Algorithms for the power-p Steiner tree problem in the Euclidean plane

Christina Burt, Alysson Costa, Charl Ras

September 15, 2015

Abstract

We study the problem of constructing minimum power-p Euclidean k-Steiner trees in the plane. The problem is to find a tree of minimum cost spanning a set of given terminals where, as opposed to the minimum spanning tree problem, at most k additional nodes (Steiner points) may be introduced anywhere in the plane. The cost of an edge is its length to the power of p (where $p \geq 1$), and the cost of a network is the sum of all edge costs. We propose two heuristics: a "beaded" minimum spanning tree heuristic; and a heuristic which alternates between minimum spanning tree construction and a local fixed topology minimisation procedure for locating the Steiner points. We show that the performance ratio κ of the beaded-MST heuristic satisfies $\sqrt{3}^{p-1}(1+2^{1-p}) \leq \kappa \leq 3(2^{p-1})$. We then provide two mixed-integer nonlinear programming formulations for the problem, and extend several important geometric properties into valid inequalities. Finally, we combine the valid inequalities with warm-starting and preprocessing to obtain computational improvements for the $p = 2$ case.

1 Introduction

The geometric Steiner tree problem involves the fundamental task of interconnecting a given set of points in the cheapest possible way. Additional points, called *Steiner points*, may be introduced anywhere in the ambient space, which makes the model significantly more general than the well-known minimum spanning tree problem.

Steiner trees are the flagship model for many network-design applications. Since the 1960's, researchers from around the world have applied a concerted effort in order to produce ever faster algorithms for constructing optimal Steiner trees [10, 15, 16, 26]. Every incremental improvement in algorithmic speed, efficiency or stability has had a tangible impact on industry—a case in point being the design of integrated circuits, where algorithmic developments have contributed to the exponential growth in the complexity-to-size ratio of microprocessors [7].

Steiner trees were originally employed as a model for designing interstate telecommunication networks [19]. Since then, they have been successfully applied to the modelling of phylogenetic trees [25], wireless sensor networks [17], underground mining networks [3], and optical fibre networks [11]. Consequently, there exists a collaborative drive within academia to produce algorithms that are faster, can handle more general definitions of network cost, and are numerically stable.

The fastest exact geometric Steiner tree algorithm to date, namely GeoSteiner [24], can comfortably handle networks with a few thousand nodes, but suffers from a significant shortcoming through the assumption that the number of Steiner points is unbounded. This premise leads to simpler geometry, but at the cost of limiting the range of applications that can utilise the algorithm. Moreover, even though in practice the cost of a network can be measured in many different ways, most Steiner tree algorithms, including GeoSteiner, assume that the network cost is calculated by adding up the lengths of the edges. Alternative algorithms—utilising alternative cost functions—have been developed (eg. [5, 23]) in order to address this shortcoming, however these algorithms are not nearly as fast or as accurate as GeoSteiner, and some may be difficult to implement directly.

In practice, Steiner points correspond to network junctions, which usually come at a cost; for instance, the set-up and maintenance costs associated with hub installation. More realistic models therefore either incorporate node costs or bound the number of Steiner points explicitly. In both cases, much of the elegant geometry that GeoSteiner exploits (such as the fact that all angles at Steiner points are 120°) is lost. The construction of Steiner trees with a bounded number of Steiner points therefore calls for a new approach. In this paper, we explore solutions to these issues by explicitly bounding the number of Steiner points, and extending the notion of network cost to incorporate any power-p, $p \geq 1$, of Euclidean distance.

We formally define the problem as follows. Let T be any tree connecting a set of points in the plane. For any $p \geq 1$ the *cost* of T is defined as

$$\|T\| = \sum_{xy \in E(T)} \|x - y\|^p,$$

where $E(T)$ is the edge-set of T. Now let Y be a set of n given points, called *terminals*, in the Euclidean plane and let $k \geq 0$ be a given integer. In the *power-p Euclidean k-Steiner tree problem* we are required to find a set $S \subset \mathbb{R}^2$ of cardinality at most k and a tree $T(S)$ spanning $Y \cup S$ such that $\|T(S)\|$ is minimised. We refer to an optimal $T(S)$ as a (p, k)-*Steiner minimal tree*, or just (p, k)-SMT. Note that classical Euclidean Steiner trees, where the degree of every Steiner point is 3, have at most $n-2$ Steiner points. Therefore the power-p Euclidean k-Steiner tree problem generalises the classical Euclidean Steiner tree problem (set $p = 1$ and $k = n - 2$). Throughout this paper we will denote the set of given terminals by $Y = \{y_1, ..., y_n\}$ and the set of Steiner points by $S = \{y_{n+1}, ..., y_{n+k}\}$.

The key contributions of this paper are the design of a fast and accurate new heuristic; a theoretical performance analysis of the beaded minimum spanning tree heuristic (which has previously been applied to the bottleneck Steiner tree problem [23]); and the proposal of two non-linear mixed-integer formulations which are strengthened by valid inequalities derived from geometric properties. Most of these properties relate to the connection between minimum spanning trees and certain proximity structures, such as Voronoi diagrams, Gabriel graphs and relative neighbourhood graphs. We also present a highly effective preprocessing routine that is able to significantly reduce the number of variables in the formulations. Finally, we present extensive computational experiments for our models and algorithms with $p = 2$.

In Section 2.1, we show that the beaded-MST heuristic has a performance ratio of at least $\sqrt{3}^{p-1}(1 + 2^{1-p})$ and at most $3(2^{p-1})$. This immediately leads to a lower-bound for the cost of a (p, k)-SMT on Y: we construct the beaded-MST on Y and then set the lower bound to $\frac{2^{1-p}}{3}\|T_{\text{bead}}\|$, where T_{bead} is the beaded-MST. Lower bounds are useful for estimating the quality of heuristics, and can be utilised in tree search algorithms to prune branches that will not lead to optimal solutions.

In our second heuristic (Section 2.2), we find locally minimal solutions to the power-p Steiner tree problem, where a locally minimal solution is defined as a tree T spanning Y and a set of k Steiner points S, such that T is an MST and is a cheapest tree (spanning Y and a set of k Steiner points) with the same topology as T. The heuristic finds locally minimal solutions by iteratively and randomly deploying Steiner points and then alternating between constructing an MST topology and locating the optimal Steiner point locations for that topology. Our experimental results for the case $p = 2$ show that this heuristic is very effective at finding the optimal solution, is fast, and can find good solutions for problems significantly larger than our exact approaches. We provide experimental results of both heuristics in Section 2.3.

In Section 3, we present two mixed-integer nonlinear programming (MINLP) models. We solve these models using MINLP and mixed-integer quadratically constrained (MIQCP) solvers for the case $p = 2$. In both models, convergence speed is improved with the use of warm-starts; preprocessing to eliminate $O(n^2)$ edge variables; and translations of several geometric properties into valid inequalities.

Since there are no alternative algorithms in the literature for geometric power-p Steiner trees for $p > 1$, our computational comparison in Section 4 focuses on our heuristics and variations on our integer programming formulations. We perform an empirical study on the effect that various improvements have on the efficiency of our approach. The algorithms we present are valid for all $p \geq 1$, however, in terms of implementation and comparison, we will focus on the case $p = 2$ only.

2 Heuristics

2.1 The beaded minimum spanning tree heuristic for $p > 1$

A *beaded-MST* [23] is a feasible solution to the power-p Euclidean k-Steiner tree problem for $p > 1$, constructed as follows. Let T' be a minimum spanning tree on the complete graph induced by the set of terminals Y. We add k degree-two Steiner points to T' by subdividing edges of T'. The first Steiner point is placed at the mid-point of a longest edge of T'. Now suppose that t Steiner points have been added to T'. To add the $t+1$th Steiner point we proceed as follows. For every edge e of the original MST on Y (note that e corresponds to a path through degree-2 Steiner points in T'), let $\ell(e) = \frac{\|e\|}{b(e)+1}$, where $b(e)$ is the number of Steiner points (*beads*) on edge e. Let $e' = \arg\max \ell(e)$. The $t+1$th Steiner point will be added to edge e'. We do this by removing all current Steiner points on the path of T' corresponding to e' and then adding them back, along with the new Steiner point, so that the Steiner points are equally spaced on e'.

Let $T_{\text{bead}}(Y)$ be a beaded-MST on Y and let $T(Y)$ be an optimal solution to the power-p Euclidean k-Steiner tree problem on Y. Let $\kappa = \sup_Y \frac{\|T_{\text{bead}}(Y)\|}{\|T(Y)\|}$. We require the following lemma.

Lemma 1 *For any non-negative real numbers c_1, c_2 it holds that $(c_1 + c_2)^p \leq 2^{p-1}(c_1^p + c_2^p)$.*

Proof. Since $p > 1$ the function $g(u) = u^p$ is convex when $u \geq 0$. Therefore for any $u_1, u_2 \geq 0$ it follows that $g\left(\frac{u_1+u_2}{2}\right) \leq \frac{1}{2}(g(u_1) + g(u_2))$. The result follows. ■

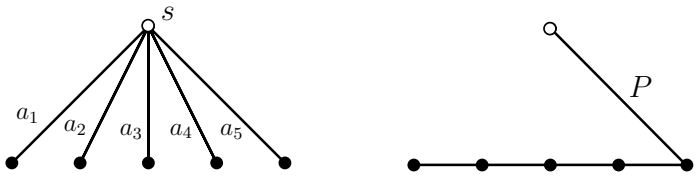

Figure 1: Base case for proof of Theorem 2

Theorem 2 $\kappa \leq 2^p + 2^{p-1} = 3 \cdot 2^{p-1}$.

