Robust Reoptimization of Steiner Trees*

Keshav Goyal[1] and Tobias Mömke[2]

1 **IIT Delhi, India,** mt5100599@maths.iitd.ac.in
2 **Saarland University, Germany,** moemke@cs.uni-saarland.de

Abstract

In reoptimization problems, one is given an optimal solution to a problem instance and a local modification of the instance. The goal is to obtain a solution for the modified instance. The additional information about the instance provided by the given solution plays a central role: we aim to use that information in order to obtain better solutions than we are able to compute from scratch.

In this paper, we consider Steiner tree reoptimization and address the optimality requirement of the provided solution. Instead of assuming that we are provided an optimal solution, we relax the assumption to the more realistic scenario where we are given an approximate solution with an upper bound on its performance guarantee.

We show that for Steiner tree reoptimization there is a clear separation between local modifications where optimality is crucial for obtaining improved approximations and those instances where approximate solutions are acceptable starting points. For some of the local modifications that have been considered in previous research, we show that for every fixed $\epsilon > 0$, approximating the reoptimization problem with respect to a given $(1+\epsilon)$-approximation is as hard as approximating the Steiner tree problem itself (whereas with a given optimal solution to the original problem it is known that one can obtain considerably improved results). Furthermore, we provide a new algorithmic technique that, with some further insights, allows us to obtain improved performance guarantees for Steiner tree reoptimization with respect to all remaining local modifications that have been considered in the literature: a required node of degree more than one becomes a Steiner node; a Steiner node becomes a required node; the cost of one edge is increased.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems, G.2.2 Graph Theory

Keywords and phrases reoptimization, approximation algorithms, Steiner tree problem, robustness

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2015.10

1 Introduction

The Steiner tree problem (STP) is one of the most studied problems in the area of network design. We are given a graph G with nodes $V(G)$, edges $E(G)$, and a cost function $c\colon E(G) \to \mathbb{R}_{\geq 0}$, as well as a set $R \subseteq V(G)$ of required nodes (also called regular nodes or terminals). The objective is to find a minimum cost tree T within G such that $R \subseteq V(T)$. The Steiner tree problem is known to be APX-hard [8], and the currently best approximation algorithm has a performance guarantee of $\ln 4 + \epsilon \approx 1.387$ [24].

* Research partially funded by Deutsche Forschungsgemeinschaft grant BL511/10-1 and by the Indo-German Max Planck Center for Computer Science (IMPECS).

We consider the Steiner tree problem with respect to reoptimization, a framework for dynamic algorithms in the context of NP-hard problems. We are given two related instances I and I' of an algorithmic problem together with a solution Sol to the instance I, and our goal is to compute a solution to I'. The relation between I and I' is determined by an operation that we call *local modification*.

The concept of reoptimization is motivated by the observation that instead of computing new solutions from scratch, oftentimes we can reuse the effort spent to solve problems similar to the one at hand. For instance, let us consider a large circuit where certain components have to be connected. The components are the required nodes and there are points that may be used by several connections, the Steiner nodes. Now suppose that a long and costly computation has led to an almost optimal solution. Afterwards the requirements change: either an additional component has to be placed to a point that previously was a Steiner node or a component is removed, which turns a required node into a Steiner node. In such a situation it would seem wasteful to discard the entire previous effort.

Classically, when considering reoptimization problems one assumes that Sol is an optimal solution. The reason for this assumption is that assuming optimality considerably reduces the formal overhead and therefore facilitates to concentrate on the main underlying properties of the reoptimization problem. We show, however, that assuming optimality is not without loss of generality. Let us assume that $c(\text{Sol})$ is a $(1+\epsilon)$ factor larger than the cost of an optimal solution. Then we say that a Steiner tree reoptimization algorithm is *robust*, if it is an approximation algorithm and its performance guarantee is $\alpha \cdot (1 + O(\epsilon))$, where α is its performance guarantee when $\epsilon = 0$. Intuitive, this definition ensures that for $\epsilon \to 0$, the performance guarantee converges smoothly towards α, independent of the given instance. We consider robustness of reoptimization algorithms to be a crucial feature, since in real world applications close to optimal solutions are much more frequent than optimal solutions.

We address all local modifications that have previously been considered for Steiner tree reoptimization. We classify these modifications into two groups, according to their robustness. The first group contains those problems where obtaining a robust reoptimization algorithm implies to provide an approximation algorithm for the (non-reoptimization) Steiner tree problem with matching performance guarantee. The second group of problems allows for improved robust reoptimization algorithms compared to STP approximation algorithms.

For all reoptimization problems of the second group that have previously been considered (and that are known to be NP-hard [15]), we provide robust reoptimization algorithms that, for $\epsilon \to 0$, obtain better performance guarantees than the previous results with optimality assumption [12, 13].

1.1 Local Modifications and Our Contribution

There are ten local modifications that previously have been considered for the Steiner tree problem. The two most studied modifications address the set of required nodes: we either declare a required node to be a Steiner node, or a Steiner node to be a required node. Here, STP^{R-} resp. STP^{R+} denote the corresponding reoptimization problems. We show, in Section 4, that finding a robust reoptimization algorithm for STP^{R-} is as hard as finding a Steiner tree approximation algorithm with matching approximation ratio. If one, however, excludes that the node t declared to be a Steiner node is a leaf in the given instance, we provide a robust reoptimization algorithm with improved performance ratio (see Table 1 for an overview of the achieved improvements). We show that in contrast to STP^{R-}, STP^{R+} always allows for improved robust reoptimization algorithms. The next interesting type of local modification is to modify the cost of a single edge. We do not require the cost function

to be metric. In particular, in the shortest path metric induced by the modified edge cost, the cost of several edges may be changed. We call the modification where the cost of one edge is increased STP^{E+}, and the converse local modification where the cost of one edge is decreased is STP^{E-}. We provide an improved robust reoptimization algorithm for STP^{E+} and show that robust reoptimization for STP^{E-} is as hard as approximating the Steiner tree problem itself (analogous to general STP^{R-}). The two local modifications to remove an edge from the graph and to add an edge to the graph reduce to STP^{E+} resp. STP^{E-} in a straightforward manner.

The remaining four local modifications are the removal or addition of a required node or a Steiner node. It is known that the local modification where required or Steiner nodes are removed is as hard as Steiner tree approximation, even if we are given an optimal solution to the old problem [15]. We show that adding a required node or a Steiner node to the graph causes robust reoptimization to be as hard as STP approximation.

One of the key insights that leads to our improved algorithms is that for all local modifications that allow for robust reoptimization algorithms, we can replace the given Steiner tree by a k-restricted Steiner tree of roughly the same cost. At the same time, we have the promise that there is an almost optimal Steiner tree for the modified instance that is k-restricted. This property allows us to handle certain subgraphs of Steiner trees called full components. (i) We remove entire full components from the given Steiner tree and perform optimal computations to obtain a feasible solutions to the modified instance, and (ii) we *guess* entire full components of the Steiner tree that we aim to compute. The new insights simplify and generalize the previous approaches to Steiner tree reoptimization and therefore give raise to more sophisticated analyses than before.[1]

Due to space constraints, we restrict the presentation to analyzing STP^{R-} and STP^{E+}, as these local modification give the best overview of the used techniques and ideas.

