Parameterized Algorithms for Stochastic Steiner Tree Problems

Denis Kurz, Petra Mutzel, and Bernd Zey

Department of Computer Science, TU Dortmund, Germany
{denis.kurz,petra.mutzel,bernd.zey}@tu-dortmund.de

Abstract. We consider the Steiner tree problem in graphs under uncertainty, the so-called two-stage stochastic Steiner tree problem (SSTP). The problem consists of two stages: In the first stage, we do not know which nodes need to be connected. Instead, we know costs at which we may buy edges, and a set of possible scenarios one of which will arise in the second stage. Each scenario consists of its own terminal set, a probability, and second-stage edge costs. We want to find a selection of first-stage edges and second-stage edges for each scenario that minimizes the expected costs and satisfies all connectivity requirements. We show that SSTP is in the class of fixed-parameter tractable problems (FPT), parameterized by the number of terminals. Additionally, we transfer our results to the directed and the prize-collecting variant of SSTP.

1 Introduction

The Steiner tree problem in graphs (STP) plays a central role in network design [14]. It asks for a minimum-cost subgraph of an undirected, weighted graph G that interconnects a given set of terminal nodes in G. It has applications in VLSI design [15,20], various communication systems, and often appears as a sub-problem of other network design problems [14].

The Steiner tree problem belongs to Karp's classical 21 NP-complete problems [16]. It is known to be NP-hard even if the input graph is unweighted and bipartite, i.e., containing only edges between terminal and non-terminal nodes [14]. Bern and Plassmann showed that the Steiner tree problem is Max-SNP-hard [1]. Therefore, there is no polynomial-time approximation scheme for STP. The best known constant-factor approximation was introduced by Byrka et al. [5] and guarantees an approximation factor of 1.39.

The most popular parameterized algorithm for STP is due to Dreyfus and Wagner [9]. In 1971, they introduced an algorithm that solves STP instances with n nodes, m edges, and t terminals in time $\mathcal{O}(3^t n^3)$, placing it in the complexity class FPT (for an introduction to the field of parameterized complexity see, e.g., [8,17]). Björklund et al. [3] were able to speed up the classical Dreyfus-Wagner algorithm to achieve a running time of $\mathcal{O}(2^t n^2 M + nm \log M)$ if all edge weights are in $\{1, \ldots, M\}$. At the same time, Fuchs et al. [10] published an algorithm with running time $(2+\delta)^t n^{\mathcal{O}(1/(\delta/\ln(1/\delta))^\zeta)}$ for any $\frac{1}{2} < \zeta \leq 1$ and sufficiently

A. Kučera et al. (Eds.): MEMICS 2012, LNCS 7721, pp. 143–154, 2013.

small $\delta > 0$. With respect to the graph's treewidth tw the best parameterized algorithm is due to Chimani et al. [7] and requires running time $\mathcal{O}(B_{tw+2}^2 \cdot tw \cdot n)$, where B_k is the k-th Bell number.

In the directed Steiner tree problem (DSTP), we are given a directed, weighted graph, a root node and a set of terminals. The objective is to find a minimum-cost subgraph that provides a directed path from the root node to each terminal. This problem is of theoretical interest as there are approximation-preserving reductions from many other problems to DSTP [6]. Unfortunately, there is no polylogarithmic approximation algorithm for DSTP unless NP $\subseteq$ ZTIME$(n^{\text{polylog}(n)})$ [13].

In practice, we often have to face uncertainty. For example, a telecommunication company has to deal with volatile cable costs or an unpredictable set of customers. One approach to tackle uncertainty is (two-stage) stochastic optimization, cf. e.g. [2]. In the two-stage stochastic Steiner tree problem (SSTP), we do not know which terminal nodes have to be connected. Instead of buying edges to connect a given terminal set, we may buy edges that seem to be a good choice given a set of future scenarios one of which will eventually arise. We have to pay the current costs in this so-called first stage.

In a second stage, one of the scenarios is drawn at random, based on a previously known distribution. Each scenario is characterized by its terminal set, second-stage edge costs and a probability. We have to buy edges at second-stage costs to extend the first-stage edges to a Steiner tree for the scenario's terminal set. The *expected cost* of a solution is the sum of first-stage edge costs plus the weighted sum of second-stage edge costs of all scenarios. The weights correspond to the scenario's probability.

Obviously, the edges that have to be bought in the second stage depend on the edges bought in the first stage. We want to find a set of first-stage edges that minimizes the expected cost.

