Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Discrete Optimization

Minimum Spanning Trees with neighborhoods: Mathematical programming formulations and solution methods

Víctor Blanco[a,*], Elena Fernández[b], Justo Puerto[c]

[a] Department of Quantitative Methods for Economics and Business, Universidad de Granada, 18011 Granada, Spain
[b] Department of Statistics and Operations Research, Universitat Politècnica de Catalunya, 08034 Barcelona, Spain
[c] Department of Statistics and Operations Research, Universidad de Sevilla, 41012 Sevilla, Spain

ARTICLE INFO

Article history:
Received 10 November 2016
Accepted 8 April 2017
Available online 15 April 2017

Keywords:
Combinatorial Optimization
Minimum Spanning Trees
Neighborhoods
Mixed Integer Non Linear Programming
Second order cone programming,

ABSTRACT

This paper studies Minimum Spanning Trees under incomplete information assuming that it is only known that vertices belong to some neighborhoods that are second order cone representable and distances are measured with a ℓ_q-norm. Two Mixed Integer Non Linear mathematical programming formulations are presented, based on alternative representations of subtour elimination constraints. A solution scheme is also proposed, resulting from a reformulation suitable for a Benders-like decomposition, which is embedded within an exact branch-and-cut framework. Furthermore, a mathheuristic is developed, which alternates in solving convex subproblems in different solution spaces, and is able to solve larger instances. The results of extensive computational experiments are reported and analyzed.

1. Introduction

Nowadays Combinatorial Optimization (CO) lies in the heart of multiple applications in the field of Operations Research. Many such applications can be formulated as optimization problems defined on graphs where some particular structure is sought satisfying some optimality property. Traditionally this type of problems assumed implicitly the exact knowledge of all input elements, and, in particular, of the precise position of vertices and edges. Nevertheless, this assumption does not always hold, as uncertainty, lack of information, or some other factors may affect the relative position of the elements of the input graph. Hence, new tools are required to give adequate answers to these challenges, which have been often ignored by standard CO tools.

A matter that, in this context, has attracted the interest of researchers over the last years is the solution of certain CO problems when the exact position of the vertices of the underlying graph is not known with certainty. If probabilistic information is available, then stochastic programming tools can be used, and optimization over expected values carried out. Moreover, even under the assumption of incomplete information one could use a uniform distribution and still apply such an approach. However, the use of probabilistic information and allowing to consider all possible locations for the vertices is not always suitable. For instance, when a unique representative associated with each point of the input graph must be determined. Scanning the related literature one can find papers applying both methodologies. Examples of stochastic approaches are for instance Bertsimas and Howell (1993) or Frank (1969). Examples of the second type of approach arise in variants of the traveling salesman problem (TSP), Minimum Spanning Tree (MST), or facility location problems that deal with *demand regions* instead of *demand points* (see Arkin and Hassin, 1994; Brimberg and Wesolowsky, 2002; Cooper, 1978; Dror, Efrat, Lubiw, and Mitchell, 2003; Juel, 1981; Nickel, Puerto, and Rodríguez-Chía, 2003; Yang, Lin, Xu, and Xie, 2007, to mention just a few).

A relevant common question raised by the latter class of problems is how to model and solve optimization problems on graphs when vertices are not points but regions in a given domain. The above mentioned case of the TSP, first introduced by Arkin and Hassin (1994, 2000), has been addressed recently by a number of authors. It generalizes the Euclidean TSP and the group Steiner tree problem, and has applications in VLSI-design and other routing problems, in which there exist constraints on the position of the vertices. Several inapproximability results and approximation algorithms have been developed for particular cases. The case of the spanning tree problem with neighborhoods (MSTN) was first addressed by Yang et al. (2007), who proved that the general case of the problem in the plane is NP-hard (result also reproved by Löffler & van Kreveld, 2010), and gave several approximation algorithms and a PTAS for the particular case of disjoint unit disks in the plane. Some extensions considering the maximization of the

* Corresponding author.
E-mail addresses: vblanco@ugr.es (V. Blanco), e.fernandez@upc.edu (E. Fernández), puerto@us.es (J. Puerto).

http://dx.doi.org/10.1016/j.ejor.2017.04.023
0377-2217/© 2017 Elsevier B.V. All rights reserved.

weights are studied in Dorrigiv et al. (2013). In particular, they proved the non existence of FPTAS for MSTN, for general disjoint disks, in the planar Euclidean case. Disser, Mihalák, Montanari, and Widmayer (2014) consider the shortest path problem and the rectilinear MSTN, and give some approximability results. To the best of our knowledge, Gentilini, Margot, and Shimada (2013) are the first authors to propose an exact Mixed Integer Non Linear Programming (MINLP) formulation for the TSP with neighborhoods, but we are not aware of any MINLP for the MSTN.

Our goal in this paper is to develop MINLP formulations and solution methods for the MSTN. We first present two MINLP formulations that allow to solve medium size MSTN planar and 3D Euclidean instances with up to 20 vertices, for neighborhoods of varying radii using an on-the-shelf solver. Furthermore, we develop an effective branch-and-cut strategy, based on a generalized Benders decomposition (Benders, 1962; Geoffrion, 1972), and compare its performance with that of the solver for the proposed formulations. For this we present an alternative formulation for the MSTN, in which the master problem consists of finding a MST with costs derived from a continuous non linear (slave) subproblem, and we develop the expression and separation of the cuts that are added in the solution algorithm. Given that both the solver (for the two MINLP formulations) and the exact branch-and-cut algorithm can be too demanding, in terms of their computing times, we have also developed an effective and efficient mathheuristic. The mathheuristic stems from the observation that the subproblems defined in the solution spaces of each of the two main sets of variables are convex (so they can be solved very efficiently); it alternates in solving subproblems in each of these solution spaces.

The paper is organized as follows. Section 2 is devoted to introduce the MSTN and to state a generic formulation. In Section 3 we present and compare two MINLP formulations for the MSTN, based on alternative representations of the spanning trees polytope. Section 4 develops the exact branch-and-cut algorithm, based on a Benders-like decomposition scheme: we define the master and the non linear subproblem, and derive the cuts and their separation. In Section 4.1 we first compare the performance of the on-the-shelf solver with the two MINLP formulations, and then we report the numerical results obtained with the exact row-generation algorithm. The mathheuristic is presented in Section 5, where we also give the numerical results that it produces. The paper ends with some concluding remarks and our list of references.

2. Minimum Spanning Trees with neighborhoods

Let $G = (V, E)$ be a connected undirected graph, whose vertices are embedded in $\mathbb{R}^d$, i.e., $v \in \mathbb{R}^d$ for all $v \in V$. Associated with each vertex $v \in V$, let $N_v \subseteq \mathbb{R}^d$ denote a convex set containing v in its interior. Let also $\|\cdot\|$ denote a given norm.