Proof. We only need to consider full trees, where every terminal is of degree 1 and every Steiner point is of degree at least 2. Let T be an optimal full Steiner tree rooted at a Steiner point s. The *height* of T is the maximum number of edges in a path from s to a terminal. We employ induction on the height h of trees. In the base case with $h = 1$ the tree T has exactly one Steiner point

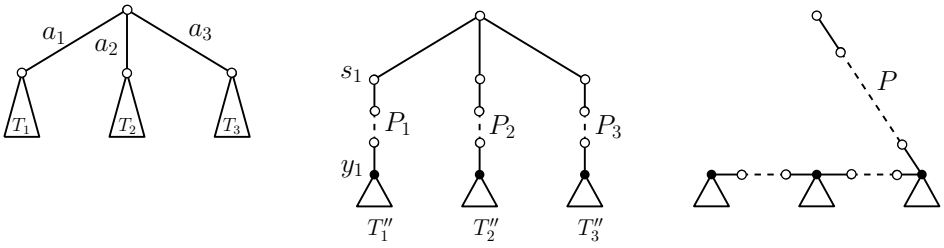

Figure 2: Inductive step for proof of Theorem 2

s (see Figure 1 where black filled circles are terminals and the open circle is a Steiner point). Let the edges of T be $a_1, ..., a_t$ joining s to vertices $y_1, ..., y_t$ respectively. We replace each edge a_i with edge $y_i y_{i+1}$, $1 \leq i < t$ and let T' be the resultant tree. Let T'' be the tree that results by removing edge sy_t from T'. Observe that T'' is a beaded spanning tree on the terminals of T, that $\|T\| = \|a_1\|^p + ... + \|a_t\|^p$, and that

$$\begin{aligned}
\|T''\| &= \|y_1y_2\|^p + ... + \|y_{t-1}y_t\|^p \\
&\leq (\|a_1\| + \|a_2\|)^p + ... + (\|a_{t-1}\| + \|a_t\|)^p \\
&\leq 2^{p-1}\|a_1\|^p + 2^p\|a_2\|^p + ... + 2^p\|a_{t-1}\|^p + 2^{p-1}\|a_t\|^p \qquad \text{(Lemma 1)} \\
&\leq 2^p\|T\| \\
&\leq (2^p + 2^{p-1})\|T\|.
\end{aligned}$$

Therefore the result follows in the base case. Note also that T' has a path $P := a_t$ with the property that $\|T''\| + 2^p\|P\| \leq (2^p + 2^{p-1})\|T\|$.

Suppose that for every optimal full tree $\widetilde{T}$ of height less than $h > 1$ the following property holds: there exists a tree $\widetilde{T}'$ spanning Y consisting of a beaded spanning tree $\widetilde{T}''$ and a path $\widetilde{P}$ connecting the root of $\widetilde{T}$ to one of its terminals such that $\|\widetilde{T}''\| + 2^p\|\widetilde{P}\| \leq (2^p + 2^{p-1})\|\widetilde{T}\|$. Suppose now that T is an optimal fulll Steiner tree of height $h > 1$ rooted at a Steiner point s with incident edges $a_1, ..., a_t$, and suppose that when we remove s we get t subtrees $T_1, ..., T_t$ rooted at Steiner points $s_1, ..., s_t$ respectively (see Figure 2 where, once again, black circles are terminals). Let $T_1', ..., T_t'$ be the respective subtrees guaranteed by the inductive hypothesis, where $T_i' = T_i'' \cup P_i$ and P_i is a path connecting s_i to terminal y_i in T_i. For each i let d_i' be the distance $\|s_i - y_i\|$. Without loss of generality we assume that $d_i' \geq d_{i+1}'$ for each $i < t$. For every $1 \leq i < t$ we now replace path P_i by edge $y_i y_{i+1}$ and place the Steiner points of P_i as beads on $y_i y_{i+1}$ (where the beads are spaced evenly). Let T'' be the resultant tree.

Note that since $d_i' \geq d_{i+1}'$ and by the triangle inequality, the cost of beaded edge $y_i y_{i+1}$ is at most $2^p\|P_i\| + (\|a_i\| + \|a_{i+1}\|)^p \leq 2^p\|P_i\| + 2^{p-1}(\|a_i\|^p + \|a_{i+1}\|^p)$. Path P_t and its Steiner points are unused and, together with edge a_t, forms the

new path P for T. Finally,

$$\begin{aligned}
\|T''\| + 2^p\|P\| &\leq \sum_{i\leq t}\|T_i''\| + 2^p\sum_{i<t}\|P_i\| + \sum_{i<t}2^{p-1}(\|a_i\|^p + \|a_{i+1}\|^p) + 2^p\|P\| \\
&\leq (2^p + 2^{p-1})\sum_{i<t}\|T_i\| + \|T_t''\| + \sum_{i<t}2^p\|a_i\|^p + 2^{p-1}\|a_t\|^p + 2^p(\|P_t\| + \|a_t\|^p) \\
&\leq (2^p + 2^{p-1})\sum_{i\leq t}\|T_i\| + (2^p + 2^{p-1})\sum_{i\leq t}\|a_i\|^p \\
&= (2^p + 2^{p-1})\|T\|,
\end{aligned}$$

where the second inequality follows from the inductive hypothesis. Therefore $\|T''\| \leq (2^p + 2^{p-1})\|T\|$, and we are done. ■

Corollary 3 *A lower bound on the cost of a (p,k)-SMT on Y is $\frac{2^{1-p}}{3}\|T_{\text{bead}}\|$.*

Lastly we will construct a lower bound for κ. Let Y be three terminals at the corners of an equilateral triangle of side length $\sqrt{3}$. It is easy to show that for $k = 1$ the optimal cost $\|T\| = 3$, since the Steiner point will be located at the centre of the triangle. On the other hand, $\|T_{\text{bead}}\| = \sqrt{3}^p + 2\left(\frac{\sqrt{3}}{2}\right)^p = \sqrt{3}^p\left(1 + 2^{1-p}\right)$. The next result now follows immediately:

Theorem 4 $\kappa \geq \sqrt{3}^{p-2}\left(1 + 2^{1-p}\right)$

2.2 The iterative alternating heuristic

The problem of finding the optimal locations of the Steiner points with respect to a topology is called the *fixed topology Steiner tree problem*. In this context, the topology refers to the connections between all combinations of terminal and Steiner nodes. Let $\mathcal{T}(S)$ be the topology of a tree spanning Y and a set S of k *variable* Steiner points. Let the edge-set of $\mathcal{T}$ be denoted by $E(\mathcal{T})$. Then the fixed topology problem for $\mathcal{T}$ solves the unconstrained problem

$$\min_S \|\mathcal{T}(S)\|$$

where $\|\mathcal{T}(S)\| := \sum_{e\in E(\mathcal{T})}\|e\|^p$. Since $p \geq 1$ and norms are convex functions it follows that $\|\mathcal{T}(S)\|$ is convex. Note that $\|\mathcal{T}(S)\|$ can be expressed as

$$\|\mathcal{T}(S)\| = \sum_{y_i,y_j\in Y\cup S} w_{ij}\|y_i - y_j\|^p, \tag{1}$$

where each w_{ij} is 0 if ij is not a connection in $\mathcal{T}(S)$, and 1 otherwise.

Solving the fixed topology problem when $p > 1$ can therefore be very efficiently achieved by Newton or gradient-based methods, since in this case $\|\mathcal{T}(S)\|$

is strictly convex. Efficient interior point methods [2] exist for solving the case $p = 1$. When $p = 2$ the fixed topology problem can be exactly solved in linear time by solving the system of linear equations that results from the first-order conditions on the optimal location of each Steiner point with respect to its neighbours. In particular, if Steiner point s is adjacent to nodes $y_1, ..., y_t$, for some $t > 1$, in an optimal tree then $\sum_{i=1}^{t} \|s - y_i\|^2$ must be a minimum. The first-order condition for s is therefore $\sum_{i=1}^{t}(s - y_i) = 0$ or $s = \frac{1}{t}\sum_{i=1}^{t} y_i$. In other words, s is located at the *centroid* of its neighbours.

The details of the heuristic are provided next.

Algorithm 1: Iterative Alternating Heuristic

input : Terminal set Y, number of Steiner points k, and number of iterations $c > 0$

output: A tree T_{IAH} spanning Y and at most k additional points

1 **while** $i < c$ **do**
2 Generate a set S' of k random points (from a uniform probability distribution) inside the convex hull of Y
3 **repeat**
4 Construct a minimum spanning tree on $Y \cup S'$, and denote its topology by $\mathcal{T}(S')$
5 Locate the optimal positions of the Steiner points with respect to $\mathcal{T}(S')$, resulting in an embedded tree T'
6 Let S' be the locations of the Steiner points of T'
7 **until** *the topology of* $\mathcal{T}(S')$ *no longer changes*
8 $i := i + 1$
9 Let T_{IAH} be the cheapest tree found

The integer c should be selected based on the values of n and k. We do not have a strong theoretical basis for how to choose c in order to achieve a required degree of accuracy. However, we claim that each Repeat loop in Line 3 finds a locally minimal solution to the problem, and it is possible to place a bound on the number of distinct locally minimal solutions. This insight may be used as a rough guide for choosing c.

Recall that a locally minimal solution is defined as a tree T spanning Y and a set of k Steiner points S, such that T is an MST and is a cheapest tree (spanning Y and a set of k Steiner points) with the same topology as T. Therefore, in a locally minimal solution, each Steiner point is optimally located with respect to its neighbours. To prove the claim, note that the Repeat loop terminates when the topology of $\mathcal{T}(S')$ does not change when an MST is calculated for the current Steiner point locations, S', at Line 4. Since S' is the optimal set of Steiner point locations for the topology of $\mathcal{T}(S')$ in the previous iteration of the loop, the set S' will also be optimal in the current iteration. Therefore both conditions for local minimality are satisfied. On the other hand, note that

if the set S' in the current iteration is different than in the previous iteration, then there must be a strict decrease in the cost of the embedded tree T'. But there are a finite number of distinct minimum spanning trees on the set Y with k Steiner points (there are at most $O(k^{k-2}n^{2k})$, see [4]), therefore the Repeat loop must terminate.

We have found that on average it requires very few iterations of the Repeat loop in order to find a locally minimal solution; see Table 1 in the next section.

2.3 Heuristic Experiments

In Table 1 we present the computational results for the two heuristics on a validation test set. Both heuristics can efficiently solve problems that, as we will see in Section 4, are too challenging for a mathematical programming approach. We have chosen $c = 100,000$ iterations for the iterative alternating heuristic, which gives a maximum run time of less than 350 seconds for each of the instances. We utilised the exact approach described in Section 3 to obtain a bound on the solution, thereby validating that the iterative-alternating heuristic finds the optimal solution for the first 8 instances. For the remaining instances, the exact approach did not converge on the optimal solution and therefore we were unable to validate the iterative-alternating heuristic on these instances.