The first constant-factor approximation for SSTP was introduced by Gupta et al. [12]. Their approach requires the second-stage costs to be globally uniform over the different scenarios. They also require an inflation factor, i.e., a fixed ratio between first- and second-stage costs for all edges. Further approximation algorithms for the SSTP were provided by Gupta et al. [11] and by Swamy and Shmoys [19]. Chimani et al. [4] introduced an ILP-based exact algorithm for SSTP.

2 Definitions

A directed graph $G = (V, E)$ consists of a finite set V of nodes and a set $E \subset V \times V$ of edges. In an undirected graph, edges are considered to have no direction, and hence $(v, w) = (w, v)$.

A weighted graph $G = (V, E, c)$ is a graph with an additional edge cost function $c : E \mapsto \mathbb{R}_0^+$. It is symmetrical for undirected graphs, i.e., $c((v, w)) = c((w, v))$ for each $(v, w) \in E$.

The optimization problems defined below are minimization problems according to [18]. We characterize them by giving their set of instances, the set of feasible solutions for each instance, and a value function for selected elements.

The undirected and directed version of the Steiner tree problem differ slightly. We will focus on the undirected case first, which is also the most special variant.

Let $G = (V, E)$ be an undirected graph. A *Steiner tree* S for a *terminal set* $T \subseteq V$ in G is a connected subgraph of G that spans T. We allow S to contain non-terminal nodes of G. Such nodes are then called *Steiner nodes*. Note the difference between non-terminal and Steiner nodes: You can tell whether a node is a non-terminal node by just looking at the input. In contrast, a Steiner node is part of the solution. Every Steiner node is also a non-terminal node.

An instance of the *Steiner tree problem* (STP) consists of an undirected, weighted graph $G = (V, E, c)$ and a terminal set $T \subseteq V$. Feasible solutions are subsets of E that form the edge set of a Steiner tree for T in G. Edge values are their costs c. The cost $c(S)$ of S is the sum of costs of its edges.

The cost of a Steiner tree is used as a quality measure when considering Steiner optimization problems. Since it only depends on the edges, we will identify Steiner trees with their corresponding edge sets.

Now, let $G = (V, E)$ be a directed graph with designated root node $r \in V$ and $T \subseteq V$ a terminal set in G. A *Steiner arborescence* is a subgraph S of G such that for every terminal $t \in T$ there exists a directed path from r to t in S. As the name suggests, minimal Steiner arborescences are arborescences rooted in r. Steiner nodes and costs are defined analogously to the undirected case.

An instance of the *Steiner arborescence problem* (*directed Steiner tree problem*, DSTP) consists of a directed, weighted graph $G = (V, E, c)$ with designated root node $r \in V$, and a terminal set $T \subseteq V$. Feasible solutions are subsets of E that form the edge set of a directed Steiner tree for T in G rooted in r. Edge values are their costs c.

An instance of the *prize-collecting Steiner tree problem* (PCSTP) consists of an undirected, weighted graph $G = (V, E, c)$ and node profits $g : V \mapsto \mathbb{R}_0^+$. The set of nodes with positive profit is denoted by $T := \{v \in V \mid g(v) > 0\}$. Feasible solutions consist of a terminal set $\tilde{T} \subset T$ and a Steiner tree $F \subseteq E$ for $\tilde{T}$ in G. The value of a terminal $t \in T$ is given as $-g(t)$, while the value of an edge is its cost c. Therefore, the value of a solution is $\sum_{e \in F} c(e) - \sum_{t \in \tilde{T}} g(t)$.

A *stochastic Steiner tree* combines several Steiner trees for different terminal sets. Let $G = (V, E)$ be a graph, $r \in V$ a designated root node, $T^0 = \{r\}$, and $\mathcal{T} = \{T^k \mid 1 \leq k \leq K\}$ a set of K terminal sets. A stochastic Steiner tree for $\mathcal{T}$ in G rooted in r consists of first stage edges $F^0 \subseteq E$ and second stage edges $F^k \subseteq E$ for each scenario k that meet the following requirements. For each $0 \leq k \leq K$, there must exist a connected Steiner tree for T^k in G whose edge set is $F^0 \cup F^k$. Note that this requires the first stage edges to be connected and to include the root node if there are any first stage edges.

