Reoptimization of Steiner Trees*

Davide Bilò, Hans-Joachim Böckenhauer, Juraj Hromkovič, Richard Královič, Tobias Mömke, Peter Widmayer, and Anna Zych

Department of Computer Science, ETH Zurich, Switzerland
{dbilo,hjb,juraj.hromkovic,richard.kralovic,
tobias.moemke,peter.widmayer,anna.zych}@inf.ethz.ch

Abstract. In this paper we study the problem of finding a minimum Steiner Tree given a minimum Steiner Tree for similar problem instance. We consider scenarios of altering an instance by locally changing the terminal set or the weight of an edge. For all modification scenarios we provide approximation algorithms that improve best currently known corresponding approximation ratios.

1 Introduction

Traditional optimization theory focuses on searching for solutions when nothing or very little is known a'priori about the problem instance. In reality, prior knowledge is often at our disposal, because a problem instance can arise from a small modification of a previous problem instance. As an example, imagine that we are given a set of nodes (points in some metric space), and the network graph of shortest length interconnecting them, where the length is the sum of the distances between adjacent nodes. Now imagine one node is excluded from the network. It is intuitively obvious that we should profit from the old network when we try to find a new one. The general idea we pursue is: given a problem instance with an optimal solution, and a variation of the problem instance obtained through a local modification, what can we learn about the new solution? We believe that looking at NP-hard problems from this perspective will allow to explore them deeper and learn more about the nature of their complexity.

The problem we deal with in this paper is reoptimization of the Minimum Steiner Tree (ST) problem. Given an edge-weighted graph and a set of terminal vertices, the ST problem asks for a minimum tree spanning the terminal set. The ST problem is a very prominent optimization problem with many practical applications, especially in network design, see for example [6,7]. The best up to date approximation ratio for non-reoptimization case is ≈ 1.55 [8]. The problem of reoptimizing ST where the local modification consists of adding/deleting one vertex to/from the input graph was considered in [5]. For the local modification of adding/removing a vertex to/from the terminal set a 1.5-approximation algorithms have been provided in [2]. We improve over these results, providing

* This work was partially supported by SBF grant C 06.0108 as part of the COST 293 (GRAAL) project funded by the European Union.

J. Gudmundsson (Ed.): SWAT 2008, LNCS 5124, pp. 258–269, 2008.

1.344-approximation algorithm for adding a terminal vertex, and 1.408 for removing a terminal vertex, while for the scenarios of increasing and decreasing the weight of an edge we obtain 4/3 and 1.302 approximation ratio respectively.

The paper is organized as follows. In Section 2 we provide basic notation, definitions and facts used throughout the paper. In Section 3 we describe the basic techniques used in Section 4, Section 5, Section 6 and Section 7 where we provide our results for the four local modification scenarios we consider.

2 Preliminaries

Let us begin with a formal definition of the Minimum Steiner Tree Problem (ST). We call a complete graph $G = (V, E, c)$ with edge weight function $c : E \to \mathbb{R}^+$ *metric*, if the edge weights satisfy the *triangle inequality*, i.e., $c((u, v)) \leq c((u, w)) + c((w, v))$ for all $u, v, w \in V$.

The ***Minimum Steiner Tree Problem (ST)*** is defined as follows:
Instance: A metric graph $G = (V, E, c)$ and a terminal set $S \subseteq V$
Solution: A Steiner tree, i.e., a subtree T of G containing S
Objective: Minimize the sum of the weights of the edges in the subtree T.

The ST problem is APX-complete even when the edge weights are restricted to the set $\{1, 2\}$ [1]. The best up to date approximation ratio is $\sigma = 1 + \frac{\log 3}{2} \approx 1.55$ for general edge costs and 1.28 for edge costs from $\{1, 2\}$ [8]. We denote the best up to date approximation algorithm for ST as $\mathrm{ApprST}(G, S)$ and view it as a function from the instance space to the solution space. The ST problem can be solved optimally in time exponential in the size of S with an algorithm proposed by Dreyfus and Wagner [4]. We denote this algorithm as $\mathrm{OptST}(G, S)$.

Below we define the reoptimization problems we investigate in this paper. The objective for all of them is to minimize the cost of Steiner tree T.

1. ***Minimum Steiner Tree Terminal Addition (ST-S+)***
   Instance: Metric graph $G = (V, E, c)$, terminal set $S_O \subseteq V$, an optimal Steiner tree T_O for (G, S_O), and terminal set $S_N = S_O \cup \{t\}$ for some $t \in V$
   Solution: Steiner tree T for (G, S_N).
2. ***Minimum Steiner Tree Terminal Removal (ST-S−)***
   Instance: Metric graph $G = (V, E, c)$, terminal set $S_O \subseteq V$, an optimal Steiner tree T_O for (G, S_O), and terminal set $S_N = S_O \setminus \{t\}$ for some $t \in V$
   Solution: Steiner tree T for (G, S_N).
3. ***Minimum Steiner Tree Edge Cost Increase (ST-E+)***
   Instance: Metric graph $G_O = (V, E, c_o)$, terminal set $S \subseteq V$, an optimal Steiner tree T_O for (G, S_O), and metric graph $G_N = (V, E, c_n)$, where $c_n = c_o$ on all but one edge $e \in E$ for which $c_n(e) \geq c_o(e)$[1]
   Solution: Steiner tree T for (G_N, S).
4. ***Minimum Steiner Tree Edge Cost Decrease (ST-E−)***
   Instance: Metric graph $G_O = (V, E, c_o)$, terminal set $S \subseteq V$, an optimal Steiner tree T_O for (G, S_O), and metric graph $G_N = (V, E, c_n)$, where $c_n = c_o$

[1] Whenever c_o and c_n coincide on some edge f, we drop the subscripts and write $c(f)$.

on all but one edge $e \in E$ for which $c_n(e) \leq c_o(e)$[1]
Solution: Steiner tree T for (G_N, S).

