ORIGINAL ARTICLE

Networks

Solving Steiner Trees – Recent Advances, Challenges and Perspectives

Ivana Ljubić

Correspondence
ESSEC Business School of Paris, 95021 Cergy Pontoise Cedex, France and LAMSADE, 75016 Paris, France
Email: ljubic@essec.edu

Funding information

The Steiner tree problem in graphs is one of the most studied problems in combinatorial optimization. Since its inception in 1970, numerous articles published in the journal *Networks* have stimulated new theoretical and computational studies on Steiner trees: from approximation algorithms, heuristics, meta-heuristics, all the way to exact algorithms based on (mixed) integer linear programming, fixed parameter tractability or combinatorial branch-and-bounds. The pervasive applicability and relevance of Steiner trees has been reinforced by the recent 11th DIMACS Implementation Challenge in 2014 and the PACE 2018 Challenge. This article provides an overview of the rich developments from the last three decades for the Steiner tree problem in graphs and highlights the most recent computational studies for some of its closely related variants.

KEYWORDS
Combinatorial Optimization, Steiner Tree Problem, Integer Programming, Exact Methods, Heuristics

1 | INTRODUCTION

The Steiner tree problem in graphs (STP) is among the most prominent problems in the history of combinatorial optimization. It plays a central role in network design problems, design of integrated circuits, location problems [47, 52, 134, 156, 200], and more recently even in machine learning, systems biology and bioinformatics [9, 136, 207, 221].

Together with the traveling salesman problem and other classical NP-hard problems, the STP also serves as a driver for discovering new generic algorithmic and methodological tools that can be easily adapted to other difficult optimization problems. In a general setting, the STP asks for designing a network that interconnects a given set of points (referred to as *terminals*) at minimum cost. The origins of this problem can be traced back to the 17th century. In a private letter of Pierre de Fermat sent to Evangelista Toricelli he raised the question of finding a point in a triangle with the minimum total distance to the three corner points. In the 19th century, Joseph Diez Gergonne generalized the Fermat-Toricelli problem into the problem of finding a set of line segments of minimum total length that interconnects a set of given points in a plane. This was the birth of the problem which is today known as the *Euclidean Steiner Tree Problem* [86, 237]. In an optimal solution such line segments will form a tree, and may intersect at additional intermediary points that are referred to as *Steiner points*. This attribution to Jakob Steiner can be traced back to the book by Courant and Robbins [62]. Many other interesting and controversial historical details related to the Euclidean Steiner tree problem can be found in [35]. In general, in *Geometric Steiner problems*, the problem consists of connecting a set of points at minimum cost according to some geometric distance metric [36].

When the choice of Steiner points is limited to a finite set of points, the problem is formulated on an edge-weighted graph (with terminal points representing a subset of nodes) with the goal of finding a subtree that interconnects all the terminals at minimum cost. The present article is devoted to the latter problem, which is known as the *Steiner Tree Problem in Graphs*. The STP is an indispensable modeling tool for a wide range of applications and has gained a lot of attention in the last 50 years. The STP was explicitly formulated for the first time in 1971 by Hakimi [121] in his seminal *Networks* article. The same year Levin independently introduced the STP in his article written in Russian [157]. In 1972 Karp [141] showed that the decision version of the STP is NP-complete. A plethora of articles on the STP published since then in the journal *Networks* paved the way and stimulated the newest methodological and computational advances not only for the STP, but also for numerous *network design* problems containing the STP as a special case.

Several reviews on the STP have been given in the past. Literature reviews up to 1987 can be found in Maculan [167] and Winter [235]. Maculan reviewed integer linear programming formulations, whereas Winter's *Networks* article gave an overview of exact algorithms, heuristics and polynomially solvable special cases. Voß [231] focused on literature related to heuristic methods up to 1992. A comprehensive survey by Hwang et al. [135] which also included Euclidean, rectilinear and phylogenetic STP variants was published in *Networks* in 1992. Additional, more recent topics on the STP up to 2000 can be found in Du et al. [71]. Extensive contributions to the STP literature have been made by Polzin [180] and Vahdati Daneshmand [228], and an overview of these results can be found in [187]. Many *node-weighted* variants of the STP have been introduced in the last decades. However, to the best of our knowledge the only available survey is by Costa et al. [61], who reviewed the *prize-collecting STP* problem.

This article aims at providing a comprehensive literature overview focusing mainly on the results for the STP achieved in the last three decades. Some important older contributions are mentioned as well, in order to highlight the origin of recent developments. The STP has been the target problem of the 11th DIMACS (the Center for Discrete Mathematics and Theoretical Computer Science) Implementation Challenge held in collaboration with ICERM (the Institute for Computational and Experimental Research in Mathematics) in 2014. Besides the STP, the challenge has spurred algorithmic advances for several other popular problem variants (like the prize-collecting STP, the node-weighted STP and the maximum connected subgraph problem). It has also motivated the creation of new generic state-of-the-art computational methods, that can simultaneously address STP and these variants [95, 102, 153]. Hence, this article also covers some of the most recent and most relevant computational studies initiated and stimulated by the DIMACS Challenge. Finally, the practical applicability of parametrized algorithms for the STP has been investigated at the recent PACE 2018 Parameterized Algorithms and Computational Experiments Challenge [33].

Outline

The remainder of this article is organized as follows. Section 2 contains problem definitions and introduces the basic notation. Mixed Integer Programming (MIP) formulations and theoretical comparisons between them are discussed in Section 3. An overview of recent reduction tests, Lagrangian relaxations, dual ascent methods, heuristics, exact methods and approximation algorithms is given in Sections 4 to 9, respectively. The most challenging and most relevant (recent) applications of the STP are reviewed in Section 10. Section 11 identifies some important combinatorial optimization problems for which the STP has been used as a powerful modeling tool. In Section 12, concluding remarks are drawn and venues for future research are identified.

2 | PROBLEM DEFINITIONS AND NOTATION

The Steiner tree problem in graphs, STP Given an undirected edge-weighted graph $G = (V, E)$, with edge weights $c : E \mapsto \mathbb{R}_{\geq 0}$, and terminal nodes $R \subseteq V$, find a subgraph $T_S = (V_S, E_S) \subseteq G$ such that T_S connects R and $c(T_S) := \sum_{e \in E_S} c_e$ is minimized.

The nodes $S := V \setminus R$ are called *potential Steiner nodes*. An example of a Steiner tree solution is given in Figure 1. Without loss of generality, the STP can be restricted to positive edge weights, since edges of non-positive weight can be preprocessed [235]. If edge weights are restricted to positive values, any optimal subgraph T_S corresponds to a tree. The nodes from $V_S \setminus R$ are referred to as *Steiner nodes*. The STP is NP-hard in general [141], and the two-well known polynomially solvable special cases are the *shortest path problem* ($|R| = 2$) and the *minimum spanning tree* (MST) problem ($R = V$). Similarly, if $|V| - |R|$ is bounded by a (small) constant, it is sufficient to enumerate all possible combinations of Steiner nodes, calculate the resulting minimum spanning trees and choose the best among them [121]. However, the STP remains NP-hard even on bipartite graphs with unit weights [103]. The problem can be approximated within a constant ratio, and the best known polynomial-time algorithm guarantees an approximation ratio of 1.39 [38, 110]. However, Bern and Plassmann [23] show that on complete graphs with edge weights 1 and 2 the STP is MAX-SNP-hard, and hence the STP on general graphs does not admit a polynomial-time approximation scheme, unless P=NP.

The STP can also be stated on a directed graph, in which case the problem is known as the *Steiner arborescence problem on graphs* (SAP). An *arborescence* $S = (V_S, A_S)$ rooted at $r \in V_S$ is a subgraph of a digraph $G = (V, A)$, such that for each node $i \in V_S \setminus \{r\}$ there exists a unique directed path from r to i. One of the first articles addressing the SAP is [240] by Wong. The problem is formally defined as follows.

The Steiner arborescence problem, SAP Given a directed graph $G = (V, A)$ with arc weights $c : A \mapsto \mathbb{R}$, terminal set $R \subseteq V$, and root node $r \in R$, find an arborescence $T_S = (V_S, A_S)$ rooted at r spanning all terminals from $R^r = R \setminus \{r\}$ and such that $\sum_{(i,j) \in A} c_{ij}$ is minimized.

An instance of the STP can be easily transformed into an instance of the SAP: Given an undirected graph $G = (V, E)$, it is sufficient to replace it by its bidirected equivalent $G_D = (V, A)$ in which arc weights are set so that $c_{ij} = c_{ji} = c_e$ for each $e = \{i, j\} \in E$. Due to the symmetry of the arc weights, an arbitrary terminal can be chosen as root r. Like the STP, the SAP is NP-hard [44], but in contrast to the STP, no algorithm with constant approximation ratio can exist unless widely accepted assumptions from complexity theory are wrong [122]. The SAP has not received nearly as much attention as the STP in the existing literature, however the SAP plays a prominent role in solution

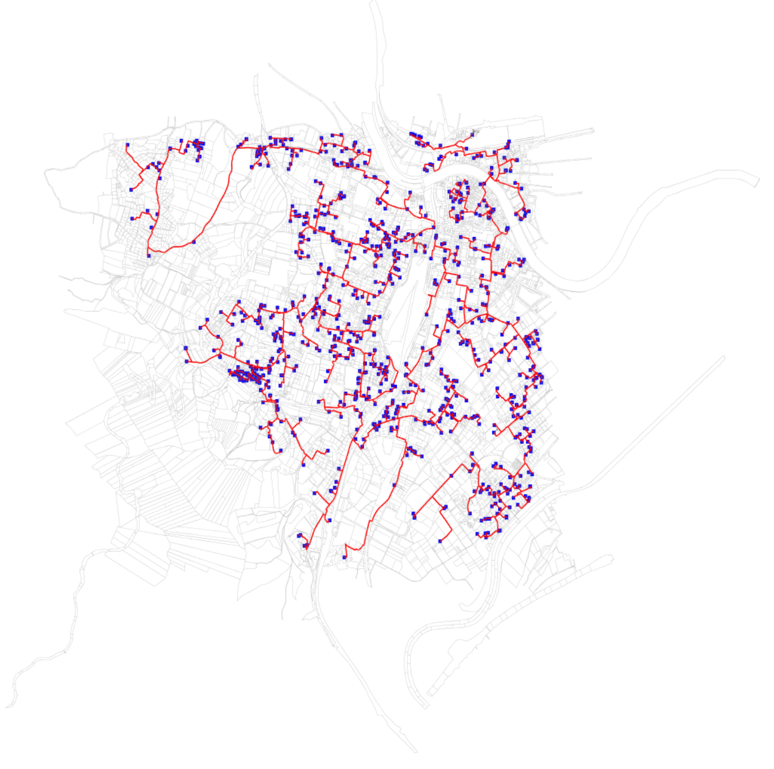

FIGURE 1 Instance `GEO-107` of the STP representing a street network of a small city in Austria, with household locations representing terminal nodes [150, 151]. Depicted is an optimal Steiner tree that connects all terminals at minimum cost.

methods for the STP (see, e.g., the exact methods from [95, 102, 145, 153, 162]). This can be attributed to the fact that MIP formulations for the SAP are in many cases stronger than their undirected counterparts [109]. In this article we follow the definition of the SAP as used in *Networks* in 1993 by Goemans and Myung [109] (see also [93]) in which arc weights can also be non-positive and the terminal set R may be a singleton (i.e., $R = \{r\}$). We notice however that some authors consider a more restricted definition of the SAP in which $|R| > 2$ and arc weights are required to be strictly positive (see, e.g., [197]).

Natural extensions of the STP involve problems with node weights and tree topology. In his 1987 *Networks* article, Segev [213] introduced the *node-weighted STP* (NWSTP), in which the set of terminals is a singleton, all node weights are negative and the goal is to find a subtree that minimizes the sum of edge- and node-weights of that tree. This problem corresponds to the rooted *prize-collecting STP* (PCSTP) and if the required root node is omitted, i.e., $R = \emptyset$, one obtains the *unrooted prize-collecting* STP (PCSTP), whose formal definition is provided below.

The Prize-Collecting Steiner Tree Problem, PCSTP
Given an undirected graph $G = (V, E)$, edge weights $c : E \mapsto \mathbb{R}_{\geq 0}$ and node prizes $p : V \mapsto \mathbb{R}_{\geq 0}$, find a connected subtree $T_S = (V_S, E_S) \subseteq G$ such that $\sum_{e \in E_S} c_e + \sum_{i \notin V_S} p_i$ is minimized.

This definition implies that the PCSTP can be seen as a generalization of the STP (i.e., the STP is the PCSTP with sufficiently large prizes set at the terminal nodes). The first reference to a prize-collecting combinatorial optimization problem goes back to 1989, to another *Networks* article by Balas [13], who introduced the Prize-Collecting Traveling Salesman Problem. In a follow-up article by Bienstock et al. [28], the PCSTP was formulated in its *minimization* form as stated in the above definition. However, the PCSTP has also been studied in its *maximization* form, in which the objective function $\sum_{i \in V_S} p_i - \sum_{e \in E_S} c_e$ is maximized [138]. Concerning MIP formulations, the two variants are known to be equivalent [138, 159]. However from the approximation algorithms point of view, there is an important difference between them. The PCSTP in its minimization form can be approximated within a constant factor (see, e.g., [7, 106]), whereas it is NP-hard to approximate the PCSTP in its maximization form within any constant factor (see e.g., Theorem 4.1 in [88]). Problem transformations into the SAP are known for both rooted and unrooted PCSTP [153, 159, 162, 197]. The *quota-constrained* PCSTP, *budget-constrained* PCSTP, and *fractional* PCSTP problems have been also studied in the literature. A survey on these and other PCSTP variants up to 2006 can be found in Costa et al. [61]. For the sake of brevity, in this article we will highlight only the most recent *computational* advances for solving the PCSTP and its closely related counterpart, the MWCS, introduced below.

The Maximum-Weight Connected Subgraph Problem, MWCS
Given an undirected graph $G = (V, E)$ with node weights $\tilde{p} : V \mapsto \mathbb{R}$, find a connected subgraph $T_S = (V_S, E_S) \subseteq G$ such that $\sum_{i \in V_S} \tilde{p}_i$ is maximized.

Without loss of generality, one can impose that the optimal MWCS solution is a tree. Dittrich et al. [66] exploit this fact to show that any MWCS instance can be transformed into an equivalent PCSTP instance (in its maximization form) by setting $c_e = -\tilde{p}_{\min}$ for all $e \in E$, and $p_v = \tilde{p}_v - \tilde{p}_{\min}$ for all $v \in V$, where $\tilde{p}_{\min} = \min_{v \in V} \tilde{p}_v$ is the smallest node weight.

Notation

For $W \subset V$, let $\delta^+(W) = \{(i, j) \in A : i \in W, j \in V \setminus W\}$ be the *outgoing* arc set, $\delta^-(W) = \{(i, j) \in A : i \in V \setminus W, j \in W\}$ the *incoming* arc set, and $\delta(W) = \{\{i, j\} \in E : i \in V \setminus W, j \in W\}$ the undirected cut set. For conciseness, if $W = \{i\}$, we write $\delta^+(i)$, $\delta^-(i)$, and $\delta(i)$, respectively. Given $s, t \in V$, a node set $N \subseteq V \setminus \{s, t\}$ is an (s, t)-*node separator* if no (s, t)-path exists after removing the nodes from N. N is minimal if for all $i \in N$, $N \setminus \{i\}$ is not a (s, t)-node separator. Given a variable vector **v** and an index set $\mathcal{I}$, let $v(\mathcal{I}) = \sum_{i \in \mathcal{I}} v_i$.

A Full Steiner Component (FSC) is a subtree in G whose internal nodes are Steiner nodes and all of whose leaves are terminals. A Full Steiner Arborescence (FSA) is an oriented FSC whose root is one of the terminals. The metric closure of an edge weighted graph is a complete weighted graph on the same node set, with weights given by shortest path distances with respect to original weights. A hypergraph $\mathcal{H} = (V, \mathcal{E})$ consists of a finite set of nodes V and *hyperedges* $\mathcal{E}$, where each edge $e \in \mathcal{E}$ corresponds to a subset of nodes from V. We will consider *simple* hypergraphs (i.e., with no edges that correspond to the same subset of nodes).

For a minimization problem with non-negative coefficients in the objective function, the *integrality gap* of a given MIP formulation is defined as the worst-case ratio between the optimal solution value and the LP-relaxation value for any given instance. For the same minimization setting, we compare MIP formulations by the quality of their LP-relaxations as follows: a formulation F_1 is said to be *stronger* than a formulation F_2 if the optimal value of the LP-relaxation of F_1, denoted by $LP(F_1) \geq LP(F_2)$ for all instances of the problem. If the converse is also true, the two formulations are *equivalent*; otherwise we say that F_1 is *strictly stronger* than F_2. If neither is stronger than the other, they are *incomparable*.

3 | MIP FORMULATIONS AND RELAXATIONS

For many NP-hard optimization problems most successful exact methods are based on strong MIP formulations. The STP is not an exception, and many MIP models have been studied in the past. A hierarchy of MIP formulations developed before 2001 can be found in Polzin and Vahdati Daneshmand [182]. In what follows, we summarize the results related to polyhedral studies and highlight the most relevant and promising formulations whose full potential is still to be exploited in some future (computational) studies.

Polyhedral studies.

We associate binary variables X_e to each edge $e \in E$ to indicate whether or not the edge belongs to a Steiner tree. The STP polytope, denoted by $\mathcal{P}_{\text{STP}}$, is the convex hull of incidence vectors X representing a feasible Steiner tree. For positive edge weights, an optimal STP solution will correspond to a vertex from the *dominant* of the STP polytope defined as $\mathcal{D}_{\text{STP}} = \mathcal{P}_{\text{STP}} + \mathbb{R}_+^{|E|}$.

In 1980 Aneja [6] proposed a *canonical formulation* for the STP. The model is a set-covering-type formulation that contains an exponential number of constraints, so-called *cut-set inequalities*, ensuring that for each subset of nodes $W \subset V$ such that $W \cap R \neq \emptyset$, $W \not\subseteq R$, at least one edge of the Steiner tree belongs to $\delta(W)$:

$$X(\delta(W)) \geq 1, \quad \forall W : W \cap R \neq \emptyset, W \not\subseteq R.$$

Various polyhedral studies of the STP polytope and its dominant can be found in [29, 54, 55, 56, 109]. Chopra and Rao [54, 55] introduced *Steiner partition* and *odd hole* inequalities (and their lifted variants) and showed that, together with cut inequalities, they define facets of $\mathcal{P}_{\text{STP}}$. The authors also showed that solving the SAP on a bidirected graph leads to better LP-relaxations and that by projecting bidirected cut inequalities (see Section 3.1 for the definition) onto the space of X variables, all above mentioned facets can be obtained. Furthermore, they also identified additional classes of facet defining inequalities that arise on grid graphs and bipartite graphs.