An instance of the *two-stage stochastic Steiner tree problem* (SSTP) consists of an undirected, weighted graph $G = (V, E, c^0)$ with designated root node $r \in V$, and a set of scenarios $\{(T^k, c^k, p^k) \mid 1 \leq k \leq K\}$, each consisting of a terminal set T^k, an edge-cost function c^k, and the scenario's probability p^k, satisfying $\sum_{k=1}^{K} p^k = 1$. Feasible solutions are stochastic Steiner trees for $\{T^k \mid 1 \leq k \leq K\}$ in G rooted in r. The value function v_1 is given as

$$v_1(e) := \begin{cases} c^0(e), & \text{if } e \in F^0 \\ p^k \cdot c^k(e), & \text{if } e \in F^k. \end{cases}$$

In other words, we ask for a stochastic Steiner tree $(F^0, \ldots, F^K)$ that minimizes the expected cost

$$\sum_{e \in F^0} c^0(e) + \sum_{k=1}^{K} p^k \sum_{e \in F^k} c^k(e).$$

Analogously to the SSTP, we define a stochastic variant of PCSTP. An instance of the *two-stage stochastic prize-collecting Steiner tree problem* (SPCSTP) consists of an undirected, weighted graph $G = (V, E, c^0)$ rooted in $r \in V$ and a set of K scenarios. Each scenario consists of a node profit function $g^k : V \mapsto \mathbb{R}_0^+$, edge costs c^k and a probability p^k with $\sum_{k=1}^{K} p^k = 1$. The set of nodes with positive profit in scenario k is denoted by $T^k := \{v \in V \mid g^k(v) > 0\}$. Feasible solutions consist of terminal sets $\mathcal{T} = \{\tilde{T}^k \subseteq T^k \mid 1 \le k \le K\}$ and a stochastic Steiner tree for $\mathcal{T}$ in G rooted in r. The value function v_2 is given as

$$v_2(x) := \begin{cases} v_1(x), & \text{if } x \in E \\ -(p^k \cdot g^k(x)), & \text{if } x \in \tilde{T}^k. \end{cases}$$

Intuitively, we do not force the solution to connect all terminals, but only the most profitable ones. Therefore, we subtract the profit of the nodes that are connected from the cost of the corresponding stochastic Steiner tree. Again, node profits g^k have to be weighted by p^k.

3 Solving SSTP

In the following section we describe an algorithm to solve the two-stage stochastic Steiner tree problem with a running time that is parameterized by the number of terminals summed up over the single scenarios. The approach is based on the algorithm by Dreyfus and Wagner [9], which can be described as follows:

The classical algorithm by Dreyfus and Wagner utilizes the method of dynamic programming: Solve a problem by formulating it recursively and solving the subproblems in increasing order. In fact, Dreyfus and Wagner realized the following recursive nature of the Steiner tree problem. Let S be an optimal Steiner tree for a given weighted, undirected graph $G = (V, E, c)$, and $p \in T \subseteq V, |T| \ge 3$, a terminal. There exists a *joining node* q in S and a partition (T_1, T_2) of T that meets the following requirement. The Steiner tree S can be split into three subtrees S_1, S_2, S_3, where S_1, S_2 are Steiner trees for $T_1 \cup \{q\}, T_2 \cup \{q\}$, respectively, and S_3 is a shortest path from p to q.

The Steiner tree problem is solved by populating a table M of values of subsolutions (p, T'). The subsolutions are solved in increasing order. For $|T'| = 1$, optimal Steiner trees connecting $T' \cup \{p\}$ are shortest paths. For $|T'| > 1$, a

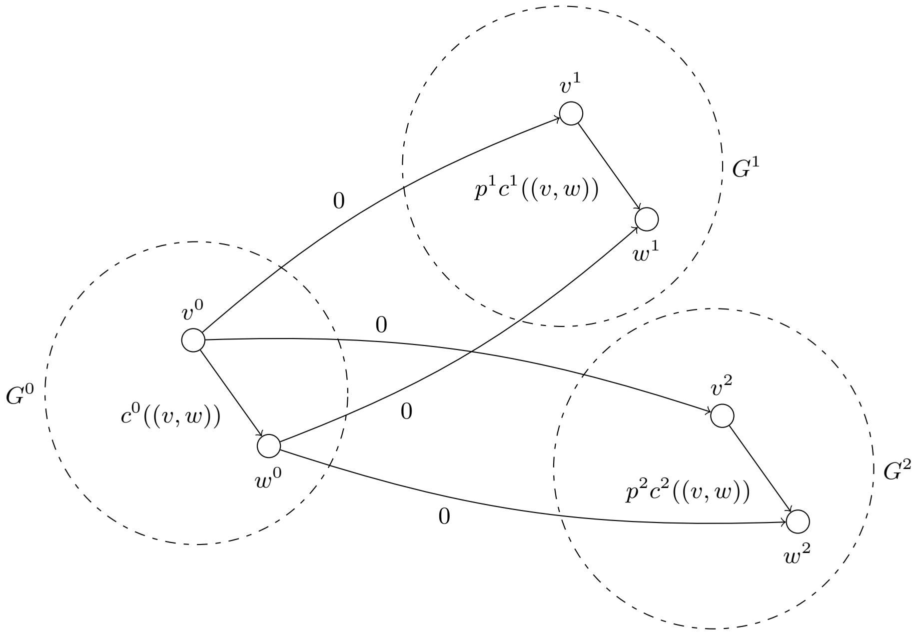

Fig. 1. Schematic diagram of the reduction from SSTP to DSTP with an example edge (v, w)

