Zuse Institute Berlin

Takustr. 7
14195 Berlin
Germany

DANIEL REHFELDT, THORSTEN KOCH

Generalized preprocessing techniques for Steiner tree and maximum-weight connected subgraph problems

ZIB Report 17-57 (October 2017)

Zuse Institute Berlin
Takustr. 7
D-14195 Berlin

Telefon: +49 30-84185-0
Telefax: +49 30-84185-125

e-mail: `bibliothek@zib.de`
URL: `http://www.zib.de`

ZIB-Report (Print) ISSN 1438-0064
ZIB-Report (Internet) ISSN 2192-7782

Generalized preprocessing techniques for Steiner tree and maximum-weight connected subgraph problems

Daniel Rehfeldt* · Thorsten Koch

Abstract

This article introduces new preprocessing techniques for the Steiner tree problem in graphs and one of its most popular relatives, the maximum-weight connected subgraph problem. Several of the techniques generalize previous results from the literature. The correctness of the new methods is shown, but also their NP-hardness is demonstrated. Despite this pessimistic worst-case complexity, several relaxations are discussed that are expected to allow for a strong practical efficiency of these techniques in strengthening both exact and heuristic solving approaches.

1 Introduction

The *Steiner tree problem in graphs* (SPG) is one of the classical $\mathcal{NP}$-hard problems [5]. However, often propelled by practical applications, also many variants of the SPG have been extensively discussed in the literature. One of the most popular of these is the *maximum-weight connected subgraph problem* (MWCSP). For both problems sophisticated exact solvers exist [7, 10, 11]. An essential component of all these approaches is preprocessing. This article introduces a series of new preprocessing technique for SPG and MWCSP (also commonly called *reduction techniques*) that often generalize previous results. These techniques will be fully integrated into the Steiner class solver SCIP-Jack [3] in the near future.

1.1 Notation

For both SPG and MWCSP we denote the underlying graph by $G := (V, E)$, with vertices V and (undirected) edges E. For any subgraph $S \subseteq G$ (e.g., a Steiner tree) we denote its vertices by V_S and its edges by E_S (please note the difference to the notation $E[W]$ defined in (1) for a set $W \subseteq V$ of vertices). While for the SPG T denotes the set of terminals, for the MWCSP we define $T := \{v \in V \mid p(v) > 0\}$; for both SPG and MWCSP we set $s := |T|$, $n := |V|$ and $m := |E|$. Furthermore, we use the notation $V = \{v_1, ..., v_n\}$ and $T = \{t_1, ..., t_s\}$.

In this article paths are invariably assumed to be simple, i.e., without cycles. The subpath of a path Q between two vertices $v_i, v_j \in V_Q$ will be denoted by $Q(v_i, v_j)$. For any $W \subseteq V$ we define

$$E[W] := \big\{\{v_i, v_j\} \in E \mid v_i, v_j \in W\big\} \tag{1}$$

and

$$\overline{W} := \big\{v_i \in V \mid \exists\{v_i, v_j\} \in E, v_j \in W\big\} \cup W. \tag{2}$$

*Zuse Institute Berlin, Takustr. 7, 14195 Berlin, Germany, koch@zib.de, rehfeldt@zib.de

Additionally, we define $\delta(W) := \big\{\{v_i, v_j\} \in E \mid v_i \in W, v_j \in V \setminus W\big\}$. Similarly, for a subgraph $G' \subseteq G$ and a set $W' \subseteq V_{G'}$ of vertices we define $\delta_{G'}(W') := \big\{\{v_i, v_j\} \in E_{G'} \mid v_i \in W', v_j \in V_{G'} \setminus W'\big\}$.

2 Preprocessing for Steiner tree problems in graphs

Given an undirected, connected graph $G = (V, E)$, costs (or weights) $c : E \to \mathbb{R}_{>0}$ and a set $T \subseteq V$ of *terminals*, the Steiner tree problem in graphs (SPG) asks for a tree $S = (V_S, E_S) \subseteq G$ such that

1. $T \subseteq V_S$ holds,
2. $\sum_{e \in E_S} c(e)$ is minimized.

A tree that satisfies condition 1 is called *Steiner tree*; a tree that additionally satisfies condition 2 is called *minimum Steiner tree*. The sum in 2 is called the *weight* of the Steiner tree S.

The SPG is a classical optimization problem, being the subject of hundreds of research articles, and can also be found in real-world applications (although applications for variations of the SPG are far more prevalent). The by far strongest preprocessing techniques for the SPG are described in [8, 9], which also form the basis of the until today strongest exact SPG solver. This section generalizes some of these techniques and suggests new ones. In the following it will be assumed that an SPG denoted by $P_{SPG} = (V, E, T, c)$ is given.

2.1 Bound-based techniques

Bound-based reductions techniques are preprocessing methods that identify edges and vertices for elimination by examining whether they induce an lower bound that exceeds a given upper bound [8, 12]. In this section a bound-based reduction concept is introduced that generalizes the Voronoi-regions concept from [8].