Lemma 1 ([2,3]). *The aforementioned problems are strongly* NP*-hard.* □

Let us adopt the following notation. With T_N we denote an optimal Steiner tree for the modified instance (G, S_N) or (G_N, S). Given a simple graph G, we denote its set of vertices with $V(G)$ and its set of edges with $E(G)$. An edge $(u, v) \in E(G)$ can be seen as a subgraph H of G with $V(H) = \{u, v\}$ and $E(H) = \{(u, v)\}$. A vertex $v \in V(G)$ can be seen as a subgraph H of G with $V(H) = \{v\}$ and $E(H) = \emptyset$. The degree of a vertex $v \in V(G)$ is $\deg_G(v)$. If H is a subgraph of G, we write $H \subseteq G$. The notation $c(G)$ denotes the *cost* of G, i.e. the sum of all its edge costs. With $G - v$ we denote graph G after removing node $v \in V(G)$ and incident edges. For two subgraphs of G: $H_1, H_2 \subseteq G$ (H_i can be a single edge) we introduce the following notation. With $H_1 - H_2$ we denote a graph such that $V(H_1 - H_2) = V(H_1)$ and $E(H_1 - H_2) = E(H_1) \setminus E(H_2)$. With $H_1 + H_2$ we denote a graph such that $V(H_1 + H_2) = V(H_1) \cup V(H_2)$ and $E(H_1 + H_2) = E(H_1) \cup E(H_2)$. We denote with $\mathit{CheapestEdge}(H_1, H_2)$ the cheapest edge in G connecting H_1 and H_2. Expression $\min\{H_1, \ldots, H_i\}$ returns a cheaper graph among $H_1, \ldots, H_i$ w.r.t. their cost. A forest F is a graph composed of node-disjoint trees $T_1, \ldots, T_i$. Such tree decomposition of F is denoted as $F = T_1 + \cdots + T_i$. For a connected subgraph $H \subseteq G$ of G, with $\mathit{Contract}(G, H, h)$ we denote a weighted graph $G' = (V', E', c')$ obtained from G by contracting H into node h, where if after contraction multiedges occur between h and any node $v \in V(G) \setminus V(H)$, then we set $c'((v, h)) = c(\mathit{CheapestEdge}(v, H))$. We describe a path in a graph as a sequence of its vertices. The length of a path is its number of edges. The cost of a path is the sum of costs of its edges. In a shortest path, the length of the path is minimized whereas in a cheapest path its cost is minimized.

For a complete graph $G = (V, E, c)$ with an arbitrary edge weight function $c : E \to \mathbb{R}^+$, we define the *metric closure* of G as the graph $\tilde{G} = (V, E, \tilde{c})$ where $\tilde{c}((u, v))$ is defined as the cost of the cheapest path in G from u to v. It is well known (see for example [7]) that a tree T is a minimum Steiner tree for (G, S) if and only if it is also a minimum Steiner tree for $(\tilde{G}, S)$ where $\tilde{G}$ is the metric closure of G. Because of this fact, for ST-S+ and ST-S−, we can assume w.l.o.g. that the given graph G is metric. For problems ST-E+ and ST-E− we assume as well that the local modification preserves metricity, however in this case this assumption is a restriction as changing the weight of one edge in a metric graph G can result in altering the cost of many edges in its metric closure. Finally, we can assume without loss of generality that the given minimum Steiner tree T_O has no non-terminal vertex of degree two; due to the metricity these vertices can be removed using the direct edge between the two adjacent vertices instead.

3 Techniques

In this section we present standard procedures used further in reoptimization algorithms. Algorithm 1 is based on the assumption, that we know a large part

T_s of an optimal solution T_N for (G, S_N) or (G_N, S). Provided that knowledge, we contract T_s to a single node, make it a terminal, and use σ-approximation algorithm ApprST to obtain the solution for the remaining part of the graph.

Algorithm 1. Shrink(G, S, T_s)

Input: A metric graph G, a terminal set $S \subseteq V(G)$, and a tree $T_s \subseteq G$
1: $G' := \mathit{Contract}(G, T_s, t_s)$
2: $T' := \mathit{ApprST}(G', \big(S \cap V(G')\big) \cup \{t_s\})$
3: Obtain T: Expand T' by substituting t_s with T_s
Output: T

Lemma 2. *Let T_{opt} be an optimal solution for (G, S). Given that $T_s \subseteq T_{opt}$, and $c(T_s) \geq \alpha c(T_{opt})$, Algorithm 1 applied to (G, S, T_s) returns $\big(\sigma - \alpha(\sigma - 1)\big)$-approximation of T_{opt}.*

Proof. Let $G' := \mathit{Contract}(G, T_s, t_s)$, $S' = \big(S \cap V(G')\big) \cup \{t_s\}$, and T' be as defined in Algorithm 1. Note, that given $T_s \subseteq T_{opt}$, solution $\mathit{Contract}(T_{opt}, T_s, t_s)$ with cost $c(T_{opt}) - c(T_s)$ is optimal for (G', S'). Thus $c(T') \leq \sigma(c(T_{opt}) - c(T_s))$. Since $\sigma \geq 1$, then the cost of solution tree T returned by $\mathit{Shrink}(G, S, T_s)$ is:

$$c(T) \leq \sigma(c(T_{opt}) - c(T_s)) + c(T_s) \leq \big(\sigma - \alpha(\sigma - 1)\big)c(T_{opt}).$$

□

The other technique, shown in Algorithm 2 is used for connecting optimally a given forest F. Provided that the number of trees in $F = T_1 + \cdots + T_i$ is logarithmic in the input size, we can connect F optimally in polynomial time.