joining node and a partition of T is found by enumerating the nodes in V and the power set of T, respectively.

This approach can be carried over to the directed Steiner tree problem with designated root by constructing the optimal solution starting from the root. For a subproblem (p, T'), p serves as the root node.

We use these results to solve some stochastic versions of the Steiner tree problem. Let $G = (V, E, c)$ be a weighted, undirected graph with dedicated root node $r \in V$, and $\mathcal{T} := \{(T^k, c^k, p^k) \mid 1 \leq k \leq K\}$ a set of K scenarios. We construct an instance of the DSTP that has the same optimal value as the SSTP instance $(G, r, \mathcal{T})$. An optimal solution to the transformed instance also implies an optimal solution to the SSTP instance, and vice versa.

The directed graph $G' = (V', E', c')$ of this instance contains $K + 1$ copies of G, cf. Figure 1. Its node set V' is the union of all $V^k := \{v^k \mid v \in V\}$, $0 \leq k \leq K$. The edge set E' includes the corresponding edge sets of G, i.e., $E^k := \{(v^k, w^k), (w^k, v^k) \mid (v, w) \in E\}$ for $0 \leq k \leq K$.

Terminals of scenario k in G also become terminals in the new graph. Further, the root node becomes a terminal in the first-stage copy of G. This yields $T' = \{r^0\} \cup \{v^k \mid v \in T^k \setminus \{r\}, 1 \leq k \leq K\}$.

We will interpret the copies like this: Whenever we buy an edge (v^0, w^0) in the transformed version, we also buy this edge in the original SSTP instance, namely in the first stage. Accordingly, buying an edge (v^k, w^k), $k > 0$, suggests buying the corresponding edge in scenario k. We call (V^0, E^0) the first-stage copy and (V^k, E^k), $1 \leq k \leq K$, the second stage copy of the k-th scenario.

Now we have to make sure that there is a way to switch between the different copies of G in H. To accomplish this, we simply add edges between them. As two scenarios k and k' only interact in the first stage, it should not be allowed to switch from k to k' directly. Therefore, we only add *transition edges* between the first-stage copy on the one side and the various second stage copies on the other side. This yields $E' = \tilde{E} \cup \bigcup_{k=0}^{K} E^k$ with $\tilde{E} := \{(v^0, v^k) \mid v \in V, 1 \leq k \leq K\}$. Note that we only add edges from the first- to the second-stage copies, but not in the reverse direction.

The edge weights c' for the edges in E' are applied straightforwardly. Every edge weighs as much as its corresponding edge in E contributes to the value of the stochastic Steiner tree. Hence, first-stage edges remain unchanged, i.e., $c'((v^0, w^0)) = c((v, w))$ for $(v, w) \in E$. Second-stage edges have to regard the probability p^k of the scenario k they belong to, i.e., $c'((v^k, w^k)) = p^k \cdot c^k((v, w))$ for $1 \leq k \leq K$, $(v, w) \in E$. We do not want to restrict the number of transitions between first- and second-stage edges and therefore, we assign cost 0 to edges (v^0, v^k), for all $v \in V$.

We can now use the classical Dreyfus-Wagner algorithm to compute a solution S' to the DSTP instance (G', T'). This solution can be used to derive a solution $S = (F^0, \ldots, F^K)$ to the SSTP instance (G, r, T). We simply choose the edges that were also chosen in the corresponding copy of G in S'. Hence, we buy the edges $F^k := \{(v, w) \in E \mid (v^k, w^k) \in S'\}$ in the k-th scenario for $1 \leq k \leq K$, or in the first stage for $k = 0$.

We have to make sure that for each optimal solution to an SSTP instance there exists an equivalent solution to the transformed STP instance. Let $S = (F^0, \ldots, F^K)$ be a stochastic Steiner tree for $\{(T^k, c^k, p^k) \mid 1 \leq k \leq K\}$ in the undirected, weighted graph $G = (V, E, c^0)$. Let G' be the graph that results from the transformation from G as described before, and T' the corresponding terminal set.