The base of the reduction technique is the following new concept: a **terminal-regions decomposition** of P_{SPG}—with underlying graph (V, E)—is a partition $H = \big\{H_t \subseteq V \mid T \cap H_t = \{t\}\big\}$ of V such that for each $t \in T$ the subgraph $(H_t, E[H_t])$ is connected. Each of the H_t will be called a *region* of H. Define for all $t \in T$

$$r_H(t) := \min\{d(t, v) \mid v \notin H_t\}. \tag{3}$$

In [8] a special terminal-regions decomposition called *Voronoi-regions decomposition* is used. The more general results presented here allow to improve on the Voronoi preprocessing methods introduced in [8]. However, it will also turn out that finding an optimal terminal-regions decomposition is $\mathcal{NP}$-hard. The following three propositions not only improve on the results from [8] by using a more general decomposition, but also by making use of the following distance function. Given vertices $v_i, v_j \in V$ define $\underline{d}(v_i, v_j)$ as the length of a shortest path between v_i and v_j without intermediary terminals. In [2] an $O(m + n \log n)$ algorithm was introduced to compute for each non-terminal v_i a constant number of $\underline{d}$-nearest terminals $v_{i,1}, v_{i,2}, ..., v_{i,k}$ (if existent) along with the corresponding paths. In the remainder of this section it will be assumed that a terminal-regions decomposition H is given. Moreover, for ease of presentation it will be assumed that the terminals of P_{SPG} are ordered such that $r_H(t_1) \leq r_H(t_2) \leq ... \leq r_H(t_s)$. The following three propositions can be proved similarly to the Voronoi reduction techniques from [8].

Proposition 1. *Let $v_i \in V \setminus T$. If there is a minimum Steiner tree S such that $v_i \in V_S$, then*

$$\underline{d}(v_i, v_{i,1}) + \underline{d}(v_i, v_{i,2}) + \sum_{q=1}^{s-2} r_H(t_q) \tag{4}$$

is a lower bound on the weight of S.

Each vertex $v_i \in V \setminus T$ such that the affiliated lower bound stated in Proposition 1 exceeds a known upper bound can be eliminated. Moreover, if a solution S corresponding to the upper bound is given and v_i is not contained in it, the latter can already be eliminated if the lower bound stated in Proposition 1 is equal to the cost of S. A similar proposition holds for edges in a minimum Steiner tree:

Proposition 2. *Let $\{v_i, v_j\} \in E$. If there is minimum Steiner tree S such that $\{v_i, v_j\} \in E_S$, then L defined by*

$$L := c(\{v_i, v_j\}) + \underline{d}(v_i, v_{i,1}) + \underline{d}(v_j, v_{j,1}) + \sum_{q=1}^{s-2} r_H(t_q) \tag{5}$$

if $base(v_i) \neq base(v_j)$ and

$$L := c(\{v_i, v_j\}) + \min\{\underline{d}(v_i, v_{i,1}) + \underline{d}(v_j, v_{j,2}), \underline{d}(v_i, v_{i,2}) + \underline{d}(v_j, v_{j,1}\} + \sum_{q=1}^{s-2} r_H(t_q) \tag{6}$$

otherwise, is a lower bound on the weight of S.

The following proposition allows to pseudo-eliminate [2] vertices, i.e., to delete a vertex and connect all its adjacent vertices by new edges.

Proposition 3. *Let $v_i \in V \setminus T$. If there is a minimum Steiner tree S such that $\delta_S(v_i) \geq 3$, then*

$$\underline{d}(v_i, v_{i,1}) + \underline{d}(v_i, v_{i,2}) + \underline{d}(v_i, v_{i,3}) + \sum_{q=1}^{s-3} r_H(t_q) \tag{7}$$

is a lower bound on the weight of S.

To efficiently apply Proposition 1, one would like to maximize (4)—and for Proposition 2 and Proposition 3 to minimize (5) and (7), respectively. Unfortunately, this problem turns out to be $\mathcal{NP}$-hard. The decision variant of the problem can be stated as follows. Let $\alpha \in \mathbb{N}_0$ and let $G_0 = (V_0, E_0)$ be an undirected, connected graph with edge cost $c : E \to \mathbb{N}$. Furthermore, set $T_0 := \{v \in V_0 \mid p(v) > 0\}$, and assume that $\alpha < |T_0|$. For each terminal-regions decomposition H_0 of G_0 define $T_0' \subsetneq T_0$ such that $|T_0'| = \alpha$ and $r_{H_0}(t') \geq r_{H_0}(t)$ for all $t' \in T_0'$ and $t \in T_0 \setminus T_0'$. Let:

$$C_{H_0} := \sum_{t \in T_0 \setminus T_0'} r_{H_0}(t). \tag{8}$$

We now define the α terminal-regions decomposition problem as follows: Given a $k \in \mathbb{N}$, is there a terminal-regions decomposition H_0 such that $C_{H_0} \geq k$? In the following proposition it is shown that this problem is $\mathcal{NP}$-complete, which forthwith establishes the $\mathcal{NP}$-hardness of finding a terminal-regions decomposition that minimizes (4), (5), (6), or (7)—which corresponds to $\alpha = 2$ and $\alpha = 3$, respectively.

Proposition 4. *For each $\alpha \in \mathbb{N}_0$ the α terminal-regions decomposition problem is $\mathcal{NP}$-complete.*

Proof. Given a terminal-regions decomposition H_0 it can be tested in polynomial time whether $C_{H_0} \geq k$. This can be done for instance by running Dijkstra's algorithm for each subgraph $(\overline{H}_{t_i}, E[\overline{H}_{t_i}])$, starting from t_i. Consequently, the terminal-regions decomposition problem is in $\mathcal{NP}$.

Next, it will be shown that the ($\mathcal{NP}$-complete [4]) independent set problem can be reduced to the terminal-regions decomposition problem. To this end, let $G_{ind} = (V_{ind}, E_{ind})$ be an undirected, connected graph and $k \in \mathbb{N}$. The problem is to determine whether an independent set in G_{ind} of cardinality at least k exists. To establish the reduction, construct a graph G_0 from G_{ind} as follows. Initially, set $G_0 = (V_0, E_0) := G_{ind}$. Next, extend G_0 by replacing each edge $e_l = \{v_i, v_j\} \in E_0$ with a vertex v_l' and the two edges $\{v_i, v_l'\}$ and $\{v_j, v_l'\}$. Define edge weights $c_0(e) = 1$ for all $e \in E_0$ (which includes the newly added edges). If $\alpha > 0$, choose an arbitrary $v_i \in V_0 \cap V_{ind}$ and add for $j = 1, ..., \alpha$ vertices $t_i^{(j)}$ to both V_0 and T_0. Finally, add for $j = 1, ..., \alpha$ edges $\{v_i, t_i^{(j)}\}$ with $c_0(\{v_i, t_i^{(j)}\}) = 2$ to E_0.