Algorithm 2. Connect(G, F)

Input: A metric graph G, a forest $F = T_1 + \cdots + T_i$, where $T_i \subset G$ and $T_1 \ldots T_i$ are pairwise node-disjoint
1: $G_1 := \mathit{Contract}(G, T_1, t_1)$
2: For $j = 2, \ldots, i$ do $G_j := \mathit{Contract}(G_{j-1}, T_j, t_j)$
3: $S := \{t_1, \ldots, t_i\}$
4: $T' := \mathrm{OptST}(G_i, S)$
5: Obtain tree T by substituting each t_i with T_i and keeping the edges of T' for connecting T_i with the rest of graph G
Output: T

Lemma 3. *If $i = O(\log |V(G)|)$, then Algorithm 2 runs in polynomial time.*

Proof. The running time of Dreyfus-Wagner algorithm OptST() is exponential in size of terminal set [4]. Since we apply OptST to S of size $O(\log |V(G)|)$, the overall running time of Algorithm 2 is polynomial. □

Remark 1. Let (G, S) be an instance of ST and T_{opt} be an optimal solution for that instance. If $F = T_1 + \cdots + T_i \subseteq G$ and $T_{j=1\ldots i} \cap S \neq \emptyset$, then solution T returned by Algorithm 2 satisfies $c(T) \leq c(F) + c(T_{opt})$.

4 Removing One Terminal

In this section we present a 1.408-approximation algorithm for ST-S−, thus improving the result in [2]. The algorithm we propose computes several feasible solutions and chooses the one of minimal cost. Let $f_1, \ldots, f_k$ be the edges adjacent in T_O to the terminal t that is supposed to be removed. We distinguish 4 cases depending on the value of k. We remark here, that Algorithm 3 improves the worst case of $k = 2$. The other cases are dealt with in the same manner as in algorithm provided in [2]. For the sake of completeness however, we provide here the full analysis.

Algorithm 3. MinSTP-S-

Input: A metric graph G, a terminal set $S_O \subseteq V(G)$, an optimal Steiner tree $T_O \subseteq G$ for (G, S_O), and $S_N = S_O \setminus \{t\}$ for some terminal $t \in S_O$

Let $f_1, \ldots, f_k$ be the list of edges adjacent to t in T_O
if $k = 3$ **then**
  Compute shortest paths p_1, p_2, p_3 connecting t with S_N, such that $f_i \in E(p_i)$ for $i = 1, \ldots, 3$
  $F := T_O - (p_1 + p_2 + p_3) - t$
  return $Connect(G, F)$
end if
if $k = 2$ **then**
  $T_1 := \min_{f \in E(G)}\{Shrink(G, S_N, f)\}$
  Compute shortest paths p_1, p_2, p_3, p_4 connecting t with S_N, such that $f_1 \in E(p_1), E(p_2)$ and $f_2 \in E(p_2), E(p_3)$
  $F := T_O - (p_1 + p_2 + p_3 + p_4) - t$
  $T_2 := Connect(G, F)$
  return $\min\{T_1, T_2\}$
end if
if $k = 1$ **then**
  Let $f_1 = (t, v)$. **if** $v \in S_N$ **then return** $T := T_O - t$
  else return MinSTP-S-$(G, S_N \cup \{v\}, T_O - t, S_N)$
end if

Output: T_O

Theorem 1. *Algorithm 3 is a 1.408-approximation algorithm for ST-S−.*

Proof. Let p be a cheapest path connecting t with S_N. Let $\alpha \geq 0$ be a parameter that shall be fixed later. We want to compute the smallest α, for which Algorithm 3 returns $(1+\alpha)$-approximation of T_N. If $c(p) \leq \alpha c(T_N)$, then $T_O \leq c(T_N) + c(p) \leq (1+\alpha)c(T_N)$, therefore we assume from now on, that $c(p) > \alpha c(T_N)$. We distinguish four cases depending on the degree of t in T_O.

Case 1. Assume $\deg_{T_O}(t) > 3$ or equivalently $k > 3$, and note that $c(T_N) \leq c(T_O)$. Since $\deg_{T_O}(t) \geq 4$, there exist four edge-disjoint paths from t to terminals in T_O and thus $c(T_O) \geq 4 \cdot c(p)$. On the other hand, since $T_N + p$ is a solution for (G, S_O), we know $c(T_N) + c(p) \geq c(T_O)$. Therefore, $c(T_N) \geq 3 \cdot c(p)$ and thus $c(T_O) \leq c(T_N) + c(p) \leq \frac{4}{3} c(T_N)$.