As S is optimal, there is exactly one path $P(t) = (r = v_1, \ldots, v_\ell = t)$ from r to a terminal t of scenario k that only uses edges in $E^0 \cup E^k$. If this path uses first-stage edges, they are all grouped together at the beginning of the path. Otherwise, $P(t)$ would contain alternating fragments of first- and second-stage paths. But since the first stage is connected, this would induce a cycle, which contradicts our assumption that S is optimal. We denote by $\tau(t)$ the index of the transition from first- to second-stage edges, i.e., if $i < \tau(t)$ then $(v_i, v_{i+1}) \in P(t)$ is a first-stage edge and otherwise, $(v_i, v_{i+1}) \in P(t)$ is a second-stage edge.

The deterministic solution S' to (G', T') can be derived as follows. For each path $P(t) = (r = v_1, \ldots, v_\ell = t)$, $t \in T^k$, $1 \leq k \leq K$, we add the edges (v_i^0, v_{i+1}^0) from the first-stage copy to S' if $i < \tau(t)$. If $i \geq \tau(t)$, we add the edge (v_i^k, v_{i+1}^k) from the second-stage copy instead. Further, we add the required transition edge $(v_{\tau(t)}^0, v_{\tau(t)}^k)$.

For every edge e in S, we added an edge e' that contributes as much to the value of S' as e contributes to the value of S. The edges (v^0, v^k) do not contribute to the value of the solution at all. Therefore, the values of S and S' are equal.

In an analogous way an optimum solution to the DSTP instance can be transformed to an SSTP instance with the same objective value.

The new graph $G' = (V', E')$ has exactly $|V| \cdot (K+1)$ nodes and $2 \cdot |E| \cdot (K+1) + K \cdot |V|$ edges. We need to connect $t^* := \sum_{k=1}^{K} |T^k|$ terminals. Using the algorithm by Dreyfus and Wagner to solve the resulting DSTP instance, we obtain a running time of $\mathcal{O}(3^{t^*} \cdot (K \cdot |V|)^3)$. We summarize the previous results in the following theorem.

Theorem 1. *The stochastic Steiner tree problem is fixed parameter-tractable by the number of terminals. It can be solved in time $\mathcal{O}(3^{t^*} \cdot (K \cdot |V|)^3)$, where t^* is the sum of the number of terminals over all scenarios.*

Since DSTP is harder to approximate than STP, the question arises whether the reduction above would work in a similar way if the transition edges were undirected.

While transferring solutions from SSTP to a transformed (undirected) STP instance works without any problems, this is not true for the other way. Consider the STP solution that buys a minimum spanning tree of G with respect to c^1 and every undirected transition edge, i.e., $S' = \{(v^0, v^k) \mid v^0 \in V, 1 \leq k \leq K\} \cup \{(v^1, w^1) \mid (v^1, w^1) \in \mathrm{MST}(G, c^1)\}$. This solution is feasible for the constructed STP instance. The path from the root r to a terminal $t = v_\ell^k$ starts by switching to the second-stage copy of scenario 1 using the edge $(r = v_1^0, v_1^1)$. Following the unique path from v_1^1 to v_ℓ^1 provided by the minimum spanning tree, we reach v_ℓ^k by using (v_ℓ^1, v_ℓ^0) and (v_ℓ^0, v_ℓ^k).

In practice, this solution might even be a good candidate for an optimal solution. Edges in second-stage copies are weighted by a scenario's probability. Thus, they are often cheaper than their counterparts in the first-stage copy, especially if the probability of scenario 1 is very low.

However, transferring S' to the SSTP instance in a way analogous to the one above yields an infeasible SSTP solution S. It only contains edges in the second stage of scenario 1. The terminals of other scenarios remain unconnected.

4 Improvements

So far, we only considered solving SSTP by using the algorithm by Dreyfus and Wagner. This algorithm can be replaced by any other one that solves DSTP.

One way to improve the running time is to use the dynamic programming algorithm by Fuchs et al. [10]. It solves the Steiner tree problem for a terminal set $T \subseteq V$ in a weighted graph $G = (V, E, c)$. First, it adds a portion of $\frac{1}{\varepsilon}|T|$ terminals to the terminal set. An optimal Steiner tree for the new terminal set can then be split into optimal Steiner trees for at most $\varepsilon|T| + 1$ terminals. Obviously, this only works for a suitable selection of $\frac{1}{\varepsilon}|T|$ new terminals. Therefore, every possible selection is considered.