First, one observes that the size $|V_0| + |E_0|$ of the new graph G_0 is a polynomial in the size $|V_{ind}| + |E_{ind}|$ of G_{ind}. Next, $r_{H_0}(v_i) = 2$ holds for a vertex $v_i \in G_0 \cap G_{ind}$ if and only if H_{v_i} contains all (newly inserted) adjacent vertices of v_i in G_0. Moreover, in any terminal-regions decomposition H_0 for (G_0, c_0), it holds that $r_{H_0}(t_i^{(j)}) = 2$ for $j = 1, ..., \alpha$. Hence, there is an independent set in G_{ind} of cardinality at least k if and only if there is a terminal-regions decomposition H_0 for (V_0, E_0, T_0, c_0) such that

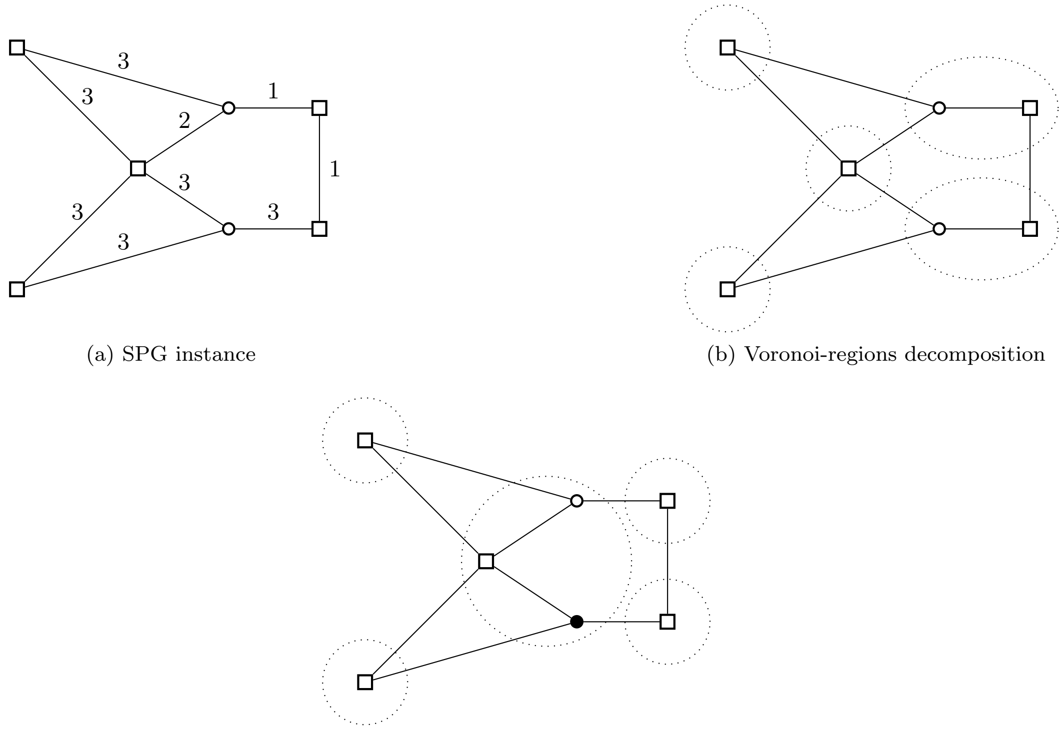
$$C_{H_0} \geq |V_{ind}| + k$$

This proves the proposition. $\square$

A straightforward practical way for computing a terminal-regions decomposition H would be to start Dijkstra's algorithm with all terminals $t \in T$ in the initial priority queue (with distance 0). However, the algorithm should be modified in such a way that it does not extend a region H_{t_i} from a vertex v_j if an upper bound $U_{t_i}^H$ on $r_H(t_i)$ is already known and $d(v_j, t_i) \geq U_{t_i}^H$. Subsequently one can use a local heuristic that checks for each edge between different terminal-regions whether including the whole edge in one of the regions would improve the lower bound C_H. Figure 1 depicts an SPG, a corresponding Voronoi-regions decomposition as described in [8], and an alternative terminal-regions decomposition. The second terminal-regions decomposition yields a stronger lower bound than the Voronoi-regions decomposition and indeed allows to eliminate a vertex if an upper bound that is sufficiently close to the optimal solution value is known. Initial computational experiments for this article have shown that it is in most cases easily possible to improve on the bound provided by the Voronoi-regions decomposition and allow for significantly stronger graph reductions.

2.2 Alternative-based techniques

In the context of Steiner tree problems, alternative-based reduction methods attempt to prove that a specified part of the problem graph is not contained in at least one optimal solution [2]. The usual procedure is to show that for each solution that contains this specified subgraph there is another, alternative, solution of equal or better objective value that does not. While the term subgraph usually means a single vertex or edge, this section describes a technique that allows to eliminate proper subgraphs. It is based on the following bottleneck distance concept. An *elementary path* is a path containing terminals only (but not necessarily) at its endpoints. Any path Q can then be broken into one or more elementary paths. The length of a longest



(a) SPG instance

(b) Voronoi-regions decomposition

(c) terminal-regions decomposition

Figure 1: Illustration of a Steiner tree instance (a), a Voronoi-decomposition (b), and a second terminal-regions decomposition (c). Terminals are drawn as squares. If an upperbound less than 11 is known, the vertex drawn filled in (c) can be deleted by means of the terminal-regions decomposition depicted in (c), but not by means of the Voronoi-regions decomposition.

elementary path in Q is called *Steiner distance* of Q. Accordingly, the *bottleneck Steiner distance* (also referred to as *special distance* [6]) $s(v_i, v_j)$ between two (distinct) vertices $v_i, v_j \in V$ is the minimum Steiner distance taken over all paths between v_i and v_j.