Goemans and Myung [109] introduced two *extended* undirected MIP formulations which are as tight as the bidirected cut formulation. In the first formulation, auxiliary variables are introduced to indicate which nodes from S are spanned by the Steiner tree. This formulation is based on *generalized subtour elimination* constraints (GSECs), which are motivated by Edmonds' [78] characterization of the spanning tree polytope. The GSEC formulation was also introduced independently by Margot et al. [169] and Lucena [163]. In the second formulation proposed by Goemans and Myung [109], the auxiliary variables keep track of the degrees of the nodes from S. In a follow-up article, Goemans [108] used projections to describe a more general class of facets for the STP polytope, coined *combinatorial design* inequalities. The latter inequalities can also be obtained by projecting the bidirected cut formulation onto the space of X variables. Moreover, Goemans [108] showed that Steiner partition and lifted odd hole facets introduced in [54] can be obtained by projecting GSECs. Biha et al. [29] provided a new class of *generalized Steiner partition* inequalities and showed that they generalize facet-defining inequalities from Chopra and Rao [54].

Motivated by the fact that the STP is solvable in polynomial time on series-parallel graphs,[1] several authors dealt with the question of obtaining a simple explicit description of $\mathcal{P}_{\text{STP}}$ and $\mathcal{D}_{\text{STP}}$ by linear inequalities. For the node-weighted STP polytope (involving edge and node variables), complete description is given by the GSEC formulation of Goemans and Myung [109] and Margot et al. [169] when G is a series-parallel graph [108]. Goemans [107] provided a complete description of the SAP polytope for series-parallel graphs. Chopra and Rao [55] conjectured that a complete description of the dominant of the STP polytope on 2-trees (which are maximal series-parallel graphs with respect to edge-inclusion) is given by Steiner partition and odd hole facets. A few years later, this conjecture was disproved by Biha et al. [29].

While there is a rich literature dealing with polyhedral studies for the STP, to our knowledge, there are very few articles focusing on the SAP polytope [56, 93, 107]. Fischetti [93] studied the SAP polytope and its dominant, and provided a new family of facet defining inequalities with arbitrarily large coefficients. Chopra and Tsai [56] used the bidirected counterpart of facet-defining odd-wheel and bipartite inequalities to strengthen the LP-relaxation in their branch-and-cut (B&C) scheme for the STP.

Besides understanding polyhedral properties of MIP formulations, another important feature is the quality of their LP-relaxations. High quality LP-relaxation bounds can be used in combination with preprocessing and reduction techniques to derive efficient branch-and-bound/branch-and-cut schemes. When it comes to successful implementations, another crucial feature of MIP formulations is the running time required to solve their LP-relaxations.

In the past, several authors proposed *extended* MIP formulations for the STP. Besides utilizing X variables, these formulations typically model the underlying tree structure by using auxiliary flow-, path- or tree-variables on G, or by stating the problem as a degree-constrained spanning tree on a larger graph (see, e.g., [22]). Beasley [22], for example, constructed an enlarged graph in which a new node v_0 is connected through zero weight edges to all nodes from S and to an arbitrary terminal from R. The STP is then stated as the minimum spanning tree problem with the additional constraint that each node from S adjacent to v_0 must have degree one. Beasley's MST formulation demonstrates a frequently seen trade-off between the computational efficiency and the theoretical effectiveness of MIP formulations: Even though the model appears attractive from the algorithmic perspective (solving the LPs could be avoided by e.g., relaxing the degree constraints in Lagrangian fashion), Polzin and Vahdati Daneshmand [182] showed that the gap between its LP/Lagrangian-relaxation and the optimal solution can be arbitrarily large. A compendium of MIP formulations (until 2001) can be found in [182].

In the remainder of this section we review four MIP formulations for the STP: (1) the *bidirected cut formulation* (DCUT), which is part of the most successful exact algorithmic frameworks; (2) *hypergraphic formulations* (HYP), which are used to derive the state-of-the-art approximation algorithms; and the two strong compact MIP formulations that

[1] A graph is series-parallel if it does not contain any subgraph homeomorphic to the complete graph K_4 on 4 vertices.

are known to dominate DCUT or HYP: (3) the *common flow formulation* (CF) of Polzin and Vahdati Daneshmand [182] and (4) the brand new *path-based formulation* (MCF-λ)) of Filipecki and Van Vyve [92] published in a recent *Networks* 2020 issue.

3.1 | Bidirected Cut Formulation

This is one of the most frequently employed formulations when it comes to practically successful or state-of-the-art algorithmic frameworks. The model is valid for the SAP as well, and can be easily adapted to the prize-collecting and other STP variants (see, e.g. [95, 102, 153, 162]). The formulation has been proposed by Wong [240], whereas its undirected counterpart was previously given by Aneja [6].

For a given root node $r \in R$, we search for the minimum-cost Steiner arborescence rooted at r and spanning all terminals $t \in R^r = R \setminus \{r\}$. A *Steiner cut* $(V \setminus W, W)$ is a bipartition of the graph's node set such that $r \notin W$ and $R \cap W \neq \emptyset$. The *bidirected cut inequalities* (1) impose that at least one arc from each Steiner cut must belong to a feasible solution. That way, connectivity between a chosen root r and any other terminal $t \in R^r$ is guaranteed.

$$\text{(DCUT)} \quad \min_{\mathbf{X} \in \{0,1\}^{|E|}} \quad \sum_{\{i,j\} \in E} c_e X_e$$

$$\text{s.t.} \quad x(\delta^-(W)) \geq 1 \qquad r \notin W, W \cap R \neq \emptyset \qquad (1)$$

$$x_{ij} + x_{ji} = X_e \qquad e = \{i,j\} \in E \qquad (2)$$

$$x_{ij} \in \{0,1\} \qquad (i,j) \in A \qquad (3)$$

Binary arc variables x_{ij} indicate whether or not an arc (i,j) is chosen as part of the Steiner arborescence rooted at r, and linking constraints (2) ensure that each edge of the Steiner tree can be oriented in exactly one direction. Despite the fact that there is an exponential number of constraints (1), they can be separated efficiently using a min-cut or max-flow implementation [53, 145, 162]. Hence, by the equivalence of optimization and separation (see, e.g., [238]), the LP-relaxation of DCUT can be calculated in polynomial time. The latter is also true for the undirected cut formulation, however there is an important fact which explains why the DCUT formulation gained so much popularity in the exact algorithmic frameworks (as opposed to the undirected one): Most of the facet-defining inequalities discovered in the polyhedral studies of the STP polytope can be obtained by projecting the bidirected cut inequalities (1) (see [54, 108]). This also explains why the aforementioned facet-defining inequalities for the STP have never been tested in practical implementations.

An equivalent model that can be obtained by the min-cut/max-flow theorem (or, by applying Benders decomposition, see Maculan [167]), is the compact *multi-commodity flow* (MCF) formulation [240]. The MCF model exploits the fact that every Steiner tree can be obtained as a union of r-t paths, for all $t \in R^r$. In the MCF model, constraints (1) are replaced by:

$$\sum_{(j,i) \in A} f_{ji}^t - \sum_{(i,j) \in A} f_{ij}^t = \begin{cases} -1, & i = r, \\ 1, & i = t, \\ 0, & \text{otherwise} \end{cases} \qquad i \in V, t \in R^r$$

$$0 \leq f_{ij}^t \leq x_{ij} \qquad (i,j) \in A, t \in R^r$$

Due to its large number of variables and constraints, MCF is not tractable even for medium-sized instances. This is why MCF has been frequently approached by Lagrangian relaxation [10, 21, 228]. By aggregating multi-commodity flows over all terminals $r \in R^r$ we obtain another compact model, known as the *single-commodity flow* (SCF) formulation [167]. However, the worst-case ratio between the LP-bounds of MCF and SCF is known to be $|R| - 1$ [72, 167].

The choice of the root node is known to not affect the quality of the LP-relaxation [109] for DCUT and MCF, respectively. This result also holds even when the models are strengthened by adding the *flow-balance constraints* from [72] (see [187]):

$$x(\delta^+(i)) \geq x(\delta^-(i)) \qquad\qquad i \in V \setminus R \tag{4}$$

Other tree-based formulations for the STP are known, they are however outperformed by the flow and cut formulations (see [182] for details).

Interestingly, formulations other than DCUT are seldom applied in recent algorithmic frameworks. Empirical studies showed that on the majority of real-world instances, LP-relaxation bounds of DCUT are already quite tight, such that little branching is required [95, 102, 145, 153]. Unfortunately, these empirical observations are not supported by the theoretical results, as the integrality gap of DCUT is known to lie between 36/31 (see [38]) and 2. The upper bound of 2 is the same as for the undirected cut formulation (it is obtained by taking a metric closure of G and connecting the terminals via an MST). Whereas this bound is tight for the undirected cut formulation, it is strongly believed that the gap of DCUT is much smaller. However finding a better-than-2 upper bound on its integrality gap is a well-known open problem.

3.2 | Hypergraphic Formulations

A prominent class of MIP formulations which are known to dominate DCUT are the *hypergraphic formulations*. They are based on the observation that the edge set of any Steiner tree can be uniquely partitioned into full Steiner components by splitting the tree at inner terminal nodes (see, e.g., [147]). In the context of the geometric Steiner tree problem, Warme [234] showed that an optimal STP can be equivalently represented as a minimum-cost spanning tree in a specially constructed hypergraph, whose nodes are terminals, and whose hyperedges correspond to full Steiner components. The weight of each hyperedge corresponds to the sum of edge weights of the underlying FSC. Finding an MST in hypergraphs is NP-hard, nevertheless, one can derive "hypergraphic MIP formulations" starting from classic MIP models for spanning trees (see, e.g., [168] for MIP models of trees in graphs). So, for example, it is not surprising that the model based on subtour elimination constraints (SECs) on undirected hypergraphs is equivalent to the model using directed cuts on its directed counterpart (see, e.g., [42, 185]). The latter model, denoted by $(\text{HYP})_{\text{CUT}}$ is illustrated below. In the following, when we refer to equivalent hypergraphic formulations, we will simply denote them by HYP.

By splitting the terminal nodes, a rooted Steiner arborescence can also be uniquely decomposed in a partition of *rooted* full Steiner arborescences (FSAs) and represented as a *spanning arborescence* on a specially designed directed hypergraph. This hypergraph $\mathcal{H} = (R, \mathcal{E})$ is created as follows: for each full Steiner component $F = (V_F, E_F)$, a collection of $|V_F \cap R|$ arborescences, one for each leaf of F being selected as a root, is added to $\mathcal{E}$. The weight of the FSA $K = (V_K, A_K)$ is the sum of its arc weights. By $\text{root}(K)$ we denote the unique root node of a full Steiner arborescence K.

Let $r \in R$ be a given root node of the input graph. The directed-cut hypergraphic formulation asks for a collection of full Steiner arborescences, the union of which corresponds to an SA in the bidirected counterpart of G. In the following, for $W \subset R$, we denote by $\Delta^-(W)$ the set of all FSAs $K = (V_K, A_K)$ such that W does not contain root(K), but it contains at least one node from V_K, i.e., root(K) $\notin W$ and $W \cap V_K \neq \emptyset$. The hypergraphic cut formulation is given as follows:

$$
\begin{aligned}
(\mathrm{HYP})_{\mathrm{CUT}} \qquad & \min_{\mathbf{X}\in\{0,1\}^{|E|}} \sum_{\{i,j\}\in E} c_e X_e && \\
\text{s.t.} \qquad & z(\Delta^-(W)) \geq 1 && W \subset R, r \notin W \quad (5)\\
& \sum_{K\in\mathcal{E}:(i,j)\in A_K} z_K + \sum_{K\in\mathcal{E}:(j,i)\in A_K} z_K = X_{\{i,j\}} && e = \{i,j\} \in E \quad (6)\\
& z_K \in \{0,1\} && K \in \mathcal{E} \quad (7)
\end{aligned}
$$

In this model, the binary variable z_K associated to an FSA $K \in \mathcal{E}$ is set to one if and only if K belongs to an optimal Steiner arborescence. Connectivity in the hypergraph between the root r and the remaining terminals is ensured by constraints (5). Finally, equations (6) provide a projection of variables z onto the space of X variables.

There are exponentially many subsets of R, and potentially superexponentially many FSAs contained in the bidirected counterpart of G. Solving even the LP-relaxation of these hypergraphic formulations is an NP-hard problem [110]. However, Warme [234] showed that if the size of the maximum FSC is bounded by a constant, then the LP-relaxation of HYP can be obtained in polynomial time. This fact has been extensively exploited in a series of articles that provide LP-based approximation algorithms for the STP [24, 26, 38, 110, 147] (cf. Section 9).

Each LP-solution of HYP is also feasible for DCUT, which follows from the transformation $x_{ij} = \sum_{K\in\mathcal{E}:(i,j)\in A_K} x_K$, however the opposite is not necessarily true. This means that HYP is strictly stronger than DCUT, and an illustrating example is given in Figure 2a (cf. [185]). The optimal LP-solution of DCUT is depicted in Figure 2b (its value is 5.5), and is infeasible for HYP (whose optimal LP-solution yields 6). Moreover, Polzin and Vahdati Daneshmand [185] showed that HYP is even stronger than DCUT enhanced by the flow-balance constraints (4). This is illustrated in the same example in Figure 2b, in which adding the flow-balance constraints does not change the optimal LP-solution of DCUT.

Warme [234] showed that not all possible FSCs have to be considered in the underlying hypergraph - it is sufficient to find their subset that guarantees to contain an optimal solution. On the other hand, the choice of the hyperedges in $\mathcal{H}$ influences the quality of the LP-relaxation of HYP (see [185] for an example).

Könemann et al. [147] proposed an alternative model, called the *partition-based formulation* on the undirected version of the hypergraph $\mathcal{H}$. Unsurprisingly, solving the LP-relaxation of this model is NP-hard as well. In a follow-up article by Chakrabarty et al. [42], the partition-based formulation has been shown to be equivalent to HYP.

The lower bound on the integrality gap of HYP (and its equivalent formulations based on SECs or partitions) is 8/7 (see, e.g., [147]). Byrka et al. [38] showed that the integrality gap is upper bounded by 1.55. Hence, the hypergraphic formulations were the first (and currently the only) MIP models for which it was possible to break a long standing integrality gap barrier of 2. On certain families of graphs, HYP and DCUT are equally strong. So, for example, Chakrabarty et al. [42] showed that the two formulations are equivalent on quasi-biparite graphs (i.e., graphs where S forms an independent set). Later, Feldmann et al. [90] improved this result by showing that DCUT and HYP are equivalent on graphs that do not contain a *Steiner claw*, i.e., a potential Steiner node which is adjacent to three other nodes from S. However, they also showed that in general, for a given STP instance, it is NP-hard to decide whether the LP-relaxation of HYP has the same value as the LP-relaxation of DCUT.

Approximation algorithms based on hypergraphic LP-relaxations have been implemented and empirically tested by Beyer et al. [24, 26], see also Section 9. However, to our knowledge, there are no empirical studies on exact methods for the Steiner tree problem in graphs derived from HYP. On the contrary, GeoSteiner, the state-of-the art

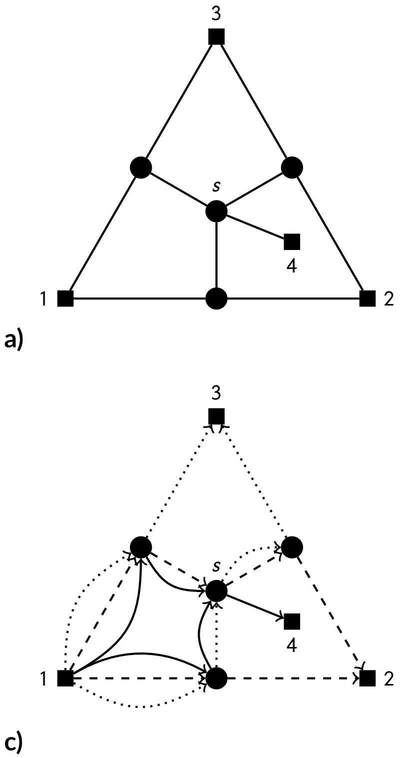

FIGURE 2 a) Input graph with all edge weights equal to one and with the set of terminals $R = \{1, 2, 3, 4\}$ (depicted as squares); b) LP-solution of DCUT/MCF. Dashed and solid edges correspond to $X_e = 1/2$, and $X_e = 1$, respectively; c) Multi-commodity flows for the LP-solution of DCUT with $r = 1$. Solid lines route the flow from r to 4, whereas dashed and dotted lines route the flow from r to 2 and 3, respectively; d) Multi-commodity flows for the LP-solution of DCUT with $r = 4$. Solid lines route the flow from r to 3, whereas dashed and dotted lines route the flow from r to 2 and 1, respectively.

algorithm for the Euclidean Steiner Tree problem in the plane, is based on HYP (see [139]). The algorithm consists of two phases: in the first phase, the hypergraph $\mathcal{H}$ is built from a subset of FSCs (while ensuring that the optimal solution is contained in this set). In the second phase, an MST on $\mathcal{H}$ is obtained. We therefore believe that implementing a competitive exact method for the STP based on HYP is an interesting open direction for future research.

3.3 | Common Flow Formulations

Several authors observed that certain graph minors are responsible for integrality gaps of MCF/DCUT formulations [158, 182]. Such structures lead to LP-solutions in which, informally speaking, two or more commodity flows join after separating. Let us consider an example given in Figure 2a, and let $1 \in R$ be a chosen root node r in the MCF formulation. Figure 2b shows the optimal MCF solution in the space of X variables. Commodity flows are depicted in Figure 2c. We notice that the three arcs which are used to simultaneously route flows for commodity 2 and 3 create a disconnected subgraph. However, this property does not hold for binary solutions. More precisely, given a subset of commodities

$B \subseteq R^r$, the *common flow* routed from r to B always follows a single path in the tree. This means that in any feasible solution the amount of the common flow entering a node must be greater or equal the amount of the common flow leaving the same node. The latter property was used by Polzin and Vahdati Daneshmand [182] to introduce a stronger flow-based formulation that we will refer to as CF2. In their model, for each pair of terminals $k, l \in R^r$, additional flow variables h_{ij}^{kl} were introduced to capture the amount of their common flow sent through an arc $(i, j) \in A$. That way, the authors managed to forbid rejoining of the flow for tuples of commodities.