1.2 Related Work

The concept of reoptimization was first mentioned by Schäffter [29] in the context of postoptimality analysis for a scheduling problem. Since then, the concept of reoptimization has been investigated for several different problems, including the traveling salesman problem [1, 5, 14, 18, 7, 28], the rural postman problem [3], fast reoptimization of the spanning tree problem [23], the knapsack problem [2], covering problems [11], the shortest common superstring problem [10], maximum weight induced heredity problems [21], and scheduling [29, 6, 20]. There are several overviews on reoptimization [4, 22, 17, 31].

The Steiner tree reoptimization problem in general weighted graphs was previously investigated in [9, 26, 16, 15, 12, 13], see Table 1.

2 Preliminaries

We denote a Steiner tree instance by (G, R, c), where G is an undirected graph, $R \subseteq V(G)$ is the set of required nodes, and $c\colon E(G) \to \mathbb{R}_{\geq 0}$ is a cost function. The *Steiner nodes* of (G, R, c) are the nodes $S = V(G) \setminus R$.

Since c is symmetric, we sometimes use the simplified notation $c(u, v) = c(v, u)$ instead of $c(\{u, v\})$.

[1] We note that with some additional effort, it would also be possible to adapt the technique of Bilò and Zych [13] and use them for our results.

Table 1 Comparison of approximation ratios of the Steiner Tree Reoptimization problem for the different types of local modifications. To increase the readability, all values ϵ, δ in the approximation ratios are omitted. The numerical values are rounded up at the third digit and we assume $\beta = 1.387$, the approximation ratio $\ln(4) + \epsilon$ of the Steiner tree approximation algorithm of Byrka et al. [24] with small enough ϵ.

Local Modification	**Our Results** **Sol: $(1+\epsilon)$-Approx.**		**Previous Results** **Sol: Optimal Solution**	
	Expression	**Value**	**Expression**	**Value**
STP^{R-} (internal node)	$\frac{10\beta-7}{7\beta-4}$	1.204	$\frac{3\beta-2}{2\beta-1}$ [13]	1.219
STP^{R-} (leaf node)	not robust	1.204 if $\epsilon = 0$	$\frac{3\beta-2}{2\beta-1}$ [13]	1.219
STP^{R+}	$\frac{10\beta-7}{7\beta-4}$	1.204	$\frac{3\beta-2}{2\beta-1}$ [12]	1.219
STP^{E+}	$\frac{7\beta-4}{4\beta-1}$	1.256	$\frac{2\beta-1}{\beta}$ [13]	1.29
STP^{E-}	not robust	1.387	$\frac{5\beta-3}{3\beta-1}$ [9] assuming metricity	1.246
Add Node	not robust	1.387	without [24]: 1.5 [26]	1.387
Remove Node	not robust	1.387	As hard as STP approx. [15]	1.387

For two graphs G, G', we define $G \cup G'$ to be the graph with node set $V(G) \cup V(G')$ and edge set $E(G) \cup E(G')$ (i. e., we do not keep multiple edges). For an edge e, $G - e$ is G with e removed from $E(G)$. We define $G - G'$ to be the graph with node set $V(G) \setminus (V(G') \cap S)$ and edge set $E(G) \setminus E(G')$. We emphasize that we do *not* remove required vertices.

In Steiner tree algorithms, it is standard to consider the edge-costs to be metric. The reason is that forming the metric closure (i. e., using the shortest path metric) does not change the cost of an optimal solution: if we replacing an edge of a Steiner tree by the shortest path between the two ends, we obtain a valid Steiner tree again.

In the context of reoptimization, however, we cannot assume the cost function to be metric without loss of generality, because the triangle inequality restricts the effect of local changes. Therefore in the following we have to carefully distinguish between metric and general cost functions.

For a given Steiner tree, its *full components* are exactly those maximal subtrees that have all leaves are in R and all internal nodes are in S. Note that for a given Steiner tree T, we may remove leaves if they are not in R; we still have a Steiner tree, and its cost did not increase. Therefore we may assume that T is composed of full components. A *k-restricted Steiner tree* is a Steiner tree where each full component has at most k nodes from R.

▶ **Lemma 1** (Borchers, Du [19]). *For an arbitrary $\epsilon > 0$ there is a $k \in O_\epsilon(1)$ such that for all Steiner tree instances (G, R, c) with optimal solution* Opt *of cost* opt *and c is a metric, there is a k-restricted Steiner tree T of cost at most $(1+\epsilon)$*opt *which can be obtained from* Opt *in polynomial time.*

We assume that in k-restricted Steiner trees T where c is a metric, the Steiner nodes $v \in V(T) \cap S$ have a degree of $\deg(v) \geq 3$. This is without loss of generality, since $\deg(v) \geq 2$ by the definition of k-restricted Steiner trees; if $\deg(v) = 2$ and u, w are the neighbors of v, $c(u, v) + c(v, w) \geq c(u, w)$. We replace $\{u, v\}, \{v, w\}$ by $\{u, w\}$ without increasing the cost of T and without changing the property that T is k-restricted.

Input : A Steiner tree instance (G, R, c),
a Steiner forest F in G with trees $F_1, F_2, \ldots, F_\ell$
Output : A Steiner tree T
Set $G' := G/F$ such that F_i is contracted to v_i;
Set $R' := \{v_i : V(F_i) \cap R \neq \emptyset\}$;
Compute a minimum Steiner tree T' of (G', R', c);
Obtain T from T' by expanding F.

Algorithm 1: Connect

Within the entire text, Opt denotes an optimal solution and opt denotes the cost of an optimal solution. We will often add sub- and superscripts to Opt and opt in order to distinguish between various types of (close to) optimal solutions.

3 Connecting Forests and Guessing Components

We state two algorithms that we will use repeatedly within the subsequent sections. The first algorithm, Connect, was introduced by Böckenhauer et al. [16] and has been used in all previous Steiner tree reoptimization results. The algorithm connects components of a Steiner forest F of G in order to obtain a feasible Steiner tree T. The idea is that we start from a partial solution with few components that together contain all required vertices, and we use an exact computation to complete the solution. In Connect we use the following notation. Denote by G/V' for $V' \subseteq V(G)$ the contraction of V' in G. We write G/F instead of $G/V(F)$, if F is a subgraph of G. Note that after contracting a component there may be multiedges. Here, we treat multigraphs as simple graphs, where we only consider the cheapest edge of each multiedge. For ease of presentation, we slightly abuse notation and use the cost function c for both the graph before and the graph after the contraction.

Clearly, the graph T computed by Connect is a Steiner tree. If the number of components ℓ of the forest F given as input is a constant, by using the Dreyfuss-Wagner algorithm [25][2] to compute T', Connect runs in polynomial time. The graph T computed by Connect is the minimum cost Steiner tree that contains F, since all Steiner trees that contain F determine feasible solutions T'.

The second algorithm of this section, Guess, which is motivated from the CONNECT algorithm of [13] and presented here in a different manner, provides a mechanism to profit from guessing full components of an optimal k-restricted Steiner tree: we compress the guessed full components to single vertices and this way we obtain a new instance to which we apply known approximation algorithms. We call Guess by simply writing Guess(ℓ), if the instance and k are clear from the context and $\mathcal{A}$ is a β-approximation algorithm. Note that for instance Guess($3k$) means that $\ell = 3k$.