For a subset $\Delta \subseteq V$ we denote by K_Δ the complete, undirected graph on Δ. Furthermore, define for each edge $\{v_i, v_j\}$ of K_Δ (i.e, $v_i, v_j \in \Delta$) weights $s_\Delta(\{v_i, v_j\}) := s(v_i, v_j)$—with s denoting the bottleneck Steiner distance in the orginal problem P_{SPG}. Note that in this way there is for each edge in K_Δ a corresponding path in (V, E). For a subset $\Delta \subseteq \overline{W} \setminus W$ we define the SPG $P_\Delta^W = (V_\Delta^W, E_\Delta^W, T_\Delta^W, c_\Delta^W)$ by

$$\begin{aligned}
V_\Delta^W &:= \Delta \cup W, \\
E_\Delta^W &:= \big\{\{v_i, v_j\} \in E \mid v_i \in W, v_j \in \Delta \cup W\big\}, \\
T_\Delta^W &:= \Delta, \\
c_\Delta^W &:= c|_{E_\Delta^W}.
\end{aligned}$$

This construction sets the stage for:

Proposition 5. *Let $W \subseteq V \setminus T$ such that $(W, E[W])$ is connected. If for each $\Delta \subseteq \overline{W} \setminus W$ with $|\Delta| \geq 2$ it holds that the weight of a minimum spanning tree in (K_Δ, s_Δ) is smaller than* *the weight of a minimum Steiner tree for P_{Δ}^{W}, then each optimal solution S to P_{SPG} satisfies $V_S \cap W = \emptyset$.*

Proof. Let S be a Steiner tree such that $V_S \cap W \neq \emptyset$. Let S' be the subgraph obtained by deleting $W \cap V_S$ and all incident edges from S. Moreover, let $\Delta_S \subseteq \Delta$ such that in each (inclusion-wise maximal) connected component of S' there is exactly one vertex of Δ_S. If $|\Delta_S| = 1$, then S' is already a Steiner tree and furthermore of smaller weight than S. Otherwise, let $C_{\Delta_S}^{SPG}$ be the weight of a minimum Steiner tree in $P_{\Delta_S}^{W}$ and $C_{\Delta_S}^{MST}$ the weight of a minimum spanning tree F_{Δ_S} in $(K_{\Delta_S}, s_{\Delta_S})$. By the assumptions of the proposition it holds that

$$0 < C_{\Delta_S}^{MST} < C_{\Delta_S}^{SPG}. \tag{9}$$

By construction S' consists of $k \in \mathbb{N}$ (inclusion-wise) maximal subtrees $S^{(1)}, ..., S^{(k)}$ such that $\Delta_S \cap V_{S^{(i)}} \neq \emptyset$ for $i = 1, ..., k$. These subtrees can be joined to a connected subgraph by paths in (V, E) corresponding to edges of F_{Δ_S}. Assume that the subtrees $S^{(1)}, ..., S^{(k)}$ are ordered such that for each $i \in \{2, ..., k\}$ there are vertices $v_q^{(i)} \in V_{S^{(j)}} \cap \Delta_S$ with $j < i$ and $v_r^{(i)} \in V_{S^{(i)}} \cap \Delta_S$ such that there is a $(v_q^{(i)}, v_r^{(i)})$-path $Q^{(i)}$ in (V, E) corresponding to an edge in F_{Δ_S}. Set $\hat{S}^{(1)} := S^{(1)}$ and proceed for $i = 2, ..., k$ as follows. Let $Q^{(i)}$ be a $(v_q^{(i)}, v_r^{(i)})$-path as defined above. This path cannot contain any vertices of W because otherwise there would be vertices $v_a, v_b \in (\overline{W} \setminus W) \cap V_{Q^{(i)}}$ such that their bottleneck Steiner distance (in P_{SPG}) is greater than or equal to the weight of a minimum Steiner tree for $P_{\{v_a, v_b\}}^{W}$ (i.e., a shortest path), which would contradict the assumptions of the proposition. Consequently, there is a subpath $Q^{(i)}(v_x, v_y)$ of $Q^{(i)}$ such that $V_{Q^{(i)}(v_x, v_y)} \cap V_{S^{(i)}} = v_x$ and $V_{Q^{(i)}(v_x, v_y)} \cap V_{\hat{S}^{(i-1)}} = v_y$. This path cannot contain any terminals apart from possibly v_x or v_y and is therefore of weight smaller than or equal to $s(v_q, v_r)$. Define $\hat{S}^{(i)} := \hat{S}^{(i-1)} \cup S^{(i)} \cup Q^{(i)}(v_x, v_y)$. Ultimately, $\hat{S} := \hat{S}^{(k)}$ is a Steiner tree with $V_{\hat{S}} \cap W = \emptyset$. Furthermore, it holds that

$$\sum_{e \in E_{\hat{S}}} c(e) \leq \sum_{e \in E_{S'}} c(e) + C_{\Delta_S}^{MST} \leq \sum_{e \in E_S} c(e) + C_{\Delta_S}^{MST} - C_{\Delta_S}^{SPG} < \sum_{e \in E_S} c(e). \tag{10}$$

In summary, $\hat{S}$ is a Steiner tree that does not contain any vertex of W and is of smaller weight than S. Thus the proposition is proven. □

One can formulate a corollary to this proposition that takes weaker assumptions, but also requires to possibly add additional edges after eliminating W. This corollary generalizes the *non-terminal of degree k* test that has been proved to be practically highly successful [2, 8]. As a prerequisite, let $W \subseteq V \setminus T$, $\Delta \subseteq V \setminus W$ and define by K_{Δ}^{W} the complete, undirected graph on Δ. Further define for each edge $\{v_i, v_j\}$ in K_{Δ}^{W} the weight $s_{\Delta}^{W}(\{v_i, v_j\})$ as the bottleneck Steiner distance between v_i and v_j in the Steiner tree problem $(V \setminus W, E[V \setminus W], T, c)$—note that $s_{\Delta}^{W}(\{v_i, v_j\}) = \infty$ is possible.