The quality of the beaded-MST heuristic is also very good: in some cases it finds the optimal solution and it has a performance low of 13.2% worse than the iterative-alternating heuristic. In Section 4, we will compare the performance of the beaded-MST with the exact solutions found from our MINLP algorithm.

3 Exact approach

In this section, we begin by providing two mixed integer nonlinear programming models for the power-p Euclidean k-Steiner tree problem. We have two main models, both of which are directed formulations employing single commodity flow in order to ensure connectivity of the network. Our first model, Model N, has a nonlinear objective function with linear constraints, whereas our second model, Model Q, is a convex formulation with nonlinear constraints. We achieve conversion from N to Q by introducing bound variables d_{ij}, each of which is bounded below by the cost of edge $y_i y_j$, as was similarly performed in [9] for the classical Steiner tree problem.

We then present a simple yet powerful preprocessing step that can be employed in either Model N or Model Q. Lastly, in Section 3.3, we derive valid inequalities based on geometric properties of optimal k-Steiner trees.

3.1 Models

We define the index sets as follows. The terminal nodes have indices: $\mathcal{I}_Y := \{1, ..., n\}$; the Steiner nodes have indices $\mathcal{I}_S := \{n+1, ..., n+k\}$. Let $\mathcal{I} := \{1, ..., n+k\}$ represent the full set of terminal and Steiner node indices. In both of

instance		beaded-MST		iterative-alternating				best bound	sol ratio
$\|S\|$	$\|T\|$	obj	time	obj	time	global it	local it		$\frac{\text{beaded-MST}}{\text{iter-alter}}$
1	10	4793.054	0.004	**4651.949**	45.300	5	2.764	**4651.951**	1.030
1	12	5266.108	0.005	**5023.772**	44.660	16	2.677	**5023.772**	1.048
1	15	4993.727	0.008	**4863.787**	47.750	13	2.762	**4863.787**	1.027
1	20	**6064.220**	0.014	**6064.221**	45.650	11	2.380	**6064.220**	1.000
2	15	5156.965	0.007	**5132.455**	66.390	376	2.712	**5132.452**	1.005
2	20	2987.430	0.015	**2893.793**	84.240	289	3.308	**2893.792**	1.032
3	10	**5852.579**	0.004	**5852.579**	80.500	29	2.617	**5852.577**	1.000
3	12	5438.205	0.005	**5235.981**	90.200	175	2.979	**5235.978**	1.039
3	20	3787.999	0.013	3667.901	107.030	–	3.297	2938.586	1.033
4	15	4944.758	0.007	4772.472	114.410	–	3.038	3204.118	1.036
4	20	4062.861	0.015	4000.243	132.460	–	3.421	1902.062	1.016
5	10	3647.391	0.003	3362.766	150.670	–	3.440	644.604	1.085
5	12	2531.375	0.005	2505.406	150.220	–	3.434	460.447	1.010
5	20	4455.516	0.014	4324.814	153.810	–	3.393	659.754	1.030
6	12	3387.858	0.006	3244.279	178.400	–	3.574	0.000	1.044
6	15	4812.687	0.007	4777.163	158.840	–	3.126	0.000	1.007
7	10	2083.748	0.003	1841.416	228.150	–	3.930	0.000	1.132
8	12	3540.096	0.005	3300.603	233.390	–	3.650	0.000	1.073
8	15	3357.239	0.017	3020.171	237.290	–	3.686	0.000	1.112
10	10	1557.291	0.004	1557.291	324.400	–	4.247	0.000	1.000
10	12	2636.205	0.005	2484.384	305.780	–	3.975	0.000	1.061
10	15	3644.812	0.007	3427.932	284.780	–	3.617	0.000	1.063
12	12	2899.817	0.005	2760.644	345.390	–	3.751	0.000	1.050

Table 1: The computational results of the beaded-MST and iterative-alternating heuristic on 24 random instances. We provide the objective value and run-time (seconds) for the iterative-alternating heuristic for 100000 iterations, along with the average number of global iterations and sub-iterations (local) required to obtain the optimal solution over 10 experiments. We provide the best known solution from the exact approaches; optimal solutions are in bold, while the remaining solutions are best bounds obtained after 3600s. Where the iterative-alternating solution is validated by the exact approaches, it is in bold.

the models in this section, we utilise the following variables. Let nodes $y_1 \dots y_n$ be the position of terminal nodes and nodes $y_{n+1} \dots y_{n+k}$ be the position of Steiner nodes. We represent the known (fixed) position of a terminal node y_i by $\overline{y}_i$. The topology of the network is captured by binary variables w_{ij}, which indicate if edge $y_i y_j$ is active in the network, or not. We enforce a spanning tree topology through the use of a single-commodity flow formulation. Here, the flow variables will represent the quantity of flow from the arbitrarily chosen root node to the leaf nodes, where the flow leaving the root node is $n+k-1$ and the flow arriving into leaf nodes is 1. The flow along each edge, g_{ij}, will reduce by 1 at every node as the branch is traced to the leaf node. These flow variables are implicitly integer, and therefore can be defined as continuous variables. That is, even if they are defined as continuous variables, the solutions are guaranteed to be integer.

We obtain the following mixed-integer nonlinear formulation:

$$
\begin{aligned}
N: \quad \min \quad & \sum_{i,j \in \mathcal{I}} w_{ij} \, ||y_i - y_j||^p & & \\
\text{s.t.} \quad & \sum_{j \in \mathcal{I}} w_{ji} = 1, & i \in \mathcal{I}, i \neq 1, \quad & (2) \\
& \sum_{j \in \mathcal{I}} g_{ji} - \sum_{j \in I, j \neq 1} g_{ij} = 1, & i \in \mathcal{I}, i \neq 1, \quad & (3) \\
& g_{ij} \leq (n+k-1) w_{ij}, & i, j \in \mathcal{I}, \quad & (4) \\
& y_i = \overline{y}_i, & i \in \mathcal{I}_Y, \quad & (5) \\
& w_{ij} \in \{0,1\}, & i, j \in \mathcal{I}, & \\
& g_{ij} \in \mathbb{R}_{\geq 0}, & i, j \in \mathcal{I}, & \\
& y_i \in \mathbb{R}^2, & i \in \mathcal{I}. &
\end{aligned}
$$

Constraint (2) ensures that every node (except the root node y_1) is visited by one incoming arc. Constraints (3) and (4) are single-commodity flow constraints that ensure that every visited node must receive one unit of flow, thus eliminating subcycles. Constraint (5) fixes the positions of the known terminal nodes.

Note that the objective function is nonlinear and potentially non-convex. Therefore a global mixed-integer nonlinear programming solver such as eg., SCIP [1] should be employed. However, we will reformulate the problem into convex form (Model Q below) so that state-of-the-art MIQCP solvers can be used for the $p = 2$ case.

As in [9], we introduce a new variable, d_{ij} for each edge $y_i y_j$. Since distance is symmetric we only introduce these variables for indices $i < j$. Let $M = \max_{i,j \in \mathcal{I}_Y} \|y_i - y_j\|^p$ and let $w'_{ij} = w_{ij} + w_{ji}$ for all i, j. We therefore obtain the

following mixed-integer convex formulation:

$$
\begin{aligned}
Q: \qquad \min \quad & \sum_{i,j\in\mathcal{I},i<j} d_{ij} \\
\text{s.t.} \quad & \|y_i - y_j\|^p - (1 - w'_{ij})M \le d_{ij}, && i,j \in \mathcal{I}, i<j, && (6)\\
& \sum_{j\in\mathcal{I}} w_{ji} = 1, && i \in \mathcal{I}, i \neq 1, && (7)\\
& \sum_{j\in\mathcal{I}} g_{ji} - \sum_{j\in I, j\neq 1} g_{ij} = 1, && i \in \mathcal{I}, i \neq 1, && (8)\\
& g_{ij} \le (n+k-1)w_{ij}, && i,j \in \mathcal{I}, && (9)\\
& y_i = \overline{y}_i, && i \in \mathcal{I}_Y, && (10)\\
& w_{ij} \in \{0,1\}, && i,j \in \mathcal{I}, \\
& g_{ij} \in \mathbb{R}_{\ge 0}, && i,j \in \mathcal{I}, \\
& d_{ij} \in \mathbb{R}_{\ge 0}, && i,j \in \mathcal{I}, i<j, \\
& y_i \in \mathbb{R}^2, && i \in \mathcal{I}.
\end{aligned}
$$

Note that Model Q is an MIQCP for $p = 2$.

3.2 Preprocessing

Here we describe a method for significantly reducing the number of w_{ij} variables for implementations of the above models. Let T be a (p,k)-SMT on Y with optimal Steiner point set S. We assume without loss of generality that the MST on Y, say MST(Y), is unique, since otherwise a small perturbation in the location of the terminals can create a terminal-set with this property. An edge in T joining two terminals is called a *terminal edge*.

The following theorem follows directly from arguments in [4].

Theorem 5 *All terminal edges of T are edges of* MST(Y).

A consequence of this is that we can remove any y_iy_j edge for $i,j \in \mathcal{I}_Y$ that is not in the MST of Y. That is, the number of admissible terminal to terminal edges drops from $O(n^2)$ to $O(n)$. Let $\mathcal{A}_Y$ be the set of *admissable arcs*, defined as follows: $\mathcal{A}_Y := \{(i,j) : y_iy_j$ is an edge of MST(Y)$\}$.

3.3 Valid inequalities

An optimal k-Steiner tree is an MST on its entire set of nodes, and this fact leads to a number of geometric restrictions on optimal trees. In this section, we derive valid inequalities corresponding to these restrictions, which can be added to models N and Q with the aim of improved computational efficiency. Some of our inequalities are nonlinear and non-convex, and therefore we also derive "relaxed" versions of the inequalities which are linear.

