \gamma_1 + \gamma_3 - 2\beta \cdot c(v_{\text{New}}, x_l) + c(v_{\text{New}}, x_l) \leq 2\beta \cdot \gamma_1 + \gamma_3. \quad (7)$$

By adding Equations (6) and (7), we get

$$2 \cdot c(T_{\text{Alg}}) \leq c(T_{\text{MST}}) + c(T_{\text{Old+}}) \leq \gamma_1 + 2\beta \cdot \gamma_3 + 2\beta \cdot \gamma_1 + \gamma_3$$
$$\leq (1 + 2\beta) \cdot (\gamma_1 + \gamma_3) = (1 + 2\beta) \cdot c(T_{\text{New}})$$

and thus $c(T_{\text{Alg}}) \leq (1 + 2\beta)/2 \cdot c(T_{\text{New}}) = (1/2 + \beta) \cdot c(T_{\text{New}})$.

Case 2: $|\Gamma_N| \geq 1$. Let $\Gamma_T = \{x_1, \ldots, x_f\}$ and $\Gamma_N = \{x_{f+1}, \ldots, x_k\}$ be the sets of terminals and non-terminals in the neighborhood of v_{New}, respectively, for some $0 \leq f < k$ in T_{Opt}. To estimate the cost of T_{MST}, we first construct, for each tree T_i with $i \in \{1, \ldots, f\}$, a spanning graph as already described in case 1. For each tree T_j with $j \in \{f + 1, \ldots, k\}$, we construct a spanning graph connected with v_{New} as described in Lemma 5. This implies

$$c(T_{\text{MST}}) \leq \gamma_1 + 2\beta \cdot \gamma_3 + \beta \cdot \gamma_2 + 2\beta \cdot \gamma_4 - \sum_{j=f+1}^{k} \beta \cdot c(x_j, x_{\text{last} j}), \quad (8)$$

where $x_{\text{last} j}$ is the last terminal of a depth-first traversal of T_j as described in Lemma 5, for $j \in \{f + 1, \ldots, k\}$. For estimating the cost of $T_{\text{Old+}}$, we connect the vertices $x_1, \ldots, x_k$ by a path P, analogously to case 1. The union of P and the trees T_i, for $i = 1, \ldots, k$, gives a feasible solution for the old instance. Thus, $c(T_{\text{Old}}) \leq c(P) + \gamma_3 + \gamma_4$. The cost of P can be bounded from above by $c(P) \leq 2\beta \cdot (\gamma_1 + \gamma_2) - \beta \cdot c(v_{\text{New}}, x_j)$, where $x_j \in \{f + 1, \ldots, k\}$.
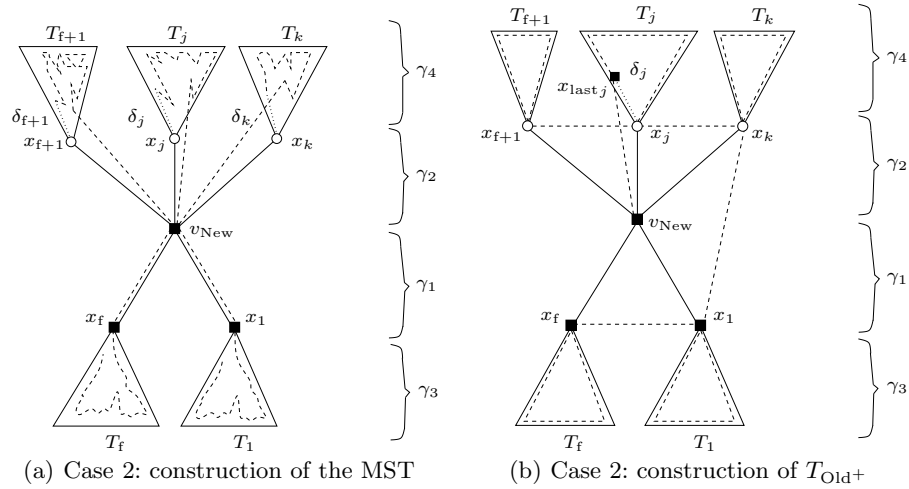
From this, we get

$$c(T_{\text{Old+}}) \leq 2\beta(\gamma_1 + \gamma_2) + \gamma_3 + \gamma_4 - \beta \cdot c(v_{\text{New}}, x_i) + c(v_{\text{New}}, x_{\text{last} i})$$
$$\leq 2\beta(\gamma_1 + \gamma_2) + \gamma_3 + \gamma_4 - \beta \cdot c(v_{\text{New}}, x_i) + \beta \cdot (c(v_{\text{New}}, x_i) + c(x_i, x_{\text{last} i}))$$
$$\leq 2\beta(\gamma_1 + \gamma_2) + \gamma_3 + \gamma_4 + c(x_i, x_{\text{last} i}). \quad (9)$$

Adding Equations (8) and (9) yields

$$2 \cdot c(T_{\text{Alg}}) \leq c(T_{\text{MST}}) + c(T_{\text{Old+}})$$

$$\leq (2\beta + 1) \cdot \gamma_1 + 3\beta \cdot \gamma_2 + (2\beta + 1) \cdot (\gamma_3 + \gamma_4) + c(x_i, x_{\text{last} i}) - \sum_{i=f+1}^{k} \beta \delta_i$$

$$\leq (1 + 2\beta)(\gamma_1 + \gamma_2 + \gamma_3 + \gamma_4) \leq (1 + 2\beta) \cdot c(T_{\text{New}}),$$

and thus $c(T_{\text{Alg}}) \leq (1 + 2\beta)/2 \cdot c(T_{\text{New}}) \leq (1/2 + \beta) \cdot c(T_{\text{New}})$. $\qquad \square$