Corollary 6. *Let $W \subseteq V \setminus T$ such that $(W, E[W])$ is connected. If for each $\Delta \subseteq \overline{W} \setminus W$ with $|\Delta| \geq 3$ it holds that the weight of a minimum spanning tree in $(K_{\Delta}^{W}, s_{\Delta}^{W})$ is greater than the weight of a minimum Steiner tree for P_{Δ}^{W}, then each optimal solution S to P_{SPG} satisfies $|E_S \cap \delta(W)| \leq 2$.*

If the conditions of Corollary 6 are satisfied for a set W, one can delete W and add for each pair of distinct vertices $v_i, v_j \in \overline{W} \setminus W$ an edge of weight equal to the length of a shortest path between v_i and v_j in the network $(\overline{W}, E[\overline{W}]), c|_{E[\overline{W}]})$. In case of parallel edges all but a shortest one are deleted.

Since both Proposition 5 and Corollary 6 contain an NP-hard subproblem, they are not practical for larger sets W. However, for the case of $|W| = 2$ or $|W| = 3$ a minimum Steiner tree for P_{Δ}^{W} can be found easily. Figure 2 shows a segment of a Steiner tree instance for which a set of two vertices (and five incident edges) can be deleted by using Proposition 5, while other common reduction techniques for the SPG do not allow for eliminations.

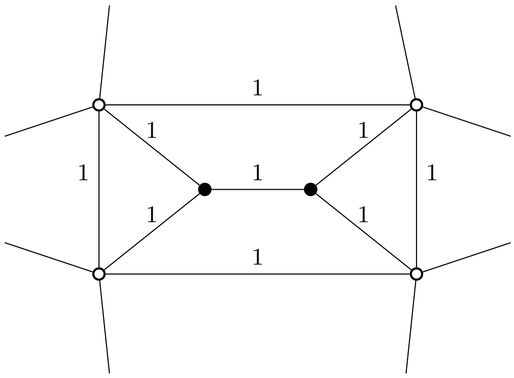


Figure 2: Segment of a Steiner tree instance (showing only non-terminals). The two central (filled) vertices can be deleted by employing Proposition 5.

3 Preprocessing for maximum-weight connected subgraph problems

Given an undirected graph $G = (V, E)$ and vertex weights $p : V \to \mathbb{R}$, the maximum-weight connected subgraph problem is to find a connected subgraph $S \subseteq G$ such that its *weight* $\sum_{v \in V_S} p(v)$ is maximized. Throughout this section we consider a MWCSP $P_{MW} = (V, E, p)$ with the property that at least one vertex is assigned a negative and one a positive weight (otherwise the problem can be solved trivially).

The most comprehensive collections of reduction techniques for the MWCSP can be found in [11] and [12]. While (arguably) interesting in their own right, reduction techniques also form the backbone of the most sucessful solvers for the MWCSP [7, 11]. In the following, results from [11] and [12] are generalized.

3.1 Dominating connected sets

The subsequently described preprocessing—by what we will call *MWCS-dominating sets*—builds on the MWCSP having vertex- but no edge weights. The first proposition generalizes the concept of MWCSP-domination introduced in [12].

Proposition 7. *Let $W \subseteq V \setminus T$ and $U \subseteq V \setminus W$ be non-empty sets such that $(U, E[U])$ is connected and the following two properties hold*

$$\overline{W} \setminus W \subseteq \overline{U}, \tag{11}$$

$$\sum_{w \in W} p(w) \leq \sum_{u \in U : p(u) < 0} p(u). \tag{12}$$

Then there exists a maximum-weight connected subgraph S such that $W \nsubseteq V_S$. The set U will be referred to as ***all-weights MWCS-dominating*** *to W.*

Proof. Let S be a connected subgraph with $W \subseteq V_S$. Note that by construction $p(w) \leq 0$ for all $w \in W$. Define

$$\Delta_S := \big\{v \in V_S \setminus W \mid \exists \{v, w\} \in E_S, w \in W\big\}.$$

Next, remove W from S to obtain a new (possibly empty) subgraph S' that contains at most $|\Delta_S|$ many (inclusion-wise maximal) connected components. If S' is connected, no further discussion is necessary. Otherwise, note that each connected component of S' contains a vertex $v_i \in \Delta_S$. Therefore, these components can be reconnected as follows. First, add $U \setminus V_{S'}$ to S' to obtain a new subgraph S''. Second, because of condition (11) and the fact that each maximal connected component contains a $v_i \in \Delta_S$ there exists a set of edges $\tilde{E}_{S''} \subseteq E[V_{S''}]$ that reconnects S''. Adding $\tilde{E}_{S''}$ to S'', one obtains a, finally connected, subgraph S'''. Finally, the construction of S''' implies:

$$\sum_{v \in V_{S'''}} p(v) \geq \sum_{v \in V_S} p(v) - \sum_{w \in W} p(w) + \sum_{u \in U : p(u) < 0} p(u) \overset{(12)}{\geq} \sum_{v \in V_S} p(v).$$

This concludes the proof. □

Unfortunately, finding to a given $W \in V \setminus T$ an all-weights MWCS-dominating set U turns out to be $\mathcal{NP}$-hard, as shown for the special case $|W| = 1$ in [11]. Nevertheless, the practically efficient heuristics described in [11] for eliminating vertices can be easily expanded by using Proposition 7 to also eliminate edges.