Feasible solutions to the Minimum Spanning Tree with Neighborhoods (MSTN) problem consist of a set of points, $Y^* = \{y_v \in N_v \mid v \in V\}$, together with a spanning tree T^* on the graph $G^* = (Y^*, E^*)$, with edge set $E^* = \{\{y_v, y_w\} : \{v, w\} \in E\}$. Edge lengths are given by the norm-based distance between the selected points relative to $\|\cdot\|$, i.e.:

$$d(y_v, y_w) = \|y_v - y_w\|, \qquad \text{for all } \{y_v, y_w\} \in E^*.$$

The overall cost of (Y^*, T^*) is therefore

$$d(T^*) = \sum_{e=\{y_v, y_w\} \in T^*} d(y_v, y_w).$$

The MSTN is to find a feasible solution, (Y^*, T^*), of minimum total cost.

Particular cases of the MSTN have been studied in the literature for planar graphs. Disser et al. (2014) studied the case when the sets N_v are rectilinear neighborhoods centered at $v \in V$. Dorrigiv et al. (2013) addressed the problem when the sets N_v are disjoint Euclidean disks. Both referenced works study the complexity of the considered problems but do not attempt to develop MINLP formulations or solution methods for it.

In this paper, we consider the general case where the graph G is embedded in $\mathbb{R}^d$. Even if our developments can be extended to generic convex sets, we focus on the case where N_v is second order cone (SOC) representable (Lobo, Vandenberghe, Boyd, & Lebret, 1998). The main reason for this is that state-of-the-art solvers incorporate mixed integer non-linear implementations of SOC constraints. Such a modeling assumption could be readily overcome if on-the-shelf solvers incorporated more general tools to deal with convex sets.

Observe that SOC representable neighborhoods allow to model, as a particular case, centered balls of a given radius r_v, associated with the standard ℓ_p-norm with $p \in [1, \infty]$ in $\mathbb{R}^d$, that we denote by $\|\cdot\|_p$, i.e., neighborhoods in the form $N_v = \{x \in \mathbb{R}^d : \|x - v\|_p \leq r_v\}$, where

$$\|z\|_p = \begin{cases} \left(\sum_{k=1}^d |z_k|^p\right)^{\frac{1}{p}} & \text{if } p < \infty \\ \max_{k\in\{1,\ldots,d\}} |z_k| & \text{if } p = \infty \end{cases}.$$

The reader is referred to Blanco, Puerto, and El-Haj Ben-Ali (2014) for further details on the SOC constraints that allow to represent (as intersections of second order cone and/or rotated second order cone constraints) such norm-based neighborhoods. Indeed, we can also easily handle neighborhoods defined as bounded polyhedra in $\mathbb{R}^d$, as well as intersections of polyhedra and balls. Hence, more sophisticated convex neighborhoods can be suitably represented or approximated using elements from the above mentioned families of sets.

Two extreme situations that can be modeled within our framework are the following. If the neighborhood for each vertex $v \in V$ is the singleton $N_v = \{v\}$, then MSTN becomes the classical MST problem with edge lengths given by the norm-based distances between each pair of vertices. On the other hand, if the considered neighborhoods are big enough so that $\bigcap_{v\in V} N_v \neq \emptyset$, then the problem reduces to finding a degenerate one-vertex tree and the solution to the MSTN is that vertex with cost 0.

Throughout this paper we use the following notation:

- ST_G as the set of incidence vectors associated with spanning trees on G, i.e. $ST_G = \{x \in \mathbb{R}_+^{|E|} : x \text{ is a spanning tree on } G\}$.
- $Y = \prod_{v\in V} N_v$, where N_v is the neighborhood associated to vertex v, which contains the possible sets of vertices for the spanning trees of MSTN.

Then, the MSTN can be stated as:

$$\min \sum_{e\in E} d(y_v, y_w)x_e \qquad \text{(MSTN)}$$
$$\text{s.t.} \quad x \in ST_G, \; y \in Y.$$

Several observations follow from the formulation above:

1. Fixing $x \in ST_G$ in MSTN results in a continuous SOC problem, which is well-known to be convex (Lobo et al., 1998). On the other hand, fixing $y \in Y$ results in a standard MST problem. It is a well-known that MST admits continuous linear programming representations (Edmonds, 1970; Martin, 1991). Thus, MSTN can be seen as a biconvex optimization problem, which is neither convex nor concave (Gorski, Pfeuffer, & Klamroth, 2007).
2. Due to the expression of its objective function, (MSTN) is not separable, even if each of its sets of variables x and y belong to convex domains in different spaces.
3. Since (MSTN) *combines* the above two subproblems, it is suitable to be represented as a MINLP.

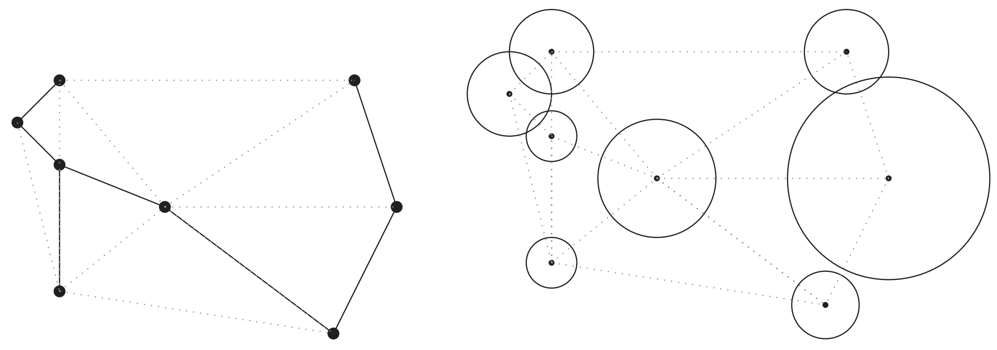
(A) Input graph G and Euclidean MST (black lines). (B) Neighborhoods of the vertices.

Fig. 1. Data for Example 2.1.

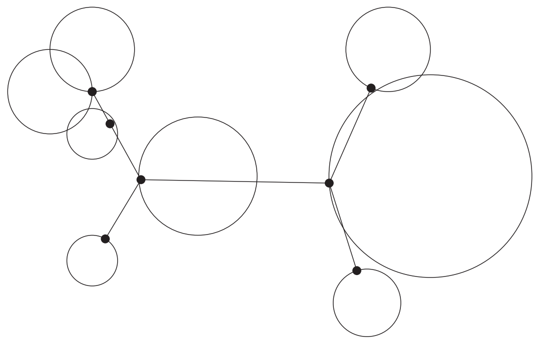
Fig. 2. A MSTN for the data in Example 2.1.