(a) Case 2: construction of the MST   (b) Case 2: construction of $T_{\mathrm{Old}+}$

Fig. 6. The two feasible solutions in case 2 of the proof of Theorem 6

References

1. C. Archetti, L. Bertazzi, and M. G. Speranza. Reoptimizing the traveling salesman problem. *Networks*, 42(3):154–159, 2003.
2. C. Archetti, L. Bertazzi, and M. G. Speranza. Reoptimizing the 0-1 knapsack problem. Technical Report 267, University of Brescia, 2006.
3. G. Ausiello, B. Escoffier, J. Monnot, and V. T. Paschos. Reoptimization of minimum and maximum traveling salesman's tours. In L. Arge and R. V. Freivalds, editors, *Proc. of the 10th Scandinavian Workshop on Algorithm Theory (SWAT 2006)*, volume 4059 of *Lecture Notes in Computer Science*, pp. 196–207. 2006.
4. M. W. Bern and P. E. Plassmann. The Steiner problem with edge lengths 1 and 2. *Information Processing Letters*, 32(4):171–176, 1989.
5. D. Bilò, H.-J. Böckenhauer, J. Hromkovič, R. Královič, T. Mömke, P. Widmayer, and A. Zych. Reoptimization of Steiner trees. In J. Gudmundsson, editor, *Proc. of the 11th Scandinavian Workshop on Algorithm Theory (SWAT 2008)*, volume 5124 of *Lecture Notes in Computer Science*, pp. 258–269. 2008.
6. D. Bilò, H.-J. Böckenhauer, D. Komm, R. Královič, T. Mömke, S. Seibert, and A. Zych. Reoptimization of the shortest common superstring problem. In *Proc. of the 20th Annual Symposium on Combinatorial Pattern Matching (CPM 2009)*, volume 5577 of *Lecture Notes in Computer Science*, pp. 78–91. 2009.
7. D. Bilò, P. Widmayer, and A. Zych. Reoptimization of weighted graph and covering problems. In *Proc. of the 6th Workshop on Approximation and Online Algorithms (WAOA 2008)*, volume 5426 of *Lecture Notes in Computer Science*, pp. 201–213. 2009.
8. H.-J. Böckenhauer, L. Forlizzi, J. Hromkovič, J. Kneis, J. Kupke, G. Proietti, and P. Widmayer. Reusing optimal TSP solutions for locally modified input instances (extended abstract). In G. Navarro, L. E. Bertossi, and Y. Kohayakawa, editors, *Proc. of the 4th IFIP International Conference on Theoretical Computer Science (TCS 2006)*, volume 209 of *IFIP*, pp. 251–270. 2006.

9. H.-J. Böckenhauer, J. Hromkovič, R. Klasing, S. Seibert, and W. Unger. Approximation algorithms for TSP with sharpened triangle inequality. *Information Processing Letters*, 75:133–138, 2000.

10. H.-J. Böckenhauer, J. Hromkovič, R. Královič, T. Mömke, and P. Rossmanith. Reoptimization of Steiner trees: Changing the terminal set. *Theoretical Computer Science*, 410(36):3428–3435, 2009.

11. H.-J. Böckenhauer, J. Hromkovič, T. Mömke, and P. Widmayer. On the hardness of reoptimization. In *Proc. of the 34th International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM 2008)*, volume 4910 of *Lecture Notes in Computer Science*, pp. 50–65. 2008.

12. H.-J. Böckenhauer and D. Komm. Reoptimization of the metric deadline TSP. In *Mathematical Foundations of Computer Science 2008. Proc. of the 33rd International Symposium (MFCS 2008)*, volume 5162 of *Lecture Notes in Computer Science*, pp. 156–167. 2008.

13. B. Escoffier, M. Milanič, and V. T. Paschos. Simple and fast reoptimizations for the Steiner tree problem. *Algorithmic Operations Research*, 4(2):86–94, 2009.

14. U. Feige. A threshold of $\ln n$ for approximating set cover. *Journal of the ACM*, 45(4):634–652, 1998.

15. M. M. Halldórsson, S. Ueno, H. Nakao, and Y. Kajitani. Approximating Steiner trees in graphs with restricted weights. *Networks*, 31(4):283–292, 1998.

16. F. K. Hwang, D. S. Richards, and P. Winter. *The Steiner Tree Problems*, volume 53 of *Annals of Discrete Mathematics*. North-Holland, 1992.

17. R. M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of computer computations*, pp. 85–103, 1972.

18. H. J. Prömel and A. Steger. *The Steiner Tree Problem*. Advanced Lectures in Mathematics. Friedr. Vieweg & Sohn, 2002.

19. G. Robins and A. Z. Zelikovsky. Improved Steiner tree approximation in graphs. In *Proc. of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2000)*, pp. 770–779. 2000.

20. M. W. Schäffter. Scheduling with forbidden sets. *Discrete Applied Mathematics*, 72(1-2):155–166, 1997.

21. V. V. Vazirani. *Approximation Algorithms*. Springer-Verlag, 2004.