Several years later, Polzin and Vahdati Daneshmand [187] proposed a slightly different and more general formulation. Let us for a moment assume that the root r is fixed and let $\mathcal{B}$ be a collection of subsets of R^r such that $\{t\} \in \mathcal{B}$, for all $t \in R^r$, and $R^r \in \mathcal{B}$ ($\mathcal{B}$ may contain many other subsets of R^r in general). We then introduce for each subset $B \in \mathcal{B}$ and each arc $(i, j) \in A$, the variables $0 \le g_{ij}^B \le 1$ to capture the amount of flow going from r to any terminal $t \in B$ through arc (i, j), i.e., we have $g_{ij}^B \ge g_{ij}^t$ for all $t \in B$, $(i, j) \in A$. For simplicity we write g^t instead of $g^{\{t\}}$ below.

The *rooted common flow* formulation, denoted by CF_r, reads as follows:

$$
\begin{aligned}
(\text{CF}_r) \quad \min_{\mathbf{X} \in \{0,1\}^{|E|}} \quad & \sum_{e \in E} c_e X_e \\
\text{s.t.} \quad & g^t(\delta^-(r)) = g^t(\delta^+(r)) - 1 && t \in R^r \quad (8)\\
& g^B(\delta^-(i)) \le g^B(\delta^+(i)) && B \in \mathcal{B}, i \in V \setminus (B \cup \{r\}) \quad (9)\\
& 0 \le g_{ij}^B \le g_{ij}^C && B \subset C, B, C \in \mathcal{B}, (i, j) \in A \quad (10)\\
& g_{ij}^{R^r} + g_{ji}^{R^r} = X_e && e = \{i, j\} \in E \quad (11)
\end{aligned}
$$

Constraints (8) ensure that one unit of flow is sent out from the root to each commodity $t \in R^r$. Inequalities (10) ensure that $g_{ij}^B \ge \max_{t \in B} g_{ij}^t$, for all $B \in \mathcal{B}$, $(i, j) \in A$. For $B = \{t\}, t \in R^r$, constraints (9) guarantee that one unit of flow reaches every terminal $t \in R^t$. Moreover, for B of size larger than one, these constraints guarantee the continuity of the common flow, i.e., they impose

$$
\sum_{ji \in \delta^-(i)} \max_{t \in B} g_{ji}^t \le \sum_{ij \in \delta^+(i)} \max_{t \in B} g_{ij}^t \qquad B \in \mathcal{B}, i \in V \setminus (B \cup \{r\}) \tag{12}
$$

Finally, equations (11) link the flow and edge variables.

If $\mathcal{B} := \mathcal{B}_2 = \{B \subseteq R^r \ : \ |B| \in \{1, 2, |R^r|\}\}$ the model CF_r is equivalent to the aforementioned CF2 formulation. By considering tuples $\{k, l\}$ of distinct commodities from R^r, the LP-relaxation of the resulting model already depends on the choice of the root node. An example is given in Figures 2c and 2d. The LP-solution of DCUT is infeasible for the model CF_r, when $r = 1$ and all pairs of commodities from $\{2, 3, 4\}$ are added to $\mathcal{B}$. However, by changing the root to $r = 4$, the solution of DCUT remains valid (for the respective collection $\mathcal{B}_2$), i.e., the choice of the root makes a difference for the quality of the LP-relaxation. Therefore, Polzin and Vahdati Daneshmand [187] used the "intersection of rooted arborescences" technique (see also [115, 118]) to derive a stronger formulation by intersecting $|R|$ rooted Steiner arborescences, one for each root $r \in R$. That way, and by considering different sizes of subsets of R^r, a hierarchy of common flow formulations was provided [187]. The strongest among these formulations, denoted by CF, is given as follows:

$$\text{(CF)} \qquad \min_{\mathbf{X}\in\{0,1\}^{|E|}} \quad \sum_{e\in E} c_e X_e$$

$$\begin{aligned}
\text{s.t.} \qquad & f^{r,t}(\delta^-(r)) = f^{r,t}(\delta^+(r)) - 1 && r \in R, t \in R^r && (13)\\
& f^{r,t}_{ij} \le f^{r,s}_{ij} + f^{s,t}_{ij} && \{r,s,t\} \subseteq R, (i,j) \in A && (14)\\
& f^{r,s}_{ji} \le f^{r,t}_{ji} + f^{s,t}_{ij} && \{r,s,t\} \subseteq R, (i,j) \in A && (15)\\
& f^{r,B}(\delta^-(i)) \le f^{r,B}(\delta^+(i)) && r \in R, B \subseteq R^r, i \in V \setminus (B \cup \{r\}) && (16)\\
& 0 \le f^{r,B} \le f^{r,C} && r \in R, B \subset C \subseteq R^r && (17)\\
& f^{r,R^r}_{ij} + f^{r,R^r}_{ji} \le X_e && r \in R, e = \{i,j\} \in E && (18)\\
& f^{r,r} = 0 && r \in R && (19)
\end{aligned}$$

In this formulation, an additional index r is added to the definition of flow variables, to indicate the source of the flow. As before, equations (13), (19) ensure that one unit of flow is sent from r to $t \in R^r$, and together with inequalities (16)-(17) the continuity of the common flow is ensured. Constraints (14)-(15), (19) indicate that by changing the root node from r to s, the flow of commodity $t \in R \setminus \{s, r\}$ will change orientation along some edges of the tree.

From the empirical perspective, the practical applicability of the CF model in the given form is quite prohibitive. For each $r \in R$, all possible commodity combinations from R^r are considered for imposing the common flow constraints (16)-(17). Thus, the model exhibits an exponential number of constraints and variables. Even by restricting the formulation to subsets of R whose size is upper bounded by a constant, the model has little chance of being competitive with the state-of-the-art. Nevertheless, there are alternative ways in which the strength of CF can be exploited in computational frameworks. Polzin and Vahdati Daneshmand [180], for example, described a restricted version of the common flow formulation that can be solved by branch-cut-and-price. Besides restricting the size of $\mathcal{B}$, advanced implementations and possible hybridizations of decomposition techniques (Lagrangian, Benders, or Dantzig-Wolfe decomposition) might be promising directions for future research.

3.4 | Path-Based Extended Formulations

Very recently, in their *Networks* article, Filipecki and Van Vyve [92] proposed new MCF-based formulations with aggregated-path variables. The authors generalized the MCF formulation by considering flows over paths, instead of just flows over edges. For a given parameter $1 \le \lambda \le |V| - 1$, the formulation denoted by MCF-λ, considers multi-commodity flows from r to $t \in R^r$ by looking at the *simple* paths connecting r to t whose length is at most λ. To ensure feasibility of the solution, the flow preservation constraints need to be imposed at paths of length $\lambda - 1$, instead of being imposed at nodes. Finally, to make the model independent on the choice of the root node, similarly as for the common flow formulation, an additional root index is introduced. In the following we simplify the original notation and only introduce the formulation MCF-2, whereas the more general model can be found in [92].

Let $\mathcal{A}^{rt}$ denote the set of directed paths of length one or two that can be used to connect r with $t \in R^r$. Each path can start at r, or end at t, though r and t are not necessarily included in the path. More precisely

$$\begin{aligned}\mathcal{A}^{rt} = &\{(l,i,j) : (l,i),(i,j) \in A,\ l,i,j \text{ are distinct}, r \notin \{i,j\}, t \notin \{l,i\}\} \cup \\ &\cup \{(r,t) : (r,t) \in A\} \qquad r \in R, t \in R^r,\end{aligned}$$

We use the notation $p \in \mathcal{A}^{rt}$ to indicate either a triple (l,i,j) or arc (r,t). Let $\mathcal{A}^r = \bigcup_{t \in R^r} \mathcal{A}^{rt}$ be the union of all paths that can be used to send flow from r to any of commodities from R^r. We define the following set of additional variables:

- w_p^{rt} indicates the amount of flow sent from r to commodity t, over the path p, $r \in R$, $t \in R^r$, $p \in \mathcal{A}^{rt}$;
- u_p^r, $(i,j) \in A$, $r \in R$ indicates the upper bound of the flow sent from $r \in R$ to any commodity $t \in R^r$ over the path $p \in \mathcal{A}^{rt}$, i.e., $u_p^r \geq \max_{t \in R^r} w_p^{rt}$.

$$\text{(MCF-2)} \quad \min_{\mathbf{X} \in \{0,1\}^{|E|}} \sum_{e \in E} c_e X_e$$

$$\text{s.t.} \quad \sum_{(l,i,j) \in \mathcal{A}^{rt}} w_{lij}^{rt} - \sum_{(i,j,l) \in \mathcal{A}^{rt}} w_{ijl}^{rt} = 0 \qquad (i,j) \in A, r \in R, t \in R^r \quad (20)$$

$$\sum_{p \in \mathcal{A}^{rt} : r \in p} w_p^{rt} = 1 \qquad r \in R, t \in R^r \quad (21)$$

$$0 \leq w_p^{rt} \leq u_p^r \qquad r \in R, t \in R^r, p \in \mathcal{A}^{rt} \quad (22)$$

$$\sum_{p \in \mathcal{A}^r : (i,j) \in p} u_p^r + \sum_{p \in \mathcal{A}^r : (j,i) \in p} u_p^r \leq X_e \qquad e = \{i,j\} \in E, r \in R \quad (23)$$

The flow preservation constraints (20)-(21) guarantee that one unit of flow is sent out from the root r to each $t \in R^r$. Constraints (22) guarantee that u_p^r provides enough capacity to route the flow for all commodities, and constraints (23) ensure that for each root node, the flow can be routed in exactly one direction. With non-negative costs on the edges, this implies that no cycles will be contained in an optimal solution.

In the generalization of this model given by Filipecki and Van Vyve [92], paths of length up to λ (where $\lambda \geq 2$ is a given input parameter) are used to route the flow and flow-preservation constraints are imposed for all subpaths of length $\lambda - 1$. The sets $\mathcal{A}^{rt}$ are comprised of paths of length λ along with all paths of length 1 to $\lambda - 1$ that start at r. For the latter paths, no flow preservation constraints are imposed, they are considered as "shortcuts" from r to t, but thanks to capacity and linking constraints (22)-(23), it is ensured that the corresponding edges will be set to one in the underlying binary solution. A hierarchy of models was derived, showing that by increasing the values of λ, the strength of the underlying MIP formulations can be strictly improved.

On some of the instances that are typically used to demonstrate the worst-case lower bounds of DCUT and HYP, the authors showed that the LP-relaxation gaps can be improved. By using the new compact model, several difficult STP instances could even be solved without branching. However, the authors provided examples of STP instances on which, for arbitrary value of λ, MCF-λ does not dominate HYP, which means that the two formulations are not comparable.

Finally, Filipecki and Van Vyve [92] did not compare their new hierarchy of formulations with the hierarchy of common flow models studied in [187], see Section 3.3. This comparison remains to be done in order to better under-stand the relationship between these flow-based models, as well as their strengths and weaknesses. With respect to the practical viability of the MCF-λ models, the same arguments as for CF apply - without advanced decomposition techniques, MCF-λ remains computationally intractable, even for $\lambda = 2$.

4 | REDUCTION TESTS

Reduction tests are algorithms that are used to transform the input graph into a smaller and simplified graph with fewer nodes and/or edges. These tests are arguably one of the most effective features for both exact and heuristic methods in solving the STP [95, 102, 143, 145, 178]. They are typically applied in a pre-processing phase and a mapping is created so that solutions obtained on the reduced graph can be uniquely mapped back into the original problem instance. In Polzin and Vahdati Daneshmand [183], the authors went beyond using the reduction tests for pre-processing purposes only, and implemented a tailor-made branch-and-bound approach in which reduction tests are applied at every node of the branching tree.

The PhD thesis by Duin [72] was the first to introduce generalizations of several earlier and relatively simple reduction tests. According to Duin [72], reduction tests take the form of *exclusions* or *inclusions*. Exclusions remove edges or nodes from the input graph. In case a node is removed, some tests may also require the addition of new edges. Inclusions modify the input graph so that a given edge or node becomes part of any feasible solution.

Moreover, according to the underlying criteria used for reduction tests, a distinction is made between *Alternative-Based* and *Bound-Based* tests [183]. The former category consists of tests which decide upon inclusion/exclusion based on the existence of alternative solutions. For example, the exclusion of edge e is valid if for any solution T_S that contains e, there exists a solution $T_{S'}$ that does not contain e and $c(T_S) \geq c(T_{S'})$. Similar arguments apply for the case of inclusion. The latter category, bound-based tests, exploits lower and upper bounds and follows a reasoning similar to the reduced cost-based variable fixing in integer programming. For example, to test for the exclusion of a potential Steiner node i, a lower bound is computed subject to the constraint that i is part of an optimal solution. If this lower bound exceeds a known upper bound (i.e., the objective value of the best incumbent solution), node i and all its incident edges can be removed [228]. Following these concepts, subsequent contributions to reduction tests are made for deterministic [102, 180, 228], stochastic [152], node-weighted variants of the STP [153, 199], or special instances stemming from VLSI design [225]. A recent comprehensive collection of reduction tests for node-weighted variants of the STP is given in [199].

More sophisticated contributions to reduction tests are due to Polzin and Vahdati Daneshmand [180, 228]. While most of the previous reduction tests rely on the inspection of simple patterns, their *extended reduction tests* introduced in [184] examine more complex patterns by exploiting the tools of conflict analysis. Given an initial assumption (e.g., an edge is contained in an optimal Steiner tree), the aim is to search for a contradiction using an enumeration procedure. Similar ideas were seen in Winter [236], Uchoa et al. [225] and Duin [73] with the major difference that the new extended tests combine both alternative-based and bound-based reductions. That way, additional reductions could be achieved. However, the authors pointed out that a careful implementation of these tests is needed, as otherwise the time invested for performing additional reductions may outweigh the achieved speedups. As a consequence, the authors exploited heuristic rules in order to truncate the search in case the effort is unlikely to pay off.

Graph connectivity and graph partitioning can also be utilized for performing reduction tests. In a trivial inclusion test, for example, one searches for a potential Steiner node i such that at least two terminals are separated by removing i from G (i.e, i is an articulation point of G). Although such nodes are rarely present in an unpreprocessed instance, node separators of small size tend to appear after a successive application of reduction tests. Polzin and Vahdati

Daneshmand [186] introduced new reduction tests in which more general cases involving *terminal node separators* $N \subset R$ such that $|N| > 1$ are addressed. By removing an articulation point from G, the graph can be decomposed into several independent subinstances, each defined on a subgraph of G. In the non-trivial case in which a separator contains more than one node ($|N| > 1$), multiple cases need to be distinguished, which is why the proposed partitioning approach is only suitable for small $|N|$. Instead of using an exact max-flow-based approach to find terminal separators, the authors employed a fast heuristic that yields a set of separators of relatively small size in time $O(|R||E|)$. If one of the resulting subinstances is too large, the computational effort of solving all cases may exceed the effort of solving the original instance. The authors applied inclusion and exclusion tests only on small subinstances instead: if an edge is present in optimal subtrees for all possible cases, it must be included, and conversely, if it is part of none of these solutions, it can be removed. In the same article, an alternative approach is suggested that does not depend on solving subinstances to optimality.

In contrast to the STP, relatively little work is devoted to studying reduction tests for node-weighted (or directed) problem variants [63, 153, 198, 199]. For a detailed overview of reduction tests for node-weighted versions of the STP until 2006, we refer to the survey by Costa et al. [61]. da Cunha et al. [63] proposed reduction tests for the PCSTP that are efficiently employed within a Lagrangian relaxation framework. Leitner et al. [153] developed alternative-based and bound-based tests for the (prize-collecting) SAP with asymmetric arc weights. Motivated by the work of [180, 228], a recent *Networks* contribution by Rehfeldt et al. [199] introduced many new reduction tests for the PCSTP and MWCS on undirected graphs. In a follow-up article, Rehfeldt and Koch [198] introduced additional NP-hard reduction tests for the MWCS and exploited them in an exact solution framework.

5 | LAGRANGIAN RELAXATIONS

The Lagrangian relaxation technique has been used in the 1980's as a useful method to derive primal and dual bounds for the STP [21, 22, 167]. Typically, easy to solve relaxed subproblem(s) are obtained after dropping a set of "complicating" constraints from the formulation at hand and transferring them to the objective function, multiplied by the assigned Lagrangian multipliers. In the past three decades however, Lagrangian relaxations were less popular for solving the STP [10, 164], but they gained more attention for approaching some of the STP variants.

In 2003, starting from the compact MCF formulation for the STP, Bahiense et al. [10] proposed to relax the flow-conservation constraints in Lagrangian fashion. The relaxed subproblems were then solved by inspection (see also [21]). Instead of using a standard subgradient procedure to solve the Lagrangian dual problem, the authors proposed a *volume algorithm*, which is an extension of the subgradient method that simultaneously calculates both dual and primal bounds. The results were compared with a standard subgradient procedure and with the B&C framework by Koch and Martin [145]. The authors noted that their approach performed poorly in the presence of many terminals, but performed reasonably well on instances with few terminal nodes. The same year, in their computational studies, Vahdati Daneshmand [228] and Polzin [180] confirmed that the scalability of Lagrangian approaches with respect to the number of terminals is an issue and refrained from using Lagrangian relaxations in their exact solution framework [183]. Soon after, in 2005, Lucena [164] proposed an alternative *non delayed relax-and-cut* Lagrangian approach for the STP. His method is a modification of the subgradient method that can handle MIP formulations with an exponential number of constraints. It can be seen as a traditional Lagrangian approach in which exponentially many constraints are dualized. To make it computationally tractable, only the polynomial number of Lagrangian multipliers are updated. Lucena tested his approach on the aforementioned GSEC formulation and on a spanning tree formulation (a stronger version of a model proposed earlier by Beasley [22]). His computational study revealed that non delayed relax-and-cut outperforms the *delayed relax-and-cut* approach (in which the exponential-size family of constraints is separated only after the Lagrangian dual problems are solved).

Lagrangian relaxations gained more popularity in solving the STP variants. Rosenwein and Wong [205] used Lagrangian relaxations for an STP variant with an additional knapsack constraint. Two Lagrangian approaches were derived starting from the MCF formulation. In the first one, the knapsack constraint is relaxed, whereas in the second one arc variables are doubled so that the Lagrangian subproblem is decomposed into a STP and the knapsack problem. The authors proved that the latter approach provides stronger dual bounds, however the computational study revealed that these bounds did not outweigh the computational burden of solving two NP-hard problems.