Some basic valid inequalities for both the N and Q models include:

$$\sum_{i,j \in \mathcal{I}} w_{ij} = n + k - 1, \tag{11}$$

since there must be exactly $n + k - 1$ edges in a spanning tree on $Y \cup S$, and

$$w'_{ij} := w_{ij} + w_{ji} \leq 1, \quad i, j \in \mathcal{I}. \tag{12}$$

We will use the notation $v = (v[1], v[2])$ for any vector $v \in \mathbb{R}^2$. The following three inequalities follow from the fact that in an optimal k-Steiner tree all Steiner points lie in the convex hull of the set of terminals:

$$y_i[1] \leq \max_{j \in \mathcal{I}_Y} y_j[1], \quad i \in \mathcal{I}_S, \tag{13}$$

$$y_i[2] \leq \max_{j \in \mathcal{I}_Y} y_j[2], \quad i \in \mathcal{I}_S, \tag{14}$$

and

$$\|y_i - \bar{m}\|^2 \leq \max_{j \in \mathcal{I}_Y} \|y_j - \bar{m}\|^2, \quad i \in \mathcal{I}_S, \tag{15}$$

where $\bar{m}$ can be chosen as any point in the convex hull of the terminals. This inequality simply says that any Steiner point must be inside the disk centred at $\bar{m}$ with radius equal to the distance to the farthest terminal from $\bar{m}$. We will choose $\bar{m}$ to be the centroid of the terminal nodes.

We also implement a set of stronger (but more numerous) linear inequalities by adding an inequality directly for each edge of the convex hull of Y. (16)

Finally, we implement the following symmetry-breaking constraints:

$$y_i[2] \leq y_{i+1}[2] \qquad i \in \mathcal{I}_S \backslash \{n + k\}. \tag{17}$$

The remaining valid inequalities arise from geometric observations.

Angle Inequalities

A basic geometric property of planar Euclidean minimum spanning trees is that the angle between any pair of adjacent edges is at least 60°. We use this fact to create a new family of valid inequalities as follows. Consider any two nodes y_i, y_j, $i, j \in I$ and let y_s be a Steiner point. If y_i and y_j are adjacent to y_s in T then y_s lies in one of the two 60° circles passing through y_i and y_j (see the first subfigure of Figure 3).

Let r_{ij} be the radius of each of the two circles. Then $r_{ij} = \|y_i - y_j\|/\sqrt{3}$. The centres of the two circles, ρ_{ij1} and ρ_{ij2} can be calculated using basic trigonometry, and are constant when y_i and y_j are terminals. Using this information we create (nonlinear) valid inequalities as follows:

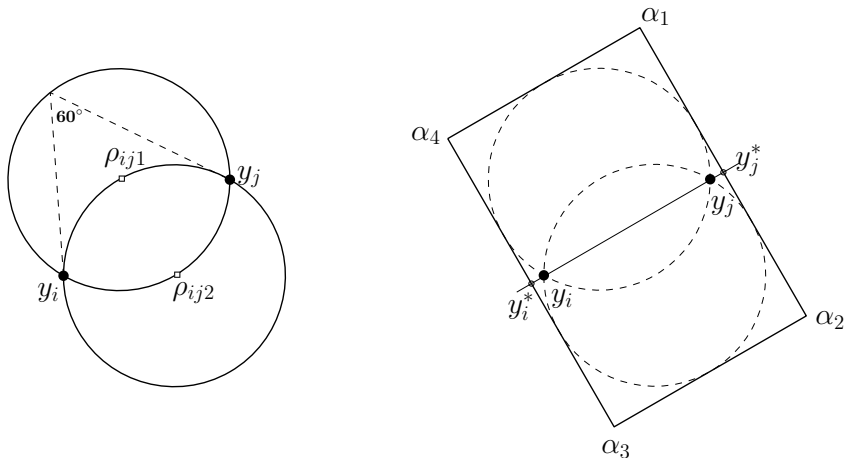

Figure 3: The Angle Inequalities

Let $z_{ijs} = 3 - \dfrac{\|y_s - \rho_{ij1}\|^p}{r_{ij}^2}$ and let $z'_{ijs} = 3 - \dfrac{\|y_s - \rho_{ij2}\|^p}{r_{ij}^2}$. If y_s is inside or on the boundary of the circle with centre ρ_{ij1} then $z_{ijs} \in [2, 3]$, otherwise $z_{ijs} \in (-\infty, 2)$. Similarly for z'_{ijs}. We therefore obtain the following valid inequalities.

$$w'_{is} + w'_{js} \leq \max\{z_{ijs}, z'_{ijs}, 1\} \qquad \forall\ i, j \in \mathcal{I}, s \in \mathcal{I}_S. \tag{18}$$

Constraint (18) cannot be implemented in the MIQCP solver, even for $p = 2$. However, it is possible to provide simpler (but more numerous) inequalities which are linear when y_i and y_j are terminals. Consider the bounding rectangle of the two 60° circles for nodes y_i, y_j, as depicted in the second subfigure of Figure 3. Let $\alpha_1, ..., \alpha_4$ be the four corners of the rectangle (the choice of these labels will depend on the orientation).

Let $q_{ij} = y_j - y_i$, let y_i^* and y_j^* be the intersection points of the line containing y_i, y_j and the bounding rectangle. We can then show that $\delta_{ij} := \|y_j^* - y_j\| = \|y_i^* - y_i\| = \left(\frac{2-\sqrt{3}}{2\sqrt{3}}\right)\|q_{ji}\|$. It is also easy to show that $\|\alpha_1 - y_j^*\| = \|\alpha_2 - y_j^*\| = \|\alpha_3 - y_i^*\| = \|\alpha_4 - y_i^*\| = \frac{\sqrt{3}}{2}\|q_{ij}\|$.

Let R be the 90° counterclockwise rotation matrix $R = [0, -1; 1, 0]$, in other words, $R(x, y) = (-y, x)$ for any vector (x, y). We then have $\alpha_1 = y_j + \delta_{ij}\frac{q_{ij}}{\|q_{ij}\|} + \frac{\sqrt{3}}{2}\|q_{ij}\|R\frac{q_{ij}}{\|q_{ij}\|} = y_j + \left(\frac{2-\sqrt{3}}{2\sqrt{3}}\right)q_{ij} + \frac{\sqrt{3}}{2}Rq_{ij}$. Similarly, $\alpha_2 = y_j + \left(\frac{2-\sqrt{3}}{2\sqrt{3}}\right)q_{ij} - \frac{\sqrt{3}}{2}Rq_{ij}$, $\alpha_3 = y_i + \left(\frac{2-\sqrt{3}}{2\sqrt{3}}\right)q_{ji} + \frac{\sqrt{3}}{2}Rq_{ji}$, $\alpha_4 = y_i + \left(\frac{2-\sqrt{3}}{2\sqrt{3}}\right)q_{ji} - \frac{\sqrt{3}}{2}Rq_{ji}$.

For any two points v_1, v_2 let $y = L(v_1, v_2, x)$ be the equation of the line passing through v_1, v_2, where x, y are variable scalars. In other words $L(v_1, v_2, x) =$

$(x - v_1[1])\left(\frac{v_2[2]-v_1[2]}{v_2[1]-v_1[1]}\right) + v_1[2]$. Now, if y_s is adjacent to both y_i and y_j then $w'_{is} + w'_{js} = 2$, otherwise $w'_{is} + w'_{js} < 2$. We set $M' = (2+\epsilon)\max_{i,j\in I} \|y_i - y_j\|^p$, where $\epsilon = 0.1$. Therefore, we obtain the following inequalities:

$$\begin{aligned}
&y_s[2] - L(\alpha_1, \alpha_2, y_s[1]) \leq -M'(w'_{is} + w'_{js} - 2) \quad i,j \in \mathcal{I}, s \in \mathcal{I}_S,\\
&y_s[2] - L(\alpha_1, \alpha_4, y_s[1]) \leq -M'(w'_{is} + w'_{js} - 2) \quad i,j \in \mathcal{I}, s \in \mathcal{I}_S,\\
&y_s[2] - L(\alpha_2, \alpha_3, y_s[1]) \geq M'(w'_{is} + w'_{js} - 2) \quad i,j \in \mathcal{I}, s \in \mathcal{I}_S,\\
&y_s[2] - L(\alpha_3, \alpha_4, y_s[1]) \geq M'(w'_{is} + w'_{js} - 2) \quad i,j \in \mathcal{I}, s \in \mathcal{I}_S.
\end{aligned} \tag{19}$$

While these inequalities are valid for all $i, j \in \mathcal{I}$, to ensure linearity we only implement them for $(i,j) \in \mathcal{I}_Y$.

Centroid Equalities for $p = 2$

For the case of $p = 2$, every Steiner point in an optimal tree is at the *centroid* (or centre of mass) of its neighbours. That is,

$$y_s = \frac{\sum_{i\in\mathcal{I}} w'_{is} y_i}{\sum_{i\in\mathcal{I}} w'_{is}} \qquad \text{for all } s \in \mathcal{I}_S.$$

Re-arranging, we obtain

$$y_s \sum_{i\in\mathcal{I}} w'_{is} = \sum_{\in\mathcal{I}} w'_{is} y_i \qquad \forall\ s \in \mathcal{I}_S. \tag{20}$$

Degree Inequalities

For any set of points in the plane there exists an MST where the maximum degree of any node is 5. Clearly the minimum degree of a Steiner point is 2 in an optimal tree. This leads to the following inequalities:

$$\sum_{i\in\mathcal{I}} w'_{ij} \leq 5 \qquad \forall\ j \in \mathcal{I}, \tag{21}$$

$$\sum_{i\in\mathcal{I}} w'_{is} \geq 2 \qquad \forall\ s \in \mathcal{I}_S, \tag{22}$$

$$\sum_{i\in\mathcal{I}} w'_{ij} \geq 1 \qquad \forall\ j \in \mathcal{I}_Y. \tag{23}$$

Gabriel Inequalities

Any MST is a subgraph of the *Gabriel graph* on the set of all nodes. The Gabriel graph G is defined as follows: edge $y_i y_j$ is in G if and only if there are no other

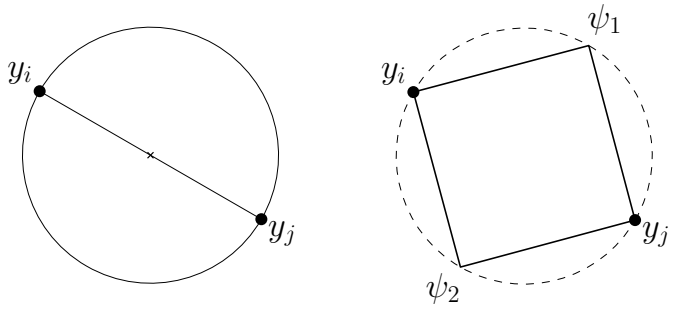

Figure 4: Gabriel Inequalities

nodes in or on the circle passing through y_i, y_j with diameter $\|y_i - y_j\|$ (see the first subfigure of Figure 4). If $y_i y_j$ is an edge of an MST and y_s is inside or on the circle then $y_i y_j$ can be replaced by one of the edges $y_i y_s$ or $y_j y_s$, giving a shorter MST, which is a contradiction. Therefore:

$$\|y_s - y_i\|^p + \|y_s - y_j\|^p > w'_{ij}\|y_i - y_j\|^p \qquad \forall\ i, j \in \mathcal{I}, i < j, s \in \mathcal{I}_S.$$

We write this as

$$\|y_s - y_i\|^p + \|y_s - y_j\|^p \geq \|y_i - y_j\|^p + \epsilon - (1 - w'_{ij})M \qquad \forall\ i, j \in \mathcal{I}, i < j, s \in \mathcal{I}_S,$$

where $M = \max\{\|y_i - y_j\|^p \text{ for all } i, j \in \mathcal{I}\}$ and ϵ is some arbitrarily small number.