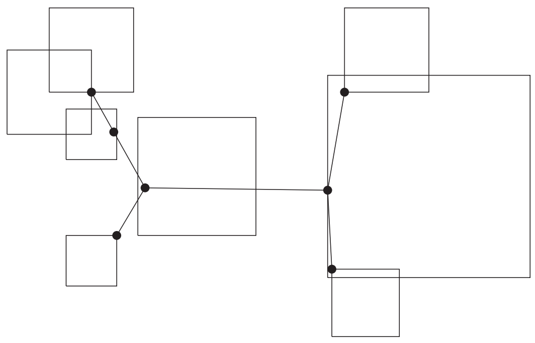
Fig. 3. A MSTN for the data in Example 2.1 for polyhedral neighborhoods.

The following example illustrates the MSTN.

Example 2.1. Let us consider a graph with eight vertices and 14 edges, $G = (V, E)$ embedded in $\mathbb{R}^2$. The graph G and an Euclidean Minimum Spanning Tree for this graph are shown in Fig. 1(A).

Fig. 1(B) shows the input graph together with the neighborhoods N_v associated with the vertices $v \in V$. The neighborhoods are (Euclidean) balls centered at the original vertices, each of them with a different radius. Fig. 2 shows an optimal MSTN solution: the location of the vertex selected in each neighborhood, as well as the final spanning tree (both in gray).

Observe that the optimal spanning tree to the classical MST problem in the original input graph shown in Fig. 1(A), with edge lengths given by the Euclidean distances between the initial vertices, is no longer valid for the MSTN. The reason is that the actual distances have been updated in order to consider the coordinates of the selected vertices, which are unknown beforehand. Note also that the structure of the original graph is somehow broken, since in the final solution some of the "initial" vertices are *merged* into a single one (note that the MST in Fig. 2 has seven vertices while the original graph had eight). This is possible only when some of the neighborhoods have a non-empty intersection.

In Fig. 3 we show an optimal solution to the MSTN in the same input graph, for a different definition of the neighborhoods. Now they are defined as boxes in the form $N_v = \{z \in \mathbb{R}^2 : |z_k - v_k| \leq r_v, k = 1, 2\}$.

As we see below, some of the properties of the standard MST extend to MSTN. In particular, the cut and cycle properties that allow reducing the dimensionality of MSTN by discarding edges that will not appear in an optimal solution as well as computing those edges that will appear in it. Before, we introduce the additional notation associated with each edge $e = \{v, w\} \in E$.

- $\widetilde{U}_e$ and $\widetilde{u}_e$ respectively denote the maximum and minimum distance between any pair of points in the neighborhoods of the end-vertices of e. That is, $\widetilde{U}_e = \max\{d(y_v, y_w) : y_v \in N_v, y_w \in N_w\}$ and $\widetilde{u}_e = \min\{d(y_v, y_w) : y_v \in N_v, y_w \in N_w\}$.

Property 1.

(a) *Let C be a cycle of $G = (V, E)$ and $e \in C$ such that $\widetilde{u}_e > \min_{e' \in E}\{\widetilde{U}_{e'} : e' \in C, e' \neq e\}$. Then, e does not belong to a MSTN.*
(b) *Let $S \subset V$ and $(S, V \setminus S) = \{e = \{v, w\} \in E \mid v \in S$ and $w \in V \setminus S\}$ be its associated cutset. Let $e = \{v, w\} \in (S, V \setminus S)$ be such that $\widetilde{U}_e < \min_{e' \in E}\{\widetilde{u}_{e'} : e' = \{v', w'\} \in E,\ e' \neq e, v' \in S, w' \in V \setminus S\}$. Then, e belongs to every MSTN.*

Proof.

(a) Let C be a cycle of $G = (V, E)$ and $e \in C$ such that $\widetilde{u}_e > \min_{e \in E}\{\widetilde{U}_{e'} : e' \in C, e' \neq e\}$.
Suppose, there is an MSTN of G, T with $e \in T$. Then, for any other edge e' in the cycle C, the tree $T' = T \cup \{e'\} \setminus \{e\}$ satisfies that:

$$d(T') \leq d(T) + \widetilde{U}_{e'} - \widetilde{u}_e < d(T).$$

Thus, the cost of T' is strictly smaller than the cost of T, contradicting the optimality of T. Hence e will not appear in T.
(b) Let T be a MSTN of G with $e \notin T$. Since T is a tree, the unique cycle of $T \cup \{e\}$ contains both e and the unique path in G connecting v and w, that does not contain e. Let e' the edge in such a path crossing the cut, i.e., $e' = \{v', w'\}$ with $v' \in S$ and w' in $V \setminus S$. Then, $T' = T \cup \{e\} \setminus \{e'\}$ is a tree and such that

$$d(T') \leq d(T) + \widetilde{U}_e - \widetilde{u}_{e'} < d(T),$$

so T' has an overall distance smaller than T, contradicting its optimality. Hence, e will appear in T. □

3. Mixed integer non linear programming formulations

In this section we present alternative MINLP formulations for the MSTN that will be compared computationally in later sections. All formulations use the following sets of decision variables:

- Binary variables $x_e \in \{0, 1\}$, $e \in E$, to represent the edges of the spanning trees.
- Continuous variables $y_v \in N_v$, $v \in V$, to represent the point selected in each neighborhood.
- Continuous variables $u_e \geq 0$, $e = \{v, w\} \in E$, to represent the distance $d(y_v, y_w)$ between the pairs of selected points.

Property 1(a) and (b) can be exploited in order to reduce the number of x variables in the formulations. In particular, we only need to define variables x_e associated with edges that do not satisfy the condition 1(a). On the other hand, we can set at value 1 all variables x_e associated with edges that satisfy 1(b).

Let $U = \{u \in \mathbb{R}_+^{|E|} : u_e \geq d(y_v, y_w)$, for all $e = \{v, w\} \in E$, for some $y \in Y\}$ denote implicitly the domain for the feasibility of the u variables. Then, a generic bilinear formulation for MSTN is

$$\min \sum_{e\in E} u_e x_e \qquad (\mathrm{P_{xu}})$$

$$\text{s.t. } x \in ST_G, \; u \in U.$$

In the following we resort to McCormick's (1976) envelopes for the linearization of the bilinear terms of the objective function. For this, we define an additional set of continuous decision variables $\theta_e \geq 0$, $e \in E$ to represent the products $u_e x_e$. Then the linearization of the generic formulation $(\mathrm{P_{xu}})$ is:

$$\min \quad \Theta = \sum_{e\in E} \theta_e \qquad (\text{RL-MSTN})$$

$$\text{s.t.} \quad \theta_e \geq u_e - \widetilde{U}_e(1 - x_e), \quad \forall e \in E, \qquad (\text{LIN-Mc})$$
$$x \in ST_G, \quad u \in U, \quad \theta_e \geq 0, e \in E.$$

Furthermore, throughout we will describe the set U using the set of constraints

$$\|y_v - y_w\| \leq u_e, \qquad \forall e = \{v, w\} \in E, \qquad (\mathrm{U_1})$$

$$y \in Y, \qquad (\mathrm{U_2})$$

which set the distance values and impose that the y points belong to the appropriate neighborhoods, respectively.