▶ **Lemma 2.** *For an arbitrary $\epsilon > 0$, let k be the parameter obtained from Lemma 1. Let $\mathcal{A}$ be a polynomial time β-approximation algorithm for the Steiner tree problem. Furthermore, let* Opt_k *be an optimal k-restricted solution of cost* opt_k *to the Steiner tree instance (G, R, c) where c is a metric. Then, for $\ell \in O_\epsilon(1)$,* Guess *runs in polynomial time and computes a Steiner tree T of cost at most $(1+\epsilon)(\beta - \beta\zeta + \zeta)\text{opt}$, where ζopt_k is the total cost of the ℓ maximum weight full components of* Opt_k *and* opt *is the cost of an optimal solution.*

[2] We refer to Hougardy et al. [27] for an overview of further exact Steiner tree algorithms that, depending on the given parameters, may be faster than Dreyfuss-Wagner.

Input : A Steiner tree instance (G, R, c) with c metric, numbers $\ell, k \in \mathbb{N}$, and a Steiner tree approximation algorithm $\mathcal{A}$
Output : A Steiner Tree T
Run $\mathcal{A}$ on (G, R, c) and obtain a Steiner tree T;
foreach $\mathcal{S} = \{S_1, S_2, \ldots, S_\ell\}$ *such that* $S_i \subseteq V$ *with* $|S_i| \leq 2k$ *and* $2 \leq |S_i \cap R| \leq k$ *for* $1 \leq i \leq \ell$ **do**
  For each i, compute a minimum spanning tree T_i with $V(T_i) = S_i$;
  Contract each T_i to a required node r_i;
  Run $\mathcal{A}$ on the resulting instance;
  Obtain T' by expanding the contracted components of each r_i;
  if $c(T') < c(T)$ **then**
    Replace T by T'.

Algorithm 2: GUESS

Proof. We first analyze the running time of the algorithm. Since $\mathcal{A}$ runs in polynomial time, we only have to consider the number of families $\mathcal{S}$ that we have to test. This number is bounded from above by $\left(\sum_{i=2}^{2k} \binom{n}{i}\right)^\ell$, since we only choose sets of size at most $2k$. Since both k and ℓ are constants, this number is polynomial in n.

Next we analyze the cost of T. Since we assume that for each Steiner node $v \in S \cap V(\text{OPT}_k)$, $\deg(v) \geq 3$, we conclude that all full components of OPT_k have at most $2k$ nodes. Therefore there is a family $\mathcal{S}$ considered by GUESS such that the classes of $\mathcal{S}$ are exactly the node sets of the ℓ maximum weight full components of OPT_k. Contracting a minimum spanning tree T_i is equivalent to contracting the full component with required nodes $R \cap S_i$ in OPT_k. We finish the proof by applying a standard argument that was used, for instance, by Böckenhauer et al. [14]. The cost of an optimal Steiner tree before expanding the full components is bounded from above by $\text{opt}_k - \zeta\text{opt}_k$, and expanding the full components adds ζopt_k. Therefore we obtain $c(T) \leq \beta(\text{opt}_k - \zeta\text{opt}_k) + \zeta\text{opt}_k = (\beta - \beta\zeta + \zeta)\text{opt}_k$. By our choice of k and Lemma 1, $\text{opt}_k \leq (1+\epsilon)\text{opt}$ and therefore $c(T) \leq (1+\epsilon)(\beta - \beta\zeta + \zeta)\text{opt}$. ◀

In the subsequent proofs, we will repeatedly obtain a value η such that $\zeta \geq (\alpha - 1 - \epsilon)\eta$, where α is the actual performance ratio of the considered approximation algorithm. By simple arithmetics and assuming that $(1+\epsilon)(\beta - \beta\zeta + \zeta)$ tends to $(\beta - \beta\zeta + \zeta)$ for ϵ chosen sufficiently small, Lemma 2 implies

$$\alpha \leq \frac{\beta + \beta\eta - \eta + \epsilon(\beta\eta - \eta)}{1 + \beta\eta - \eta}. \tag{1}$$

The reason for our assumption is that we can choose k in Lemma 1 and therefore the additional error is arbitrarily small.[3] We avoid complicated formalisms and instead slightly relax the approximation ratios in theorem statements by adding an arbitrarily small value $\delta > 0$ whenever the proofs use (1).

4 A Required Node Becomes a Steiner Node

The variant of the minimum Steiner tree reoptimization problem where a node is declared to be a Steiner node (STP_ϵ^{R-}) is defined as follows.

[3] Note that in contrast to the error from Lemma 1, we *cannot* control the error of the given solutions.

Input : An instance $(G, R, c, \textsc{Opt}_\epsilon^{\text{old}}, t)$ of STP_ϵ^{R-}
Output : A Steiner tree T
while $\deg_{\textsc{Opt}_\epsilon^{\text{old}}}(t) = 1$ **do** `// We assume that either` $\epsilon = 0$ `or` $\deg_{\textsc{Opt}_\epsilon^{\text{old}}}(t) > 1$
  Set $t' := \text{child}(t)$; `// The node adjacent to` t `in` $\textsc{Opt}_\epsilon^{\text{old}}$
  Remove t from $\textsc{Opt}_\epsilon^{\text{old}}$ and R, rename t' to t, and set $t \in R$;
  `// Now` $(G, R, c, \textsc{Opt}_\epsilon^{\text{old}}, t)$ `is the changed instance`
Transform $\textsc{Opt}_\epsilon^{\text{old}}$ to a k-restricted solution $\textsc{Opt}_{\epsilon,k}^{\text{old}}$ such that $\text{opt}_{\epsilon,k}^{\text{old}} \leq (1+\epsilon_k)\text{opt}_\epsilon^{\text{old}}$, where ϵ_k tends to 0 for large enough k;
Set $T_1 := \textsc{Opt}_\epsilon^{\text{old}}$; `// Note that` $\text{opt}_\epsilon^{\text{old}} \leq \text{opt}_{\epsilon,k}^{\text{old}}$
Let $C_1^t, C_2^t, \ldots$ be the full components of $\textsc{Opt}_{\epsilon,k}^{\text{old}}$ such that $t \in V(C_i^t)$ for all i and $c(C_i^t) \leq c(C_j^t)$ for $i < j$;
Set $F := \textsc{Opt}_{\epsilon,k}^{\text{old}} - C_1^t - C_2^t - C_3^t$; `// Ignore` C_3^t `if it does not exist`
Set $T_2 := \textsc{Connect}(F)$;
Set $T_3 := \textsc{Guess}(3k)$;
Set $T = T_i$ with $i = \min \arg_{j \in \{1,2,3\}}\{c(T_j)\}$.

Algorithm 3: DeclareSteiner

Given: A parameter $\epsilon > 0$, a Steiner tree instance (G, R, c), a solution $\textsc{Opt}_\epsilon^{\text{old}}$ to (G, R, c) such that $\text{opt}_\epsilon^{\text{old}} \leq (1+\epsilon)\text{opt}^{\text{old}}$, and a node $t \in R$.
Solution: A Steiner tree solution to $(G, R \setminus \{t\}, c)$.