Case 2. Assume $\deg_{T_O}(t) = 3$ or equivalently $k = 3$. Let p_1, p_2, p_3 be as in the Algorithm 3 in case $k = 3$ (see Figure 1). Since p_1, p_2, p_3 are paths minimal in the number of nodes ending in a terminal, and each non-terminal is branching ($deg_{T_O}(v) > 2$ for $v \notin S$), thus $|p_i| \leq \log |V(G)|$. Therefore the number of trees in F is bounded by $3 \log |V(G)|$, and thus by Lemma 3 $Connect(G, F)$ runs in polynomial time. For connecting the forest F optimally we pay at most $c(T_N)$, because each tree of F contains a terminal from S_N. Therefore for tree T_2 computed by the algorithm we have that $c(T_2) \leq c(T_O) - c(p_1) - c(p_2) - c(p_3) + c(T_N) \leq c(T_N) + c(p) - 3c(p) + c(T_N) \leq 2c(T_N)(1 - \alpha)$. This value is bounded from above by $(1 + \alpha)c(T_N)$ for any $\alpha \geq \frac{1}{3}$.

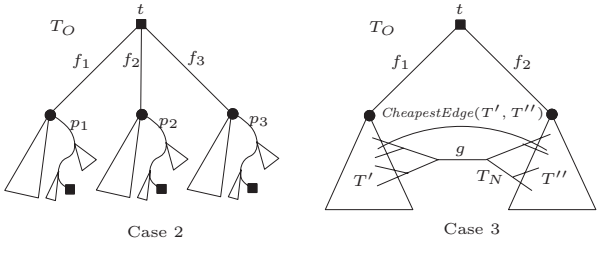

Fig. 1. Illustration for Case 2 and first case of Case 3 (A)

Case 3. Assume $\deg_{T_O}(t) = 2$ or equivalently $k = 2$. Let now β be a parameter that shall be fixed later. We distinguish two cases: $c(f_1) + c(f_2) \leq \beta c(p)$ and $c(f_1) + c(f_2) > \beta c(p)$.

Assume (A) $c(f_1) + c(f_2) > \beta c(p)$. Let μ be a parameter which we will fix later. Let T' and T'' be the trees attached to t in T_O with f_1 and f_2 respectively, as shown in Figure 1. Let p' be the cheapest path in T_N connecting T' with T''. Note, that $T' + T'' + p'$ is a feasible solution in (G, S_N) and that $c(T' + T'' + p') \geq c(T_2)$. If there exists an edge g contained in all paths in T_N from T' to T'', then such an edge is unique. Otherwise let g be an imaginary edge of cost 0. By minimality of p' and due to the fact, that T_N is branching on each endpoint of g into disjoint paths to T' or T'', we have $c(p') \leq \frac{c(T_N) - c(g)}{2} + c(g) = \frac{1}{2}(c(T_N) + c(g))$. If $c(g) > \mu c(T_N)$, then by Lemma 2 $Shrink(G, S_N, g)$ returns solution T_1 of cost $c(T_1) \leq (\sigma - \mu(\sigma - 1))c(T_N)$. This value is bounded from above by $(1 + \alpha)c(T_N)$ for $\alpha \geq (\sigma - 1)(1 - \mu)$. If $c(g) \leq \mu c(T_N)$, then $c(T_2) \leq c(T_O) - c(f_1) - c(f_2) + c(p') \leq c(T_N) + c(p) - \beta c(p) + \frac{1}{2}(c(T_N) - c(g))$, therefore $c(T_2) \leq (\frac{3}{2} + \alpha(1 - \beta) + \frac{\mu}{2})c(T_N)$. This in turn is bounded from above by $(1 + \alpha)c(T_N)$ for $\alpha \geq \frac{1+\mu}{2\beta}$. Setting $\frac{1+\mu}{2\beta} = (\sigma - 1)(1 - \mu)$ to obtain later minimal α satisfying both inequalities gives $\mu = \frac{2\beta(\sigma-1)-1}{2\beta(\sigma-1)+1}$ and $\alpha \geq \frac{2(\sigma-1)}{2\beta(\sigma-1)+1}$.

Now assume (B) $c(f_1)+c(f_2) \leq \beta c(p)$. Let p_1, p_2, p_3, p_4 be the shortest paths p_1, p_2, p_3, p_4 connecting t with S_N, such that $p_1 \cap p_2 = f_1$ and $p_2 \cap p_3 = f_2$. If $f_1 = (t, v_1)$ and v_1 is a terminal, then $p_1 = p_2 = f_1$ (same holds for f_2), otherwise there must be two edge disjoint paths from v_1 to S_N not passing trough t. The same argument as in *Case 2* shows that after removing these paths from T_O we can reconnect obtained forest F in polynomial time. The cost of T_2 is bounded from above by $c(T_2) \leq c(T_O)-c(p_1)-c(p_2)-c(p_3)-c(p_4)+c(f_1)+c(f_2)+c(T_N)$, because for reconnecting the obtained forrest optimally we again pay at most $c(T_N)$. Thus $c(T_2) \leq c(T_N) + c(p) - 4c(p) + \beta c(p) + c(T_N)$, what gives $c(T_2) \leq 2c(T_N)-(3-\beta)\alpha c(T_N)$ (assuming $\beta < 3$). This is upper bounded by $(1+\alpha)c(T_N)$ when $\alpha \geq \frac{1}{4-\beta}$. To compute the minimal α for which this inequality and the inequality obtained for Case (A) are satisfied, we set $\frac{1}{4-\beta} = \frac{2(\sigma-1)}{2\beta(\sigma-1)+1}$. That gives $\beta = 2 - \frac{1}{4(\sigma-1)}$ and the minimum value of α is $\frac{4(\sigma-1)}{8(\sigma-1)+1}$. Plugging in $\sigma = 1 + \frac{\log 3}{2}$ guarantees 1.408 approximation ratio for Case 3.