We employ the d_{ij} variables in our Q model to obtain a simpler set of inequalities:

$$\|y_s - y_i\|^p + \|y_s - y_j\|^p \geq d_{ij} + \epsilon \qquad \forall\ i, j \in \mathcal{I}, i < j, s \in \mathcal{I}_S.$$

As for the Angle Inequalities, it is possible to create simpler inequalities which approximate the Gabriel circle by using a polygon. Once again, these inequalities are linear when y_i and y_j are terminals.

Consider the example from Figure 4. Then $\psi_1 = (y_i + y_j)/2 + \frac{1}{2}Rq_{ij}$ and $\psi_2 = (y_i + y_j)/2 - \frac{1}{2}Rq_{ij}$. Based on the orientation we use the labels $\alpha_1, ..., \alpha_4$ for the four corners of the square. In the example of the figure, we let $\alpha_1 := \psi_1$ and let $\alpha_2, \alpha_3, \alpha_4$ be the corners we meet in clockwise order.

Now, if edge $y_i y_j$ is in a feasible tree, i.e., if $w'_{ij} = 1$, then the region bounded by $\alpha_1, ..., \alpha_4$ must be empty. Therefore, if $w'_{ij} = 1$ then for every y_s we have:

$$\begin{aligned}
&y_s[2] > L(\alpha_1, \alpha_2, y_s[1]) \text{ or}\\
&y_s[2] < L(\alpha_2, \alpha_3, y_s[1]) \text{ or}\\
&y_s[2] < L(\alpha_3, \alpha_4, y_s[1]) \text{ or}\\
&y_s[2] > L(\alpha_4, \alpha_1, y_s[1]).
\end{aligned}$$

This can be expressed in the following way:

$$\begin{aligned}
\sum_{h=1}^{4} \beta_{ijhs} &= w'_{ij} && \forall\ i,j \in \mathcal{I}_Y, s \in \mathcal{I}_S, \\
y_s[2] - L(\alpha_1, \alpha_2, y_s[1]) &\geq -M'(1-\beta_{ij1}) + \epsilon && \forall\ i,j \in \mathcal{I}_Y, s \in \mathcal{I}_S, \\
L(\alpha_2, \alpha_3, y_s[1]) - y_s[2] &\geq -M'(1-\beta_{ij2}) + \epsilon && \forall\ i,j \in \mathcal{I}_Y, s \in \mathcal{I}_S, \\
L(\alpha_3, \alpha_4, y_s[1]) - y_s[2] &\geq -M'(1-\beta_{ij3}) + \epsilon && \forall\ i,j \in \mathcal{I}_Y, s \in \mathcal{I}_S, \\
y_s[2] - L(\alpha_4, \alpha_1, y_s[1]) &\geq -M'(1-\beta_{ij4}) + \epsilon && \forall\ i,j \in \mathcal{I}_Y, s \in \mathcal{I}_S,
\end{aligned} \tag{24}$$

and where $\beta_{ijhs} \in \{0,1\}$.

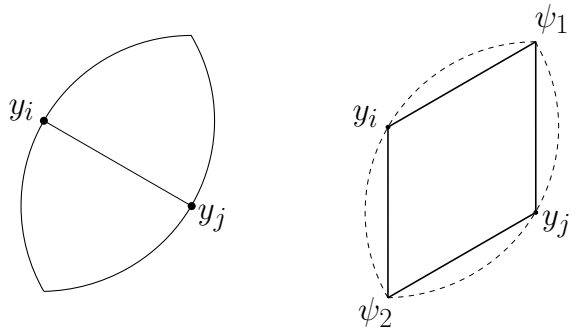

Figure 5: Lens inequalities

Lens Inequalities

Every MST is also a subgraph of the *Relative Neighbourhood graph* on its nodes, which, in turn, is a subgraph of the Gabriel graph. The Relative Neighbourhood graph (RNG) is defined as follows: edge y_iy_j is an edge of RNG if and only if there are no nodes inside or on the boundary of the *lens* specified by line segment y_iy_j. The lens is the intersection of the two disks centred at y_i and y_j respectively, both of radius $\|y_i - y_j\|$ (see Figure 5). This leads to the following inequalities:

$$\max\left\{\|y_s - y_i\|^p, \|y_s - y_j\|^p\right\} > w'_{ij}\|y_i - y_j\|^p \quad \forall\ i,j \in \mathcal{I}, i < j, s \in \mathcal{I}_S.$$

As with the Gabriel property, we can linearise these inequalities when y_i, y_j are terminals. This time we place a parallelogram inside the lens specified by y_i, y_j. In the example from Figure 5 we have $\psi_1 = (y_i + y_j)/2 + \frac{\sqrt{3}}{2}Rq_{ij}$ and $\psi_2 = (y_i + y_j)/2 - \frac{\sqrt{3}}{2}Rq_{ij}$. Based on the orientation of the lens we use the labels $\alpha_1, ..., \alpha_4$ for the extreme points of the parallelogram. In terms of the α_i this yields the same set of inequalities as for the Gabriel property.

4 Experiments

4.1 Validation Experiments

All experiments performed are for the case $p = 2$ only. We perform our validation experiments on a Dell Optiplex 990 with a 3.4 GHz Intel Core i7 CPU and 8 GB RAM. We implement model N in the Pyomo modelling tool (version 3.5 for python 2.7) [14], and solve it with the SCIP Optimisation Suite [1, 6, 22], version 3.1.0, using Ilog Cplex version 12.6 [8] as a linear programming solver, and Ipopt version 3.11.8 as a nonlinear programming solver. We implement model Q in the Gurobi python API, and solve it using Gurobi version 6.0.4 [12]. Since we are solving a quadratic integer program, we turn off presolve, as recommended by Gurobi. We do not use any other non-default parameters, preprocessing or valid inequalities in this test.

Name	H obj	N obj	Q obj	H time	N time	Q time
T4-2	283992.4	283992.4	283992.4	66.56	1.13	0.20
T5-3	239005.5	239005.5	239005.5	94.04	14.69	6.22
T6-3	241416.0	241416.0	241415.7	98.46	64.15	12.90
T8-3	298657.0	319462.3*	298657.0	84.33	T/O	341.45
T10-1	4652.0	4652.0	4652.0	45.30	4.33	7.68
T10-3	5852.6	6971.3*	5852.6*	80.50	T/O	T/O
T10-5	3362.7	8975.7*	3499.0*	150.67	T/O	T/O
T12-1	5023.7	5023.8	5023.8	44.66	10.97	21.36
T12-3	5236.0	6842.5*	5261.6*	90.20	T/O	T/O
T12-5	2505.4	5198.6*	2768.0*	150.22	T/O	T/O

Table 2: The objective value and run time (seconds) for validation test set for both exact approaches and the iterative-alternating heuristic with 100,000 runs. The experiment time limit was set to 1800s. Experiments that timed-out are marked T/O. The iterative-alternating heuristic solution is in column H; the non-linear model solution marked with N; and Q is the quadratically constrained model solved by Gurobi. Both N and Q have no improvements added. Objective functions where the solver timed out are marked with $(*)$.

The results of the validation tests are in Table 2. They confirmed empirically that the reformulation is sound, and that the solvers agree on the global solution. The tests revealed two more important things. The first is that the iterative-alternating heuristic is very effective at finding the optimal solution. The second is that the MIQCP solver Gurobi is far more efficient at solving the model Q than the global MINLP solver SCIP is at solving the model N. We therefore complete the remaining experiments on improvements to the formulations only on the Q implementation in Gurobi.

4.2 Valid Inequality Experiments

We perform the remaining experiments as a single process on a Dell Server with Intel Xenon E5-2440 @ 2.4 GHz processor; 24 cores including 6 physical cores with 48 threads; 64 GB RAM. We solve the MIQCP, Q, with Gurobi version 6.0.2. This version restricted the process to 12 virtual CPUs by default. We have not tuned Gurobi, but use the following non-default parameters to improve numerical stability in the final solutions:

- Quad precision for the Simplex Algorithm turned on. This is made necessary by the number of large and small coefficients present in our model.
- Feasibility Tolerance set to 1e-9. All constraints must be satisfied by this tolerance.
- Presolve is turned off, as recommended by Gurobi for MIQCP models.

For the first three experiments, we solved the MIQCP model on 50 random instances for each pairing of terminal node and Steiner node set sizes. We generated a random number from the Uniform distribution between 0 and 100 for both the x and y coordinates using the Python `random` library [20]. We then varied the number of terminal nodes between 5 and 9, and the number of Steiner nodes between 1 and 3. This resulted in 750 instances. For the later experiments, where we use the fastest combinations of improvements (including valid inequalities), we increase the number of terminal nodes to 10, resulting in 900 instances.

In Table 3, we present the performance of the preprocessing and warm start heuristics for the 50 instances of each k–t combination. The preprocessing is very strong, obtaining on average between 21% and 62% arc removal. The strength of the preprocessing improves as the ratio of terminal nodes to Steiner nodes increases. The warm start heuristic is obtained by solving the beaded heuristic for the arc decisions and allowing Gurobi to complete the solution as described in [13]. We compare the warm start value with the optimal objective value, the latter which we obtain from solving an exact model to optimality. The warm start value was also very strong, ranging from 73% of optimal on average to 98%. From the range of these percentages, we can see that in every set, there was at least one instance for which the warm start heuristic found the optimal solution. Only once did the warm start fail. Since the beaded heuristic is guaranteed to be feasible, this failure is due to Gurobi primal heuristics finding a better solution and cutting off the proposed start solution.