An instance of STP_ϵ^{R-} is a tuple $(G, R, c, \textsc{Opt}_\epsilon^{\text{old}}, t)$. If $\epsilon = 0$, we skip the index and write STP^{R-}. Without loss of generality we assume that c is a metric: we may use the metric closure since the local modification does not change G or c.

The algorithm DeclareSteiner starts with reducing the instance to one where the changed required node has a degree of at least two, using a known technique. Afterwards it transforms the given solution to a k-restricted Steiner tree (note that the order of these two steps is important). The remaining algorithm outputs the best of three solutions that intuitively can be described as follows: we either keep the old solution; or we remove up to three full components incident to t to obtain a partial solution that we complete again using Connect; or we guess a partial solution that is at least as large as the $3k$ largest full-components of an optimal solution and complete these components to a solution using the best available approximation algorithm.

The following theorem indicates that in general we have to require $\epsilon = 0$ for instances of STP_ϵ^{R-} with $\deg(t) = 1$.

▶ **Theorem 3.** *For an arbitrary $\epsilon > 0$, let $\mathcal{A}$ be a polynomial time α-approximation algorithm for STP_ϵ^{R-}. Then there is a polynomial time α-approximation algorithm for the Steiner tree problem.*

Proof. Given a Steiner tree instance (G, R, c), let opt^{new} be the cost of an optimal solution. We construct a STP_ϵ^{R-} instance $(G', R', c', \textsc{Opt}_\epsilon^{\text{old}}, t)$ from (G, R, c). We first compute a minimum spanning tree $\tilde{T}$ of $G[R]$. Note that $G[R]$ is a complete graph since we assume c to be metric, and $c(\tilde{T}) \leq 2\text{opt}^{\text{new}}$, as shown by Takahashi and Matsuyama [30]; we assume w. l. o. g. that $\alpha < 2$. We obtain G' by combining G and a new node t as follows. We set $V(G') := V(G) \cup \{t\}$ and $E(G') = E(G) \cup \{t, t'\}$ for a node $t' \in R$. Then we obtain c' from c by setting $c'(t, t') = c(\tilde{T}) \cdot (1-\epsilon)/\epsilon$ and forming the metric closure. We set $R' = R \cup \{t\}$ and obtain a solution $\textsc{Opt}_\epsilon^{\text{old}}$ to (G', R', c') by adding $\{t, t'\}$ to $\tilde{T}$. Finally, we obtain the Steiner tree T by applying $\mathcal{A}$ to $(G', R', c', \textsc{Opt}_\epsilon^{\text{old}}, t)$.

Observe that T cannot contain an edge incident to t, since all of those edges are more expensive than $\tilde{T}$. Therefore T is a Steiner tree of (G, R, c). Conversely, all Steiner trees of (G, R, c) are feasible solutions to $(G', R', c', \textsc{Opt}_\epsilon^{\text{old}}, t)$. We conclude that T provides an α approximation, i. e., T is a feasible solution to (G, R, c) and $c(T) \leq \alpha \text{opt}^{\text{new}}$.

To finish the proof, we have to show that $\textsc{Opt}_\epsilon^{\text{old}}$ was a valid solution given to $\mathcal{A}$, i. e., its cost $\text{opt}_\epsilon^{\text{old}}$ is at most a factor $(1+\epsilon)$ larger than optimum. Clearly, $\textsc{Opt}_\epsilon^{\text{old}}$ is a Steiner tree of (G', R', c'). Let opt^{old} be the cost of an optimal Steiner trees for (G', R', c').

$$\frac{\text{opt}_\epsilon^{\text{old}}}{\text{opt}^{\text{old}}} = \frac{c(\tilde{T}) + c(t,t')}{\text{opt}^{\text{new}} + c(t,t')} \leq \frac{2\text{opt}^{\text{new}} + c(t,t')}{\text{opt}^{\text{new}} + c(t,t')} \leq 1 + \frac{\text{opt}^{\text{new}}}{\text{opt}^{\text{new}} + \text{opt}^{\text{new}} \cdot \frac{1-\epsilon}{\epsilon}} = 1 + \epsilon.$$

◀

For all remaining cases, DeclareSteiner profits from knowing $\textsc{Opt}_\epsilon^{\text{old}}$.

▶ **Theorem 4.** *Let $(G, R, c, \textsc{Opt}_\epsilon^{\text{old}}, t)$ be an instance of STP_ϵ^{R-} with $\deg_{\textsc{Opt}_\epsilon^{\text{old}}}(t) \geq 2$ or $\epsilon = 0$. Then, for an arbitrary $\delta > 0$, DeclareSteiner is an approximation algorithm for STP_ϵ^{R-} with performance guarantee*

$$\frac{(10\beta - 7 + 2\epsilon - 2\epsilon\beta)(1+\epsilon)}{7\beta - 4 + 5\epsilon - 2\epsilon\beta} + \delta.$$

For the approximation ratio $\beta = \ln(4) + \epsilon''$ from [24] with ϵ'' and δ chosen sufficiently small, we obtain an approximation ratio of less than $1.204 \cdot (1+\epsilon)$.

4.1 Proof of Theorem 4

Since k is a constant, all steps of DeclareSteiner except for the call of Connect clearly run in polynomial time. To see that also the call of Connect does, observe that removing the edges and Steiner nodes of a full component increases the number of connected components by at most $k-1$.

We continue with showing the claimed upper bound on the performance guarantee. Before we show the main result, we introduce two simplification steps. First, we show that we can restrict our attention to the case $\deg(t) = 2$ in $\textsc{Opt}_{\epsilon,k}^{\text{old}}$. Our analysis simultaneously gives a new proof for the previous best reoptimization result [13]. Subsequently we reduce the class of considered instances to those where all optimal solution to $(G, R \setminus \{t\}, c)$ have a special structure.

We start with analyzing the case where $\deg(t) = 1$. If this case appears in the while loop, by our assumption we have $\epsilon = 0$ and thus $\textsc{Opt}_\epsilon^{\text{old}}$ is an optimal solution. The transformation of DeclareSteiner within the while loop reduces the instance to one where $\deg(t) \geq 2$ [16]. When transforming the resulting solution $\textsc{Opt}_\epsilon^{\text{old}}$ to $\textsc{Opt}_{\epsilon,k}^{\text{old}}$, generally t could become a degree-one vertex. We use, however, that this is not the case when applying the algorithm of Borchers and Du [19]: The algorithm considers the full components separately, which implies that initially the degrees of all required vertices are one. Each full component is replaced by a graph where each required vertex has a degree of at least one. Consequently, the degree of no required vertex is decreased.

For the remaining proof, we assume $\deg_{\textsc{Opt}_{\epsilon,k}^{\text{old}}}(t) \geq 2$. We prove the following technical lemma, which is needed for our subsequent argumentation.