Engevall et al. [82] published a Lagrangian approach for solving the NWSTP in *Networks* in 1997. The authors used Beasley's MST-based formulation for the STP [22], adapted it as suggested by Duin and Volgenant [74], and employed a subgradient algorithm enhanced by the separation of subtour elimination constraints. For the same problem, Cordone and Trubian [60] extended the approach of Engevall et al. [82], modified the subgradient method and integrated it into a branch-and-bound procedure.

For the quota-constrained PCSTP problem, Haouari and Siala [124] combined a volume algorithm with a genetic algorithm. In a subsequent work, Haouari et al. [123] considered a generalized version of the PCSTP, formulated it as a spanning tree problem with side constraints, and compared the strength of four Lagrangian relaxations. They also provided a computational comparison of nine different implementations of the subgradient method with the volume algorithm and two variants of the variable target value method. For the PCSTP, da Cunha et al. [63] proposed a non-delayed relax-and-cut approach derived from the MIP formulation based on GSECs. Recently Álvarez-Miranda and Sinnl [5] applied a similar methodology for the MWCS and its budget-constrained version.

For *two-stage stochastic STP*, Leitner et al. [152] combined two types of decomposition techniques which can be seen as dual to each other, namely Benders decomposition and Lagrangian relaxation. The authors demonstrated that using these two powerful decompositions in a combined framework creates synergetic effects and allows one to attack the stochastic STP from multiple angles so as to exploit different types of problem-structures.

6 | DUAL-ASCENT METHODS

Dual-ascent algorithms are effective combinatorial procedures for computing heuristic solutions to the LP dual of a given MIP formulation. They can also be seen as a combinatorial alternative to Lagrangian dual approaches (see, e.g., the PhD thesis of Raghavan [194]). The history of dual-ascent methods for the STP begins in 1984 with the method of Wong [240] published in *Mathematical Programming* which is by far the most prominent and the most frequently cited dual-ascent approach to the STP. Wong's algorithm works on the dualization of the MCF problem formulation for the SAP. Dual ascent was also formulated in terms of the equally strong DCUT formulation [68, 72]. Duin [72] proposed an efficient implementation whose running time is $O(|A| \min\{|V||R|, |A|\})$. Raghavan [194] also studied and computationally compared four efficient implementations of Wong's dual-ascent algorithm for solving the SAP. Even though the dual-ascent algorithm applied to the strong MCF/DCUT formulation generally performs well in practice, its lower bounds depend on the choice of the root node (see [181] for an example). Moreover, the solution of Wong's dual-ascent algorithm can be arbitrarily far from the optimum (Vahdati Daneshmand [228] shows this result for undirected graphs and Candia-Véjar and Bravo-Azlán [39] prove this result for directed graphs).

Dual-ascent algorithms typically construct a dual feasible solution in an iterative fashion – in each iteration some dual variables are increased while dual feasibility is preserved. A similar idea is employed in many *primal-dual* approximation algorithms, in which primal solutions are additionally constructed on the fly, and the approximation guarantee is derived from the gap between the primal and dual solution. Goemans and Williamson [111] proposed a primal-dual heuristic for the (PC)STP with a performance guarantee of $2 - 2/|R|$. Lower bounds in [111] are de facto obtained by a dual-ascent procedure applied to the undirected cut formulation. One of implementations proposed by Raghavan [194] simultaneously increases the values of dual variables associated to cut-sets, and when applied to undirected graphs, it resembles the primal-dual heuristic of Goemans and Williamson [111]. On directed graphs, however, Raghavan's implementation effectively changes the rate at which the dual variables are increased on different arcs, as an arc can be in multiple cut-sets. Several years later, Polzin and Vahdati Daneshmand [181] independently provided an implementation very similar to the one given by Raghavan [194] proving that it guarantees the worst case approximation ratio of $2 - 2/|R|$, while providing empirically high quality bounds.

Poggi de Aragão et al. [178] used heuristics and reduced cost fixing to improve the performance of dual ascent. The authors also considered two techniques to make an efficient use of dual ascent within an exact solution framework: 1) *branch-and-ascent*, i.e., a branch-and-bound in which dual ascent replaces the usual LP solver; and 2) initialization of a branch-and-cut scheme (derived on top of the DCUT formulation) using cuts derived from the basis of the dual ascent solution.

Raghavan [194] extended the dual-ascent method to the network design problem with connectivity requirements (the STP/SAP is a special case of this more general problem). He also emphasized the choice of cut-sets (also known as root components) and the importance of determining the order in which dual variables associated to these root components are updated. These arguments have been reinforced by Pajor et al. [177] who proposed a new and efficient implementation based on a specific scoring function which can be evaluated in a lazy fashion.

The SAP/STP finds important applications in the multicast routing in telecommunication networks (see, e.g., [173, 174]). In such a context, complete information about the topology and the state the network is not available at every single node, and hence *distributed* algorithms for Steiner trees need to be considered. Drummond et al. [69] gave a distributed implementation of Wong's algorithm and demonstrated its advantages over the existing construction heuristics in the distributed environment.

When it comes to the PCSTP, a first implementation of Goemans and Williamson's primal-dual method was computationally tested by Johnson et al. [138]. Fifteen years later, at the DIMACS Challenge, Hegde et al. [126] presented a new highly efficient implementation of the same method whose running time is $O(d|E|\log|V|)$ (where d bits of precision are used to specify edge weights and node prizes, c and p, respectively). The impressive computing times were reported, along with very large duality gaps. The latter can be explained by the the fact that dual ascent from Goemans and Williamson [106] is built on top of the undirected cut formulation for the PCSTP, which is known to be much weaker than its directed counterpart (see, e.g., Ljubić et al. [162]). The first dual-ascent approach derived from the DCUT-like formulation for the PCSTP is the algorithm of Leitner et al. [153]. Unlike the primal-dual method of Goemans and Williamson [106], the algorithm of Leitner et al. [153] is also valid for the asymmetric PCSTP, and can also be used for the MWCS and the NWSTP. In a recent *Networks* article, Rehfeldt et al. [199] transformed the PCSTP and the MWCS into the SAP and used dual ascent as a tool for reducing the input size. For two STP generalizations, namely the stochastic STP and the Steiner tree-star problem, dual-ascent techniques have been successfully applied by Leitner et al. [152] and Bardossy and Raghavan [16], Leitner et al. [154], respectively.

7 | HEURISTICS

Comprehensive surveys on heuristics for the STP up to 2000 can be found in [27, 71, 228, 231]. In the following paragraphs we focus on more recent contributions.

Construction heuristics for STP typically build a feasible solution by mimicking a minimum spanning tree algorithm. One of the widely used MST-based construction heuristics is a generalization of Prim's MST algorithm, also known as the shortest path heuristic of Takahashi and Matsuyama [218] (TM). Starting with a single terminal node, the TM algorithm generates a feasible solution by iteratively adding the shortest path from the closest terminal to the current tree. Efficient implementations of TM and other MST-based construction heurisitics were given by Poggi de Aragão et al. [179]. Even though the proposed implementations did not improve the worst-case running time, the authors managed to significantly improve their empirical performance, pointing out that in terms of the solution quality, TM outperformed alternative MST-based heuristics. The quality of TM solutions heavily depends on the starting terminal, and hence, a multi-start approach may yield a better solution. An efficient two-phase implementation which allows to run the TM heuristic in a multi-start fashion without sacrificing the worst-case time complexity was given by Vahdati Daneshmand [228].

In a subsequent work, Ribeiro et al. [203] implemented a hybrid GRASP multi-start heuristic in which solutions are generated using the MST-based heuristics from [179]. They also employed neighborhood search heuristics, and reduction tests from [74, 225] as a preprocessing, and path-relinking [201] as a postprocessing procedure.

When it comes to local search heuristics, four basic move operators, namely *node-insertion*, *node-elimination*, *keypath-exchange* and *keynode-elimination* are frequently used in the literature [77, 203]. A keynode is a Steiner node whose degree is at least three in a given feasible solution. A keypath is a subpath of a feasible solution whose end nodes are terminals or keynodes, and such that all intermediate nodes are Steiner nodes with degree two [229]. Evaluation procedures and specific data structures that enable a fast enumeration of these neighborhoods were introduced by Uchoa and Werneck [227]. The authors showed that the enumeration of each neighborhood can be performed in $O(|E|\log|V|)$. In the same study, the neighborhood search procedures (all but the node-elimination) were embedded in a variable neighborhood descent (VND). The authors stated that the keynode-elimination dominates the node-elimination neighborhood. However in a later study by Pajor et al. [177], the authors explained that this is not the case in general.

Several heuristic methods based on reduction tests were described in Vahdati Daneshmand [228] and Polzin and Vahdati Daneshmand [183]. Iterative applications of reduction tests usually reach a point at which no further reductions are possible. In that case, a few nodes or edges may be excluded in a heuristic fashion, thus enabling further bound-based reductions.

Metaheuristics have been developed for the STP as well: genetic algorithms [84, 133, 140, 189], tabu search [19, 104, 202], or more recently, hybrid ant colony optimization [34], or particle swarm optimization [193]. Most of these algorithms have been empirically evaluated on a set of instances from the OR-library [20]. However, already in 2001, all instances from the the OR-library were solved to optimality by the method of Pozin and Vahdati Daneshmand [183] in less than a second of CPU time on a Pentium II machine. Hence, competitiveness of the above-mentioned metaheuristics with respect to state-of-the-art exact methods remains questionable. Even more so, after taking into account that none of these methods participated at the DIMACS Challenge in 2014. In the remainder of this section, we review the heuristic developments initiated by this challenge.

Gamrath et al. [102] exploited a variable neighborhood descent procedure as a primal heuristic in their exact method. The authors integrated some of the most successful STP heuristics from the literature (TM, the dual ascent heuristic, and the improvements from [179, 183, 228]) into their `SCIPJack` framework. In addition, they proposed a new recombination heuristic that creates a subgraph from a collection of promising solutions, applies reduction techniques, modifies the edge weights and solves the resulting instance heuristically.

A battery of heuristics [151, 179, 183, 228] was also used in the computational framework of Fischetti et al. [95], which was the winner of the DIMACS 2014 Challenge in most of the categories, including the best found heuristic solutions (see [65]). The framework was particularly effective for the very challenging family of PUC instances which were originally developed for the STP in [206], and were later adapted for the PCSTP [159, 162]. For these and other instances with uniform (or almost uniform) edge weights, a *local branching* heuristic [96] was applied to the node-based counterpart of the DCUT formulation. To create a starting feasible solution for the local branching, the authors employed a *Benders-like set covering heuristic*. The latter interprets a partially generated set of cut-set inequalities as a set covering problem, in which the terminal nodes are *elements*, and the adjacent potential Steiner nodes are interpreted as *sets*. The set covering instance was solved heuristically using the method from Caprara et al. [40]. If the resulting solution was infeasible, the chosen Steiner nodes (i.e., sets of the set-covering instance) were fixed to one, and the resulting significantly reduced (uniform) STP instance was solved using the branch-and-cut scheme.

Fu and Hao [99] introduced a *swap-vertex* move operator which can be applied within a local search heuristic not only for STP, but also for its node-weighted variants (PCSTP/MWCS). The operator exchanges a node from a given feasible solution with another node out of it, and then repairs the solution by constructing a spanning tree over the selected nodes. Fu and Hao [99] introduced a series of dynamic data structures to enable an efficient evaluation of the swap-vertex moves. Their heuristic was particularly effective for the aforementioned difficult PUC dataset. In a subsequent article by Fu and Hao [98], the authors embedded the local search procedures into a knowledge-guided framework, thus improving upon their previous computational results published in [99].

Pajor et al. [177] introduced a multi-start heuristic which employs the aforementioned four neighborhoods and improves upon several aspects originally studied by Ribeiro et al. [203]. As an additional intensification strategy, the authors proposed a cascaded combination procedure in which newly constructed solutions are iteratively combined with the members of a solution pool, i.e., a subset of all previously constructed solutions. The combination is performed heuristically in a manner similar to one of the path-relinking strategies proposed in [203].

Finally, there is a parallel stream of research whose major focus is on efficiently utilizing the constantly increasing computational power in developing *parallel* or *distributed* algorithms for solving the STP. Bezenšek and Robič [27] reviewed parallel and distributed algorithms for solving the STP up to 2014.

8 | EXACT METHODS

An overview of exact algorithms up to 1990 can be found in Hwang et al. [134]. Common ingredients for the majority of the exact methods developed in the last three decades are: preprocessing tools based on reduction tests (cf. Section 4), the use of Lagrangian or dual ascent relaxations for calculating strong lower bounds (cf. Section 5 and 6, respectively), sophisticated local-search-based heuristics for calculating primal bounds (cf. Section 7) and finally branch-and-cut methods based on the MIP formulations described in Section 3.

In their 1998 *Networks* article, Lucena and Beasley [165] designed a branch-and-cut scheme based on Beasley's MST formulation [22]. They augmented this undirected formulation with additional valid inequalities, including subtour elimination constraints and Steiner connectivity cuts. In addition, preprocessing in the form of reduction tests [75] was applied. The authors gave a side-by-side comparison with Beasley's original branch-and-bound based on the Lagrangian relaxation [22], a branch-and-cut of Chopra et al. [53] based on the DCUT formulation, and an exact algorithm by Duin [72] based on reduction tests and dual ascent. Their results demonstrated that the approach by Duin outperformed the others by approximately three orders of magnitude. The authors attributed this to Duin's more advanced reduction tests, and an efficient overall implementation. Most notably, Duin's approach did not rely on branch-and-cut, but focused solely on reduction tests and dual ascent bounds.

The same year, another exact method for the STP was published in *Networks*: Koch and Martin [145] proposed a branch-and-cut scheme built on top of the original methodology of Chopra et al. [53]. The most important enhancements were the inclusions of reduction tests [75], *flow-balance inequalities*, and the development of an efficient separation procedure that combined *back cuts* (introduced by Chopra et al. [53]), *nested cuts* and *arc-minimal cuts*. Although no updated comparison with Duin's results was provided, the authors managed to solve almost all benchmark instances from the OR-Library to optimality. Based on these results, two years later, a new collection of difficult STP instances at the time, SteinLib [146], was born.

Poggi de Aragão et al. [178] developed a dual-ascent-based branch-and-bound procedure. The results indicated that their approach is particularly effective on the *incidence instances* from the SteinLib (e.g., dataset I320). Moreover, the authors also showed how to improve the performance of a branch-and-cut based on the DCUT formulation by warm-starting it using the active cuts obtained from the dual-ascent solution. Starting from a sequential branch-and-bound developed by Poggi de Aragão et al. [178] and in the PhD thesis of Uchoa [223], Drummond et al. [70] proposed a *fully distributed branch-and-bound* for STP.

The efforts of Duin [72] were further improved by Polzin and Vahdati Daneshmand [183] whose exact solution framework for the STP remained state-of-the-art for many SteinLib instances until today. An overview of the algorithmic ideas used in their framework can be found in [187]. With a careful implementation, the authors managed to speed up the execution of known reduction tests. Moreover, they introduced expanded reduction tests, reduction by partitioning and a dynamic programming algorithm [186], a polynomial variant of the common flow formulation (cf. Section 3), and the ascend-and-prune heuristic. Thanks to the integration of all these techniques, the majority of SteinLib instances were solved to optimality in very short computing times. However, it is worth mentioning that their framework is extremely well-tuned for undirected STP instances, which makes it less flexible when it comes to minor modifications in the problem definition. So for example, node weighted, budget-, quota- or degree-constrained problem variants cannot be addressed by their method. Moreover, the PUC instances, which were specially designed to defy the majority of standard reduction tests, along with some of the largest VLSI instances (dataset LIN), remained out of reach of their method. A recently updated computational study on the results achieved by their framework can be found in [188].

The next milestone in the development of exact methods for the STP was the 11th DIMACS Challenge, during which ten teams competed in solving the STP and five other problem variants. A large number of difficult and real-world benchmark instances was made available by many contributors [1]. These instances constitute the current battlefield for development of new methods, even though a structured overview of state-of-the-art results per each instance (like it is given for the SteinLib, for example) is missing. Among the exact methods, the winner of the challenge for the STP, PCSTP, MWCS and the degree-constrained STP (DCSTP) was the code `mozartballs` from Fischetti et al. [95]. The only exception was the rooted PCSTP variant, for which the `SCIPJack` code of Gamrath et al. [102] was slightly faster.

The codes `mozartballs` and `staynerd` from Fischetti et al. [95] refer to a general-purpose branch-and-cut approach (for solving the STP/SAP/PCSTP/NWSTP/ DCSTP) based on an earlier exact solution framework for the PCSTP and MWCS introduced by Ljubić et al. [162] and Álvarez-Miranda et al. [2, 3], respectively. Starting from a rule-based parameter selection procedure, the framework chooses between the DCUT model (after transforming the instance into the SAP) and the node-based connectivity model. Next, the framework chooses between the set-covering-based heuristic (cf. Section 7), a variable neighborhood descent close to the one developed by Uchoa and Werneck [226] and a partition-based heuristic of Leitner et al. [151]. Once the pool of feasible solutions is generated, a local branching procedure [96] is applied. Connectivity cuts separated during the local branching phase are subsequently used to initialize the LP before the final branch-and-cut phase is entered. The commercial MIP solver CPLEX is used for the branch-and-cut scheme, and separation procedures for DCUT rely on implementations developed in Ljubić et al. [162].

The package is publicly available. [2]

An improved version of the `SCIPJack` solver was published in Gamrath et al. [102]. `SCIPJack` is a general-purpose exact framework for the STP, its variants mentioned above, and for the hop-constrained STP and the group STP problem. The solver is based on an earlier version of the branch-and-cut scheme by Koch and Martin [145]. It is implemented on top of the academic MIP solver SCIP. Notable algorithmic improvements with respect to Koch and Martin [145] include additional reduction tests [180, 196, 224, 228], VND [226], the Ascend-and-Prune heuristic [180, 228] and the dual-ascent procedure from Pajor et al. [177]. Most importantly, the framework exploits transformations from various variants of the STP to the SAP, to which reduction tests for the SAP [75, 196] are applied. However, the SAP appears to be more difficult to preprocess than the STP, which is why the authors also included several problem-specific reduction tests in their framework. The code is available open source and can benefit from a massively parallel MIP-framework enabled by the SCIP solver. Recent implementation details and computational results of using `SCIPJack` in a distributed environment with 43 000 cores are given in Shinano et al. [214].