This approach yields an algorithm for STP with running time $(2+\delta)^{|T|} n^{\mathcal{O}(1/(\delta/\ln(1/\delta))^{\zeta})}$ for any $\frac{1}{2} < \zeta \leq 1$ and sufficiently small $\delta > 0$ if we allow the Steiner tree to be constructed from parts of varying size. To utilize this algorithm to solve SSTP, we first need to show that it is capable of finding optimal directed Steiner trees.

Directed Steiner trees can be decomposed at inner terminals just like undirected Steiner trees. In contrast to the undirected STP, where sub-problems are fully characterized by their terminal sets, we also have to determine a root node within each terminal set for DSTP. Consider the directed Steiner tree problem for a terminal set $T \subseteq V$ in a weighted graph $G = (V, E, c)$ with designated root node $r \in T$, and a fixed partition $\mathcal{T}$ into subproblems. The following scheme provides us with a suitable choice of the root node for a given sub-problem with terminal set U.

Let $(\mathcal{T}_1, \dots, \mathcal{T}_\ell)$ be a sequence of sub-problems with $\mathcal{T}_i \in \mathcal{T}$ for $1 \leq i \leq \ell$, $\mathcal{T}_i \cap \mathcal{T}_{i+1} \neq \emptyset$ and $\mathcal{T}_i \neq \mathcal{T}_\ell$ for $1 \leq i \leq \ell - 1$, $r \in \mathcal{T}_1$, and $\mathcal{T}_\ell = U$. In other words, the sequence is a path of sub-problems that starts in the sub-problem that contains r and ends in U, where adjacent sub-problems share at least one terminal.

If $\ell = 1$, we may choose r as the new root. Otherwise, there is exactly one node r' in the cut set $\mathcal{T}_{\ell-1} \cap \mathcal{T}_\ell$. If this cut set contained more than one node, combining the Steiner trees of all sub-problems would induce a cycle. This node r' has to be chosen as the root node for the sub-problem U.

Another way to speed up the computation of stochastic Steiner trees is to speed up the Dreyfus-Wagner algorithm itself. The fast subset convolution introduced by Björklund et al. [3] is one way to achieve this. A careful implementation of an underlying min-sum semiring allows them to compute an optimal Steiner tree in time $\tilde{\mathcal{O}}(2^k n^2 + nm)$ if edge weights are bounded by a constant, where $\tilde{\mathcal{O}}$ hides polylogarithmic factors.

Björklund et al. described how to apply their fast subset convolution to solve STP. However, they utilize the same recursive nature of the Steiner tree problem as Dreyfus and Wagner did. Therefore, the modifications made to the Dreyfus-Wagner algorithm to solve DSTP can also be applied to their approach.

5 Extensions

5.1 Directed SSTP

The reduction technique introduced in Section 3 can not only be used to solve the undirected SSTP. As the resulting graph is directed anyway, we might as well start with a directed graph. It is easy to see that a directed version of SSTP that asks for an arborescence for every scenario can be solved in much the same way as SSTP itself.

5.2 Prize-Collecting SSTP

To solve the prize-collecting version of the Steiner tree problem with the algorithm by Dreyfus and Wagner it needs some modifications. First, in contrast to the Steiner tree problems that connect terminals instead of collecting prizes, we do not have to connect every node of a given set. Instead, we may concentrate on the most profitable ones. Second, the value of a solution not only depends on the selected edges, but also on their incident nodes.

The first difference does not seem to be a problem at all. The Dreyfus-Wagner algorithm computes the value of every sub-problem, anyway. On this problem, every sub-solution that contains the root node is a feasible solution to the problem itself. Therefore, we have to memorize the value of the best solution we have seen so far. This value is only updated if the currently considered sub-problem connects the root node. Notice that it is very well possible that we get to see the optimal solution very early in the computation; This is not possible for terminal-connecting Steiner tree problems.

The latter difference requires some modifications to the Dreyfus-Wagner algorithm. We propose a modification that does not necessarily compute the correct values for every sub-solution. It does, however, compute correct values for the relevant subset of these sub-solutions.

The original Dreyfus-Wagner algorithm computes values $M(v, T')$ of sub-solutions, where v is an arbitrary node of the input graph and T' is a subset of its terminal set T. This value is computed as

$$M(v, T') = \min_{\substack{q \in V \\ T_1 \subset T \setminus \{v\}}} (d_G(v, q) + M(q, T_1) + M(q, T \setminus T_1)),$$

where d_G denotes distances in G.