▶ **Lemma 5.** *There is a collection $\mathcal{C}$ of at most $3k$ full components of $\textsc{Opt}_k^{\text{new}}$ such that $F \cup \mathcal{C}$ is a connected graph.*

Proof. Observe that F has less than $3k$ connected components, and each of them contains nodes from R. We use that the full components of $\textsc{Opt}_k^{\text{new}}$ only intersect in R. Since $\textsc{Opt}_k^{\text{new}}$ is connected, by the pigeonhole principle it has a full component C that contains required nodes from two distinct components of F. Thus adding C to F reduces the number of components. Now the claim follows inductively. ◀

Let $\alpha = c(T)/\text{opt}^{\text{new}} \geq 1$ be the performance ratio of DeclareSteiner. Thus, in the following we want to determine an upper bound on α. We may assume

$$\text{opt}_{\epsilon,k}^{\text{old}} \geq \alpha \text{opt}^{\text{new}} \tag{2}$$

since otherwise, T_1 already gives an approximation ratio better than α.

We define $\gamma = c(\textsc{Connect}(F)) - c(F)$, the cost to connect F. Let d be the number of full components removed from $\textsc{Opt}_{\epsilon,k}^{\text{old}}$ to obtain F, i. e., $d \in \{2, 3\}$.

▶ **Lemma 6.** *For an arbitrary $\delta > 0$, the performance ratio α of* DeclareSteiner *is bounded from above by $1 + \frac{\beta-1+\epsilon(\beta-1)(d+1)}{1+(\beta-1)d+\epsilon} + \delta$.*

Proof. We have $c(C_1^t) + c(C_2^t) + c(C_3^t) \geq d \cdot c(C_1^t)$ assuming $c(C_1^t) \leq c(C_i^t)$ for $i \leq d$.

We determine the following constraints. Since $C_1^t + \textsc{Opt}^{\text{new}}$ contains a feasible solution to (G, R, c),

$$\text{opt}^{\text{new}} + c(C_1^t) \geq \text{opt}^{\text{old}}. \tag{3}$$

Furthermore,

$$\text{opt}_{\epsilon,k}^{\text{old}} - c(C_1^t) - c(C_2^t) - c(C_3^t) + \gamma \geq \alpha \text{opt}^{\text{new}} \tag{4}$$

since $c(T_2)$ is at most as large as the left hand side of (4).

We assume $\text{opt}_{\epsilon,k}^{\text{old}}$ tends to $\text{opt}_{\epsilon}^{\text{old}}$ for large enough k and then use (3) to replace opt^{old} in (2) to obtain

$$c(C_1^t) \geq \frac{\alpha - 1 - \epsilon}{1 + \epsilon} \text{opt}^{\text{new}}. \tag{5}$$

By applying (3) and (5) to (4), we obtain

$$\gamma \geq \frac{d}{1 + \epsilon} \cdot (\alpha - 1 - \epsilon) \text{opt}^{\text{new}}. \tag{6}$$

Finally, $\zeta \geq \gamma/\text{opt}_k^{\text{new}}$, by Lemma 5. Therefore, due to Lemma 2 and assumption that ϵ due to transformation to k-restricted tree tends to zero for large enough k,

$$\beta - \beta\gamma/\text{opt}_k^{\text{new}} + \gamma/\text{opt}_k^{\text{new}} \geq \alpha. \tag{7}$$

Now the claim follows if we assume $\text{opt}_k^{\text{new}}$ tends to opt^{new} for large enough k and replace γ in (7) by the right hand side of (6), where we used that $\beta \geq 1$. ◀

We note that for $d = 2$ and $\epsilon = 0$, the upper bound on the performance guarantee due to Lemma 6 matches the previously best performance guarantee [13]. For $d = 3$, the value is better than the aimed-for value from Theorem 4. Observe that a straightforward extension of DeclareSteiner would allow us to consider values of d larger than three.

Due to Lemma 6, in the following we may assume that $\deg(t) = 2$. Next, we analyze the structure of $\textsc{Opt}_{\epsilon,k}^{\text{old}}$ and $\textsc{Opt}^{\text{new}}$. Let $R_1 = (R \cap V(C_1^t)) \setminus \{t\}$ and $R_2 = (R \cap V(C_2^t)) \setminus \{t\}$.

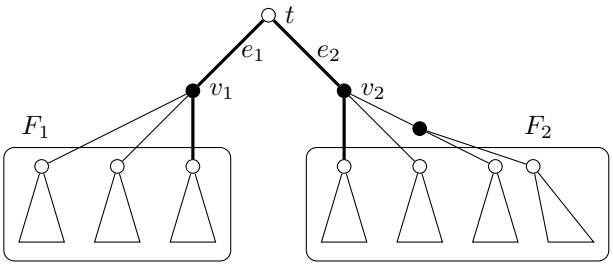

Figure 1 Structure of $\text{OPT}^{\text{old}}_{\epsilon,k}$. The paths P_1 and P_2 are drawn with thick lines.

We partition F into forests F_1 and F_2 such that F_1 contains exactly the trees T of F with $V(T) \cap R_1 \neq \emptyset$ and F_2 contains the remaining trees T', with $V(T') \cap R_2 \neq \emptyset$ (see Fig. 1).

Let $v_1 \in V(C_1^t)$ and $v_2 \in V(C_2^t)$ such that $e_1 = \{t, v_1\} \in E(C_1^t)$ and $e_2 = \{t, v_2\} \in E(C_2^t)$. Let P_1 be a minimum cost path in $\text{OPT}^{\text{old}}_{\epsilon,k}$ from t to R_1 and let P_2 be a minimum cost path in $\text{OPT}^{\text{old}}_{\epsilon,k}$ from t to R_2. Observe that P_1 contains e_1 and that P_2 contains e_2. We define $\kappa_1 := c(P_1)$, $\kappa_1' := c(e_1)$, and $\kappa_1'' = c(P_1) - c(e_1)$. Analogously, $\kappa_2 := c(P_2)$, $\kappa_2' := c(e_2)$, and $\kappa_2'' = c(P_2) - c(e_2)$. Note that we do not exclude that $v_1 \in R_1$ or $v_2 \in R_2$. In this case κ_1'' resp. κ_2'' are zero.

To simplify the presentation, we define $\kappa' := (\kappa_1' + \kappa_2')/2$ and $\kappa'' := (\kappa_1'' + \kappa_2'')/2$. Since P_1, P_2 are minimum cost paths, $c(C_1^t) \geq \kappa_1' + 2\kappa_1''$ and $c(C_2^t) \geq \kappa_2' + 2\kappa_2''$, which implies

$$c(C_1^t) + c(C_2^t) \geq 2\kappa' + 4\kappa''. \tag{8}$$

We have $\text{opt}^{\text{new}} + \kappa_1' + \kappa_1'' \geq \text{opt}^{\text{old}}$ and $\text{opt}^{\text{new}} + \kappa_2' + \kappa_2'' \geq \text{opt}^{\text{old}}$. Therefore,

$$\text{opt}^{\text{new}} + \kappa' + \kappa'' \geq \text{opt}^{\text{old}}. \tag{9}$$

▶ **Lemma 7.** *Suppose there are at least two edge disjoint paths in* $\text{OPT}^{\text{new}}_k$ *between* $V(F_1)$ *and* $V(F_2)$*. Then, for an arbitrary* $\delta > 0$*, the performance guarantee of* DeclareSteiner *is bounded from above by* $\frac{(11\beta-8)(1+\epsilon)}{8\beta-5+3\epsilon} + \delta$*.*