Note that the above formulation (RL-MSTN) can be reinforced by adding the following valid inequalities: $\theta_e \geq \tilde{u}_e x_e$, for all $e \in E$.

The two formulations below differ from each other in the representation of subtour elimination constraints (SEC). One of them uses the classical representation of Edmonds (1970), which consists of a family with an exponential number of inequalities.The second one uses a compact formulation based on the well-known MTZ constraints (Miller, Tucker, & Zemlin, 1960). Despite having a weaker linear programming bound than the subtour elimination representation for the classical MST problem, we use this formulation since, in practice, it has given quite good results for other problems related to spanning trees (Fernández, Pozo, Puerto, & Scozzari, 2016; Landete & Marín, 2014). Indeed, other compact representations could be used, like for instance, the one by Martin (1991). In our experience, Miller et al. (1960) gives a good tradeoff between the number of variables it requires and the bounds it produces.

3.1. MSTN formulation based on classical representation of SECs

$$\min \quad \Theta = \sum_{e\in E} \theta_e \qquad (\text{SEC-MSTN})$$

$$\text{s.t. (LIN–Mc), } (\mathrm{U_1}), (\mathrm{U_2}),$$

$$\sum_{e\in E} x_e = |V| - 1, \qquad (\mathrm{ST_1})$$

$$\sum_{e=\{v,w\}:v,w\in S} x_e \leq |S| - 1, \quad \forall S \subset V, \qquad (\mathrm{ST_2})$$

$$u, \theta \in \mathbb{R}_+^{|E|}, y \in \mathbb{R}^{|V|\times d}, x \in \{0, 1\}^{|E|}. \qquad (\mathrm{D_1})$$

Constraints $(\mathrm{ST_1})$ impose that exactly $|V| - 1$ edges are selected and subtours are prevented by $(\mathrm{ST_2})$. $(\mathrm{D_1})$ define the domain of the variables.

As mentioned, the number of constraints in $(\mathrm{ST_2})$ is exponential on $|V|$, so a separation procedure (e.g. max flow – min cut) to certify whether a solution is feasible or otherwise, to provide a violated constraint, is needed to solve this formulation. This is avoided in the next formulation, which uses the MTZ compact representation of SECs (Miller et al., 1960).

3.2. MSTN formulation based on Miller–Tucker–Zemlin

The formulation based on the MTZ representation of SECs builds a tree rooted at an arbitrarily selected vertex where the arcs of the tree are oriented *towards* the root. In our case we set vertex 1 as the root of the trees. Associated with each edge $\{v, w\} \in E$ we define two additional binary decision variables, z_{vw} and z_{wv}, to indicate whether or not (v, w) (resp. (w, v)) is used as a directed arc. The set of such arcs is denoted by A. As it is usual for the representation of the SEC constraints we use continuous variables s_v, $v \in V$, associated with the vertices. The (MTZ-MSTN) formulation is:

$$\min \qquad \Theta = \sum_{e\in E} \theta_e \qquad (\text{MTZ-MSTN})$$

$$\text{s.t. (LIN–Mc), } (\mathrm{U_1}), (\mathrm{U_2}),$$

$$x_e = z_{uv} + z_{vu}, \quad \forall e = \{u, v\} \in E, \qquad (\mathrm{MTZ_1})$$

$$\sum_{(v,1)\in\delta^-(1)} z_{v1} \geq 1, \qquad (\mathrm{MTZ_2})$$

$$\sum_{(v,w)\in\delta^-(u)} z_{vw} = 1, \quad \forall v \in V\backslash\{1\}, \qquad (\mathrm{MTZ_3})$$

$$|V|z_{vw} + s_v - s_w \leq |V| - 1, \quad \forall (v, w) \in A, \qquad (\mathrm{MTZ_4})$$

$$s_1 = 1; 2 \leq s_u \leq |V|, \quad \forall u \in V\backslash\{1\}, \qquad (\mathrm{MTZ_5})$$

$$u, \theta \in \mathbb{R}_+^{|E|}, \quad y \in \mathbb{R}^{|V|\times d}, \quad x \in \{0, 1\}^{|E|}, \qquad (\mathrm{D_1})$$

$$z \in \{0, 1\}^{|E|}, s \in \mathbb{R}_+^{|V|}. \qquad \mathrm{D_2}$$

The meaning of the new constraints is as follows. Constraints $(\mathrm{MTZ_1})$ relate the *edge* and *arc* decision variables. The connectivity with the root is guaranteed by $(\mathrm{MTZ_2})$ and $(\mathrm{MTZ_3})$. Subtours are eliminated by $(\mathrm{MTZ_4})$ and $(\mathrm{MTZ_5})$, where the later set appropriate bounds for the vertex variables s. The domain of the new variables is set by $(\mathrm{D_2})$.

As mentioned, the two formulations presented above use the *norm* constraints $(\mathrm{U_1})$ and $(\mathrm{U_2})$ to represent the distance measure for the edges and for the neighborhoods, respectively. As we see below both sets of constraints can also be handled by using either SOC or linear constraints. The following remarks show the explicit representation of some general cases of this type of constraints.

Remark 3.1 (ℓ_q-norm representation)**.** As shown in Blanco et al. (2014, Lemma 3), if the norm $\|\cdot\|$ is a ℓ_q-norm with $q \in \mathbb{Q}$ and $q = \frac{r}{s} > 1$ (with $gcd(r, s) = 1$), then the constraints of the form $\|X -$

$Y\|_q \leq Z$ as those of (U$_1$) can be rewritten as the following set of inequalities:

$$\left.\begin{array}{ll} Q_k + X_k - Y_k \geq 0, & k=1,\ldots,d,\\ Q_k - X_k + Y_k \geq 0, & k=1,\ldots,d,\\ (Q_k)^r \leq (R_k)^s Z^{r-s}, & k=1,\ldots,d,\\ \sum_{k=1}^d R_k \leq Z, & \\ R_k \geq 0, & k=1,\ldots,d, \end{array}\right\} \tag{3.1}$$

where for $k=1,\ldots,d,$ $Q_k = |X_k - Y_k|$ and $R_k = |X_k - Y_k|^q Z^{-1/\rho}$, with $\rho = \frac{r}{r-s}$.

The above gives a representation of (U$_1$) with a number of SOC inequalities that is polynomial in the dimension d and q.