To make this work for PCSTP, we need to consider the profit of connected nodes. Instead of detecting which profitable nodes with respect to a node-profit function g have been connected, we simply ignore the nodes that were included accidentally. In other words, we only consider profits of nodes that we intended to connect, but not those that were connected by default rather than by design. This yields the following recursive form to compute the values of sub-solutions:

$$M_{\mathrm{PC}}(v, \tilde{T}) = \min_{\substack{q \in V \\ T_1 \cup T_2 = \tilde{T} \setminus \{q\}}} (M_{\mathrm{PC}}(v, \{q\}) + M_{\mathrm{PC}}(q, T_1) + M_{\mathrm{PC}}(q, T_2)) + 2g(q))$$

The profit $g(q)$ of the joining node q has to be added twice because it would otherwise have been subtracted thrice, once for each M_{PC} that q is involved in.

The initialization of M has to be adjusted, too. Dreyfus and Wagner initialize $M(v, \{t\})$ for each node v and each terminal t. We also need $M_{\mathrm{PC}}(v, \{q\})$ for arbitrary node pairs v, q. Further, we have to consider the node profits of the involved nodes. Therefore, we initialize M_{PC} with $M_{\mathrm{PC}}(v, \{q\}) = d_G(v, q) - g(v) - g(q)$ for each $v, q \in V$, $v \neq q$, and $M_{\mathrm{PC}}(v, \{v\}) = -g(v)$ for each $v \in V$.

Consider the value of $M_{\mathrm{PC}}(e, \{a, b\})$ in the example in Figure 2. It is computed as the sum of three sub-solutions, minimized over all joining nodes and

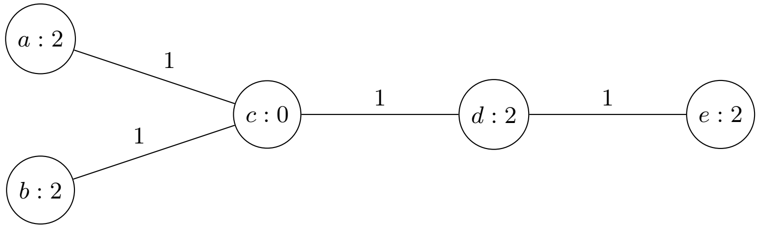

Fig. 2. PCSTP instance that includes sub-problems whose optimal values are not computed correctly; nodes are labelled with their name and their profit; edges are labelled with their cost

all partitions of $\{a, b\}$ with exactly two non-empty sets. Obviously, it can only be split into the sets $\{a\}$ and $\{b\}$. The sum of all sub-solutions for joining node q is $M_{\text{PC}}(e, \{q\}) + M_{\text{PC}}(q, \{a\}) + M_{\text{PC}}(q, \{b\}) + 2g(q)$. This sum evaluates to -2 or -3 for $q = c$ or $q = d$, respectively. In the first case, the edge (d, c) is used twice. In the second case, the profit of node d is ignored. In either case, the computed value is not optimal although the optimal selection of profitable nodes is connected.

However, the correct value of the optimal solution is still computed. We make sure that d is not ignored by including it in the set of nodes we want to connect. Although d is ignored when trying to connect e to $\{a, b\}$ without using it as a joining node, we do not ignore it when we try to connect e to $\{a, b, d\}$. In this case, the value of $M_{\text{PC}}(e, \{a, b, d\})$ would be computed as -4, which is the correct value of the optimal solution that connects all nodes in this example.

This observation can be formulated more general. Let $M_{\text{PC}}(v, \tilde{T})$ be a table entry that has not been computed correctly. The only way $M_{\text{PC}}(v, \tilde{T})$ can be incorrect is by ignoring profits of nodes that have actually been included. Let w be such a node. Then, the value of $M_{\text{PC}}(v, \tilde{T} \cup \{w\})$ is smaller than that of $M_{\text{PC}}(v, \tilde{T})$. This argument can be applied repeatedly until there are no ignored nodes left. The optimal set of profitable nodes is clearly found as we consider every subset of T.

These modifications also work for the directed case. They can thus be used in combination with the reduction technique from Section 3 to solve SPCSTP.

5.3 SSTP without a Root Node

One last extension allows us to provide not only one root for an SSTP instance, but a set of root candidates or no root at all. To allow for this, we do not add r^0 as a terminal during our reduction to the DSTP. Instead, we add a new node ρ that does not correspond to any node in the input graph G. This new node is then connected to the equivalent of each root candidate in the first-stage copy. These unidirectional edges are then equipped with high edge costs with the result that only one of the candidates is chosen.

These three extensions can be combined freely. It is therefore possible to solve, e.g., a directed stochastic prize-collecting Steiner tree problem using the classical algorithm and its modifications.