Proof. Let P' and P'' be two edge-disjoint paths within $\text{OPT}^{\text{new}}_k$ between $V(F_1)$ and $V(F_2)$ such that none of their internal nodes are in $V(\text{OPT}^{\text{old}}_{\epsilon,k})$. Without loss of generality, we assume that $c(P') \leq c(P'')$. We will also assume that $\text{opt}^{\text{new}}_k$ tends to opt^{new} for large enough k. Then, additionally to the previous constraints, we obtain the following.

$$\text{opt}^{\text{old}}_{\epsilon,k} - 2\kappa' + c(P') = \text{opt}^{\text{old}}_{\epsilon,k} - \kappa_1' - \kappa_2' + c(P') \geq \alpha \text{opt}^{\text{new}} \tag{10}$$

$$\zeta \cdot \text{opt}^{\text{new}}_k \geq c(P') + c(P'') \geq 2c(P') \tag{11}$$

From (9) and (10) and assuming $\text{opt}^{\text{old}}_{\epsilon,k}$ tends to $\text{opt}^{\text{old}}_{\epsilon}$ for large enough k, we obtain

$$c(P') \geq (\alpha - 1 - \epsilon)\text{opt}^{\text{new}} + (1-\epsilon)\kappa' - (1+\epsilon)\kappa'', \tag{12}$$

and thus, due to (11) and (2),(9),

$$\zeta \text{opt}^{\text{new}}_k \geq 2((\alpha-1-\epsilon)\text{opt}^{\text{new}} + (1-\epsilon)\kappa' - (1+\epsilon)\kappa'') \geq \frac{4}{(1+\epsilon)}(\alpha-1-\epsilon)\text{opt}^{\text{new}} - 4\kappa''. \tag{13}$$

Furthermore, by using (8) and (9) in (4), we obtain

$$\gamma \geq (\alpha - 1 - \epsilon)\text{opt}^{\text{new}} + (1-\epsilon)\kappa' + (3-\epsilon)\kappa''.$$

and thus, due to (2) and (9) and the fact that $\zeta \text{opt}_k^{\text{new}} \geq \gamma$,

$$\zeta \text{opt}_k^{\text{new}} \geq \frac{2}{(1+\epsilon)}(\alpha - 1 - \epsilon)\text{opt}^{\text{new}} + 2\kappa''. \tag{14}$$

A linear combination of (13) and (14) with coefficients one and two gives $\zeta \geq \frac{8(\alpha-1-\epsilon)}{3(1+\epsilon)}$, by assuming that $\text{opt}_k^{\text{new}}$ tends to opt^{new} for large enough k. Using (1) we obtain

$$\alpha \leq \frac{(11\beta - 8)(1+\epsilon)}{8\beta - 5 + 3\epsilon} + \delta. \qquad \blacktriangleleft$$

Since the value obtained by Lemma 7 is better than the aimed-for ratio, from now on we can restrict our focus to instances where in $\text{OPT}_k^{\text{new}}$, there are no two edge-disjoint paths between F_1 and F_2. In particular, this means that there is exactly one full component L in $\text{OPT}_k^{\text{new}}$ that connects F_1 and F_2. Since we assumed that there are no Steiner nodes of degree two in $\text{OPT}_k^{\text{new}}$, there is exactly one edge e_L in L such that removing e_L leaves two connected components of $\text{OPT}_k^{\text{new}}$, one containing R_1 and the other one containing R_2. Let P_L be a minimum cost path between $V(F_1)$ and $V(F_2)$ in L (and thus P_L clearly contains e_L). Let P_L^1 be the subpath of P_L between F_1 and e_L and let P_L^2 be the subpath of P_L between F_2 and e_L. We define $\lambda := c(P_L)$, $\lambda' := c(e_L)$, $\lambda_1'' := c(P_L^1)$, and $\lambda_2'' := c(P_L^2)$. Similar to above, we define $\lambda'' := (\lambda_1'' + \lambda_2'')/2$. Note that $\lambda - \lambda' = 2\lambda''$. It follows easily that $c(L) \geq \lambda' + 4\lambda''$. Let L' be a forest with a minimum number of full components from $\text{OPT}_k^{\text{new}}$ such that $\text{OPT}_{\epsilon,k}^{\text{old}} - C_1^t - C_2^t + L'$ is connected. From Lemma 5, we obtain that L' contains at most $3k$ full components and thus we considered guessing L' when computing T_3 in DeclareSteiner. We define $\xi := c(L') - \lambda' - 4\lambda''$. Since L' contains L, ξ is non-negative.

To find an upper bound on the value of α, we maximize α subject to the constraints (2), (9) and the following constraints.

By removing e_L from $\text{OPT}_k^{\text{new}}$ and adding the paths P_1 and P_2, we obtain a feasible solution to (G, R, c); conversely, by removing e_1 and e_2 from $\text{OPT}_{\epsilon,k}^{\text{old}}$ and adding P_L, we obtain a feasible solution to $(G, R \setminus t, c)$ that is considered in T_2. Therefore

$$\text{opt}_k^{\text{new}} + 2\kappa' + 2\kappa'' - \lambda' \geq \text{opt}^{\text{old}}, \tag{15}$$

$$\text{opt}_{\epsilon,k}^{\text{old}} - 2\kappa' + \lambda' + 2\lambda'' \geq \alpha \text{opt}^{\text{new}}. \tag{16}$$

In T_2 we also consider to remove C_1^t, C_2^t completely and to add L'. Therefore

$$\text{opt}_{\epsilon,k}^{\text{old}} - 2\kappa' - 4\kappa'' + \lambda' + 4\lambda'' + \xi \geq \alpha \text{opt}^{\text{new}}. \tag{17}$$

Due to Lemma 2 and assumption that ϵ due to transformation to k-restricted tree tends to zero for large enough k, we may assume

$$\beta - \beta\zeta + \zeta \geq \alpha. \tag{18}$$

In T_3, one of the considered guesses is L' and therefore

$$\zeta \cdot \text{opt}_k^{\text{new}} \geq \lambda' + 4\lambda'' + \xi. \tag{19}$$

We assume that $\text{opt}_k^{\text{new}}$ and $\text{opt}_{\epsilon,k}^{\text{old}}$ tends to opt^{new} and $\text{opt}_{\epsilon}^{\text{old}}$ respectively for large enough k and then scale the values such that $\text{opt}^{\text{new}} = 1$. Then we perform the following

replacements. We replace opt^{old} in (9) and in (15) by using (2); we use (9) to replace opt^{old} in (16); we use (9) to replace opt^{old} in (17). We keep (18) and (19). This way we obtain a linear program that maximizes α subject to the following constrains.

$$\begin{aligned} -\kappa' - \kappa'' + \alpha/(1+\epsilon) &\leq 1 \\ -2\kappa' - 2\kappa'' + \lambda' + \alpha/(1+\epsilon) &\leq 1 \\ (1-\epsilon)\kappa' - (1+\epsilon)\kappa'' - \lambda' - 2\lambda'' + \alpha &\leq 1+\epsilon \\ (1-\epsilon)\kappa' + (3-\epsilon)\kappa'' - \lambda' - 4\lambda'' - \xi + \alpha &\leq 1+\epsilon \\ (\beta-1)\zeta + \alpha &\leq \beta \\ \lambda' + 4\lambda'' + \xi - \zeta &\leq 0 \end{aligned}$$