Case 4. Assume $\deg_{T_O}(t) = 1$ or equivalently $k = 1$. In this case, there is exactly one $v \in V(G)$ such that $f_1 = (t, v) \in E(G)$ is incident to t. Tree $T_O - t$ is an optimal Steiner tree for $(G, (S \cup \{v\}) \setminus \{t\})$. Therefore we get a new, smaller problem instance. Since we exclude non-terminals of degree two, either v is a terminal or $\deg_{T_O}(v) \geq 3$. If v is a terminal, the algorithm yields a solution that costs at most $c(T_O - t)$, which is the optimum. Otherwise, $\deg_{T_O - t}(v) \geq 2$ and we have to continue with one of the other three cases. □

5 Adding One Terminal

In this section we present a 1.344-approximation algorithm for the scenario of adding a vertex to the terminal set S, thus improving the result in [5,2].

Algorithm 4. MinSTP-S+

Input: A metric graph G, a terminal set $S_O \subseteq V(G)$, an optimal Steiner tree $T_O \subseteq G$ for (G, S_O) and a new terminal set $S_N := S_O \cup \{t\}$ for some non-terminal t.

1: $T_1 := T_O +$ *CheapestEdge*(T_O, t)
2: Let T_2 be any spanning tree in G
3: **for** $t' \in V(G) \setminus V(T_O), u \in V(G) \setminus \{t, t'\}, v \in V(G) \setminus \{t, t', u\}$ **do**
4: Let T' be a tree on $V(T') = \{t, t', u, v\}$ with edges $E(T') = \{(t, t'), (t', u), (t', v)\}$
5: $T_2 := \min\{T_2,$ *Shrink*$(G, S_N, T')\}$
6: **end for**

Output: $\min\{T_1, T_2\}$

Theorem 2. *Algorithm 4 is a 1.344-approximation algorithm for ST-S+.*

Proof. Let $\alpha > \frac{1}{3}$ be a parameter which we fix later. Because T_N is a feasible solution for the old instance, there holds $c(T_O) \leq c(T_N)$. If $t \in T_O$ then $T_1 = T_O$

is optimal. Otherwise, let $f_{min} = (w, t)$ be a cheapest edge connecting T_O with t. We can assume that $c(f_{min}) > \alpha c(T_N)$, otherwise $c(T_1) \leq c(T_O) + c(f_{min}) \leq (1+\alpha)c(T_N)$ gives $(1+\alpha)$-approximation. Let $f_1, \ldots, f_k$ be the edges that are adjacent to t in T_N. There must be k edge disjoint paths in T_N from t to S_O, and by metricity the cost of each of them is greater then $c(f_{min}) > \alpha c(T_N) > \frac{1}{3}c(T_N)$. Therefore $c(T_N) > \frac{k}{3}c(T_N)$ implies $k \leq 2$. We distinguish two cases.

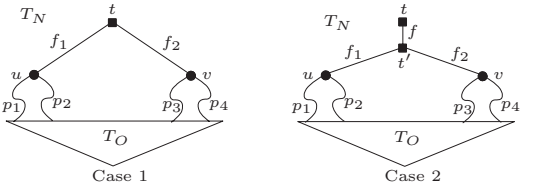

Fig. 2. Pattern for adding a terminal

Case 1: $k = 2$. Let $f_1 = (t, u)$ and $f_2 = (t, v)$. Algorithm 4 exhaustively searches trough all triples of nodes that are candidates for t, u and v. When it hits the triple t, u, v with $t' = t$, it applies $Shrink(G, S_N, T')$ for tree $T' \subseteq T_N$ which contains only edges f_1 and f_2. It looks for minimum T_2 over all triples, thus $c(T_2) \leq Shrink(G, S_N, T')$ for that particular T'. Let p_1 and p_2 be two edge disjoint paths in T_N connecting u with T_O. If $u \notin T_O$, its degree in T_N is at least 2 and therefore such paths exist. If $u \in T_O$, then let $p_1 = p_2 = \{u\}$ and $c(p_1) = c(p_2) = 0$. We define p_3 and p_4 analogically with respect to vertex v. The situation is shown in Figure 2. By metricity the following inequalities hold:

$$\begin{cases} c(f_1) + c(p_1) > \alpha c(T_N) \\ c(f_1) + c(p_2) > \alpha c(T_N) \\ c(f_2) + c(p_3) > \alpha c(T_N) \\ c(f_2) + c(p_4) > \alpha c(T_N) \end{cases} \tag{1}$$

Summing all the inequalities in (1) gives $c(f_1) + c(f_2) + c(T_N) \geq c(f_1) + c(f_2) + c(f_1) + c(f_2) + \sum_{i=1}^{4} c(p_i) > 4\alpha c(T_N)$ and thus $c(f_1) + c(f_2) > (4\alpha - 1)c(T_N)$. Therefore by Lemma 2 we get $c(T_2) \leq c(Shrink(G, S_N, T')) \leq (\sigma - (4\alpha - 1)(\sigma - 1))c(T_N)$. This is bounded by $(1+\alpha)c(T_N)$ for any α satisfying $\alpha \geq \frac{2(\sigma-1)}{4(\sigma-1)+1}$. Plugging in $\sigma = 1 + \frac{\log 3}{2}$ ensures 1.344 approximation ratio in this case.