Remark 3.2 (Polyhedral norm representation)**.** When the norm $\|\cdot\|$ is a polyhedral (or block) norm, a (linear) representation, much simpler than the one given in Remark 3.1 is possible. Let B^* be the unit ball of its dual norm and Ext(B^*) the set of extreme points of B^*. The constraint $Z \geq \|X-Y\|$ is then equivalent to

$$Z \geq e^t(X-Y),\ \forall e \in \text{Ext}(B^*),$$

where e^t denotes the transpose of e.

3.3. Computational comparison of the two formulations

We have performed a series of computational experiments in order to compare the performance of the two formulations (SEC-MSTN) and (MTZ-MSTN), as well as to explore the limitations of each of them. For this we have generated several batteries of instances with different settings. We consider complete graphs with a number of vertices ranging in [5, 20], and randomly generated coordinates in $\mathbb{R}^2$ and $\mathbb{R}^3$ ranging in [0, 100]. Distances are measured using the Euclidean norm and Euclidean balls are used as neighborhoods of the vertices. In addition, we consider four different scenarios for generating the radii to define the neighborhoods of each vertex in a given instance:

Small size neighborhoods ($r=1$): Radii randomly generated in [0, 5].

Small–medium size neighborhoods ($r=2$): Radii randomly generated in [5, 10].

Medium–large size neighborhoods ($r=3$): Radii randomly generated in [10, 15].

Large size neighborhoods ($r=4$): Radii randomly generated in [15, 20].

The above four cases allow us to observe the performance of the formulations for neighborhoods of varying sizes and to analyze how these sizes affect the computation the MSTN in each case. Finally, five different instances were generated for each combination of number of vertices and radii, both in the plane and in the 3D-space. The generated data are available at bit.ly/mstneigh.

All the formulations were coded in C, and solved using Gurobi 6.5 (Gurobi Optimization Inc., 2015) in a Mac OSX El Capitan with an Intel Core i7 processor at 3.3 gigahertz and 16 gigabytes of RAM. A time limit of 2 hours was set in all the experiments.

Tables 1 and 2 summarize the results of these experiments. In these tables the column *CPU*, under the heading of each formulation, reports the average computing time (in seconds) to attain optimality. Whenever the time limit of 2 hours is reached without certifying optimality, columns under *GAP* report the average percentage deviation of the best solution found during the exploration with respect to the lower bound at termination. Columns under *#Nodes* report the average number of nodes explored in the branch-and-bound search, whereas column *SEC* gives the average number of constraints (ST$_2$) incorporated to formulation (SEC-MSTN) throughout the solution process. Finally, the last column in each block reports the percentage of instances optimally solved with each formulation.

Observe that the computing times required by (SEC-MSTN) are in most cases smaller than those required by (MTZ-MSTN). Furthermore, some instances that could not be solved with (MTZ-MSTN), were optimally solved with (SEC-MSTN). In most of the cases where (SEC-MSTN) did not succeed, (MTZ-MSTN) was also not able to solve the corresponding instance. Note that, for the instances with $n=20$, we only report the results for the first scenario ($r=1$), since neither (SEC-MSTN) nor (MTZ-MSTN) were able to solve any of such instances for $r \geq 2$. We would like to highlight that, even if the 3-dimensional instances have a higher number of variables than the planar ones, the results, in terms of computing times, percentage deviations, and number of optimally solved instances are better for these instances than for the 2-dimensional ones. Observe that the difficulty of an instance is highly related to whether or not the neighborhoods have non-empty intersections; in such cases, the continuous relaxation tends to *collapse* the vertices of intersecting neighborhoods into a single one, which is not necessarily an optimal strategy. This justifies the higher difficulty of planar instances since, with uniform randomly generated points and given radii, the probability of intersection of neighborhoods is higher in case of the plane than in the space (Dufour, 1973).

4. Branch-and-cut solution algorithm

In this section we describe the branch-and-cut solution algorithm that we propose for solving MSTN. The special structure of MSTN, with disjoint domains for each set of variables – x and u – and a bilinear objective function makes it possible to apply well-known Benders-like decomposition methods (Benders, 1962; Geoffrion, 1972). This type of well-known solution schemes have been widely applied to problems with two sets of structural decision variables, in which the subproblem that results when fixing one of the sets of variables can be *efficiently* solved. Note that, as mentioned before, this requisite is satisfied in the case of MSTN.

In order to warrant the convergence properties of the approach, we also apply reformulation techniques to the bilinear objective function. For a given spanning tree $\bar{x} \in ST_G$, the "optimal" vertices and distances of its associated MSTN, can be computed by solving the following convex subproblem:

$$u(\bar{x}) = \min \sum_{e\in E} u_e \bar{x}_e \qquad \text{PU}_{\bar{x}}$$
$$\text{s.t. } u \in U$$

As already mentioned, (PU$_{\bar{x}}$) is a continuous SOC problem, which can be efficiently solved with on-the-shelf solvers. Note also that the number of u variables in (PU$_{\bar{x}}$) reduces to $n-1$, because only distances associated with the edges $e \in E$ with $\bar{x}_e = 1$ need to be computed. Hence, (generalized) Benders decomposition is a suitable methodology for solving the MSTN problem. The following result states explicitly the form of the Benders cuts that allow to use particular solutions of (PU$_{\bar{x}}$) to solve MSTN.

Theorem 4.1. *Let $\bar{x} \in ST_G$ and $u(\bar{x})$ its associated (PU$_{\bar{x}}$) solution. Then,*

$$\Theta \geq u(\bar{x}) + \sum_{e:\bar{x}_e=1} \widehat{U}_e(x_e - 1) + \sum_{e:\bar{x}_e=0} \widehat{u}_e x_e,$$

is a valid cut for MSTN, where, as before, $\Theta = \sum_{e\in E}\theta_e$ with $\theta_e \geq 0$, $e \in E$; and $\widehat{U}_e$ and $\widehat{u}_e$ are strict upper and lower bounds on the maximum and minimum values of the distance of edge e, respectively, i.e. $\widehat{U}_e > \widetilde{U}_e$ and $\widehat{u}_e < \widetilde{u}_e$ for all $e \in E$.

Proof. Let us consider the following equivalent reformulation of (PU$_{\bar{x}}$) based on the McCormick linearization of the bilinear terms

Table 1
Results of (MTZ-MSTN) and (SEC-MSTN) for $\mathbb{R}^2$ instances.