Now we obtain the dual linear program

$$\begin{aligned} \text{minimize} \quad & y_1 + y_2 + (1+\epsilon)y_3 + (1+\epsilon)y_4 + \beta y_5 & \\ \text{s.t.:} \quad & -y_1 - 2y_2 + (1-\epsilon)y_3 + (1-\epsilon)y_4 & \geq 0 \\ & -y_1 - 2y_2 - (1+\epsilon)y_3 + (3-\epsilon)y_4 & \geq 0 \\ & y_2 - y_3 - y_4 + y_6 & \geq 0 \\ & -2y_3 - 4y_4 + 4y_6 & \geq 0 \\ & -y_4 + y_6 & \geq 0 \\ & (\beta-1)y_5 - y_6 & \geq 0 \\ & y_1/(1+\epsilon) + y_2/(1+\epsilon) + y_3 + y_4 + y_5 & \geq 1 \end{aligned}$$

To finish the proof, we consider the following feasible solution. We set

$$y_1 = \frac{2(\beta-1)(1+\epsilon)(1-2\epsilon)}{7\beta - 4 + 5\epsilon - 2\epsilon\beta}; \qquad y_2 = y_1/2;$$

$$y_3 = y_1/(1-2\epsilon); \qquad y_4 = y_1/(1-2\epsilon);$$

$$y_5 = \frac{3(1+\epsilon)(1-2\epsilon)}{(1-2\epsilon)(7\beta - 4 + 5\epsilon - 2\epsilon\beta)}; \qquad y_6 = \frac{3y_1}{2(1-2\epsilon)}.$$

With these values, the objective function value matches the claimed value in Theorem 4. By weak duality, we obtained an upper bound on the value of α in the primal linear program, which finishes the proof.

5 Increased Edge Cost

We now consider the reoptimization variant where the edge cost of one edge is increased, $\text{STP}_{\epsilon}^{E+}$. If e is the edge of G with increased cost, we define $c^{\text{new}}\colon E(G) \to \mathbb{R}_{\geq 0}$ as $c^{\text{new}}(e') = c(e')$ for all edges $e' \in E(G)\setminus\{e\}$ and $c^{\text{new}}(e)$ is the increased cost. Then the formal definition of the reoptimization variant is as follows.

Given: A parameter $\epsilon > 0$, a Steiner tree instance (G, R, c), a solution $\text{OPT}_{\epsilon}^{\text{old}}$ to (G, R, c) such that $\text{opt}_{\epsilon}^{\text{old}} \leq (1+\epsilon)\text{opt}^{\text{old}}$, and a cost $c^{\text{new}}(e) \geq c(e)$ for an edge $e \in E(G)$.
Solution: A Steiner tree solution to (G, R, c^{new}).

Observe that the cost function obtained after applying the local modification in general is not a metric, and $\text{OPT}_{\epsilon,k}^{\text{new}}$ is assumed to live in the metric closure according to the new cost function.

▶ **Theorem 8.** *Let* $(G, R, c, \text{OPT}_{\epsilon}^{\text{old}}, e, c^{\text{new}}(e))$ *be an instance of* $\text{STP}_{\epsilon}^{E+}$. *Then, for an arbitrary* $\delta > 0$, EdgeIncrease *is a* $\left(\frac{7\beta-4+\epsilon(4\beta-4)}{4\beta-1} + \delta\right)$*-approximation algorithm for* $\text{STP}_{\epsilon}^{E+}$.

Input : An instance $(G, R, c, \textsc{Opt}_\epsilon^{\text{old}}, e, c^{\text{new}}(e))$ of STP_ϵ^{E+}
Output : A Steiner tree T
Transform $\textsc{Opt}_\epsilon^{\text{old}}$ to a k-restricted solution $\textsc{Opt}_{\epsilon,k}^{\text{old}}$ such that $\text{opt}_{\epsilon,k}^{\text{old}} \leq (1+\epsilon_k)\text{opt}_\epsilon^{\text{old}}$ where ϵ_k tends to 0 for large enough k;
Set $T_1 := \textsc{Opt}_\epsilon^{\text{old}}$;
Set $T_2 := \textsc{Guess}(k+1)$, with respect to $c^{\text{new}}(e)$;
Set $T = T_i$ with $i = \min \arg_{j\in\{1,2\}}\{c(T_j)\}$.

Algorithm 4: EdgeIncrease

Proof. Let us introduce the following notation. To emphasize which of the two instances we consider, we write $c^{\text{old}}(e)$ instead of $c(e)$, where e is the edge with increased cost. We assume that $e \in E(\textsc{Opt}_{\epsilon,k}^{\text{old}})$, as otherwise T_1 would be good enough already. Therefore the graph $\textsc{Opt}_{\epsilon,k}^{\text{old}} - e$ has exactly two connected components F_1 and F_2. Similar to the previous proof, we define $R_1 := R \cap V(F_1)$ and $R_2 := R \cap V(F_2)$.

In $\textsc{Opt}_{\epsilon,k}^{\text{old}}$, let K be the full component that contains e. Let P be a minimum cost path from R_1 to R_2 within K. Then we set $\kappa := c(P) - c^{\text{old}}(e)$.

In $\textsc{Opt}_k^{\text{new}}$, there is a full component L of cost λ such that $V(L)$ contains nodes from both R_1 and R_2. If L has two edge-disjoint paths between R_1 and R_2, we define $\lambda' = 0$. Otherwise there is an edge $e_L \in E(L)$ such that e_L is a cut edge in $F_1 \cup F_2 \cup L$, and $\lambda' := c(e_L)$. We obtain the following inequalities, where as before $\alpha = c(T)/\text{opt}^{\text{new}}$.

Removing e and adding a shortest path between R_1 and R_2 within L gives a feasible solution to (G, R, c^{new}). Therefore T_2 is good enough unless

$$\text{opt}_{\epsilon,k}^{\text{old}} - c^{\text{old}}(e) + \lambda/2 + \lambda'/2 \geq \alpha\text{opt}^{\text{new}}. \tag{20}$$

One feasible solution to the original instance is to remove e_L and to add P. Therefore we obtain

$$\text{opt}_k^{\text{new}} - \lambda' + c^{\text{old}}(e) + \kappa \geq \text{opt}^{\text{old}}. \tag{21}$$

We obtain an additional constraint by observing that in addition to using e_L, within $\textsc{Opt}_k^{\text{new}}$ the required vertices of K have to be connected. Let K_1 be the tree of $K - e$ that contains $R_1 \cap V(K)$. We see K_1 as a rooted tree with the root r_1 contained in $e = \{r_1, r_2\}$. Let us fix any two vertices $u \neq u' \in V(K_1) \setminus \{r\}$, with parents v, v'. Then the minimum distance between the two subtrees rooted at u, u' is at least $\max\{c(u,v), c(u',v')\}$. The same argumentation holds for K_2, which we define analogous to K_1 (it contains $R_2 \cap V(K)$, and has the root r_2). By traversing a path from $V(K_1) \cap R_1$ to r_1 within K_1 and from $V(K_2) \cap R_2$ to r_2, and adding the distances, we conclude that there is a collection of at most k full components in $\textsc{Opt}_k^{\text{new}}$ that without counting e_L have a total cost of at least κ. Therefore, using T_2,

$$\zeta\text{opt}_k^{\text{new}} \geq \lambda' + \kappa.$$

From these constraints, we obtain the claimed result using arguments similar to those of the previous section. ◀

Acknowlegment. We would like to thank Anna Zych for some helpful comments and the anonymous reviewers for helping to improve the presentation.