Leitner et al. [153] provided the most recent exact solution framework for the asymmetric PCSTP (which can also solve the STP/SAP, MWCS and NWSTP, thanks to a specific transformation into the SAP with both positive and negative arc weights). The authors provided a generalization of the Wong's dual ascent algorithm that shares similarities with the primal-dual 2-approximation algorithm for the PCSTP from Goemans and Williamson [106]. New bound-based and alternative-based reduction tests for the asymmetric PCSTP are introduced as well. This is the first exact framework which is capable of handling Steiner tree problems that include asymmetric arc costs. On most of the available benchmark instances the code is several orders of magnitude faster than the code of Fischetti et al. [95], and it is even faster than state-of-the-art heuristics from Fu and Hao [98, 99]. The code is available as open source. [3]

Rehfeldt and Koch [198] studied the MWCS and provided a combination of (NP-hard) reduction tests, primal heuristics, a problem transformation into the SAP and the integration of these techniques into a branch-and-cut framework. They further improved the computational times from Leitner et al. [153] and solved one more difficult problem instance to optimality.

Finally, in a very recent *Networks* issue, Siebert et al. [215] presented a novel MIP-based approach for solving the SAP. Each feasible rooted SA can be represented using a *laminar family* of subsets of terminals. The authors exploited this fact to derive an LP model which finds a least-expensive SA for a given admissible laminar family. They also proved that it is sufficient to consider laminar families whose tree representation corresponds to a full binary tree, thus showing that at most $O((\frac{2|R|}{e})^{|R|})$ LPs need to be solved to find the optimal solution value. Hence, their approach results in a poly-time algorithm when the number of terminals $|R|$ is bounded by a constant. Nevertheless, computational performance of this model remains fairly limited.

Dynamic programming methods and parametrized algorithms.

We now briefly summarize the most relevant fixed-parameter tractable (FPT) methods for the STP. FPT methods are exact algorithms which are computable in a time that is polynomial in the input size and exponential in a given parameter, so that they become tractable after the value of the parameter is fixed. For example, the STP is fixed-parameter tractable with respect to $|R|$: the Dreyfus-Wagner algorithm [67] (see also [157]) is a classical exact method based on dynamic programming (DP) which runs in $O(3^{|R|}|V|)$. Subsequent improvements are given in [100] and in [30, 172] – the latter two approaches reduce the worst-case time complexity to $O(2^{|R|}|V|^2 c_{\max} + |V||E|c_{\max})$ for the special case that all edge weights are from the set $\{1, 2, \dots, c_{\max}\}$. Vygen [233] provides an improved DP approach which currently has the best running time bound for a wide range of $|R|/|V|$ values. The first polynomial-

[2] See `homepage.univie.ac.at/ivana.ljubic/research/staynerd/StayNerd.html`

[3] See `github.com/mluipersbeck/dapcstp`

time algorithm for the SAP (when $|R|$ is a fixed constant) is given by Feldman and Ruhl [89]. The only FPT algorithms for the MWCS that we are aware of are given by Buchanan et al. [37]: the one solves the MWCS in time $O(2^q(|E|+|V|))$, where q is the number of nodes with a negative weight, and the other runs in time $O(4^\rho|V|^3)$ where ρ is the number of nodes with a positive weight. Finally, for graphs whose independence number α is bounded, the authors showed that the optimal solution can be found in $O(4^\alpha|V|^3)$ time.

Similarly, DP algorithms applied to tree decompositions to solve the STP use time polynomial in $|V|$ and exponential in the treewidth, resulting in polynomial-time algorithms on graphs with a bounded treewidth (see [32, 49, 129] and further references therein). Roughly speaking, the treewidth of a graph G measures how "similar" is G to a tree. Experimental evaluation of the several variants of the DP algorithm of Bodlaender et al. [32] (enhanced by weighted bit string representation) is given by Fafianie et al. [85] and their software is publicly available.[4]

While the majority of the aforementioned DP algorithms is of limited practical viability, the work of Hougardy et al. [129] stands out as an exception. The authors provided a more efficient implementation of the Dreyfus-Wagner algorithm in which shortest path computations are enhanced by the concepts of *label-pruning* and *future costs*. The proposed algorithm is shown to be practically efficient, especially for large-scale instances with very few terminals.

The recent PACE 2018 Challenge on Steiner trees [33] attracted a lot of researchers from the FPT community. One of the goals of the challenge was to test the computational performance and practical applicability of parametrized and heuristic algorithms for the STP. Three competition tracks were created, derived from the instances: with a small number of terminals (track A) , with a low treewidth and a high number of terminals (track B), and with a large number of terminals and a high treewidth (track C). The most successful implementation for the track A, given by Iwata and Shigemura [137], is based on the separator-based improvement of the DP method of Erickson et al. [83]. This implementation, which is also available as an open-source,[5] is competitive against the aforementioned state-of-the-art branch-and-cut approaches for the instances with a small number of terminals. The track B was dominated by the results obtained by the `SCIPJack` solver (whose team was the only one to participate in both, DIMACS and PACE challenge), and the best results for the track C are obtained using a memetic algorithm whose implementation is also publicly available. [6] Further details about the competition and the obtained results can be found in Bonnet and Sikora [33].

9 | APPROXIMATION ALGORITHMS

Approximation algorithms are polynomial-time heuristics that provide a provable guarantee on the quality of the obtained solution with respect to the optimal one. Numerous results are available concerning the STP and its variants, many of them being derived on restricted graph topologies. An online compendium by Hauptmann and Karpiński [125] provides an up-to-date overview of the best known approximation results, and addresses, besides the STP and SAP, 38 additional problem variants! Detailed reviews of approximation algorithms for the STP can be found in [119, 192].

Chlebík and Chlebíková [51] showed that it is NP-hard to approximate the STP within a factor of 96/95. The majority of approximation algorithms developed before 2010 were of a combinatorial nature [142, 191, 245, 246]. The strongest known combinatorial approximation algorithm by Robins and Zelikovsky [204] yields an approximation ratio of $1.55+\varepsilon$. Construction heuristics which are much easier to implement and are therefore frequently used in computational frameworks (cf. Sections 7 and 8) like the TM heuristic [218], the distance network heuristic [170], the primal-dual algorithm [111], the dual-ascent approach from [181], or the local search from [226] yield $(2-2/|R|)$-

[4] See `http://www.staff.science.uu.nl/~bodla101/java/steiner.zip`

[5] `https://github.com/wata-orz/steiner_tree`

[6] `https://github.com/HeathcliffAC/SteinerTreeProblem`

approximations. Despite the fact that the DCUT model is known to be stronger than its undirected counterpart [109], no better LP-based approximation algorithm based on this formulation is known.

While the MIP community is concerned with the relative strength of MIP formulations for the STP, the approximation community studies the worst possible gap between the LP-relaxation of an MIP model and the optimal solution. Until the work of Byrka et al. [38], the question whether there exist an MIP formulation for the STP whose (efficiently solvable) LP relaxation is bounded away from 2, remained open. The authors proved that the integrality gap of the HYP formulation (cf. Section 3) is 1.55 (a simpler proof is given in [42]). In the same article, Byrka et al. provided the currently best known constant approximation ratio of $1.39 + \varepsilon$ for the STP. The solution was obtained by using the HYP_{CUT} formulation (over a subset of FSAs of a restricted size) in combination with an iterative randomized rounding technique and it required to repeatedly solve the LP-relaxation of HYP_{CUT}. Goemans et al. [110] gave an alternative, deterministic and much faster method in which the solution of a single LP-relaxation of the *undirected* HYP relaxation is sufficient to guarantee the same approximation ratio.

The major idea behind combinatorial [142, 204, 245, 246] as well as LP-based [38, 42, 110] approximation algorithms with the ratio better than 2 is based on a decomposition of a Steiner tree into k-restricted full Steiner components (for a given parameter $k \geq 3$). Among a set of k-restricted components, the algorithms search for a cost efficient way of putting them together to obtain a feasible Steiner tree. For the majority of the algorithms, the strongest approximation ratios are only obtained for $k \mapsto \infty$. However, the running time (which is exponential in k) heavily affects the potential practical applicability of these algorithms. There are very few computational studies that address these issues. In a first study provided by Chimani and Woste [50], the authors provided a side-by-side comparison (with respect to the CPU time and the solution quality) of two construction heuristics [170, 218], with the combinatorial approaches by Zelikovsky [246] and the algorithm of Robins and Zelikovsky [204]. The study showed that on a set of benchmark instances from the SteinLib, the older and simpler approach by Zelikovsky [246] was comparable with heuristics from [170, 218], and that all three methods computationally outperformed the 1.55 approximation algorithm from Robins and Zelikovsky [204]. They also showed that the exact method by Polzin and Vahdati Daneshmand [183] was faster and applicable to larger graphs than the combinatorial approximation algorithms. In a subsequent study, Beyer and Chimani [24] implemented the $(1.39 + \varepsilon)$-approximation algorithm following the recipe of Goemans et al. [110]. They concluded that a practical implementation can be achieved for $k = 3$, however the algorithm was still not competitive against the TM heuristic or a simple branch-and-cut derived from the DCUT formulation (see also [57] for similar conclusions). In a follow-up article, Beyer and Chimani [25] showed that even after applying reduction tests as a preprocessing step, no implementation of these strong approximation algorithms could compete with state-of-the-art exact and heuristic algorithms. Finally, Beyer and Chimani [26] gave a comprehensive and detailed overview of different implementation aspects and parameter choices for combinatorial and LP-based strong approximation algorithms for the STP. They concluded that one of the major bottlenecks for the computational efficiency of the method by Goemans et al. [110] seems to be the generation and storage of k-restricted FSCs and the computation of the LP-relaxation of HYP.

10 | OLD AND NEW APPLICATIONS OF THE STP

The book by Cheng and Du [47] from 2001 was devoted to applications of Steiner trees in industry. In the following, we briefly overview some of the traditional applications, and highlight the more recent and challenging ones from the fields of bioinformatics, systems biology and machine learning.

Multicast routing.

Multicasting (also known as selective broadcasting) is a technology that is used to efficiently distribute large amount of data among a group of nodes in a telecommunicaton network. Typical applications include video on demand, distance learning, or teleconferencing. The minimum-cost routing of multicast connections in which there is a central (root) node from which the information has to be distributed to a selected subset of destination nodes (terminals) is frequently modeled as a STP. Surveys on multicast routing (that also addressed additional practical side constraints) were given by Novak et al. [173] and Oliveira and Pardalos [174], see also [69, 195, 210].

Physical design of circuits.

Integrated circuits are created by combining millions of transistors onto a single (micro)chip. Herein, a set of given points (representing components) need to be connected through wires with a Steiner tree of minimum length, using only horizontal and vertical line segments (see [156, 244] for a detailed electrical background on constructing the layouts). The problem is known as the *rectilinear Steiner tree problem*, and can be modeled as a STP on the underlying Hanan grid induced by the given points. Instead of approaching the problem on such induced large graphs, many specialized, better performing algorithms have been developed in the past (see [139] for a detailed overview). Currently, state-of-the-art results are obtained by Juhl et al. [139] and their GeoSteiner 5.0 implementation. Their code is publicly available. [7] Many problem variants involving more realistic side constraints (such as avoiding obstacles, or additionally minimizing flow-costs) were also addressed in the literature (see [36, 52, 244]).

Design of service networks.

Design of *access networks* in telecommunications is frequently modeled as a STP. An access network connects subscribers (terminal nodes) to a central office of their service provider (root node) through copper or fiber optic cables (or a combination of the two). The street junctions (potential Steiner nodes) along with aforementioned nodes make up the network. The nodes are connected with links corresponding to street segments and the weight of each link represents the cost for digging the cable ducts. In contrast to *core networks* (that need to ensure network connectivity in case of potential link failures), the access networks typically have a tree topology, hence the optimization problem that minimizes the infrastructure cost becomes a (rooted) STP (see, e.g., [31] for a more detailed description and further practical constraints/costs that might be included in the network planning). A survey on optimization problems for the deployment of last-mile optical access networks can be found in Grötschel et al. [120], and some older examples of using the STP for telecommunicaton networks were given by Voß [232] (see also the book by Resende and Pardalos [200]).

Sometimes, service providers may be given a choice to decide (based on expected customers' revenues) to which customers they want to offer the service. In this case the problem becomes the PCSTP. The design of district heating networks was addressed in Ljubić et al. [162] and a prize-collecting version of the local access network design was studied in Ljubić et al. [161].

Frequently, network design and location decisions are combined. So, for example Prodon et al. [190] used the PCSTP to locate sensors for leak detection in water distribution networks. In the design of local access networks, the *facility location problem* and the STP are combined and the underlying optimization problem is known as the *Connected Facility Location* Problem (ConFLP). The most recent theoretical and computational studies for the ConFL are given in Bardossy and Raghavan [16], Gollowitzer and Ljubić [114] and Leitner et al. [149, 154]. In Gollowitzer et al. [112] the authors additionally took the cable capacities into account, and Arulselvan et al. [8] considered the problem in a multi-period planning context. A bi-objective problem variant of the ConFLP was studied by Leitner et al. [155]. The

[7] See `geosteiner.com`

ConFLP under uncertainty was addressed in Bardossy and Raghavan [17] and in a 2017 *Networks* article by Bardossy and Raghavan [18]. Ljubić and Gollowitzer [160] studied the ConFLP with hop constraints.

Bioinformatics and Systems Biology.

A sequence of articles in bioinformatics and systems biology, initiated with the work of Ideker et al. [136] used Steiner trees in networks (and in particular, the PCSTP and MWCS) as a successful modeling tool for analysis of high-throughput data. New metabolic pathways (series of interactions among molecules in a cell) are detected by looking at a collective interaction between differentially expressed genes, proteins or other biological factors. Node weights correspond to a statistical *score* associated to the gene expression and (possible) edge weights express the level of confidence for the interaction between pairs of nodes. The *subnetwork extraction algorithms* are used to favor subgraphs containing genes with higher scores over the others, and precisely this is achieved by modeling the underlying optimization problem as the MWCS (in case of node-weighted networks) or the PCSTP (in the more general case). Using this model, Ideker et al. [136] implemented an algorithm based on simulated annealing to find suboptimal solutions. A few years later, Scott et al. [212] provided the first study that explicitly used the STP as a modeling tool for investigating gene interaction networks. Contrary to Ideker et al. [136], who performed a global search based on node scores only, Scott et al. [212] allowed the users to specify the set of terminals (calling them “distinguished” nodes, which are typically genes/elements that are most likely to be involved in regulatory interactions).

A plethora of follow-up articles in bioinformatics and systems biology based on the STP/MWCS modeling strategies repeatedly reported that the obtained subnetworks (i.e., pathways) are often unexpected but for the biologists the most interesting ones. The discovered pathways can provide new insights into biological processes and potentially lead towards new drug discoveries and, more generally speaking, new therapeutic approaches (see, e.g., [9, 11, 12, 81, 87, 208, 242, 243], and the PhD theses of El-Kebir [79], Huang [130] and Hume [132]).

Dittrich et al. [66] proposed a transformation from the MWCS into the rooted PCSTP problem. The authors used the exact algorithm of Ljubić et al. [162] in their software package to determine (sub)optimal solutions. Likewise, Huang and Fraenkel [131] (see also [222]) integrated the exact PCSTP algorithm from Ljubić et al. [162] into their software to detect novel pathways in integrated protein–protein and protein–DNA networks. El-Kebir and Klau [80] (see also [79]) developed a branch-and-cut code combined with graph decomposition techniques for the MWCS/PCSTP, called `Heinz`[8] specifically targeted for bioinformatics applications. Tuncbag et al. [220] (see also [221]) used the prize-collecting Steiner *forest* to effectively reconstruct multiple pathways. Practically speaking, the authors modeled the problem as the PCSTP with a degree constraint at the root node. Their problem was solved using the code from Bailly-Bechet et al. [12]. Most recently, Klimm et al. [144] combined single-cell RNA-sequencing data with protein-protein interaction networks to detect active modules in cells of different transcriptional states. The open-source code by Leitner et al. [153] was employed to solve their underlying MWCS model.

The PhD thesis of Won [239] (see also [43]) dealt with the prevention of a delayed detection of dementia, by analyzing the strength and structure of brain connectivity within certain functional regions which are associated with cognitive decline. The authors modeled the problem as the k-cardinality tree problem, which can also be seen (and efficiently solved) as a cardinality-constrained SAP (see next section, and also [48, 216]).

Wildlife Conservation, Forestry Planning.

The MWCS and PCSTP, sometimes extended by cardinality or budget constraints, are common models for addressing network design questions in wildlife conservation. The underlying problems typically ask to select a subset of contiguous land parcels for conservation to ensure the mobility of threatened species between existing reserves (see

[8] https://github.com/ls-cwi/heinz

[58, 59, 64]) or to connect known habitats of a given set of different species (see [175, 176]). The problems are also known as the reserve network design or the corridor design problem. The input graph is made up by nodes representing land parcels, which are connected by an edge if and only if the two land parcels are adjacent. Connections to the SAP are emphasized by Dilkina and Gomes [64] and Álvarez-Miranda et al. [2, 3]. Similar applications of the connected subgraph problem arise in forestry planning [41], or off-shore oil-drilling [128].

Machine Learning and Robotics.

Applications of the SAP in machine learning can be found in image recognition and feature extraction [45, 207, 211, 230]. Russakovsky and Ng [207] proposed a reduction of a feature selection problem to the SAP, to efficiently select parameters for image segmentation. The SAP was solved heuristically, using the approximation algorithm of Charikar et al.[44]. Feature extraction on 2D seismic images was modeled as the PCSTP over a specially designed graph described in Schmidt et al. [211]. A more general modeling approach for extracting sparse features using graph transformations into the PCSTP was discussed in Hegde et al. [127]. The authors employed their own efficient implementation of the 2-approximation algorithm of Goemans and Williamson [106] (see also [126]). In Vijayanarasimhan and Grauman [230], the MWCS was used for detecting the contours of objects in a given image. In a follow-up article, Chen and Grauman [45] extended this approach, and applied it to the detection of moving objects in video streams. In both cases, problem instances were transformed to the PCSTP and solved using the exact solver from Ljubić et al. [162].

Banfi et al. [15] used the PCSTP and MWCS to model deployment of robots for search and rescue operations in communication-restricted environments (i.e., where communication between robots and the base station is enabled through an ad hoc network). More details can be found in the PhD thesis of Banfi [14].