		(MTZ-MSTN)				(SEC-MSTN)				
r	n	CPU	#Nodes	GAP	%Solved	CPU	#SECs	#Nodes	GAP	%Solved
1	5	0.0652	5.40		100	0.0250	3.40	9.00		100
	6	0.0965	7.60		100	0.0334	6.40	21.20		100
	7	0.1403	84.60		100	0.0456	9.60	54.00		100
	8	0.1917	201.60		100	0.0677	9.20	41.40		100
	9	0.2592	37.60		100	0.0826	29.60	76.00		100
	10	0.4843	434.80		100	0.1318	64.60	241.40		100
	11	0.6472	568.20		100	0.3922	123.80	552.60		100
	12	0.9159	712.00		100	0.3083	156.40	547.80		100
	13	10.9525	3145.80		100	1.1175	419.00	1314.80		100
	14	4.7581	4014.80		100	1.1627	300.40	1043.60		100
	15	657.1666	41153.60		100	444.5906	1474.20	17828.00		100
	20	2915.1011	110070.80		100	840.0096	2431.20	32173.80		100
2	5	0.0820	47.00		100	0.0263	7.40	54.60		100
	6	0.1226	44.10		100	0.0451	11.90	84.80		100
	7	0.1571	123.20		100	0.0582	18.60	95.60		100
	8	0.4895	480.80		100	0.2000	98.40	457.40		100
	9	0.5531	415.80		100	0.3984	128.40	666.20		100
	10	1.3820	915.40		100	0.7600	174.40	1125.00		100
	11	1.6639	835.60		100	1.2961	235.80	1050.20		100
	12	32.8139	12301.20		100	8.2899	832.80	9301.60		100
	13	143.7873	16259.40		100	9.7330	4685.40	68409.20		100
	14	1467.5540	44337.00	7.64%	80	661.3465	3252.60	36310.60		100
	15	3428.0761	423135.80	4.97%	80	3424.9741	15712.80	179939.00	6.29%	60
3	5	0.0958	44.20		100	0.0354	9.40	79.80		100
	7	0.2645	414.60		100	0.2772	189.70	1133.40		100
	8	1.6716	2097.80		100	1.1393	338.60	1894.20		100
	9	3.7345	3827.40		100	3.8655	407.60	3515.40		100
	10	5.9807	3465.20		100	3.8294	333.80	2426.20		100
	11	713.2283	172376.20		100	976.5382	61128.20	363205.60		100
	12	1054.4171	479364.20		100	2828.2251	97800.80	576762.00		100
	13	3323.6210	279362.20	13.45%	60	4626.0085	116751.40	953914.60	20.98%	80
	14	>7200	1385623.40	30.04%	0	>7200	27120.40	162667.60	38.07%	0
	15	>7200	1473884.40	19.43%	0	>7200	87730.20	392951.00	23.65%	0
4	5	0.0886	33.20		100	0.0288	4.80	47.40		100
	6	0.1688	307.20		100	0.1797	95.80	709.20		100
	8	2.0333	1976.60		100	1.1078	289.80	1562.40		100
	9	4.4483	4936.00		100	9.3935	444.60	6657.20		100
	10	67.5709	33224.80		100	194.9068	1224.20	28680.60		100
	11	469.3033	198141.80		100	315.9130	6463.80	70995.60		100
	12	2471.0749	403914.60	6.45%	80	822.4408	105,361.40	906147.00		100
	13	4609.7707	874785.60	16.88%	40	5134.5084	8477.00	163847.00	19.64%	40
	14	>7200	807955.40	44.52%	0	>7200	37016.40	192311.20	51.26%	0
	15	>7200	948641.60	34.07%	0	>7200	29946.80	168779.80	43.33%	0

of the objective function in the original MSTN formulation:

$$u(\bar{x}) = \min \sum_{e\in E} \theta_e$$
$$\text{s.t. } \theta_e \geq u_e + \widehat{U}_e(\bar{x}_e - 1), \quad e \in E \qquad (\text{RPU}_{\text{x}})$$
$$\theta_e \geq \widehat{u}_e \bar{x}_e, \quad e \in E$$
$$u \in U.$$

Note that the reformulation (RPU_{x}) is a convex optimization problem, and Slater condition holds (Slater, 1950). Hence, (necessary and sufficient) optimality conditions can be derived from the following Lagrangian function associated with $(\text{PU}_{\bar{\text{x}}})$:

$$L(\bar{x}, \theta, u; \lambda, \mu, \nu) = \sum_{e\in E} \theta_e - \sum_{e\in E} \lambda_e(\theta_e - u_e + \widehat{U}_e(1 - \bar{x}_e)) - \sum_{e\in E} \mu_e(\theta_e - \widehat{u}_e \bar{x}_e) + \nu^t G(u),$$

where $G(u) \leq 0$ are the constraints (only involving u-variables) defining U.

Let θ_e^*, u_e^*, $e \in E$, be an optimal solution to (RPU_{x}) and λ^*, μ^* and ν^* the associated optimal multipliers. Then, λ^* and μ^* must satisfy:

$$1 - \lambda_e^* - \mu_e^* = 0, \quad \forall e \in E, \tag{4.1}$$

together with the complementary slackness constraints:

$$\lambda_e^*(\theta_e^* - u_e^* + \widehat{U}_e(1 - \bar{x}_e)) = 0, \quad \forall e \in E, \tag{4.2}$$

$$\mu_e^*(\theta_e^* - \widehat{u}_e \bar{x}_e) = 0, \quad \forall e \in E. \tag{4.3}$$

From the equations above, we get that if $\bar{x}_e = 1$, then $\mu_e^* = 0$ by (4.3), since $\theta_e^* \geq u_e^* > \hat{u}_e$. Hence, by (4.1), $\lambda_e^* = 1$. Besides, if $\bar{x}_e = 0$, by (4.2) and because $u_e^* < \widehat{U}_e$, we get that $\theta_e^* = 0$ and $\lambda_e^* = 0$. Again, applying (4.1), we derive that $\mu_e^* = 1$. Thus, we conclude that, the values of the optimal Lagrangian multipliers are:

$$\lambda_e^* = \bar{x}_e \text{ and } \mu_e^* = 1 - \bar{x}_e, \quad \forall e \in E. \tag{4.4}$$

On the other hand, since $u(x) = \Theta = \sum_{e\in E} \theta_e = \max_{\lambda\geq 0, \mu\geq 0} \min_{\theta, u} L(x, \theta, u; \lambda, \mu, \nu)$ also holds for any $x \in ST_G$, we have that

$$\Theta \geq \min_{\theta, u} L(\bar{x}, \theta, u; \lambda^*, \mu^*, \nu^*)$$
$$= \sum_{e\in E} \theta_e^* - \sum_{e\in E} \lambda_e^*(\theta_e^* - u_e^* + \widehat{U}_e(1 - \bar{x}_e)) - \sum_{e\in E} \mu_e^*(\theta_e^* - \widehat{u}_e \bar{x}_e) + \nu^{*t} G(u^*)$$
$$= \sum_{e\in E} \theta_e^* - \sum_{e\in E} \lambda_e^*(\theta_e^* - u_e^* + \widehat{U}_e(1 - x_e))$$

Table 2
Results of (MTZ-MSTN) and (SEC-MSTN) for $\mathbb{R}^3$ instances.