References

1 Claudia Archetti, Luca Bertazzi, and Maria Grazia Speranza. Reoptimizing the traveling salesman problem. *Networks*, 42(3):154–159, 2003.

2 Claudia Archetti, Luca Bertazzi, and Maria Grazia Speranza. Reoptimizing the 0-1 knapsack problem. *Discrete Applied Mathematics*, 158(17):1879–1887, 2010.

3 Claudia Archetti, Gianfranco Guastaroba, and Maria Grazia Speranza. Reoptimizing the rural postman problem. *Computers & OR*, 40(5):1306–1313, 2013.

4 Giorgio Ausiello, Vincenzo Bonifaci, and Bruno Escoffier. *Complexity and approximation in reoptimization*. Imperial College Press/World Scientific, 2011.

5 Giorgio Ausiello, Bruno Escoffier, Jérôme Monnot, and Vangelis Th. Paschos. Reoptimization of minimum and maximum traveling salesman's tours. *Journal of Discrete Algorithms*, 7(4):453–463, 2009.

6 Michael A. Bender, Martin Farach-Colton, Sándor P. Fekete, Jeremy T. Fineman, and Seth Gilbert. Reallocation problems in scheduling. In *Proc. of the 25th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA 2013)*, pages 271–279. ACM, 2013.

7 Tobias Berg and Harald Hempel. Reoptimization of traveling salesperson problems: Changing single edge-weights. In *Proc. of the Third International Conference on Language and Automata Theory and Applications (LATA 2009)*, volume 5457 of *LNCS*, pages 141–151. Springer, 2009.

8 Marshall W. Bern and Paul E. Plassmann. The Steiner problem with edge lengths 1 and 2. *Information Processing Letters*, 32(4):171–176, 1989.

9 Davide Bilò, Hans-Joachim Böckenhauer, Juraj Hromkovič, Richard Královič, Tobias Mömke, Peter Widmayer, and Anna Zych. Reoptimization of Steiner trees. In *Proc. of the 11th Scandinavian Workshop on Algorithm Theory (SWAT 2008)*, volume 5124 of *LNCS*, pages 258–269, 2008.

10 Davide Bilò, Hans-Joachim Böckenhauer, Dennis Komm, Richard Królovic, Tobias Mömke, Sebastian Seibert, and Anna Zych. Reoptimization of the shortest common superstring problem. *Algorithmica*, 61(2):227–251, 2011.

11 Davide Bilò, Peter Widmayer, and Anna Zych. Reoptimization of weighted graph and covering problems. In *Proc. of the 6th International Workshop on Approximation and Online Algorithms (WAOA 2008)*, volume 5426 of *Lecture Notes in Computer Science*, pages 201–213. Springer, 2008.

12 Davide Bilò and Anna Zych. New reoptimization techniques employed to Steiner tree problem. In *Proceedings of the 6th Latin-American Algorithms, Graphs and Optimization Symposium (LAGOS 11)*, 2011.

13 Davide Bilò and Anna Zych. New advances in reoptimizing the minimum Steiner tree problem. In *Mathematical Foundations of Computer Science 2012*, pages 184–197. Springer, 2012.

14 Hans-Joachim Böckenhauer, Luca Forlizzi, Juraj Hromkovič, Joachim Kneis, Joachim Kupke, Guido Proietti, and Peter Widmayer. On the approximability of TSP on local modifications of optimally solved instances. *Algorithmic Operations Research*, 2(2):83–93, 2007.

15 Hans-Joachim Böckenhauer, Karin Freiermuth, Juraj Hromkovič, Tobias Mömke, Andreas Sprock, and Björn Steffen. The Steiner tree reoptimization problem in graphs with sharpened triangle inequality. *Journal of Discrete Algorithms*, 11:73–86, 2012.

16 Hans-Joachim Böckenhauer, Juraj Hromkovič, Richard Královič, Tobias Mömke, and Peter Rossmanith. Reoptimization of Steiner trees: Changing the terminal set. *Theoretical Computer Science*, 410(36):3428–3435, 2009.

17 Hans-Joachim Böckenhauer, Juraj Hromkovič, Tobias Mömke, and Peter 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 *LNCS*, pages 50–65, 2008.

18 Hans-Joachim Böckenhauer and Dennis Komm. Reoptimization of the metric deadline TSP. *Journal of Discrete Algorithms*, 8(1):87–100, 2010.

19 Al Borchers and Ding-Zhu Du. The k-Steiner ratio in graphs. *SIAM Journal on Computing*, 26(3):857–869, 1997.

20 Nicolas Boria and Federico Della Croce. Reoptimization in machine scheduling. *Theoretical Computer Science*, 540:13–26, 2014.

21 Nicolas Boria, Jérôme Monnot, and Vangelis Th Paschos. Reoptimization of maximum weight induced hereditary subgraph problems. *Theoretical Computer Science*, 514:61–74, 2013.

22 Nicolas Boria and Vangelis T Paschos. A survey on combinatorial optimization in dynamic environments. *RAIRO-Operations Research*, 45(03):241–294, 2011.

23 Nicolas Boria and Vangelis Th Paschos. Fast reoptimization for the minimum spanning tree problem. *Journal of Discrete Algorithms*, 8(3):296–310, 2010.

24 Jaroslaw Byrka, Fabrizio Grandoni, Thomas Rothvoß, and Laura Sanità. Steiner tree approximation via iterative randomized rounding. *Journal of the ACM*, 60(1):6, 2013.

25 S. E. Dreyfus and R. A. Wagner. The Steiner problem in graphs. *Networks*, 1:195–207, 1971/72.

26 Bruno Escoffier, Martin Milanič, and Vangelis Th. Paschos. Simple and fast reoptimizations for the Steiner tree problem. *Algorithmic Operations Research*, 4(2):86–94, 2009.

27 Stefan Hougardy, Jannik Silvanus, and Jens Vygen. Dijkstra meets Steiner: a fast exact goal-oriented Steiner tree algorithm. *arXiv preprint arXiv:1406.0492*, 2014.

28 Jérôme Monnot. A note on the traveling salesman reoptimization problem under vertex insertion. *Inf. Process. Lett.*, 115(3):435–438, 2015.

29 Markus W Schäffter. Scheduling with forbidden sets. *Discrete Applied Mathematics*, 72(1):155–166, 1997.

30 Hiromitsu Takahashi and Akira Matsuyama. An approximate solution for the Steiner problem in graphs. *Mathematica Japonica*, 24(6):573–577, 1979/80.

31 Anna Zych. *Reoptimization of NP-hard problems.* PhD thesis, Diss., Eidgenössische Technische Hochschule ETH Zürich, Nr. 20257, 2012, 2012.