Case 2: $k = 1$. The situation is shown in Figure 2. Let $f = (t, t')$ be the only edge adjacent to t in T_N. In this case $T_N - t$ is a feasible solution for the old instance, and thus $c(T_O) \leq c(T_N) - c(f)$. Let f_{min} be the cheapest edge connecting t' with T_O. If $c(f_{min}) \leq \alpha c(T_N)$ we have $c(T_1) \leq c(T_O) + c(f_{min}) + c(f) \leq (1+\alpha)c(T_N)$. Assume $c(f_{min}) > \alpha c(T_N)$. If $t' \in T_O$, then T_1 is an optimal solution. Otherwise t' is non terminal and there must be two edges $f_1 = (t', u)$ and $f_2 = (t', v)$ adjacent to t' in T_N. Further analysis is identical as in Case 1, taking t' instead of t, and gives 1.344 approximation ratio. □

6 Increasing the Weight of One Edge

Throughout this and the next section we will assume that a modification of edge weights do not affect the metricity of the graph, as already discussed in Section 2. In this section we consider the local modification where the cost of one edge $e \in E(G)$ increases: $c_n(e) > c_o(e)$. As a consequence $c(T_N) \geq c(T_O)$.

Algorithm 5. MinSTP-E+

Input: A metric graph $G_O = (V, E, c_o)$, a terminal set $S \subseteq V(G)$, an optimal Stainer Tree T_O for (G_O, S), and a new metric graph $G_N = (V, E, c_n)$, where $c_n = c_o$ on all but one edge e for which $c_o(e) \leq c_n(e)$.

1: Let $T_O{}'$ and $T_O{}''$ be the subtrees obtained from T_O by removing e.
2: $T_1 := T_O - e + \mathit{CheapestEdge}(T_O{}', T_O{}'')$
3: Let $e = (u, v)$ and $f_1, \ldots, f_k$ be the edges adjacent to e in T_O
4: $F := T_O - u - v$
5: **if** $k < 3$ **then** $T_3 := \mathit{Connect}(G_N, F)$
6: $T_4 := \min_{x \in V(G) \setminus \{u,v\}, y \in V(G) \setminus \{x,u\}} \{\mathit{Shrink}(G_N, S, T_{(u,x,y)})\}$
$T_5 := \min_{x \in V(G) \setminus \{u,v\}, y \in V(G) \setminus \{x,v\}} \{\mathit{Shrink}(G_N, S, T_{(v,x,y)})\}$
where $T_{(w,x,y)}$ denotes tree on vertices $\{u, v, x, y\}$ spanning edges $\{(u, x), (x, v)(w, y)\}$

Output: $\min\{T_O, T_1, T_3, T_4, T_5\}$

Theorem 3. *Algorithm 5 is a $\frac{4}{3}$-approximation algorithm for ST-E+.*

Proof. Let $\alpha = \frac{1}{3}$. If $e \notin E(T_O)$ or $e \in E(T_N)$, then T_O is an optimal solution for the new instance. Therefore we consider the only non trivial case when $e = (u, v) \in E(T_O)$ and $e \notin E(T_N)$. Let $f_1, \ldots, f_k$ be the edges adjacent to e in T_O. Assume there is an edge $f_i = (u, w) \in E(T_O)$, such that $c(f_i) \leq \alpha c(T_N)$. Let $g = (w, v)$. Then $c(T_1) \leq c(T_O) - c_O(e) + c(g)$, thus by metricity we have $c(T_1) \leq c(T_N) + c(f_i) \leq (1 + \alpha)c(T_N)$. Therefore we can assume that $c(f_i) > \alpha c(T_N) \geq \frac{1}{3}c(T_N)$. Hence $deg_{T_N}(u) + deg_{T_N}(v) \leq 3$. Moreover, since $c(T_N) \geq c(T_O)$, there are at most two such edges in T_O. We can also assume

$$c_n(e) - c_o(e) > \alpha c(T_N) \tag{2}$$

otherwise $c(T_O) \leq (1 + \alpha)c(T_N)$. Observe that for each f_i adjacent to e holds:[1]

$$c(f_i) \geq \frac{c_n(e) - c_o(e)}{2}. \tag{3}$$

If $k = 2$, then $c(T_3) \leq c(T_O) - c(f_1) - c(f_2) - c_o(e) + c(T_N) \leq 2c(T_N) - \frac{2}{3}c(T_N) = \frac{4}{3}c(T_N)$. The remaining case is when there is only one edge f adjacent to e in T_O. In this particular case both u and v are terminal vertices. We distinguish further cases regarding the number of edges in T_N adjacent to e.

[1] Let h be an edge that forms a triangle with e and f_3. By metricity: $c_n(e) + c(f_3) \geq c(h) \geq c_o(e) - c(f_3)$ which gives the inequality we use.

Case 1: $deg_{T_N}(u) + deg_{T_N}(v) = 3$. Let p_{uv} be the path from u to v in T_N. If (A) there are two nodes $x, y \in V(p_{uv})$ other than u and v (see Figure 3, Case 1A), then there must edge disjoint paths from x and y to terminals. Moreover there must be another path edge disjoint with these two from u or v to a terminal, since $deg_{T_N}(u) + deg_{T_N}(v) = 3$. This gives three paths of cost greater than $\frac{1}{3}c(T_N)$ which can not be the case. Assume (B) there is only one node $x \in V(p_{uv})$ other then u and v. This situation is shown in Figure 3, Case 1B. Let $f_1 = (u, x)$, $f_2 = (x, v)$ and w.l.o.g let $f_3 = (v, y)$, $f_1, f_2, f_3 \in E(T_N)$. When computing T_4, the algorithm exhaustively searches trough all candidates for x and y, therefore $c(T_4) \leq c(Shrink(G_N, S, T_{(v,x,y)}))$, where $T_{(v,x,y)}$ spans f_1, f_2, f_3. By metricity and (2) we get $c(f_1) + c(f_2) > \alpha c(T_N)$. By (3) and (2), we get $c(f_3) \geq \frac{c_n(e)-c_o(e)}{2} \geq \frac{\alpha}{2}c(T_N)$. Therefore $c(f_1 + f_2 + f_3) > \frac{3\alpha}{2}c(T_N)$ and Lemma 2 gives the following bound: $c(T_4) \leq \sigma c(T_N) - \frac{3\alpha}{2}(\sigma - 1)c(T_N)$. This guarantees $(1 + \alpha)$-approximation for any $\alpha \geq \frac{2(\sigma-1)}{3(\sigma-1)+2}$. Plugging in best up to date σ ensures in this case 4/3-approximation ratio.