11 | THE STP/SAP AS A POWERFUL MODELING TOOL

The STP and the SAP were frequently exploited as a powerful tool to model other difficult combinatorial optimization problems. We highlight some of the recent successful examples, in which the original problems were transformed into a (constrained) SAP/STP or PCSTP, allowing one to exploit structural properties of these better understood problems within computational frameworks.

The Connected Facility Location Problem.

The goal of the ConFLP is to install a set of facilities on a network and assign customers to the installed facilities. In addition, the set of installed facilities has to be connected by a Steiner tree. Bardossy and Raghavan [16] transformed the problem into the SAP with a unit-degree constraint at the root node and then applied a dual-ascent heuristic combined with local search to obtain strong lower and upper bounds. A simpler transformation into the SAP, using a much smaller directed graph was given in Tomazic and Ljubić [219] and Leitner et al. [154]. More general problems with a tree-star topology and their transformation into a constrained SAP were addressed by Leitner et al. [154]. Even more general setting with a tree-tree topology (and a set of facilities being installed to provide transition from the primary subtree into the secondary subtrees) was considered by Gollowitzer et al. [113]. The authors showed how to transform the problem into the SAP with additional node-degree constraints. A survey by Fortz [97] covers the ConFLP and other more recent applications that combine network design and facility location decisions in the context of telecommunication networks. Another problem closely related to the ConFLP is the *p-arborescence star problem* for which Morais et al. [171] exploited a connection to the SAP to derive a strong cut-set based MIP formulation.

Hop-constrained and diameter-constrained trees.

The goal in these settings is to find a minimum-cost spanning/Steiner tree such that the number of edges between the root and any terminal (between any two terminals, respectively) is bounded by a given constant. These problems can be transformed into the SAP on specially constructed layered graphs (see [69, 117]).

Gouveia et al. [115] solved a hop-constrained problem variant in which multiple root nodes are given. They modeled the problem as an intersection of Steiner arborescences – one Steiner arborescence on a layered graph is used to connect each root with its given set of terminals. Ljubić and Gollowitzer [160] used two different SAP models (extended by node-degree constraints) to model the hop-constrained ConFLP. A node-based SAP modeling approach to the hop- and budget-constrained PCSTP problem was proposed by Sinnl and Ljubić [217]. A recent survey on layered graph transformations for combinatorial optimization problems can be found in Gouveia et al. [116].

The k-cardinality tree problem.

The problem asks for a minimum-cost subtree in an edge-weighted graph with exactly k edges. Fischetti et al. [94] introduced a first exact approach for the problem in 1994 in *Networks*, and Chimani et al. [48] used a transformation into the SAP with an additional arc-cardinality constraint to derive an exact approach which even outperformed state-of-the-art metaheuristics at the time. The authors also showed how to exploit this SAP transformation for node-weighted graphs and for the prize-collecting variant as well. Simonetti et al. [216] later extended the SAP model with several families of facet-defining inequalities and improved the results of Chimani et al. [48].

The Regenerator Location Problem.

The problem deals with transmission of signals in optical networks: a signal can only travel a limited distance before it has to be re-amplified by installing regenerators at nodes of the network. The goal is to deploy the minimal number of regenerators, while ensuring all nodes can communicate with each other. The problem was introduced in *Networks* by Chen et al. [46]. The authors developed a branch-and-cut based on a problem transformation into the SAP with a unit-degree constraint at the root node. The authors also pointed out the connection to the *Maximum Leaf Spanning Tree* (MLST) and a similar transformation of the latter to the SAP. The MLST problem asks for a spanning tree that maximizes the number of leaves. A polyhedral study published in *Networks* by Fujie [101], together with the work of Fernandes and Gouveia [91] led the way to the work of Lucena et al. [166] who exploited the connection to the SAP to derive two strong models for the MLST problem on directed graphs. Both problems are closely related to the *Minimum Connected Dominating Problem* (MCDP) which asks for a dominating set of a graph of minimum cardinality. In another *Networks* article, Wu et al. [241] proposed a tabu search approach and highlighted the importance of the MCDP in the design of mobile ad hoc networks and sensor grids. Finally, Gendron et al. [105] exploited the connection to the STP to derive an efficient Benders decomposition approach.

The Group Steiner Tree Problem (GSP).

In this problem we are given a family of node subsets and the goal is to find a minimum-cost tree spanning (at least) one node from each subset. The problem can be easily transformed into a STP on an enlarged graph (see [135, 209]). Duin et al. [76] investigated the computational performance of this transformation and showed that it compares favorably with specialized GSP heuristics at the time. Leitner [148] introduced the GSP with hop constraints and exploited the connection to the SAP to derive strong MIP formulations on a specially designed layered graph for which he also implemented a branch-and-cut approach.

12 | CONCLUSIONS

The STP continues to fascinate researchers from many disciplines, including Applied Mathematics, Computer Science, Operations Research and Management Science. New and challenging applications are regularly being discovered, with recent examples stemming from bioinformatics, systems biology or machine learning.

Historically speaking, the 1990s were marked by polyhedral studies, in the early 2000s we saw a rise of exact solution methods and a sequence of combinatorial approximation algorithms. In the 2010s, a breakthrough has been achieved by developing new LP-based approximation algorithms and finally the DIMACS Challenge in 2014 and the 2018 PACE Challenge brought to light some new sophisticated generic or FPT-based exact solution frameworks. However, there is no unique answer to the question: what is the state-of-the-art exact solver for the STP? Several generic computational frameworks (Fischetti et al. [95], Gamrath et al. [102], Leitner et al. [153]) emerged in the past few years, but none of them dominate the others. Some are stand-alone open-source implementations ([153]), the others rely on MIP solvers, like CPLEX ([95]) or SCIP ([102]), or can be applied only to specific problem variants, like the MWCS (Rehfeldt and Koch [198]). It is worth mentioning that the implementation of Polzin and Vahdati Daneshmand [187] from 2009 remains competitive or even state-of-the-art for many instances from the SteinLib, but their code is unfortunately not publicly available.

It is hard to predict what the next breakthrough developments will be, however, there are certainly some promising directions for future research that we will highlight below.

- Despite the fact that the bidirected cut formulation (DCUT) has an exceptional empirical performance, finding a better-than-2 upper bound on its integrality gap remains a well-known open problem.
- In this article we gave an overview of recent MIP formulations that are strictly stronger than DCUT. Unfortunately, the size of these models is prohibitive and makes it impossible to simply plug them into an off-the-shelf solver. Hence, the development of advanced implementations and decomposition techniques to make these models competitive with the state-of-the-art is an interesting venue for future research. Moreover, the relationship between the two flow-based hierarchies, the one based on common-flows of Polzin and Vahdati Daneshmand [187] and the other based on paths of Filipecki and Van Vyve [92] remains to be studied.
- Further understanding of the structural properties of LP-relaxations of the aforementioned models, and potential combination of hypegraphic formulations with flow-based ones can lead to new theoretical insights and possibly better approximation ratios.
- Recent computational studies have shown that the strongest approximation algorithms currently existing are not competitive with the state-of-the-art exact methods. Addressing these issues either through some advanced algorithm engineering techniques or by developing faster approximation algorithms is another interesting open direction for future research.
- This article showcases numerous examples in which the SAP was used as a powerful modeling tool for solving other network design problems. While there exists a plethora of very advanced reduction techniques for the STP, the same cannot be said of the SAP, and in particular not of the setting in which negative arc weights are allowed. Given that the reduction techniques are an indispensable tool for modern exact and heuristic methods, it is expected that the development of more advanced reduction techniques could further boost the performance of SAP-based approaches.
- In the future we may expect to see more advanced STP algorithms developed for parallel computing. An excellent pioneering example is a very recent `PARASCIP` implementation of Shinano et al. [214] with which new STP results are obtained in a distributed environment with 43 000 cores.

- We might also see a revival of “sleeping beauties”, like Lagrangian, Benders or Dantzig-Wolfe decompositions (and combinations thereof), in the context of distributed/parallel computing. With data veracity in big-data applications, there is a diminishing demand in producing STP solutions with zero gap. Instead, practitioners desire stable solutions, robust with respect to minor data modifications (see, e.g., [4]), and with small duality gaps. Decomposition approaches might be promising candidates to meet these demands stemming from more computationally demanding real-world applications.

ACKNOWLEDGMENTS

The author is grateful to the Guest Editors of this special issue, S. Raghavan and Cole Smith, for the opportunity to write this article, and to Martin Luipersbeck, for collecting many of the references used in this survey.

References

[1] 11th DIMACS Implementation Challenge: Steiner tree problems, 2015..

[2] E. Álvarez Miranda, I. Ljubić, and P. Mutzel, “*The maximum weight connected subgraph problem*," *Facets of Combinatorial Optimization: Festschrift for Martin Grötschel*, M. Jünger and G. Reinelt (eds.), Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 245–270.

[3] E. Álvarez-Miranda, I. Ljubić, and P. Mutzel, *The rooted maximum node-weight connected subgraph problem*, Proceedings of CPAIOR 2013, Vol. 7874 of *LNCS*, Springer, 2013, pp. 300–315.

[4] E. Álvarez-Miranda, I. Ljubić, and P. Toth, *Exact approaches for solving robust prize-collecting steiner tree problems*, Eur. J. Oper. Res. **229** (2013), 599 – 612.

[5] E. Álvarez-Miranda and M. Sinnl, *A relax-and-cut framework for large-scale maximum weight connected subgraph problems*, Comput. Oper. Res. **87** (2017), 63–82.

[6] Y.P. Aneja, *An integer linear programming approach to the Steiner problem in graphs*, Networks **10** (1980), 167–178.

[7] A. Archer, M. Bateni, M. Hajiaghayi, and H. Karloff, *Improved approximation algorithms for prize-collecting Steiner tree and TSP*, SIAM J. Comput. **40** (2011), 309–332.

[8] A. Arulselvan, A. Bley, and I. Ljubić, *The incremental connected facility location problem*, Comput. Oper. Res. **112** (2019).

[9] C. Backes, A. Rurainski, G.W. Klau, O. Müller, D. Stöckel, A. Gerasch, J. Küntzer, D. Maisel, N. Ludwig, M. Hein, et al., *An integer linear programming approach for finding deregulated subgraphs in regulatory networks*, Nucleic Acids Res. **40** (2012), e43–e43.

[10] L. Bahiense, F. Barahona, and O. Porto, *Solving Steiner tree problems in graphs with Lagrangian relaxation*, J. Combinatorial Optim. **7** (2003), 259–282.

[11] M. Bailly-Bechet, C. Borgs, A. Braunstein, J. Chayes, A. Dagkessamanskaia, J. Francois, and R. Zecchina, *Finding undetected protein associations in cell signaling by belief propagation*, Proc. National Acad. Sci. (2010), 1–6.

[12] M. Bailly-Bechet, A. Braunstein, and R. Zecchina, *A prize-collecting Steiner tree approach for transduction network inference*, Computational Methods in Systems Biology, 7th International Conference, CMSB 2009, Bologna, Italy, August 31-September 1, 2009. Proceedings, Vol. 5688 of *Lecture Notes in Computer Science*, Springer, 2009, pp. 83–95.

[13] E. Balas, *The prize-collecting traveling salesman problem*, Networks **19** (1989), 621–636.

[14] J. Banfi, Multirobot exploration of communication-restricted environments, Ph.D. thesis, Politecnico di Milano, Italy, 2018.

[15] J. Banfi, A. Quattrini Li, I. Rekleitis, F. Amigoni, and N. Basilico, *Strategies for coordinated multirobot exploration with recurrent connectivity constraints*, Autonomous Robots **42** (2018), 875 – 894.

[16] M.G. Bardossy and S. Raghavan, *Dual-based local search for the connected facility location and related problems*, INFORMS J. Computi **22** (2010), 584–602.

[17] M.G. Bardossy and S. Raghavan, *Approximate robust optimization for the connected facility location problem*, Discr. Appl. Math. **210** (2016), 246–260.

[18] M.G. Bardossy and S. Raghavan, *An inexact sample average approximation approach for the stochastic connected facility location problem*, Networks **70** (2017), 19–33.

[19] M.P. Bastos and C.C. Ribeiro, "*Reactive tabu search with path-relinking for the Steiner problem in graphs*," *Essays and Surveys in Metaheuristics*, Springer, Boston, MA, 2002, pp. 39–58.

[20] J. Beasley, 1990. Collection of test data sets for a variety of operations research (OR) problems.

[21] J.E. Beasley, *An algorithm for the Steiner problem in graphs*, Networks **14** (1984), 147–159.

[22] J.E. Beasley, *An SST-based algorithm for the Steiner problem in graphs*, Networks **19** (1989), 1–16.

[23] M. Bern and P. Plassmann, *The Steiner problem with edge lengths 1 and 2*, Informat. Process. Lett. **32** (1989), 171–176.

[24] S. Beyer and M. Chimani, *Steiner tree 1.39-approximation in practice*, Mathematical and Engineering Methods in Computer Science, Springer International Publishing, Cham, 2014, pp. 60–72.

[25] S. Beyer and M. Chimani, *The influence of preprocessing on Steiner tree approximations*, Proceedings of the 9th International Conference on Combinatorial Optimization and Applications - Volume 9486, Springer-Verlag, Berlin, Heidelberg, 2015, COCOA 2015 pp. 601–616.

[26] S. Beyer and M. Chimani, *Strong Steiner tree approximations in practice*, J. Exp. Algorithmics **24** (2019).

[27] M. Bezenšek and B. Robič, *A survey of parallel and distributed algorithms for the Steiner tree problem*, Int. J. Parallel Program. **42** (2014), 287–319.

[28] D. Bienstock, M. Goemans, D. Simchi-Levi, and D. Williamson, *A note on the prize collecting traveling salesman problem*, Math. Progamming **59** (1993), 413–420.

[29] M.D. Biha, H. Kerivin, and A.R. Mahjoub, *Steiner trees and polyhedra*, Discr. Appl. Math. **112** (2001), 101–120.

[30] A. Björklund, T. Husfeldt, P. Kaski, and M. Koivisto, *Fourier meets Möbius: fast subset convolution*, Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007, ACM, 2007, pp. 67–74.

[31] A. Bley, I. Ljubić, and O. Maurer, *Lagrangian decompositions for the two-level FTTx network design problem*, EURO J. Comput. Optimizati **1** (2013), 221–252.

[32] H.L. Bodlaender, M. Cygan, S. Kratsch, and J. Nederlof, *Deterministic single exponential time algorithms for connectivity problems parameterized by treewidth*, Informat. Computation **243** (2015), 86 – 111, 40th International Colloquium on Automata, Languages and Programming (ICALP 2013).

[33] É. Bonnet and F. Sikora, *The PACE 2018 Parameterized Algorithms and Computational Experiments Challenge: The Third Iteration*, 13th International Symposium on Parameterized and Exact Computation (IPEC 2018), Vol. 115 of *Leibniz International Proceedings in Informatics (LIPIcs)*, Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2019, pp. 26:1–26:15.

[34] A. Bouchachia and M. Prossegger, *A hybrid ensemble approach for the Steiner tree problem in large graphs: A geographical application*, Appl. Soft Comput. **11** (2011), 5745 – 5754.

[35] M. Brazil, R.L. Graham, D.A. Thomas, and M. Zachariasen, *On the history of the Euclidean Steiner tree problem*, Archive History Exact Sci. **68** (2014), 327–354.

[36] M. Brazil and M. Zachariasen, *Optimal Interconnection Trees in the Plane: Theory, Algorithms and Applications*, Mathematics and Statistics, Springer, Berlin, 2015.

[37] A. Buchanan, Y. Wang, and S. Butenko, *Algorithms for node-weighted Steiner tree and maximum-weight connected subgraph*, Networks **72** (2018), 238–248.

[38] J. Byrka, F. Grandoni, T. Rothvoss, and L. Sanità, *Steiner tree approximation via iterative randomized rounding*, J ACM **60** (2013), 6:1–6:33.

[39] A. Candia-Véjar and H. Bravo-Azlán, *Worst-case performance of Wong's Steiner tree heuristic*, Discr. Appl. Math. **154** (2006), 730 – 737, IV ALIO/EURO Workshop on Applied Combinatorial Optimization.

[40] A. Caprara, M. Fischetti, and P. Toth, *A heuristic method for the set covering problem*, Oper. Res. **47** (1999), 730–743.

[41] R. Carvajal, M. Constantino, M. Goycoolea, J. Vielma, and A. Weintraub, *Imposing connectivity constraints in forest planning models*, Oper. Res. **61** (2013), 824–836.

[42] D. Chakrabarty, J. Könemann, and D. Pritchard, *Hypergraphic LP relaxations for Steiner trees*, SIAM J. Discr. Math. **27** (2013), 507–533.

[43] W.A. Chaovalitwongse, D. Won, O. Seref, P. Borghesani, M.K. Askren, S. Willis, and T.J. Grabowski, *Network optimization of functional connectivity within default mode network regions to detect cognitive decline*, IEEE Trans Neural Syst Rehabil E **25** (2017), 1079–1089.

[44] M. Charikar, C. Chekuri, T.y. Cheung, Z. Dai, A. Goel, S. Guha, and M. Li, *Approximation algorithms for directed Steiner problems*, J. Algorithms **33** (1999), 73–91.

[45] C.Y. Chen and K. Grauman, *Efficient activity detection with max-subgraph search*, 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012, pp. 1274–1281.

[46] S. Chen, I. Ljubić, and S. Raghavan, *The regenerator location problem*, Networks **55** (2010), 205–220.

[47] X. Cheng and D.Z. Du, *Steiner trees in industry* Vol. 11 of *Combinatorial Optimization*, Springer, Boston, MA, 2001.

[48] M. Chimani, M. Kandyba, I. Ljubić, and P. Mutzel, *Obtaining optimal k-cardinality trees fast*, ACM J. Experimental Algorithmics **14** (2009).

[49] M. Chimani, P. Mutzel, and B. Zey, *Improved Steiner tree algorithms for bounded treewidth*, J. Discr. Algorithms **16** (2012), 67–78.

[50] M. Chimani and M. Woste, *Contraction-based Steiner tree approximations in practice*, Algorithms and Computation, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 40–49.

[51] M. Chlebík and J. Chlebíková, *The Steiner tree problem on graphs: inapproximability results*, Theor. Comput. Sci. **406** (2008), 207–214.

[52] J.D. Cho, "*Steiner tree problems in VLSI layout designs*," *Steiner Trees in Industry*, X.Z. Cheng and D.Z. Du (eds.), Springer, Boston, MA, 2001, pp. 101–173.