		(MTZ-MSTN)				(SEC-MSTN)				
r	n	CPU	#Nodes	GAP	%Solved	CPU	#SECs	#Nodes	GAP	%Solved
1	5	0.0677	3.60		100	0.0282	2.40	17.20		100
	6	0.1049	11.80		100	0.0429	3.00	14.00		100
	7	0.2137	24.40		100	0.0694	5.60	24.80		100
	8	0.2439	52.40		100	0.0813	6.20	38.40		100
	9	0.3733	166.80		100	0.1298	13.40	127.40		100
	10	0.3803	56.20		100	0.1442	34.00	127.40		100
	11	1.0249	281.40		100	0.3568	27.60	336.20		100
	12	0.6932	235.20		100	0.2772	62.00	225.00		100
	13	1.3241	763.40		100	0.9351	113.60	819.60		100
	14	4.1596	1112.00		100	2.6353	200.80	1164.60		100
	15	4.2952	1286.20		100	2.5708	197.00	812.40		100
	20	67.5323	6555.20		100	8.9617	372.20	1441.00		100
2	5	0.0983	12.40		100	0.0431	6.80	37.40		100
	6	0.1479	27.40		100	0.0497	4.70	35.30		100
	7	0.2058	51.80		100	0.0770	9.20	55.80		100
	8	0.3084	211.40		100	0.1645	49.80	263.00		100
	9	0.8943	382.00		100	0.4596	86.20	593.80		100
	10	0.5047	170.60		100	0.2185	50.60	267.80		100
	11	1.4917	653.40		100	0.5416	134.00	679.60		100
	12	3.2860	1814.40		100	5.4726	462.80	2440.20		100
	13	5.3095	1956.40		100	5.6612	437.20	2344.40		100
	14	16.8888	4485.20		100	13.0737	1108.60	9084.40		100
	15	100.5050	14664.20		100	54.8965	1524.20	12674.20		100
3	5	0.1034	12.00		100	0.0450	3.00	39.60		100
	7	0.2737	199.30		100	0.1663	79.70	428.00		100
	8	1.0901	972.40		100	1.6812	230.40	1323.80		100
	9	15.9457	3589.40		100	2.0036	295.00	3520.80		100
	10	2.0609	1124.00		100	2.2637	259.80	1459.20		100
	11	29.7077	5477.80		100	34.5579	549.20	7713.00		100
	12	330.0074	19946.80		100	531.3279	1580.20	20383.00		100
	13	1069.2640	37625.20		100	668.1420	2349.60	30331.40		100
	14	3875.3014	152561.80	15.19%	60	2519.3367	11488.00	112377.40	6.87%	80
	15	1001.7704	47758.80		100	160.5466	4114.40	37114.80		100
4	5	0.0875	21.60		100	0.0469	6.80	42.60		100
	6	0.2094	134.20		100	0.1156	28.00	255.40		100
	8	0.8188	832.20		100	1.1261	204.00	1188.60		100
	9	2.8822	2408.60		100	1.7530	329.40	4937.60		100
	10	6.4525	3461.40		100	7.0799	525.80	3539.00		100
	11	32.0012	9411.20		100	37.8657	1084.40	9208.20		100
	12	70.9765	12658.60		100	37.6467	1104.00	11910.80		100
	13	710.0275	100078.40		100	1679.7648	52401.40	287336.00		100
	14	4635.9384	287990.20	27.48%	60	6433.5763	39467.20	192079.80	25.48%	40
	15	5741.0396	115401.20	7.12%	20	3609.2785	11392.80	75087.00	10.55%	60

$$
\begin{aligned}
&\quad - \sum_{e\in E} \mu_e^*(\theta_e^* - \widehat{u}_e x_e) + \nu^{*t} G(u^*)\\
&\quad - \sum_{e\in E} \lambda_e^*(\widehat{U}_e(1-\bar{x}_e)) + \sum_{e\in E} \lambda_e^*(\widehat{U}_e(1-x_e))\\
&\quad - \sum_{e\in E} \mu_e^*(\widehat{u}_e x_e) + \sum_{e\in E} \mu_e^*(\widehat{u}_e \bar{x}_e)\\
&= u(\bar{x}) + \sum_{e\in E} \lambda_e^* \widehat{U}_e (x_e - \bar{x}_e) + \sum_{e\in E} \mu_e^* \widehat{u}_e (x_e - \bar{x}_e)\\
&= u(\bar{x}) + \sum_{e\in E:\bar{x}_e=1} \widehat{U}_e(x_e - 1) + \sum_{e\in E:\bar{x}_e=0} \widehat{u}_e x_e.
\end{aligned}
$$

This concludes the proof. □

Note that, by construction, the above generalized Benders cuts imply that, we can compare the value of the subproblem (PU$_{\bar{x}}$) associated with a given spanning tree $\bar{x} \in ST_G$, $u(\bar{x})$, with the value of the subproblem (RPU$_x$) associated with a different spanning tree $x \in ST_G$, $u(x)$. In particular, if there exist $e_1, e_2 \in E$ with $\bar{x}_{e_1} = 1$ and $x_{e_1} = 0$, and $\bar{x}_{e_2} = 0$ and $x_{e_2} = 1$, then the value of $u(x)$ is at least $u(\bar{x}) - \widehat{U}_{e_1} + \widehat{u}_{e_2}$. In other words, the difference between the values of the two subproblems is bounded by the maximum amount that can be saved (in the cost function) by removing e_1, plus the minimum gain that can be attained by adding e_2. Therefore, the relaxed master problem at the Kth iteration of the row-generation solution algorithm can be stated as:

$$
\begin{aligned}
\Theta^* = \min \quad & \Theta\\
& \Theta \geq u(\bar{x}^k) + \sum_{e:\bar{x}_e^k=1} \widehat{U}_e(x_e - 1) + \sum_{e:\bar{x}_e^k=0} \widehat{u}_e x_e, \; k = 1, \ldots, K,\\
& x \in ST_G. \qquad (4.5)
\end{aligned}
$$

The reader may note that the cuts (PU$_x$) can be interpreted as some form of lifting of the surrogated McCorminck inequalities (LIN-Mc), after projecting out the u variables in formulation (RL-MSTN).

Using the above cuts algorithmically gives rise to the solution scheme described in Algorithm 1:

The stopping criterion is that the gap between the upper and lower bound does not exceed the fixed threshold value ε.