6 Conclusion and Outlook

We showed that SSTP is fixed-parameter tractable by the number of terminals. This result was subsequently extended to cover the directed and the prize-collecting SSTP.

It remains an open question whether SSTP and its variants can be parameterized by other parameters. One interesting parameter is the treewidth of the input graph. There exist parameterized algorithms for STP that utilize the treewidth. We have yet to investigate if these algorithms can be transferred to SSTP.

Another promising parameter is the number of non-terminals: A simple algorithm for STP tests every subset of non-terminals as the set of Steiner nodes and computes the corresponding minimum spanning tree. This approach might be transferable to the SSTP.

References

1. Bern, M.W., Plassmann, P.E.: The Steiner problem with edge lengths 1 and 2. Information Processing Letters 32(4), 171–176 (1989)
2. Birge, J.R., Louveaux, F.: Introduction to Stochastic Programming. Springer, New York (1997)
3. Björklund, A., Husfeldt, T., Kaski, P., Koivisto, M.: Fourier meets Möbius: Fast subset convolution. In: STOC, pp. 67–74. ACM (2007)
4. Bomze, I., Chimani, M., Jünger, M., Ljubić, I., Mutzel, P., Zey, B.: Solving Two-Stage Stochastic Steiner Tree Problems by Two-Stage Branch-and-Cut. In: Cheong, O., Chwa, K.-Y., Park, K. (eds.) ISAAC 2010, Part I. LNCS, vol. 6506, pp. 427–439. Springer, Heidelberg (2010)
5. Byrka, J., Grandoni, F., Rothvoß, T., Sanitá, L.: An improved LP-based approximation for Steiner tree. In: STOC, pp. 583–592. ACM (2010)
6. Charikar, M., Chekuri, C., Cheung, T., Dai, Z., Goel, A., Guha, S., Li, M.: Approximation algorithms for directed Steiner problems. In: SODA, pp. 192–200. SIAM (1998)
7. Chimani, M., Mutzel, P., Zey, B.: Improved Steiner Tree Algorithms for Bounded Treewidth. In: Iliopoulos, C.S., Smyth, W.F. (eds.) IWOCA 2011. LNCS, vol. 7056, pp. 374–386. Springer, Heidelberg (2011)
8. Downey, R.G., Fellows, M.R.: Parameterized Complexity. Springer (1999)
9. Dreyfus, S.E., Wagner, R.A.: The Steiner problem in graphs. Networks 1(3), 195–207 (1971)
10. Fuchs, B., Kern, W., Mölle, D., Richter, S., Rossmanith, P., Wang, X.: Dynamic programming for minimum Steiner trees. Theory Computing Systems 41(3), 493–500 (2007)
11. Gupta, A., Hajiaghayi, M.T., Kumar, A.: Stochastic Steiner Tree with Non-uniform Inflation. In: Charikar, M., Jansen, K., Reingold, O., Rolim, J.D.P. (eds.) APPROX/RANDOM 2007. LNCS, vol. 4627, pp. 134–148. Springer, Heidelberg (2007)

12. Gupta, A., Pál, M., Ravi, R., Sinha, A.: Boosted sampling: Approximation algorithms for stochastic optimization. In: STOC, pp. 417–426. ACM (2004)
13. Halperin, E., Krauthgamer, R.: Polylogarithmic inapproximability. In: STOC, pp. 585–594. ACM (2003)
14. Hwang, F., Richards, D., Winter, P.: The Steiner tree problem. Annals of discrete mathematics, vol. 53. North-Holland (1992)
15. Kahng, A.B., Robins, G.: On Optimal Interconnections for VLSI. Kluwer Academic Publishers (1995)
16. Karp, R.M.: Reducibility among combinatorial problems. In: Complexity of Computer Computations, pp. 85–103. Plenum (1972)
17. Niedermeier, R.: Invitation to Fixed-Parameter Algorithms. Habilitation, Universität Tübingen (2002)
18. Papadimitriou, C.H., Steiglitz, K.: Combinatorial Optimization: Algorithms and Complexity. Dover Publications (1998)
19. Swamy, C., Shmoys, D.B.: Approximation Algorithms for 2-Stage Stochastic Optimization Problems. In: Arun-Kumar, S., Garg, N. (eds.) FSTTCS 2006. LNCS, vol. 4337, pp. 5–19. Springer, Heidelberg (2006)
20. Uchoa, E., de Aragão, M.P., Ribeiro, C.C.: Preprocessing Steiner problems from VLSI layout. Networks 40(1), 38–50 (2002)