As a variation of Proposition 7 one can formulate the following proposition that allows to eliminate arbitrary subgraphs of $(V \setminus T, E[V \setminus T])$, but also involves a more restricting test condition.

Proposition 8. *Let $W \subseteq V \setminus T$ and $U \subseteq V \setminus W$ be non-empty sets such that $(U, E[U])$ is connected and the following two properties hold*

$$\overline{W} \setminus W \subseteq \overline{U} \tag{13}$$

$$\max_{w \in W} p(w) \leq \sum_{u \in U : p(u) < 0} p(u) \tag{14}$$

Then there exists a maximum-weight connected subgraph S such that $W \cap V_S = \emptyset$. The set U will be referred to as ***max-weight MWCS-dominating*** *to W.*

Proof. Let S be a connected subgraph with $W \cap V_S \neq \emptyset$. Further, define Δ_S as in the proof of Proposition 7. Remove $W \cap V_S$ from S to obtain a new (possibly empty) subgraph S' that contains at most $|\Delta_S|$ many maximal connected components. Assume that there are at least two connected components. Each of these components contains a vertex $v_i \in \Delta_S$. These components can therefore be reconnected as in the proof of Proposition 7 by adding $U \setminus V_{S'}$ to S'. Because of condition (14) it holds for the resulting connected subgraph S''' that $\sum_{v \in V_{S'''}} p(v) \geq \sum_{v \in V_S} p(v)$ and furthermore it holds by construction that $W \cap V_{S'''} = \emptyset$. □

For the special case of $|W| = 1$ a vertex set U is max-weight MWCS-dominating to W if and only if it is all-weights MWCS-dominating to W. Due to the above remarks, finding to a set $W \subseteq V \setminus T$ a max-weight MWCS-dominating connected set $U \subseteq V \setminus W$ is therefore $\mathcal{NP}$-hard. As above, however, the authors expect heuristic methods that only consider sets W of small cardinality (e.g. smaller than 4) to be efficient in practice. Figure 3 shows a MWCSP instance that can be reduced by Proposition 8.

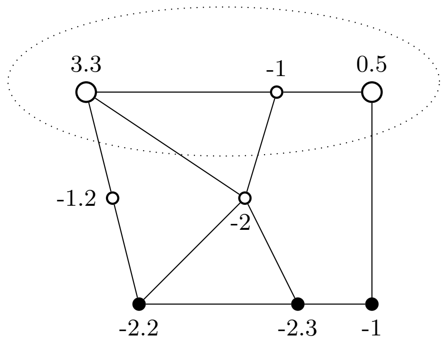


Figure 3: An MWCSP instance. Considering the vertices enclosed by the dotted ellipse as the set U, one can verify with Proposition 8 that the three filled vertices can be deleted.

3.2 From connected sets to bottleneck distances

Instead of showing that a subgraph is dominated by a connected subgraph, as in the previous section, one can use a perhaps more refined argument based on the concept of vertex weight bottleneck distances. This distance function for the MWCSP was first introduced in [12]. Let $v_i, v_j \in V$ be two distinct vertices and $\mathcal{Q}(v_i, v_j)$ the set of all simple paths between v_i and v_j. We define the *interior cost* of such a subpath as:

$$C^-(Q(x,y)) := \sum_{v \in V_{Q(x,y)} \setminus \{x,y\}} p_v. \tag{15}$$

Furthermore, we define the *vertex weight bottleneck length* of Q as:

$$\hat{l}(Q) := \min_{x,y \in V_Q} C^-(Q(x,y)). \tag{16}$$

Finally, the *vertex weight bottleneck distance* between v_i and v_j is defined as

$$\hat{d}(v_i, v_j) := \max\{\hat{l}(Q) \mid Q \in \mathcal{Q}(v_i, v_j)\}. \tag{17}$$

As shown in [11], computing the vertex weight bottleneck distance is $\mathcal{NP}$-hard, but heuristic relaxations can still lead to powerful reduction methods. One such method is the *Non-Positive Vertex of degree k* (NPV_k) test introduced in [12]. The following proposition considerably generalizes the underlying test condition. Initially, for any $U \subseteq V$ denote by K_U the complete graph on U. Furthermore, define for each edge $\{v_j, v_k\}$ of K_U weights $\hat{d}_U(\{v_j, v_k\}) := \hat{d}(v_j, v_k)$—with $\hat{d}$ being the vertex weight bottleneck distance in the original problem P_{MW}. Let $W \subseteq V \setminus T$. For any subset $\Delta \subseteq \overline{W} \setminus W$ we define the MWCSP $P^W_\Delta = (V^W_\Delta, E^W_\Delta, p^W_\Delta)$ by

$$\begin{aligned} V^W_\Delta &:= \Delta \cup W, \\ E^W_\Delta &:= \big\{\{v_i, v_j\} \in E \mid v_i \in W, v_j \in \Delta \cup W\big\}, \\ p^W_\Delta(v) &:= \begin{cases} 0, & \text{if } v \in \Delta \\ p(v), & \text{if } v \in W \setminus \Delta. \end{cases} \end{aligned}$$

This construction sets the stage for:

Proposition 9. *Let $W \subseteq V \setminus T$ such that $(W, E[W])$ is connected. If for each $\Delta \subseteq \overline{W} \setminus W$ with $|\Delta| \geq 2$ it holds that the weight of a maximum spanning tree on $(K_\Delta, \hat{d}_\Delta)$ is greater than the* *weight of a maximum-weight connected subgraph S_W in P_Δ^W that satisfies the additional condition that $\Delta \subset V_{S_W}$, then there is an optimal solution S to P_{MW} that satisfies $W \cap V_S = \emptyset$.*