[53] S. Chopra, E.R. Gorres, and M. Rao, *Solving the Steiner tree problem on a graph using branch and cut*, INFORMS J. Computi **4** (1992), 320–335.

[54] S. Chopra and M.R. Rao, *The Steiner tree problem I: Formulations, compositions and extension of facets*, Math. Program. **64** (1994), 209–229.

[55] S. Chopra and M.R. Rao, *The Steiner tree problem II: Properties and classes of facets*, Math. Program. **64** (1994), 231–246.

[56] S. Chopra and C.Y. Tsai, "*Polyhedral approaches for the Steiner tree problem on graphs*," *Steiner Trees in Industry*, Springer, Boston, MA, 2001, pp. 175–201.

[57] K. Ciebiera, P. Godlewski, P. Sankowski, and P. Wygocki, *Approximation algorithms for Steiner tree problems based on universal solution frameworks*, CoRR **abs/1410.7534** (2014).

[58] J. Conrad, C. Gomes, J. van Hoeve, A. Sabharwal, and J. Suter, *Wildlife corridors as a connected subgraph problem*, J. Environmental Econ. Manage. **63** (2012), 1–18.

[59] J. Conrad, C.P. Gomes, W.J. van Hoeve, A. Sabharwal, and J. Suter, *Connections in networks: Hardness of feasibility versus optimality*, Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, Springer, Berlin, Heidelberg, 2007, pp. 16–28.

[60] R. Cordone and M. Trubian, *An exact algorithm for the node weighted Steiner tree problem*, 4OR **4** (2006), 124–144.

[61] A. Costa, J.F. Cordeau, and G. Laporte, *Steiner tree problems with profits*, INFOR **44** (2006), 99–115.

[62] R. Courant and H. Robbins, *What is mathematics?*, Oxford University Press, London, 1941.

[63] A.S. da Cunha, A. Lucena, N. Maculan, and M.G. Resende, *A relax-and-cut algorithm for the prize-collecting Steiner problem in graphs*, Discr. Appl. Math. **157** (2009), 1198–1217.

[64] B. Dilkina and C. Gomes, *Solving connected subgraph problems in wildlife conservation*, Proceedings of CPAIOR, Vol. 6140 of *LNCS*, 2010, pp. 102–116.

[65] .Results of the 11th DIMACS Competition on Steiner Tree Problems . Accessed: 2019-12-30.

[66] M.T. Dittrich, G.W. Klau, A. Rosenwald, T. Dandekar, and T. Müller, *Identifying functional modules in protein–protein interaction networks: An integrated exact approach*, Bioinformatics **24** (2008), i223–i231.

[67] S.E. Dreyfus and R.A. Wagner, *The Steiner problem in graphs*, Networks **1** (1971), 195–207.

[68] M. Dror, B. Gavist, and J. Choquette, *Directed Steiner tree problem on a graph: models, relaxations and algorithms*, INFOR **28** (1990), 268–281.

[69] L.M. Drummond, M. Santos, and E. Uchoa, *A distributed dual ascent algorithm for Steiner problems in multicast routing*, Networks **53** (2009), 170–183.

[70] L.M. Drummond, E. Uchoa, A.D. Goncalves, J.M. Silva, M.C. Santos, and M.C.S. de Castro, *A grid-enabled distributed branch-and-bound algorithm with application on the Steiner problem in graphs*, Parallel Comput. **32** (2006), 629 – 642.

[71] D.Z. Du, J. Smith, and J.H. Rubinstein, *Advances in Steiner trees* Vol. 6 of *Combinatorial Optimization*, Springer, Boston, MA, 2000.

[72] C. Duin, Steiner's problem in graphs, Ph.D. thesis, University of Amsterdam, 1993.

[73] C. Duin, "*Preprocessing the steiner problem in graphs*," *Advances in Steiner Trees*, D.Z. Du, J.M. Smith, and J.H. Rubinstein (eds.), Springer, Boston, MA, 2000, pp. 175–233.

[74] C. Duin and A. Volgenant, *Some generalizations of the Steiner problem in graphs*, Networks **17** (1987), 353–364.

[75] C. Duin and A. Volgenant, *Reduction tests for the Steiner problem in graphs*, Networks **19** (1989), 549–567.

[76] C. Duin, A. Volgenant, and S. Voß, *Solving group Steiner problems as Steiner problems*, Eur. J. Oper. Res. **154** (2004), 323 – 329.

[77] C. Duin and S. Voß, *Efficient path and vertex exchange in Steiner tree algorithms*, Networks **29** (1997), 89–105.

[78] J. Edmonds, "*Submodular functions, matroids, and certain polyhedra*," *Combinatorial Optimization — Eureka, You Shrink!: Papers Dedicated to Jack Edmonds 5th International Workshop Aussois, France, March 5–9, 2001 Revised Papers*, M. Jünger, G. Reinelt, and G. Rinaldi (eds.), Springer Berlin Heidelberg, Berlin, Heidelberg, 2003, pp. 11–26.

[79] M. El-Kebir, Networks, modules and breeding schedules: Applications of combinatorial optimization to computational biology, Ph.D. thesis, Vrije Universiteit Amsterdam, 2015.

[80] M. El-Kebir and G.W. Klau, 2014. Solving the maximum-weight connected subgraph problem to optimality. 11th DIMACS Implementation Challenge Workshop. Available at http://dimacs11.zib.de/workshop/ElKebirKlau.pdf. Accessed: 2017-06-15.

[81] M. El-Kebir, H. Soueidan, T. Hume, D. Beisser, M. Dittrich, T. Müller, G. Blin, J. Heringa, M. Nikolski, L.F. Wessels, et al., *xheinz: an algorithm for mining cross-species network modules under a flexible conservation model*, Bioinformatics **31** (2015), 3147–3155.

[82] S. Engevall, M. Göthe-Lundgren, and P. Värbrand, *A strong lower bound for the node weighted Steiner tree problem*, Networks **31** (1998), 11–17.

[83] R.E. Erickson, C.L. Monma, and A.F. Veinott, *Send-and-split method for minimum-concave-cost network flows*, Math. Oper. Res. **12** (1987), 634–664.

[84] H. Esbensen, *Computing near-optimal solutions to the Steiner problem in a graph using a genetic algorithm.*, Networks **26** (1995), 173–185.

[85] S. Fafianie, H.L. Bodlaender, and J. Nederlof, *Speeding up dynamic programming with representative sets: an experimental evaluation of algorithms for Steiner tree on tree decompositions*, Algorithmica **71** (2015), 636–660.

[86] M. Fampa, *Insight into the computation of Steiner minimal trees in Euclidean space of general dimension*, Discr. Appl. Math. (2019), in press.

[87] K. Faust, P. Dupont, J. Callut, and J. Van Helden, *Pathway discovery in metabolic networks by subgraph extraction*, Bioinformatics **26** (2010), 1211–1218.

[88] J. Feigenbaum, C.H. Papadimitriou, and S. Shenker, *Sharing the cost of multicast transmissions*, J. Comput. System Sci. **63** (2001), 21 – 41.

[89] J. Feldman and M. Ruhl, *The directed steiner network problem is tractable for a constant number of terminals*, SIAM J. Comput. **36** (2006), 543–561.

[90] A.E. Feldmann, J. Könemann, N. Olver, and L. Sanità, *On the equivalence of the bidirected and hypergraphic relaxations for Steiner tree*, Math. Program. **160** (2016), 379–406.

[91] L.M. Fernandes and L. Gouveia, *Minimal spanning trees with a constraint on the number of leaves*, Eur. J. Oper. Res. **104** (1998), 250–261.

[92] B. Filipecki and M. Van Vyve, *Stronger path-based extended formulation for the Steiner tree problem*, Networks **75** (2020), 3–17.

[93] M. Fischetti, *Facets of two Steiner arborescence polyhedra*, Math. Program. **51** (1991), 401–419.

[94] M. Fischetti, K. Hamacher, H.and Jørnsten, and F. Maffioli, *Weighted k-cardinality trees: Complexity and polyhedral structure*, Networks **24** (1994), 11–21.

[95] M. Fischetti, M. Leitner, I. Ljubić, M. Luipersbeck, M. Monaci, M. Resch, D. Salvagnin, and M. Sinnl, *Thinning out Steiner trees: a node-based model for uniform edge costs*, Math. Program. Computation **9** (2017), 203–229.

[96] M. Fischetti and A. Lodi, *Local branching*, Math. Program. **98** (2003), 23–47.

[97] B. Fortz, "*Location problems in telecommunications*," *Location Science*, G. Laporte, S. Nickel, and F. Saldanha da Gama (eds.), Springer International Publishing, Cham, 2015, pp. 537–554.

[98] Z.H. Fu and J.K. Hao, *Knowledge-guided local search for the prize-collecting Steiner tree problem in graphs*, Knowledge-Based Syst. (2017).

[99] Z.H. Fu and J.K. Hao, *Swap-vertex based neighborhood for Steiner tree problems*, Math. Program. Computation **9** (2017), 297 – 320.

[100] B. Fuchs, W. Kern, D. Molle, S. Richter, P. Rossmanith, and X. Wang, *Dynamic programming for minimum steiner trees*, Theory Comput. Syst. **41** (2007), 493–500.

[101] T. Fujie, *The maximum-leaf spanning tree problem: Formulations and facets*, Networks **43** (2004), 212–223.

[102] G. Gamrath, T. Koch, S.J. Maher, D. Rehfeldt, and Y. Shinano, *SCIP-Jack — a solver for STP and variants with parallelization extensions*, Math. Program. Computation **9** (2017), 231–296.

[103] M.R. Garey and D.S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness*, W. H. Freeman & Co., USA, 1990.

[104] M. Gendreau, J.F. Larochelle, and B. Sansò, *A tabu search heuristic for the Steiner tree problem.*, Networks **34** (1999), 162–172.

[105] B. Gendron, A. Lucena, A.S. da Cunha, and L. Simonetti, *Benders decomposition, branch-and-cut, and hybrid algorithms for the minimum connected dominating set problem*, INFORMS J. Comput. **26** (2014), 645–657.

[106] M. Goemans and D. Williamson, "*The Primal-dual method for approximation algorithms and its application to network design problems*," *Approximation Algorithms for NP-hard Problems*, D. Hochbaum (ed.), PWS Publishing Company, USA, 1997, pp. 144–191.

[107] M.X. Goemans, *Arborescence polytopes for series-parallel graphs*, Discr. Appl. Math. **51** (1994), 277–289.

[108] M.X. Goemans, *The Steiner tree polytope and related polyhedra*, Math. Program. **63** (1994), 157–182.

[109] M.X. Goemans and Y.S. Myung, *A catalog of Steiner tree formulations*, Networks **23** (1993), 19–28.

[110] M.X. Goemans, N. Olver, T. Rothvoß, and R. Zenklusen, *Matroids and integrality gaps for hypergraphic Steiner tree relaxations*, Proceedings of the forty-fourth annual ACM symposium on Theory of computing, 2012, pp. 1161–1176.

[111] M.X. Goemans and D.P. Williamson, *A general approximation technique for constrained forest problems*, SIAM J. Comput. **24** (1995), 296–317.

[112] S. Gollowitzer, B. Gendron, and I. Ljubić, *A cutting plane algorithm for the capacitated connected facility location problem*, Comput. Optim. Appl. **55** (2013), 647–674.

[113] S. Gollowitzer, L. Gouveia, and I. Ljubić, *Enhanced formulations and branch-and-cut for the two level network design problem with transition facilities*, Eur. J. Oper. Res. **225** (2013), 211 – 222.

[114] S. Gollowitzer and I. Ljubić, *MIP models for connected facility location: A theoretical and computational study*, Comput. Oper. Res. **38** (2011), 435 – 449.

[115] L. Gouveia, M. Leitner, and I. Ljubić, *Hop constrained Steiner trees with multiple root nodes*, Eur. J. Oper. Res. **236** (2014), 100 – 112.

[116] L. Gouveia, M. Leitner, and M. Ruthmair, *Layered graph approaches for combinatorial optimization problems*, Comput. Oper. Res. **102** (2019), 22–38.

[117] L. Gouveia, L. Simonetti, and E. Uchoa, *Modeling hop-constrained and diameter-constrained minimum spanning tree problems as Steiner tree problems over layered graphs*, Math. Program. **128** (2011), 123–148.

[118] L. Gouveia and J. Telhada, *The multi-weighted Steiner tree problem: A reformulation by intersection*, Comput. Oper. Res. **35** (2008), 3599–3611.

[119] C. Gröpl, S. Hougardy, T. Nierhoff, and H.J. Prömel, "*Approximation algorithms for the Steiner tree problem in graphs*," *Steiner Trees in Industry*, X.Z. Cheng and D.Z. Du (eds.), Springer, Boston, MA, 2001, pp. 235–279.

[120] M. Grötschel, C. Raack, and A. Werner, *Towards optimizing the deployment of optical access networks*, EURO J. Comput. Optim. **2** (2014), 17–53.

[121] S.L. Hakimi, *Steiner's problem in graphs and its implications*, Networks **1** (1971), 113–133.

[122] E. Halperin and R. Krauthgamer, *Polylogarithmic inapproximability*, Proceedings of the thirty-fifth annual ACM Symposium on Theory of Computing, 2003, pp. 585–594.

[123] M. Haouari, S.B. Layeb, and H.D. Sherali, *The prize collecting Steiner tree problem: Models and Lagrangian dual optimization approaches*, Comput. Optim. Appl. **40** (2008), 13–39.

[124] M. Haouari and J. Siala, *A hybrid Lagrangian genetic algorithm for the prize collecting Steiner tree problem*, Comput. Oper. Res. **33** (2006), 1274–1288.

[125] M. Hauptmann and M. Karpiński, *A compendium on Steiner tree problems*, Tech. report, Department of Computer Science and Hausdorff Center for Mathematics, University of Bonn, 2013.

[126] C. Hegde, P. Indyk, and L. Schmidt, 2014. A fast, adaptive variant of the Goemans-Williamson scheme for the prize-collecting Steiner tree problem. 11th DIMACS Implementation Challenge Workshop. Available at `http://people.csail.mit.edu/ludwigs/papers/dimacs14_fastpcst.pdf`. Accessed: 2017-06-15.

[127] C. Hegde, P. Indyk, and L. Schmidt, *A nearly-linear time framework for graph-structured sparsity*, Proceedings of the 32nd International Conference on Machine Learning (ICML-15), 2015, pp. 928–937.

[128] D.S. Hochbaum and A. Pathria, *Node-optimal connected k-subgraphs*, Tech. report, UC Berkeley, 1994.

[129] S. Hougardy, J. Silvanus, and J. Vygen, *Dijkstra meets Steiner: a fast exact goal-oriented Steiner tree algorithm*, Math. Program. Computation **9** (2017), 135–202.

[130] S.s.C. Huang, A constraint optimization framework for discovery of cellular signaling and regulatory networks, Ph.D. thesis, Massachusetts Institute of Technology, 2011.

[131] S.s.C. Huang and E. Fraenkel, *Integrating proteomic, transcriptional, and interactome data reveals hidden components of signaling and regulatory networks*, Sci. Signal. **2** (2009), ra40–ra40.

[132] T. Hume, Practical and theoretical approaches for module analysis of protein-protein interaction networks, Ph.D. thesis, École Doctorale de Mathématiques et Informatique de Bordeaux, 2016.

[133] N.V. Huy and N.D. Nghia, *Solving graphical Steiner tree problem using parallel genetic algorithm*, 2008 IEEE International Conference on Research, Innovation and Vision for the Future in Computing and Communication Technologies, 2008, pp. 29–35.

[134] F. Hwang and D.S. Richards, *Steiner tree problems*, Networks **22** (1992), 55–89.

[135] F.K. Hwang, D.S. Richards, and P. Winter, *The Steiner Tree Problem*, *Annals of Discrete Mathematics*, *Volume 53* Vol. 53, North Holland, 1992.

[136] T. Ideker, O. Ozier, B. Schwikowski, and A.F. Siegel, *Discovering regulatory and signalling circuits in molecular interaction networks*, Bioinformatics **18** (2002), S233–S240.

[137] Y. Iwata and T. Shigemura, *Separator-based pruned dynamic programming for steiner tree*, The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019, AAAI Press, 2019, pp. 1520–1527.

[138] D.S. Johnson, M. Minkoff, and S. Phillips, *The prize collecting Steiner tree problem: theory and practice*, Symposium on Discrete Algorithms, ACM/SIAM, 2000, pp. 760–769.

[139] D. Juhl, D.M. Warme, P. Winter, and M. Zachariasen, *The GeoSteiner software package for computing Steiner trees in the plane: an updated computational study*, Math. Program. Computation **10** (2018), 487–532.

[140] A. Kapsalis, V.J. Rayward-Smith, and G.D. Smith, *Solving the graphical Steiner tree problem using genetic algorithms*, J. Oper. Res. Soc. **44** (1993), 397–406.

[141] R.M. Karp, "*Reducibility among combinatorial problems*," *Complexity of Computer Computations*, Springer US, Boston, MA, 1972, pp. 85–103.

[142] M. Karpinski and A. Zelikovsky, *New approximation algorithms for the Steiner tree problems*, J. Combinatorial Optim. **1** (1997), 47–65.

[143] G.W. Klau, I. Ljubić, P. Mutzel, U. Pferschy, and R. Weiskircher, *The fractional prize-collecting Steiner tree problem on trees: Extended abstract*, Algorithms - ESA 2003, 11th Annual European Symposium, Budapest, Hungary, September 16-19, 2003, Proceedings, Vol. 2832 of *Lecture Notes in Computer Science*, Springer, 2003, pp. 691–702.

[144] F. Klimm, E.M. Toledo, T. Monfeuga, F. Zhang, C.M. Deane, and G. Reinert, *Functional module detection through integration of single-cell RNA sequencing data with protein-protein interaction networks*, BMC Genomics **21** (2020), 756.

[145] T. Koch and A. Martin, *Solving Steiner tree problems in graphs to optimality*, Networks **32** (1998), 207–232.

[146] T. Koch, A. Martin, and S. Voß, "*Steinlib: An updated library on steiner tree problems in graphs*," *Steiner Trees in Industry*, X.Z. Cheng and D.Z. Du (eds.), Springer, Boston, MA, 2001, pp. 285–325.

[147] J. Könemann, D. Pritchard, and K. Tan, *A partition-based relaxation for Steiner trees*, Math. Program. **127** (2011), 345–370.

[148] M. Leitner, *Layered graph models and exact algorithms for the generalized hop-constrained minimum spanning tree problem*, Comput. Oper. Res. **65** (2016), 1 – 18.