In the remaining experiments, we will test the impact of the improvements on the computational performance of the model. The key for the improvements we applied in subsequent experiments is as follows:

- pr: preprocessing variables w_{ij} using the MST over the terminal nodes
- ws: warm-starting using the *beaded-MST solution*
- de5: the degree-inequality constraints as upper bound on the Steiner and terminal nodes, from inequalities (21)

$\|S\|$	$\|T\|$	Inst.	total arcs	% removed	obj value	ws σ	ws % optimal	% range
1	5	50	30.00	40.00	8846.44	3969.19	94.84	[0.81,1.00]
	6	50	42.00	47.62	8802.63	3607.59	95.53	[0.78,1.00]
	7	50	56.00	53.57	8727.56	3073.20	97.07	[0.87,1.00]
	8	50	72.00	58.33	9519.23	2515.56	97.98	[0.88,1.00]
	9	50	90.00	62.22	9737.62	3010.95	97.99	[0.90,1.00]
2	5	50	42.00	28.57	6503.13	3276.83	89.24	[0.01,1.00]
	6	50	56.00	35.71	7038.66	2513.83	88.59	[0.42,1.00]
	7	50	72.00	41.67	7941.53	2944.81	87.51	[0.57,1.00]
	8	50	90.00	46.67	8678.70	3500.01	84.29	[0.52,1.00]
	9	50	110.00	50.91	8245.22	2703.72	88.38	[0.50,1.00]
3	5	50	56.00	21.43	5989.44	3232.32	76.57	[0.50,1.00]
	6	50	72.00	27.78	5982.65	3131.02	77.22	[0.39,1.00]
	7	50	90.00	33.33	6340.25	3482.81	73.09	[0.39,1.00]
	8	50	110.00	38.18	7076.64	3541.99	76.63	[0.47,1.00]
	9	50	132.00	42.42	7442.89	3733.31	77.05	[0.39,1.00]

Table 3: Performance of preprocessing and warm start heuristic. For 50 random instances per k–t combination, we provide the total arcs and percentage removed. The warm start (ws) values are obtained from the beaded heuristic. We provide the standard deviation (σ). Obj value is the optimal solution obtained in later experiments, but repeated here to evaluate the performance of the warm start heuristic.

- de2: the degree-inequality constraints as lower bound on the Steiner nodes, from inequalities (22)
- de1: the degree-inequality constraints as lower bound on the terminal nodes, from inequalities (23)
- sym: symmetry breaking constraints, from inequalities (17)
- sd: angle (sixty-degree) inequalities, from inequalities (19)
- nce: nonlinear centroid inequality constraints, from inequalities (20)
- ge: general valid inequalities, from inequalities (11)–(12)
- ch: convex hull inequalities, as described by (16)
- gl: Gabriel inequalities, from inequalities (24)
- ll: lens inequalities, as described on page 16

We excluded constraints (15) because they are dominated by the convex-hull constraints (16) almost everywhere.

In Tables 4 and 5, we present the results of the experiment with only one improvement at a time. We excluded 64 instances from the complete set of 750 due to time-outs. That is, if an instance times out for any one combination of improvements, we exclude that instance from the results of all combinations of improvements. Most time-outs were due to `gl` and `ll` and occurred for instances (3,8) and (3,9). We attribute these time-outs to the large number of binary variables added, as elaborated later. Under the column heading 'default' we show the result for no improvements, while the remaining headings are as

$\|S\|$	$\|T\|$	$\|Inst.\|$	default	pr	ws	de5	de1	de2	sym	sd	nce	ge	ch	gl	ll
1	5	50	943.57	**0.56**	**0.10**	1.09	1.10	2.11	1.00	**0.99**	**0.16**	1.16	**0.59**	2.81	2.60
	6	50	2526.49	**0.39**	**0.08**	**0.96**	1.00	1.75	1.00	**0.83**	**0.25**	**0.81**	**0.55**	1.68	1.75
	7	50	5647.27	**0.32**	**0.09**	**0.91**	**0.99**	1.47	1.00	**0.73**	**0.25**	**0.90**	**0.61**	**0.98**	1.08
	8	50	8675.32	**0.40**	**0.16**	**0.95**	1.00	1.28	1.00	**0.72**	**0.30**	**0.90**	**0.71**	**0.72**	**0.74**
	9	50	9869.73	**0.53**	**0.27**	1.04	1.09	1.24	1.00	**0.78**	**0.39**	**0.99**	**0.81**	**0.69**	**0.74**
2	5	50	8546.32	**0.68**	**0.15**	**0.99**	**0.98**	**0.97**	**0.93**	**0.89**	**0.20**	**0.86**	**0.67**	1.00	1.03
	6	50	11665.95	**0.80**	**0.34**	1.01	1.01	1.02	**0.95**	**0.93**	**0.27**	**0.89**	**0.93**	1.26	1.17
	7	50	13251.35	**0.97**	**0.72**	**0.99**	1.02	**0.99**	**0.92**	1.04	**0.54**	1.00	**0.99**	1.97	2.24
	8	50	16952.57	**0.83**	**0.82**	**0.96**	**0.99**	1.01	**0.85**	1.01	**0.72**	**0.93**	**0.95**	4.26	4.90
	9	50	25455.48	**0.74**	**0.84**	1.03	1.00	**0.99**	**0.84**	**0.96**	**0.69**	**0.91**	**0.96**	6.44	6.86
3	5	50	14930.91	**0.96**	**0.76**	1.03	1.04	1.01	**0.83**	1.05	**0.69**	1.04	1.04	1.95	2.18
	6	50	27059.46	**0.84**	**0.94**	**1.00**	1.01	1.01	**0.68**	1.00	**0.94**	**0.96**	**0.93**	3.54	3.70
	7	50	57288.15	**0.80**	1.12	**0.99**	1.02	1.00	**0.59**	1.07	1.09	**0.97**	**0.99**	7.05	6.62
	8	32	119115.06	**0.73**	1.23	1.01	1.03	**0.93**	**0.51**	**1.00**	1.29	**0.93**	**0.98**	8.81	10.69
	9	4	354131.55	**0.50**	1.20	**0.84**	**0.76**	**0.92**	**0.28**	**0.70**	1.42	**0.71**	**0.89**	7.73	7.85

Table 4: Geometric mean of node count for 50 instances for each size. Of the total 750 instances, 64 were excluded due to time-outs (1800s). The default setting has no improvements added. We provide the node count for default mode, and percentage relative node count for remaining experiments. Each column heading corresponds to an improvement that was turned on in addition to default settings. **Bold** results beat default settings, and highlighted results are the best.

$\|S\|$	$\|T\|$	$\|Inst.\|$	default	pr	ws	de5	de1	de2	sym	sd	nce	ge	ch	gl	ll
1	5	50	0.14	**0.62**	**0.43**	**0.97**	**0.95**	1.76	**0.92**	**0.86**	**0.71**	1.00	**0.56**	1.80	1.55
	6	50	0.24	**0.57**	**0.38**	1.04	1.14	2.04	1.00	**0.90**	**0.76**	1.10	**0.72**	2.20	1.96
	7	50	0.39	**0.56**	**0.37**	1.17	1.17	2.19	1.07	**0.95**	**0.73**	1.30	**0.82**	2.33	2.06
	8	50	0.64	**0.65**	**0.38**	1.29	1.38	1.99	1.22	1.16	**0.75**	1.47	**0.83**	1.80	1.85
	9	50	1.04	**0.54**	**0.39**	1.26	1.27	1.76	1.25	1.29	**0.78**	1.45	**0.83**	1.52	1.55
2	5	50	0.87	**0.75**	**0.24**	1.07	1.08	1.20	1.01	**0.94**	**0.52**	1.11	**0.45**	1.61	1.64
	6	50	2.04	**0.59**	**0.19**	1.21	1.24	1.48	1.15	1.17	**0.48**	**0.73**	**0.70**	1.58	1.44
	7	50	2.67	**0.86**	**0.29**	1.29	1.33	1.32	1.09	1.66	**0.77**	1.46	**0.95**	2.19	2.58
	8	50	3.63	1.03	**0.39**	1.25	1.30	1.34	1.11	1.73	1.26	1.33	1.03	4.59	4.76
	9	50	5.02	1.23	**0.49**	1.52	1.42	1.45	1.22	1.58	1.50	1.40	1.14	7.89	8.19
3	5	50	5.38	1.40	**0.24**	1.36	1.48	1.54	**0.96**	1.95	**0.90**	1.76	1.15	2.27	2.21
	6	50	9.57	1.13	**0.39**	1.30	1.40	1.53	**0.88**	1.59	1.59	1.33	1.02	3.53	3.62
	7	50	24.02	1.03	**0.38**	1.36	1.34	1.50	**0.73**	1.52	1.39	1.23	1.05	5.82	5.11
	8	32	53.47	1.09	**0.38**	1.42	1.69	1.31	**0.70**	1.61	1.46	1.29	1.06	7.44	8.67
	9	4	187.69	**0.46**	**0.30**	**0.77**	**0.80**	1.27	**0.26**	**0.97**	**0.85**	**0.67**	**0.57**	3.22	4.05

Table 5: Geometric mean of solve time for 50 instances for each size. Of the total 750 instances, 64 were excluded due to time-outs (1800s). The default setting has no improvements added. We provide the solve time in seconds for default mode, and percentage relative solve time for the remaining experiments. Each column heading corresponds to an improvement that was turned on in addition to default settings. **Bold** results beat default settings, and highlighted results are the best.

described above. If a result beats 'default', then it is in bold. If a result is the best, it is highlighted. In Table 4, we present the geometric mean of node count for branch-and-bound search. We see that most improvements have impact on the node count when the problem is small and easy (relative to number of Steiner nodes), except `sym` which should have no impact when $k = 1$ as there is no symmetry to break. The impact for `de5`, `de1` and `de2` is negligible. While `gl` and `ll` have some evidence of impact for low values of k, the node count increases dramatically as the terminal nodes and Steiner nodes increase. This is due to the way that we obtained linear versions of these constraints: we introduce binary variables for each constraint—specifically, for each Steiner node we add $4 \times k \times n$ variables and $5 \times k \times n$ constraints. Reducing the admissible terminal-to-terminal arcs therefore has direct impact on the number of these variables and constraints. The improvements obtained from `pr` and `ws` are very strong, followed by some strong improvements from `sym`, `sd`, and `nce`.