Proof. Let S be a connected subgraph with $W \cap V_S \neq \emptyset$. Let S' be the subgraph obtained obtained from S by removing the vertices $W \cap V_S$ and their incident edges and let $S^{(1)}, ..., S^{(k)}$ be its (inclusion-wise maximal) connected components. If $k = 1$, then S' is already a connected subgraph and no further discussion is necessary. Otherwise, let $\Delta_S \subseteq \Delta$ such that $|S^{(i)} \cap \Delta| = 1$ for $i = 1, ..., k$. Further, let F_{Δ_S} be a maximum spanning tree on $(K_{\Delta_S}, \hat{d}_{\Delta_S})$ and denote its weight by $C_{\Delta_S}^{MST}$. Let S_W be a maximum-weight connected subgraph S_W that satisfies the additional condition $\Delta_S \subseteq V_{S_W}$. Let C_W^{MWCS} be the weight of S_W. It holds that

$$C_W^{MWCS} < C_{\Delta_S}^{MST} \leq 0. \tag{18}$$

Assume that the maximal connected subgraphs $S^{(i)}$ are ordered such that for each $i \in \{2, ..., k\}$ there are vertices $v_q^{(i)} \in V_{S^{(j)}} \cap \Delta_S$ with $j < i$ and $v_r^{(i)} \in V_{S^{(i+1)}} \cap \Delta_S$ such that there is a $(v_q^{(i)}, v_r^{(i)})$-path $Q^{(i)}$ in (V, E) corresponding to an edge in K_{Δ_S} (i.e., $\hat{l}(Q^{(i)}) = \hat{d}(v_q^{(i)}, v_r^{(i)})$). Set $\hat{S}^{(1)} := S^{(1)}$ and proceed for $i = 2, ..., k$ as follows. First, observe that $V_{Q^{(i)}} \cap W = \emptyset$ due to the assumptions of the proposition. Consequently, there is a subpath $Q^{(i)}(v_x, v_y)$ of $Q^{(i)}$ (with $v_x, v_y \in V$) such that $V_{Q^{(i)}(v_x,v_y)} \cap V_{S^{(i)}} = v_x$ and $V_{Q^{(i)}(v_x,v_y)} \cap V_{\hat{S}^{(i-1)}} = v_y$. This path is of weight greater than or equal to $\hat{d}(v_q^{(i)}, v_r^{(i)})$. Define $\hat{S}^{(i)} := \hat{S}^{(i-1)} \cup S^{(i)} \cup Q^{(i)}(v_x, v_y)$. Ultimately, $\hat{S} := \hat{S}^{(k)}$ is a connected subgraph and it holds that

$$P(\hat{S}) \geq P(S') + C_{\Delta_S}^{MST} \geq P(S) + C_{\Delta_S}^{MST} - C_W^{MWCS} \overset{(18)}{>} P(S). \tag{19}$$

In summary, $\hat{S}$ is a maximum-weight connected subgraph with $V_{\hat{S}} \cap W = \emptyset$ and $P(\hat{S}) > P(S)$. Thus the proposition is proven. □

Besides the bottleneck distance, Proposition 9 includes another $\mathcal{NP}$-hard component, the computation of a connected subgraph in W of maximum weight that contains a predefined subset of vertices. Therefore, practical tests should only be performed for sets W of small cardinality.

3.3 Combining dominating sets and bottleneck distances

While both the concept of MWCS-dominating sets and vertex weight bottleneck distances can be used individually for designing preprocessing methods, their combination leads to yet another result.

Proposition 10. *Let $W \subseteq V \setminus T$ and define*

$$\Delta := \overline{W} \setminus W.$$

If $\Delta = \emptyset$, then no optimal solution to P_{MW} contains any vertex of W. Otherwise, let $U \subseteq V \setminus W$ such that

$$\Delta_1 := \Delta \cap \overline{U}$$

is non-empty and $(U, E[U])$ is connected. Define

$$C_1 := \sum_{u \in U : p(u) < 0} p(u). \tag{20}$$

Further, let $\Delta_2 := \Delta \setminus \Delta_1$ and choose for each $v_k \in \Delta_2$ an, arbitrary, $v'_k \in U$. Define

$$C_2 := \sum_{v_k \in \Delta_2} \hat{d}(v_k, v'_k). \tag{21}$$

If

$$C := C_1 + C_2 > \sum_{w \in W} p(w), \tag{22}$$

then each optimal solution S to P_{MW} satisfies $W \nsubseteq V_S$.

Proof. Let S be a connected subset with $W \subseteq V_S$. In the following it will be demonstrated how to construct a connected subgraph S''' that does not contain all vertices of W and satisfies $P(S''') \geq P(S)$. Start with the subgraph S' obtained from S by removing W and all incident edges. It holds that

$$P(S') = P(S) - \sum_{w \in W} p(w). \tag{23}$$

First, let $\Delta_1^S := \Delta_1 \cap V_{S'}$ and initially set $S'' := S'$. Then, add $(U, E[U])$ to S''. Moreover, add for each $v_i \in \Delta_1^S \setminus V_{S''}$ an edge $\{v_i, v_j\}$ for a $v_j \in U$ to S''. Consequently, it holds for S'' that

$$P(S'') \geq P(S') + C_1 \overset{(23)}{=} P(S) - \sum_{w \in W} p(w) + C_1. \tag{24}$$