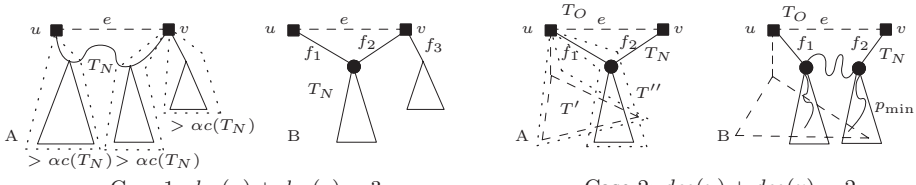

Fig. 3. Pattern for increasing the cost of edge e

Case 2: $\deg_{T_N}(u) + \deg_{T_N}(v) = 2$. In this case we distinguish two subcases (see Figure 3 Case 2), mainly (A) when there is only one node on the path $p_{uv} \subseteq T_N$ from u to v, and when (B) there are at least two nodes on this path. For both cases, let w.l.o.g. $f = (u, w)$ be the only edge in T_O adjacent to e.

Assume that (A) $x \in V(p_{uv})$ is the only node on the path other than u and v. This implies that v is a leaf in T_O. Let $f_1 = (u, x), f_2 = (v, y)$ be the edges adjacent to e in T_N. Since $\deg_{T_N}(u) + \deg_{T_N}(v) = 2$, both u and v are leaves in T_N. Thus, tree $T' = T_O - v$ is an optimal solution for $(G_N, S \setminus \{v\})$ (otherwise we could improve T_O in (G_O, S)). Since v is a leaf in T_N, tree $T'' = T_N - v$ is a feasible solution for $(G_N, S \setminus \{v\})$, and therefore $c(T') \leq c(T'')$. The situation is presented in Figure 3 Case 2A. This gives $c(T_O) - c_o(e) + c(f_2) \leq c(T_N)$. But $c(T_1) \leq c(T_O) - c_o(e) + c(f_1) + c(f_2) \leq c(T_N) + c(f_1)$. It follows immediately, that if $c(f_1) \leq \alpha c(T_N)$, then T_1 is $(1+\alpha)$ - approximation. Otherwise $c(f_1) > \alpha c(T_N)$. From (3) and (2) holds $c(f_2) > \frac{\alpha}{2}c(T_N)$, what together with the above inequality implies $c(f_1) + c(f_2) > \frac{3}{2}\alpha c(T_N)$, and applying $Shrink(G_N, S, T_{(u,x,v)})$ when $y = v$ guarantees by Lemma 2 that $c(T_4) \leq (1 + \alpha)c(T_N)$ for any $\alpha \geq \frac{2(\sigma-1)}{3(\sigma-1)+2}$. Plugging in best up to date σ ensures also in this case 4/3 approximation ratio.

Now assume (B) $f_1 = (u, x), f_2 = (v, y) \in E(p_{uv}) \subseteq T_N$, and $x \neq y$. We may assume $y \notin T_O$, otherwise T_1 is optimal. Since $\alpha \geq \frac{1}{3}$, there must be

$c(T_N) \leq 3\alpha c(T_N)$. By (3) and (2) holds $c(f_1) \geq \frac{\alpha}{2}c(T_N)$ and $c(f_2) \geq \frac{\alpha}{2}c(T_N)$, what implies $c(T_N) - c(f_1) - c(f_2) \leq 2\alpha c(T_N)$. Because y is a non terminal, we have $deg_{T_N}(y) \geq 3$, thus there must be two edge disjoint paths in $T_N - f_1 - f_2$ from y to $S \backslash \{u, v\}$. Minimal such path p_{min} must satisfy $c(p_{min}) \leq \frac{c(T_N)-c(f_1)-c(f_2)}{2} \leq \alpha c(T_N)$. Therefore $c(T_1) \leq c(T_O) - c_o(e) + c(f_2) + c(p_{min}) \leq c(T_N) + c(p_{min}) \leq (1+\alpha)c(T_N)$ gives the desired bound. □

7 Decreasing the Weight of One Edge

In this subsection, we present a 1.302-approximation algorithm for ST-E−. The local modification is the decrease of the cost of one edge: $c_n(e) \leq c_o(e)$.

Algorithm 6. MinSTP-E-

Input: A metric graph $G_O = (V, E, c_o)$, a terminal set $S \subseteq V(G)$, an optimal Stainer Tree T_O for (G_O, S), and a new metric graph $G_N = (V, E, c_n)$, where $c_n = c_o$ on all but one edge $e = (u, v)$ for which $c_o(e) \geq c_n(e)$.