In Table 5, we present the geometric mean of solve times in seconds. Since most instances excluded are from (3,9), this row should be interpreted with caution. As single improvements, `pr` and `ws` have the strongest impact on solve time. The improvements `sym`, `nce` and `ch` also have consistent impact as single improvements, although to a lesser degree.

Next, we are interested in seeing the combined improvement. We set up this experiment with preprocessing and warm-start as default settings, and add a single further improvement. These results are in Tables 6 and 7. We excluded 58 instances from the total 750 due to time-outs. All of these time-outs occurred in instances where $k = 3$, and were caused by improvements `gl` and `ll`.

In Table 6, we present the geometric mean of the node count during branch-and-bound search. The two most important improvements are `ge` and `sym`. We see the impact by nonlinear centroid inequalities (`nce`) has diminished now that `pr` and `ws` are both on. The symmetry-breaking constraints (`sym`) become important as the Steiner nodes increase. The convex hull inequalities (`ch`) have impact across all problem sizes, while the angle inequalities (`sd`) have negligible impact. The inequalities `de1`, `de2`, `de5`, and `ge` sporadically have impact with no consistent behaviour. The Gabriel and lens inequalities continue their negative performance in the node count during branch and bound search.

In Table 7, we present the geometric mean of the solve time of each experiment. The combined impact of `pr` and `ws` is better than the individual impact of these improvements. The most important additional improvements are convex hull constraints (`ch`) and general constraints (`ge`), which beat default for most experiments. However, the difference is not very large. A more impressive difference is visible with the symmetry breaking constraints (`sym`), which improve in their impact as the Steiner points increase. The nonlinear centroid inequalities (`nce`) improve on the individual solve times reported in Table 5, but the improvement is much worse than the default setting of `pr` and `ws`. This result suggests that the time required to solve the additional nonlinear constraints is not aided by the preprocessing and warm-start. Again, the inequalities `de1`, `de2`, and `de5` sporadically have impact, although to a lesser degree than we would expect from the node count. This suggests that even though the branch

$\|S\|$	$\|T\|$	$\|Inst.\|$	default	de5	de1	de2	sym	sd	nce	ge	ch	gl	ll
1	5	50	70.02	**1.00**	1.01	1.01	1.00	1.04	**0.96**	**0.92**	**0.90**	1.51	1.54
	6	50	131.98	**0.98**	**0.97**	**0.97**	1.00	1.03	1.03	**0.89**	**0.93**	1.70	1.69
	7	50	223.54	**0.98**	1.05	1.03	1.00	1.09	1.19	**0.89**	**0.96**	1.89	2.05
	8	50	353.07	**0.98**	1.00	**0.99**	1.00	1.07	1.22	**0.87**	**0.96**	2.13	2.20
	9	50	544.07	1.01	1.01	**1.00**	1.00	1.11	1.12	**0.85**	**0.97**	2.13	2.17
2	5	50	862.93	1.03	1.02	1.00	1.13	1.15	1.18	**0.95**	**0.97**	2.06	2.08
	6	50	2053.96	1.01	1.01	1.03	1.12	1.12	1.06	**0.91**	**0.99**	2.11	2.14
	7	50	4810.72	**0.96**	**0.99**	1.00	**0.98**	1.04	**0.90**	**0.87**	**0.98**	1.83	1.93
	8	50	8686.89	**0.98**	1.04	**0.99**	**0.96**	1.02	**0.92**	**0.88**	1.02	1.74	1.77
	9	50	13720.89	**0.98**	1.06	1.01	**0.90**	1.06	1.01	**0.90**	1.01	2.04	1.84
3	5	50	10018.58	1.01	**0.89**	1.00	**0.88**	**0.90**	**0.90**	**0.77**	**0.89**	1.30	1.33
	6	50	21667.64	1.01	**0.99**	**0.98**	**0.70**	**0.98**	1.06	**0.84**	**0.99**	1.63	1.46
	7	50	46831.46	**0.99**	1.03	1.01	**0.57**	1.04	1.20	**0.93**	**0.98**	5.83	8.60
	8	34	98537.33	**0.99**	1.01	**0.99**	**0.43**	1.06	1.15	**0.94**	**0.98**	13.17	19.21
	9	8	155963.79	1.00	1.05	1.03	**0.42**	1.13	1.30	**0.91**	**0.98**	15.73	15.32

Table 6: Geometric mean of node count for 50 instances for each size. Of the total 750 instances, 58 were excluded due to time-outs (1800s). Default settings have both preprocessing and warm-start turned on. We provide node count for default settings, and percentage relative performance for remaining experiments. Each column heading corresponds to an improvement that was turned on in addition to default settings. Results that beat default settings are in bold. The cell of the best result is highlighted.

$\|S\|$	$\|T\|$	$\|Inst.\|$	default	de5	de1	de2	sym	sd	nce	ge	ch	gl	ll
1	5	50	0.03	1.07	1.01	1.06	1.02	1.10	1.75	1.10	**0.97**	1.05	1.02
	6	50	0.05	**0.95**	**1.00**	**0.99**	**0.97**	1.09	1.93	1.03	**0.89**	1.08	1.05
	7	50	0.06	1.01	1.04	**1.00**	1.02	1.15	2.23	1.08	**0.94**	1.24	1.31
	8	50	0.08	1.04	**0.99**	**0.98**	1.02	1.11	2.31	1.05	**0.97**	1.34	1.43
	9	50	0.09	1.01	1.04	**0.99**	1.02	1.17	2.46	1.02	**0.99**	1.57	1.59
2	5	50	0.12	**0.99**	1.03	1.06	1.19	1.09	3.44	1.09	**0.89**	1.67	1.68
	6	50	0.20	**0.99**	1.07	1.10	1.30	1.28	3.75	**0.97**	**0.88**	2.12	2.07
	7	50	0.35	1.04	1.18	1.15	1.32	1.37	3.68	**0.92**	**0.92**	2.33	2.20
	8	50	0.60	1.02	1.14	1.11	1.34	1.30	3.71	**0.98**	1.04	2.89	2.82
	9	50	1.14	1.06	1.05	1.00	1.04	1.29	3.52	**0.97**	**0.99**	2.66	2.30
3	5	50	1.16	**0.98**	**0.85**	1.02	1.18	1.03	3.14	**0.81**	**0.74**	1.84	1.82
	6	50	3.70	1.05	1.00	1.03	**0.93**	1.18	3.30	**0.94**	**0.92**	2.05	1.80
	7	50	8.19	1.01	1.03	1.02	**0.82**	1.18	3.35	1.06	**0.90**	6.01	9.10
	8	34	16.86	**0.99**	1.01	**0.99**	**0.61**	1.17	3.06	**0.98**	**0.92**	14.99	21.99
	9	8	25.83	1.05	1.09	1.05	**0.42**	1.21	3.46	**1.00**	**0.92**	17.31	17.12

Table 7: Geometric mean of solve time for 50 instances for each size. Of the total 750 instances, 58 were excluded due to time-outs (1800s). Default settings have both preprocessing and warm-start turned on. We provide the time in seconds for default settings, and percentage relative performance for remaining experiments. Each column heading corresponds to an improvement that was turned on in addition to default settings. Results that beat default settings are in bold. The cell of the best result is highlighted.

and bound tree was smaller, more time was required to solve the relaxation with these constraints added. The Gabriel and lens inequalities decrease performance of the model, as expected from the node count results.

We combine the results from these two tables to create a final experiment. We create combinations of improvements that we expect to perform well with respect to node count and time. From the results in Tables 6 and 7, which are consistent, we create a set with `pr`, `ws`, `sym`, `ge`, and `ch`. In set B, we take all improvements from set A and add two degree inequality constraints that appeared to have negligible impact in previous experiments. For a third experiment set, we look at the non-negligible impact improvements from Table 4. We do this because those sets that perform best for node count are likely to become the best sets in the future when algorithms for the nonlinear programming relaxation improve. If the inequalities are added dynamically via separation, then this set is also likely to be the best performer. However, given the current state-of-the-art in algorithm performance, we expect the time-based improvement sets to be the best sets. This results in three improvement sets, which we outline in Table 8.

A	B	C
`pr`, `ws`, `ge`, `ch` `sym`	`pr`, `ws`, `ge`, `ch` `sym`, `de1`, `de2`	`pr`, `ws`, `sd`, `nce` `sym`, `ch`, `ge`

Table 8: The improvement sets in the final experiment.

We compare each of these sets against `default`, which has no improvements. We increase the test instances to include 10 terminal nodes, and decrease the time-out to 600s. Of the resulting 900 instances, we excluded 3 due to time-out–all of which were caused by `None`.

We present the node count for all five sets in Table 9. As we expected, the set that we selected to perform well with respect to node count (set C) has the best outcome for most instance sizes. The ratio of best average performance over the average default performance ranges from 0.04 to 0.39, with the strongest results occurring when the Steiner points are few. All improvement sets outperformed the default settings for node count.