Second, let $\Delta_2^S := \Delta_2 \cap V_S$. Initially, set $S''' := S''$. Consider each $v_k \in \Delta_2^S \setminus V_{S'''}$ consecutively and choose a (v_k, v'_k)-path Q^k (with v'_k as defined in the statement of this proposition) such that $\hat{l}(Q^k) = \hat{d}(v_k, v'_k)$. Next, if v_k and v'_k are in different connected components of S''', there are vertices $v_q \in V_{Q^k}$ and $v'_q \in V_{Q^k}$ in the connected components of v_k and v'_k, such that $V_{Q(v_q, v'_q)} \cap V_{S'''} = \{v_q, v'_q\}$. Add $Q(v_q, v'_q)$ to S'''. Because of condition (22) there is at least on vertex of W that is not contained in any of these newly added paths. Moreover, because of condition (21) the overall procedure reduces the weight of S'' by at most $|C_2|$. Hence, it holds for the new (now connected) subgraph S''' that

$$P(S''') \geq P(S'') + C_2 \overset{(24)}{\geq} P(S) - \sum_{w \in W} p(w) + C_1 + C_2. \tag{25}$$

Finally, $W \nsubseteq V_{S'''}$ holds and due to (22) it follows from (25) that

$$P(S''') > P(S). \tag{26}$$

Hence the proposition is proven. □

Corollary 11. *Assume that the conditions of Proposition 10 hold, but instead of (22) assume*

$$C_1 + C_2 > \max_{w \in W} p(w). \tag{27}$$

Then each optimal solution S to P_{MW} satisfies $W \cap V_S = \emptyset$.

Once again, for the special case of $|W| = 1$, corollary and proposition coincide. This special case was already introduced in [11] and proved to be an effective preprocessing tool. Figure 4 shows an MWCSP for which a vertex can be deleted by means of this special case.

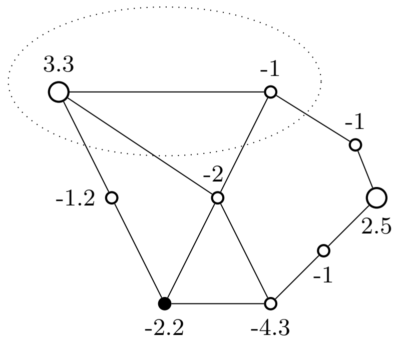


Figure 4: An MWCSP instance. Considering the vertices enclosed by the dotted ellipse as the set U, one can verify with Proposition 10 that the (bottom left) filled vertex can be deleted.

4 Conclusion and outlook

This paper has introduced various new preprocessing techniques for both the Steiner tree problem in graphs and the maximum-weight connected subgraph problem. Although no computational results (and not too many implementation details) have been provided here, initial computational experiments with the Steiner tree solver SCIP-Jack [3] already revealed a great potential of these approaches for strengthening exact solving. We plan to fully integrate the new methods into SCIP-Jack and expect significant improvements both in solving time and solvability, especially for MWCSP. A development version of SCIP-Jack that includes some of the reduction techniques introduced in this article can already solve all MWCSP instances from the 11th DIMACS Challenge [1] (which took place in December 2014) in less than 0.1 seconds—at least three orders of magnitude faster than any solver participating in the Challenge. Furthermore, the new preprocessing techniques could also be useful for other—exact as well as heuristic—solvers for Steiner tree problem in graphs or maximum-weight connected subgraph problem.

5 Acknowledgements

This work was supported by the BMWi project *Realisierung von Beschleunigungsstrategien der anwendungsorientierten Mathematik und Informatik fur optimierende Energiesystemmodelle - BEAM-ME* (fund number 03ET4023DE). The work for this article has been conducted within the Research Campus Modal funded by the German Federal Ministry of Education and Research (fund number 05M14ZAM).

References

[1] 11th DIMACS Challenge. `http://dimacs11.zib.de/`. Accessed: August 10. 2017.

[2] C. Duin. *Steiner Problems in Graphs.* PhD thesis, University of Amsterdam, 1993.

[3] Gerald Gamrath, Thorsten Koch, Stephen Maher, Daniel Rehfeldt, and Yuji Shinano. Scip-jack a solver for stp and variants with parallelization extensions. *Mathematical Programming Computation*, 9(2):231 – 296, 2017.

[4] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman & Co., New York, NY, USA, 1979.

[5] R. Karp. Reducibility among combinatorial problems. In R. Miller and J. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

[6] Thorsten Koch and Alexander Martin. Solving Steiner tree problems in graphs to optimality. *Networks*, 32:207–232, 1998.

[7] M. Leitner, M. Luipersbeck, I. Ljubic, and M. Sinnl. A dual-ascent-based branch-and-bound framework for the prize-collecting Steiner tree and related problems. Technical report, Department of Statistics and Operations Research, 2016.

[8] Tobias Polzin and Siavash Vahdati Daneshmand. Improved Algorithms for the Steiner Problem in Networks. *Discrete Appl. Math.*, 112(1-3):263–300, September 2001.

[9] Tobias Polzin and Siavash Vahdati Daneshmand. *Extending Reduction Techniques for the Steiner Tree Problem*, pages 795–807. Springer Berlin Heidelberg, Berlin, Heidelberg, 2002.

[10] Tobias Polzin and Siavash Vahdati-Daneshmand. The Steiner Tree Challenge: An updated Study. Unpublished manuscript at http://dimacs11.cs.princeton.edu/downloads.html, 2014.

[11] Daniel Rehfeldt and Thorsten Koch. Combining NP-Hard Reduction Techniques and Strong Heuristics in an Exact Algorithm for the Maximum-Weight Connected Subgraph Problem. Technical Report 17-45, ZIB, Takustr.7, 14195 Berlin, 2017.

[12] Daniel Rehfeldt, Thorsten Koch, and Stephen Maher. Reduction techniques for the prize-collecting steiner tree problem and the maximum-weight connected subgraph problem. Technical Report 16-47, ZIB, Takustr.7, 14195 Berlin, 2016.