1: **if** $u \in S$ **and** $v \in S$ **then**
2: Let f_{max} be the most expensive edge on the path from u to v in T_O
3: $T_A := T_O - f_{max} + e$
4: **end if**
5: $T_S := \min_{t,w \in \{u,v\}, x,y \in V(G), x \neq y}\{Shrink(G_N, S, T_{(x,t,w,y)}\}$ where $T_{(x,t,w,y)}$ is a tree on $\{x, t, w, y\}$ spanning edges $\{(t, x), (w, y), e\}$

Output: $\min\{T_O, T_A, T_S\}$

Theorem 4. *Algorithm 6 for ST-E− achieves an approximation ratio of* 1.302.

Proof. Let $\alpha = \frac{22}{73}$. Clearly $c(T_N) \leq c(T_O)$. If $e \notin E(T_N)$, then T_N is a feasible solution for (G, S, c_o), and thus T_O is optimal for the new instance. If $e \in E(T_N)$ and $e \in E(T_O)$, then $c_n(T_N) - c_n(e) + c_o(e) \geq c_o(T_O)$ because T_N is feasible in G_O. But that implies $c_n(T_O) \leq c_n(T_N)$ and thus T_O is optimal for (G_N, S). Further we analyze the only non trivial case when $e \notin E(T_O)$ and $e \in E(T_N)$.

Let $f_1, \ldots, f_k \in E(T_N)$ be the edges adjacent to e in T_N. W.l.o.g. let $f_i = (u, x_i)$. Let $g_i = (x_i, v)$. A feasible solution for (G_O, S) is $T_N - e + g_i$, thus $c(T_O) \leq c(T_N) - c_n(e) + c(g_i)$. By metricity $c(g_i) \leq c(f_i) + c_n(e)$, and therefore for each edge f_i adjacent to e in T_N holds

$$c(T_O) \leq c(T_N) + c(f_i). \tag{4}$$

For the remaining part of the proof we can assume that

$$c(f_i) > \alpha c(T_N) \tag{5}$$

$$c_o(e) - c_n(e) > \alpha c(T_N), \tag{6}$$

otherwise, from (4) and from the fact that $c(T_O) \leq c(T_N) - c_n(e) + c_o(e)$ we have that T_O is a $(1+\alpha)$-approximation. We distinguish two cases.

Case 1: $k > 1$ or ($k = 1$ and $c(f_1) + c_n(e) > \frac{3}{2}\alpha c(T_N)$). If there are at least two edges $f_1, f_2 \in E(T_N)$ adjacent to e, $Shrink(G_N, S, T_{(u,v,x,y)})$ must be called at some step for $T_{(u,v,x,y)}$ spanning e, f_1, f_2. Note, that by (5) we get at this step $c(T_{(u,v,x,y)}) \geq c(f_1) + c(f_2) \geq 2\alpha c(T_N) \geq \frac{3}{2}\alpha c(T_N)$. If there is only one edge f_1 adjacent to e in T_N, $Shrink(G_N, S, T_{(u,v,x,y)})$ is called at some step for $T_{(u,v,x,y)}$ spanning e, f_1. Then, $c(T_{(u,v,x,y)}) \geq c(f_1) + c_n(e) > \frac{3}{2}\alpha c(T_N)$. Thus, in both cases from Lemma 2 after calculations and plugging in best up to date σ, we obtain $(1+\alpha)$-approximation for any $\alpha \geq \frac{22}{73}$.

Case 2: $k = 1$ and $c(f_1) + c_n(e) \leq \frac{3}{2}\alpha c(T_N)$. In this case both u and v are terminals, thus Algorithm 6 computes solution T_A. For any edge f adjacent to e in G, by (3), there must hold $c(f) \geq \frac{c_o(e)-c_n(e)}{2}$. Since on the path from u to v in T_O there are two edges adjacent to e, we are guaranteed $c(f_{max}) \geq \frac{c_o(e)-c_n(e)}{2} \geq^{(6)} \frac{\alpha}{2} c(T_N)$. Therefore

$$c(T_A) = c(T_O) - c(f_{max}) + c_n(e) \leq^{(4)} c(T_N) + c(f_1) + c_n(e) - c(f_{max}) \leq (1+\alpha)c(T_N).$$

□

References

1. Bern, M.W., Plassmann, P.E.: The Steiner problem with edge lengths 1 and 2. Inf. Process. Lett. 32(4), 171–176 (1989)
2. Böckenhauer, H.-J., Hromkovič, J., Královič, R., Mömke, T., Rossmanith, P.: Reoptimization of steiner trees: Changing the terminal set. Theoretical Computer Science (to appear)
3. Böckenhauer, H.-J., Hromkovič, J., Mömke, T., Widmayer, P.: On the hardness of reoptimization. In: 34th International Conference on Current Trends in Theory and Practice of Computer Science, SOFSEM 2008, pp. 50–65 (2008)
4. Dreyfus, S.E., Wagner, R.A.: The Steiner problem in graphs. Networks 1, 195–207 (1971/1972)
5. Escoffier, B., Milanic, M., Paschos, V.T.: Simple and fast reoptimizations for the Steiner tree problem. Technical Report 2007-01, DIMACS (2007)
6. Hwang, F., Richards, D., Winter, P.: The Steiner Tree Problems. Annals of Discrete Mathematics, vol. 53. North-Holland, Amsterdam (1992)
7. Prömel, H.J., Steger, A.: The Steiner Tree Problem. Advanced Lectures in Mathematics. Friedr. Vieweg & Sohn, Braunschweig (2002)
8. Robins, G., Zelikovsky, A.: Improved Steiner tree approximation in graphs. In: Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2000, pp. 770–779. ACM Press, New York (2000)