The time results for all four sets are in Table 10, where we see an improvement in the performance ratio from best performance to default performance. This now ranges from 0.07 to 0.26, and illustrates the impact of the improvements on the ancillary algorithms—such as linear and nonlinear programming subproblem methods—gained from the improvement sets. When $k = 1$, both A and B perform equally as well. However, set B becomes dominant as the problem difficulty increases with respect to Steiner nodes. This illustrates the importance of testing the computational impact of valid inequalities in the context of each other. The nodecount based set, C, outperformed the default settings. However, the combined impact of this set could not keep up with the performance of B or C due to a loss in performance of subproblem algorithms.

k	n	None	A	B	C	$\frac{\text{best}}{\text{default}}$
1	5	943.57	**58.38**	**55.87**	**38.40**	0.04
	6	2526.49	**109.26**	**107.10**	**80.76**	0.03
	7	5647.27	**195.19**	**194.08**	**162.45**	0.03
	8	8675.32	**294.82**	**298.24**	**254.31**	0.03
	9	9869.73	**469.27**	**466.26**	**381.80**	0.04
	10	13370.08	**682.18**	**705.11**	**556.11**	0.04
2	5	8546.32	**788.64**	**787.46**	**594.45**	0.07
	6	11665.95	**2027.54**	**1869.03**	**1277.22**	0.11
	7	13251.35	**4183.71**	**3909.23**	**2378.86**	0.18
	8	16952.57	**7048.67**	**7172.73**	**4465.16**	0.26
	9	25455.48	**11099.15**	**10694.99**	**7838.36**	0.31
	10	39870.09	**16802.27**	**17055.77**	**13853.20**	0.35
3	5	14930.91	**6516.68**	**5052.74**	**4274.75**	0.29
	6	27059.46	**13557.92**	**11054.63**	**10646.82**	0.39
	7	57288.15	**23904.48**	**22380.26**	**21086.59**	0.37
	8	129623.67	**42021.82**	**39528.68**	**42837.50**	0.30
	9	277804.60	**79135.59**	**70965.90**	**85061.84**	0.26
	10	557109.07	**130913.13**	**122668.59**	**170658.52**	0.22

Table 9: Geometric mean of node count for no improvements ('None') and three test sets. Of the 900 instances, 3 were excluded due to time-outs (600s). **Bold** results beat default settings, and highlighted results are the best.

k	n	None	A	B	C	$\frac{\text{best}}{\text{default}}$
1	5	0.14	**0.04**	**0.04**	**0.06**	0.26
	6	0.24	**0.05**	**0.05**	**0.09**	0.20
	7	0.39	**0.06**	**0.06**	**0.12**	0.15
	8	0.64	**0.08**	**0.07**	**0.15**	0.11
	9	1.04	**0.09**	**0.10**	**0.22**	0.09
	10	1.47	**0.11**	**0.12**	**0.29**	0.08
2	5	0.87	**0.13**	**0.14**	**0.35**	0.15
	6	2.04	**0.24**	**0.24**	**0.57**	0.12
	7	2.67	**0.45**	**0.38**	**0.88**	0.14
	8	3.63	**0.72**	**0.68**	**1.45**	0.19
	9	5.02	**1.13**	**1.03**	**2.39**	0.21
	10	8.31	**2.22**	**1.86**	**4.57**	0.22
3	5	5.38	**0.73**	**0.63**	**1.63**	0.12
	6	9.57	**2.89**	**1.72**	**4.05**	0.18
	7	24.02	**6.08**	**4.61**	**10.18**	0.19
	8	65.38	**9.41**	**7.04**	**17.01**	0.11
	9	161.90	**14.44**	**11.00**	**30.64**	0.07
	10	300.31	**32.74**	**23.12**	**58.04**	0.08

Table 10: Geometric mean of solve time (seconds) for no improvements ('None') and four improvement sets. Of the 900 test instances, 3 were excluded due to time-outs (600s).

4.3 Discussion

While we observed greater numerical stability in the global MINLP solver, SCIP, for solving model N, this came at a significant cost of solve time—to the extent where we would not expect to solve interesting sized networks with default settings. In contrast, the MIQCP solver, Gurobi, was very efficient for model Q, but with unstable performance that had to be tamed with strict tolerances for precision, both on quadratic precision and feasibility tolerances.

The extensive set of computational experiments lead to a new perspective on the efficiency of the proposed procedures and valid inequalities. In particular, we could obtain combinations of strategies that were able to efficiently reduce the size of the branch and bound tree and computational time needed to solve instances of up to 10 terminal nodes and 3 Steiner points. Although these instances are still small from the perspective of most real applications, this study opens a path of research in the use of black box solvers for the solution of this family of geometric Steiner problems.

We have devoted much attention to the analysis of the performance of different valid inequalities when included a priori in the model. As the results indicate, a stronger formulation is not always equivalent to a superior computational performance. Further investigation in this area includes both a deeper theoretical analysis of the strength of such valid inequalities and extended computational experiments that could, for example, add violated cuts on the fly, reducing the burden of solving the complete nonlinear problem at each node.

With respect to the heuristics, our results indicate that even simple procedures such as the beaded heuristic can obtain relatively good solutions in very short computational times. This indicates that the use of such heuristic as a warm start procedure (as presented here) can be an easy and efficient way of reducing the solution times of exact methods. Thus both heuristics can be useful for large-scale problems, where obtaining a good-quality solution is preferable to obtaining no solution at all. Moreover, the beaded heuristic appears as a strong candidate as the solution method of choice when time available for solution is restricted, either because the application requires a near-to-immediate answer or because the method has to be solved many times (for example, when the Steiner problem is a subproblem in a decomposition algorithm for a more complex formulation).

Finally, in between the exact approaches and the beaded heuristic sits our proposed iterative-alternating heuristic, which is capable of providing optimal solutions in reasonable computational times (i.e., in minutes). Two features of this heuristic are note worthy. First, each iteration of the heuristic relies on alternating an MST procedure and the solving of a system of linear equations. These "local" iterations are very fast and converge in very few steps to a local optimum. Other methods could make use of such an idea while possibly adding some long term memory. Indeed, any kind of general metaheuristic framework such as GRASP [21] or Iterated Local Search [18] could be used in the upper decision layer that defines the initial positions of the candidate Steiner points. This is likely to accelerate convergence to good solutions. Second, the optimality

of these local iterations suggest that if one is to find a decomposition method which relies on such subproblems, then it is likely that much larger instances will become solvable.

References

[1] T. Achterberg, SCIP: solving constraint integer programs, *Mathematical Programming Computation*, 1 (2009) 1–41.

[2] K.D. Andersen, E. Christiansen, A.R. Conn and M.L. Overton, An efficient primal-dual interior-point method for minimizing a sum of Euclidean norms, *SIAM Journal on Scientific Computing*, 22 (2000) 243–262.

[3] M. Brazil, P. Grossman, J.H. Rubinstein and D. Thomas, Improving Underground Mine Access Layouts Using Software Tools, *Interfaces* 44 (2013) 195–203.

[4] M. Brazil, C.J. Ras, K.J. Swanepoel and D.A. Thomas, Generalised k-Steiner Tree Problems in Normed Planes, *Algorithmica* 71 (2015) 66–86.

[5] P. Berman and A. Zelikovsky, On approximation of the power-p and bottleneck Steiner trees. In D.–Z. Du, J.M. Smith, J.H. Rubinstein (Eds.), *Advances in Steiner trees*, pp. 117-–135, Kluwer Academic Publishers, 2000.

[6] T. Berthold, S. Heinz, and S. Vigerske, Extending a CIP framework to solve MIQCPs. In Lee, Jon and Leyffer, Sven, Eds, Mixed Integer Nonlinear Programming, Springer, 2012, 427–444.

[7] J. Cong, L. He, C.–K. Koh and P.H. Madden, Performance optimization of VLSI interconnect layout, *Integration, the VLSI journal*, 21 (1996) 1–94.

[8] Cplex Optimizer. `http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/`

[9] M. Fampa and N. Maculan, A new relaxation in conic form for the Euclidean Steiner problem in $\mathbb{R}^n$, *RAIRO-Operations Research* 35, 4 (2001) 383–394.

[10] E.N. Gilbert and H. O. Pollak, Steiner minimal trees, *SIAM Journal on Applied Mathematics*, 16 (1968) 1–29.

[11] M. Grotschel, C.L. Monma and M. Stoer, Design of survivable communications networks. In Ball et al. (Eds.), *Network Models*, Chapter 10, Handbook of Management Science and Operations Research, Volume 7, pp. 617–672, Elsevier, 1995.

[12] Gurobi Optimization, `http://www.gurobi.com/`

[13] Gurobi Optimization, Variable attributes: start, `http://www.gurobi.com/documentation/6.0/refman/start.html`

[14] W. Hart, C. Laird, J-P. Watson and D. Woodruff, Pyomo–Optimization Modeling in Python, 67, Springer, 2012.

[15] L. Kou, G. Markowsky and L. Berman, A fast algorithm for Steiner trees, *Acta informatica*, 15 (1981) 141–145.

[16] J.W. Laarhoven and J.W. Ohlmann, A randomized Delaunay triangulation heuristic for the Euclidean Steiner tree problem in R_d, *Journal of Heuristics*, 17 (2011) 353–372.

[17] S. Lee and M. Younis, Recovery from multiple simultaneous failures in wireless sensor networks using minimum Steiner tree, *Journal of Parallel and Distributed Computing*, 70 (2010) 525–536.

[18] H.R. Lourenço, O. Martin and T. Stützle, Iterated Local Search. In F. Glover and G. Kochenberger (Eds.), *Handbook of Metaheuristics*, pp. 321–353, Kluwer Academic Publishers, 2003.

[19] M. Minoux, Networks synthesis and optimum network design problems: Models, solution methods and applications, *Networks* 19 (1989) 313–360.

[20] Python random library. `https://docs.python.org/2/library/random.html`

[21] M.G.C. Resende and C.C Ribeiro, Greedy randomized adaptative search procedures. In F.Glover and G.Kochenberger (Eds.), *Handbook of Metaheuristics*, pp. 219–249, Kluwer Academic Publishers, 2003.

[22] S. Vigerske, Decomposition of Multistage Stochastic Programs and a Constraint Integer Programming Approach to Mixed-Integer Nonlinear Programming, Ph.D Thesis, Humboldt-University Berlin, 2013.

[23] L. Wang and Z. Li, An approximation algorithm for a bottleneck k-Steiner tree problem in the Euclidean plane, *Information Processing Letters*, 81 (2002) 151–156.

[24] D.M. Warme, P. Winter and M. Zachariasen, Exact Algorithms for Plane Steiner Tree Problems: A Computational Study. In D.–Z. Du, J.M. Smith, J.H. Rubinstein (Eds.), *Advances in Steiner Trees*, pp. 81-–116, Kluwer Academic Publishers, 2000.

[25] J.F. Weng, D.A. Thomas and I. Mareels, Maximum Parsimony, Substitution Model, and Probability Phylogenetic Trees, *Journal of Computational Biology*, 18 (2011) 67–80.

[26] P. Winter and M. Zachariasen, Euclidean Steiner Minimum Trees: An Improved Exact Algorithm, *Networks* 30 (1997) 149–166.