[149] M. Leitner, I. Ljubić, J.J.S. González, and M. Sinnl, *The connected facility location polytope*, Discr. Appl. Math. **234** (2018), 151–167.

[150] M. Leitner, I. Ljubić, M. Luipersbeck, M. Prossegger, and M. Resch, *New real-world instances for the Steiner tree problem in graphs*, Tech. report, University of Vienna, Austria, 2014.

[151] M. Leitner, I. Ljubić, M. Luipersbeck, and M. Resch, *A partition-based heuristic for the Steiner tree problem in large graphs*, Hybrid Metaheuristics - Proceedings, Vol. 8457 of *Lecture Notes in Computer Science*, 2014, pp. 56–70.

[152] M. Leitner, I. Ljubić, M. Luipersbeck, and M. Sinnl, *Decomposition methods for the two-stage stochastic Steiner tree problem*, Comput. Optim. Appl. **69** (2018), 713–752.

[153] M. Leitner, I. Ljubić, M. Luipersbeck, and M. Sinnl, *A dual-ascent based branch-and-bound framework for the prize-collecting Steiner tree and related problems*, INFORMS J. Comput. **30** (2018), 402–420.

[154] M. Leitner, I. Ljubić, J.J. Salazar-González, and M. Sinnl, *An algorithmic framework for the exact solution of tree-star problems*, Eur. J. Oper. Res. **261** (2017), 54 – 66.

[155] M. Leitner, I. Ljubić, M. Sinnl, and A. Werner, *ILP heuristics and a new exact method for bi-objective 0/1 ILPs: Application to FTTx-network design*, Comput. Oper. Res. **72** (2016), 128 – 146.

[156] T. Lengauer, *Combinatorial algorithms for integrated circuit layout*, Springer Science & Business Media, 2012.

[157] A.Y. Levin, *Algorithm for the shortest connection of a group of graph vertices*, Dokl. Akad. Nauk SSSR **200** (1971), 773–776.

[158] W. Liu, *A lower bound for the Steiner tree problem in directed graphs*, Networks **20** (1990), 765–778.

[159] I. Ljubić, Exact and memetic algorithms for two network design problems, Ph.D. thesis, Vienna University of Technology, 2004.

[160] I. Ljubić and S. Gollowitzer, *Layered graph approaches to the hop constrained connected facility location problem*, INFORMS J. Computi **25** (2013), 256–270.

[161] I. Ljubić, P. Putz, and J.J.S. González, *A MIP-based approach to solve the prize-collecting local access network design problem*, Eur. J. Oper. Res. **235** (2014), 727–739.

[162] I. Ljubić, R. Weiskircher, U. Pferschy, G.W. Klau, P. Mutzel, and M. Fischetti, *An algorithmic framework for the exact solution of the prize-collecting Steiner tree problem*, Math. Program. **105** (2006), 427–449.

[163] A. Lucena, *Tight bounds for the Steiner problem in graphs*, Preprint, IRC Process Syst. Eng., Imperial College, Lond. (1993).

[164] A. Lucena, *Non delayed relax-and-cut algorithms*, Ann. Oper. Res. **140** (2005), 375–410.

[165] A. Lucena and J. Beasley, *A branch and cut algorithm for the Steiner problem in graphs*, Networks **31** (1998), 39–59.

[166] A. Lucena, N. Maculan, and L. Simonetti, *Reformulations and solution algorithms for the maximum leaf spanning tree problem*, Comput. Manage. Sci. **7** (2010), 289–311.

[167] N. Maculan, *The Steiner problem in graphs*, Ann. Discr. Math. **31** (1987), 185–211.

[168] T.L. Magnanti and L.A. Wolsey, "*Optimal trees*," *Network Models*, Elsevier, Amsterdam, 1995, pp. 503 – 615.

[169] F. Margot, A. Prodon, and T.M. Liebling, *Tree polytope on 2-trees*, Math. Program. **63** (1994), 183–191.

[170] K. Mehlhorn, *A faster approximation algorithm for the Steiner problem in graphs*, Informat. Process. Lett. **27** (1988), 125–128.

[171] V. Morais, B. Gendron, and G.R. Mateus, *The p-arborescence star problem: Formulations and exact solution approaches*, Comput. Oper. Res. **102** (2019), 91 – 101.

[172] J. Nederlof, *Fast polynomial-space algorithms using Möbius inversion: Improving on Steiner tree and related problems*, Automata, Languages and Programming, 36th International Colloquium, ICALP 2009, Rhodes, Greece, July 5-12, 2009, Proceedings, Part I, Vol. 5555 of *Lecture Notes in Computer Science*, Springer, 2009, pp. 713–725.

[173] R. Novak, J. Rugelj, and G. Kandus, *A note on distributed multicast routing in point-to-point networks*, Comput. Oper. Res. **28** (2001), 1149 – 1164.

[174] C.A. Oliveira and P.M. Pardalos, *A survey of combinatorial optimization problems in multicast routing*, Comput. Oper. Res. **32** (2005), 1953 – 1981.

[175] H. Önal and R.A. Briers, *Optimal selection of a connected reserve network*, Oper. Res. **54** (2006), 379–388.

[176] H. Önal and Y. Wang, *A graph theory approach for designing conservation reserve networks with minimal fragmentation*, Networks **51** (2008), 142–152.

[177] T. Pajor, E. Uchoa, and R.F. Werneck, *A robust and scalable algorithm for the Steiner problem in graphs*, Math. Program. Computation **10** (2018), 69–118.

[178] M. Poggi de Aragão, E. Uchoa, and R.F. Werneck, *Dual heuristics on the exact solution of large Steiner problems*, Electron. Notes Discr. Math. **7** (2001), 150–153.

[179] M. Poggi de Aragão and R.F. Werneck, *On the implementation of MST-based heuristics for the Steiner problem in graphs*, ALENEX, Vol. 2409 of *Lecture Notes in Computer Science*, Springer, 2002, pp. 1–15.

[180] T. Polzin, Algorithms for the Steiner problem in networks., Ph.D. thesis, Saarland University, 2003.

[181] T. Polzin and S. Vahdati Daneshmand, "*Primal-dual approaches to the Steiner problem*," *Approximation Algorithms for Combinatorial Optimization*, K. Jansen and S. Khuller (eds.), Springer, Boston, MA, 2000, pp. 214–225.

[182] T. Polzin and S. Vahdati Daneshmand, *A comparison of Steiner tree relaxations*, Discr. Appl. Math. **112** (2001), 241–261.

[183] T. Polzin and S. Vahdati Daneshmand, *Improved algorithms for the Steiner problem in networks*, Discr. Appl. Math. **112** (2001), 263–300.

[184] T. Polzin and S. Vahdati Daneshmand, *Extending reduction techniques for the Steiner tree problem*, Algorithms - ESA 2002, 10th Annual European Symposium, Rome, Italy, September 17-21, 2002, Proceedings, Vol. 2461 of *Lecture Notes in Computer Science*, Springer, 2002, pp. 795–807.

[185] T. Polzin and S. Vahdati Daneshmand, *On Steiner trees and minimum spanning trees in hypergraphs*, Oper. Res. Lett. **31** (2003), 12–20.

[186] T. Polzin and S. Vahdati Daneshmand, *Practical partitioning-based methods for the Steiner problem*, Experimental Algorithms, 5th International Workshop, WEA 2006, Cala Galdana, Menorca, Spain, May 24-27, 2006, Proceedings, Vol. 4007 of *Lecture Notes in Computer Science*, Springer, 2006, pp. 241–252.

[187] T. Polzin and S. Vahdati Daneshmand, "*Approaches to the Steiner problem in networks*," *Algorithmics of Large and Complex Networks*, Springer, Boston, MA, 2009, pp. 81–103.

[188] T. Polzin and S. Vahdati Daneshmand, *The Steiner tree challenge: An updated study*, Tech. report, Theoretische Informatik, Universität Mannheim, Germany, 2014, http://dimacs11.zib.de/papers/PolzinVahdatiDIMACS.pdf.

[189] G.L. Presti, G.L. Re, P. Storniolo, and A. Urso, *A grid enabled parallel hybrid genetic algorithm for spn*, Computational Science - ICCS 2004, Springer Berlin Heidelberg, Berlin, Heidelberg, 2004, pp. 156–163.

[190] A. Prodon, S. De Negre, and T. Liebling, *Locating leak detecting sensors in a water distribution network by solving prize-collecting Steiner arborescence problems*, Math. Program. **Series B** (2010), 119–141.

[191] H.J. Prömel and A. Steger, *A new approximation algorithm for the Steiner tree problem with performance ratio* $5/3$, J. Algorithms **36** (2000), 89–101.

[192] H.J. Prömel and A. Steger, *The Steiner tree problem: a tour through graphs, algorithms, and complexity*, Vieweg+Teubner Verlag, Braunschweig/Wiesbaden, Germany, 2002.

[193] R. Qu, Y. Xu, J.P. Castro, and D. Landa-Silva, *Particle swarm optimization for the Steiner tree in graph and delay-constrained multicast routing problems*, J. Heuristics **19** (2013), 317–342.

[194] S. Raghavan, Formulations and algorithms for network design problems with connectivity requirements, Ph.D. thesis, Massachusetts Institute of Technology, 1995.

[195] S. Ramanathan, *Multicast tree generation in networks with asymmetric links*, IEEE/ACM Trans. Networking (TON) **4** (1996), 558–568.

[196] D. Rehfeldt, A generic approach to solving the Steiner tree problem and variants, Master's thesis, Technische Universitat Berlin, Germany, 2015.

[197] D. Rehfeldt and T. Koch, *Transformations for the prize-collecting Steiner tree problem and the maximum-weight connected subgraph problem to SAP*, Tech. report 16–36, ZIB, Berlin, Germany, 2016.

[198] D. Rehfeldt and T. Koch, *Combining NP-Hard Reduction Techniques and Strong Heuristics in an Exact Algorithm for the Maximum-Weight Connected Subgraph Problem*, SIAM J. Optim. **29** (2019), 369–398.

[199] D. Rehfeldt, T. Koch, and S.J. Maher, *Reduction techniques for the prize collecting Steiner tree problem and the maximum-weight connected subgraph problem*, Networks **73** (2019), 206–233.

[200] M.G. Resende and P.M. Pardalos, *Handbook of optimization in telecommunications*, Springer Science & Business Media, 2008.

[201] M.G. Resende and C.C. Ribeiro, "*GRASP with path-relinking*," *Optimization by GRASP*, Springer New York, New York, NY, 2016, pp. 189–204.

[202] C.C. Ribeiro and M.C. De Souza, *Tabu search for the Steiner problem in graphs*, Networks **36** (2000), 138–146.

[203] C.C. Ribeiro, E. Uchoa, and R.F. Werneck, *A hybrid GRASP with perturbations for the Steiner problem in graphs*, INFORMS J. Comput. **14** (2002), 228–246.

[204] G. Robins and A. Zelikovsky, *Tighter bounds for graph Steiner tree approximation*, SIAM J. Discr. Math. **19** (2005), 122–134.

[205] M.B. Rosenwein and R.T. Wong, *A constrained Steiner tree problem*, Eur. J. Oper. Res. **81** (1995), 430–439.

[206] I. Rosseti, M.P. De Aragão, C.C. Ribeiro, E. Uchoa, and R.F. Werneck, "*New benchmark instances for the Steiner problem in graphs*," *Metaheuristics: Computer Decision-Making*, Springer, Boston, MA, 2003, pp. 601–614.

[207] O. Russakovsky and A.Y. Ng, *A Steiner tree approach to efficient object detection*, 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010, pp. 1070–1077.

[208] A. Sadeghi and H. Fröhlich, *Steiner tree methods for optimal sub-network identification: an empirical study*, BMC Bioinformati **14** (2013), 144.

[209] J.J. Salazar, *A note on the generalized Steiner tree polytope*, Discr. Appl. Math. **100** (2000), 137–144.

[210] M. Santos, L.M. Drummond, and E. Uchoa, *A distributed dual ascent algorithm for the hop-constrained Steiner tree problem*, Oper. Res. Lett. **38** (2010), 57–62.

[211] L. Schmidt, C. Hegde, P. Indyk, L. Lu, X. Chi, and D. Hohl, *Seismic feature extraction using Steiner tree methods*, 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2015, pp. 1647–1651.

[212] M.S. Scott, T. Perkins, S. Bunnell, F. Pepin, D.Y. Thomas, and M. Hallett, *Identifying regulatory subnetworks for a set of genes*, Molecular Cellular Proteomics **4** (2005), 683–692.

[213] A. Segev, *The node-weighted Steiner tree problem*, Networks **17** (1987), 1–17.

[214] Y. Shinano, D. Rehfeldt, and T. Koch, *Building optimal Steiner trees on supercomputers by using up to 43,000 cores*, Integration of Constraint Programming, Artificial Intelligence, and Operations Research, Springer International Publishing, 2019, pp. 529–539.

[215] M. Siebert, S. Ahmed, and G. Nemhauser, *A linear programming based approach to the Steiner tree problem with a fixed number of terminals*, Networks **75** (2020), 124–136.

[216] L. Simonetti, A.S. Da Cunha, and A. Lucena, *Polyhedral results and a branch-and-cut algorithm for the k-cardinality tree problem*, Math. Program. **142** (2013), 511–538.

[217] M. Sinnl and I. Ljubić, *A node-based layered graph approach for the Steiner tree problem with revenues, budget and hop-constraints*, Math. Program. Computation **8** (2016), 461–490.

[218] H. Takahashi and A. Matsuyama, *An approximate solution for the Steiner problem in graphs*, Mathematica Japonica **24** (1980), 573–577.

[219] A. Tomazic and I. Ljubić, *A GRASP algorithm for the connected facility location problem*, 2008 International Symposium on Applications and the Internet, 2008, pp. 257–260.

[220] N. Tuncbag, A. Braunstein, A. Pagnani, S.S.C. Huang, J. Chayes, C. Borgs, R. Zecchina, and E. Fraenkel, *Simultaneous reconstruction of multiple signaling pathways via the prize-collecting Steiner forest problem*, J. Comput. Biol. **20** (2013), 124–136.

[221] N. Tuncbag, S.J. Gosline, A. Kedaigle, A.R. Soltis, A. Gitter, and E. Fraenkel, *Network-based interpretation of diverse high-throughput datasets through the omics integrator software package*, PLoS Comput. Biol. **12** (2016), e1004879.

[222] N. Tuncbag, S. McCallum, S.S.C. Huang, and E. Fraenkel, *Steinernet: A web server for integrating 'omic'data to discover hidden components of response pathways*, Nucleic Acids Res. (2012), gks445.

[223] E. Uchoa, Algoritmos para problemas de Steiner com aplicações em projeto de circuitos VLSI, Ph.D. thesis, Catholic University of Rio de Janeiro, 2001.

[224] E. Uchoa, *Reduction tests for the prize-collecting Steiner problem*, Oper. Res. Lett. **34** (2006), 437–444.

[225] E. Uchoa, M. Poggi de Aragão, and C.C. Ribeiro, *Preprocessing Steiner problems from VLSI layout*, Networks **40** (2002), 38–50.

[226] E. Uchoa and R.F. Werneck, *Fast local search for Steiner trees in graphs*, Proceedings of the Twelfth Workshop on Algorithm Engineering and Experiments, ALENEX 2010, Austin, Texas, USA, January 16, 2010, SIAM, 2010, pp. 1–10.

[227] E. Uchoa and R.F. Werneck, *Fast local search for the Steiner problem in graphs*, ACM J. Experimental Algorithmi **17** (2012).

[228] S. Vahdati Daneshmand, Algorithmic approaches to the Steiner problem in networks, Ph.D. thesis, University of Mannheim, Germany, 2003.

[229] M. Verhoeven and M. Severens, *Parallel local search for Steiner trees in graphs*, Ann. Oper. Res. **90** (1999), 185–202.

[230] S. Vijayanarasimhan and K. Grauman, *Efficient region search for object detection*, Proceedings of IEEE Conference on Computer Vision and Pattern Recognition 2011, IEEE, 2011, pp. 1401–1408.

[231] S. Voß, *Steiner's problem in graphs: heuristic methods*, Discr. Appl. Math. **40** (1992), 45 – 72.

[232] S. Voß, "*Steiner tree problems in telecommunications*," *Handbook of Optimization in Telecommunications*, M.G.C. Resende and P.M. Pardalos (eds.), Springer, Boston, MA, 2006, pp. 459–492.

[233] J. Vygen, *Faster algorithm for optimum Steiner trees*, Informat. Process. Lett. **111** (2011), 1075–1079.

[234] D.M. Warme, Spanning trees in hypergraphs with applications to Steiner trees, Ph.D. thesis, University of Virginia, 1998.

[235] P. Winter, *Steiner problem in networks: a survey*, Networks **17** (1987), 129–167.

[236] P. Winter, *Reductions for the rectilinear Steiner tree problem*, Networks **26** (1995), 187–198.

[237] P. Winter and M. Zachariasen, *Euclidean Steiner minimum trees: An improved exact algorithm*, Networks **30** (1997), 149–166.

[238] L.A. Wolsey, *Integer Programming* Vol. 42, Wiley-Interscience, 1998.

[239] D. Won, Optimization and machine learning frameworks for complex network analysis, Ph.D. thesis, University of Washington, 2016.

[240] R.T. Wong, *A dual ascent approach for Steiner tree problems on a directed graph*, Math. Program. **28** (1984), 271–287.

[241] X. Wu, Z. Lü, and P. Galinier, *Restricted swap-based neighborhood search for the minimum connected dominating set problem*, Networks **69** (2017), 222–236.

[242] T. Yamamoto, H. Bannai, M. Nagasaki, and S. Miyano, *Better decomposition heuristics for the maximum-weight connected graph problem using betweenness centrality*, Discovery Science, Vol. 5808 of *LNCS*, 2009, pp. 465–472.

[243] N. Yosef, L. Ungar, E. Zalckvar, A. Kimchi, M. Kupiec, E. Ruppin, and R. Sharan, *Toward accurate reconstruction of functional protein networks*, Molecular Syst. Biol. **5** (2009), 248.

[244] M. Zachariasen and A. Rohe, *Rectilinear group Steiner trees and applications in VLSI design*, Math. Program. **94** (2003), 407–433.

[245] A.Z. Zelikovsky, *An 11/6-approximation algorithm for the network Steiner problem*, Algorithmica **9** (1993), 463–470.

[246] A.Z. Zelikovsky, *A faster approximation algorithm for the Steiner tree problem in graphs*, Informat. Process. Lett. **46** (1993), 79 – 83.