Theorem 2.4 in Geoffrion (1972) states the finite convergence of the decomposition approach under the following assumptions: convexity and finiteness of the "separable" feasible domains, closeness of the "linking" constraints between the sets, and convexity of the objective functions. In our case, the finiteness of the number of underlying spanning trees of ST_G, the convexity of (PU$_{\bar{x}}$) for any $\bar{x} \in ST_G$, and the linear separability of the problem allows to apply the above result, which assures that Algorithm 1 terminates

```
Algorithm 1: Decomposition algorithm for solving MSTN.
Initialization: Let $x^0 \in ST_G$ be an initial solution and $\varepsilon$ a given threshold value.
    Set $LB = 0$, $UB = +\infty$, $\bar{x} = x^0$.
while $|UB - LB| > \varepsilon$ do
    1. Solve (4.5) for $\bar{x}$ to get $u(\bar{x})$.
    2. Add the cut $\Theta \geq u(\bar{x}) + \sum_{e:\bar{x}_e=1} \widehat{U}_e(x_e - 1) + \sum_{e:\bar{x}_e=0} \widehat{u}_e x_e$ to the current master problem.
    3. Obtain the optimal value $\bar{\Theta}$ to the current master problem, and its associated solution $\bar{x}$.
    4. Update $LB = \max\{LB, \bar{\Theta}\}$ and $UB = \min\{UB, \sum_{e\in E} u(\bar{x})_e \bar{x}_e\}$
end
```

in a finite number of steps (for any given $\varepsilon \geq 0$). Moreover, if $\varepsilon \leq \min\{\tilde{U}_{e_1} - \tilde{u}_{e_2} \geq 0 : e_1 \neq e_2 \in E\}$, it outputs an optimal MSTN.

To avoid the enumeration of all spanning trees of G, and to reduce the number of iterations, several recipes can be applied. One of them is to start with a non-empty set of cuts which give a suitable initial representation of the lower envelope of Θ. Hence, if $\overline{ST}_G$ denotes the set of trees associated with the current set of constraints (PU$_x$), the representation we use for the master problem is:

$$\min \quad \sum_{e\in E} \theta_e \tag{4.6}$$

$$\text{s.t.} \sum_{e\in E} \theta_e \geq u(\bar{x}) + \sum_{e:\bar{x}_e=1} \widehat{U}_e(x_e - 1) + \sum_{e:\bar{x}_e=0} \widehat{u}_e x_e, \forall \bar{x} \in \overline{ST}_G, \tag{4.7}$$

$$\theta_e \geq \tilde{u}_e x_e, \quad e \in E,$$

$$x \in ST_G.$$

Given that the master problem exhibits a combinatorial nature, the performance of a Benders-like algorithm can be improved by embedding the cut generation mechanism within a branch-and-cut scheme. This is the current trend nowadays (Fischetti, Ljubic, & Sinnl, 2016a; 2016b). This requires to separate the optimality cuts in addition to any other generated cuts, at the nodes of the enumeration tree. Note that this approach is also valid in our case, as the cuts (4.7) are also valid if $\bar{x}$ is the solution to a linear programming relaxation of a valid MST formulation.

4.1. Computational experiments

The proposed decomposition approach has been tested over the same set of benchmark instances used to compare the compact formulations (see Section 3.3). Based on the results obtained in such a comparison, and also to take advantage of the possibility of adding dynamically violated SECs within the branch-and-cut, we combine the decomposition approach with the classical SEC representation (SEC-MSTN). In addition to the average statistics reported in the previous tables (CPU, #SECs, #Nodes, GAPs, and %Solved), we also report now the average number of Benders' type cuts, #BendersCuts, and the gap after the exploration of the root node of the branch-and-cut tree, %GAP$_0$. Average results for the 4 scenarios are reported in Tables 3 and 4.

As can be seen, the computing times required by the decomposition approach are smaller than those obtained with the MINLP formulations for the small size radii scenario and also in the small-medium size radii scenario for the 3D case. However, the results obtained for the medium-large and large size scenarios reveal that the MINLP formulations have a better performance than the decomposition scheme. Note that the cuts induced by our approach depends of the available upper and lower bounds on the lengths of the edges in the graph. These bounds are tight for the small size radii scenarios, but far from being a representative value of the actual length of the edge in the remaining scenarios. Hence, a large number of cuts are needed to certify optimality of the solution in these cases.

Table 3
Average results for the decomposition approach for $\mathbb{R}^2$ instances.

r	n	CPU	#SEC	#BendersCuts	#NodesB%B	%GAP$_0$	%GAP	%Solved
1	5	0.0065	1.20	0.20	0.00	5.45		100
	6	0.0196	3.60	2.40	10.40	18.99		100
	7	0.0328	5.60	4.00	22.80	12.07		100
	8	0.0347	3.60	3.80	23.40	15.41		100
	9	0.0646	12.80	7.60	64.60	18.79		100
	10	0.1796	26.60	23.40	180.00	20.06		100
	11	0.5341	116.60	68.40	950.60	28.48		100
	12	0.6484	213.20	71.80	1129.00	32.67		100
	13	1.5531	246.20	167.60	2573.80	37.76		100
	14	1.6703	300.60	177.00	2204.20	32.39		100
	15	45.3193	1016.40	1637.40	23077.60	47.74		100
	20	333.5085	1628.60	3721.80	59876.80	39.75		100
2	5	0.0464	4.20	6.40	25.60	29.32		100
	6	0.0730	6.70	11.40	50.90	24.67		100
	7	0.0678	12.20	10.80	78.20	28.19		100
	8	0.2743	21.60	43.60	311.60	41.06		100
	9	0.3111	55.20	46.80	492.20	30.63		100
	10	0.4646	78.00	66.60	721.40	33.22		100
	11	1.3472	245.40	167.80	2382.60	35.11		100
	12	160.8519	864.80	3027.00	36569.60	61.72		100
	13	326.1787	1598.20	2800.40	50047.80	50.47		100
	14	226.5067	2024.20	6463.60	96243.00	43.07		100
	15	5824.7652	8023.00	18775.80	284590.80	73.80	3.76	20
3	5	0.1152	3.80	5.80	24.40	27.67		100
	7	0.4851	58.10	93.50	712.60	50.67		100
	8	3.2475	158.20	526.80	3963.60	59.22		100
	9	17.3417	521.00	1492.40	14560.00	67.32		100
	10	5.8312	226.20	595.00	5933.40	50.17		100
	11	2603.6210	4308.40	12569.00	168712.00	75.77	40.36	80
	12	>7200	5223.40	23172.40	275986.80	81.98	22.01	0
	13	>7200	7191.60	20230.60	282031.60	85.37	20.33	0
	14	>7200	15425.00	14481.80	311567.60	90.64	53.59	0
	15	>7200	11379.40	13846.80	310549.80	83.69	35.16	0
4	5	0.0476	3.80	5.80	24.20	33.07		100
	6	0.3993	36.20	83.80	428.60	56.12		100
	8	2.9985	187.20	424.40	3055.80	62.99		100
	9	53.7040	418.00	2631.80	23586.40	67.46		100
	10	1013.3837	1444.00	7611.00	72987.20	82.73		100
	11	4256.8194	4636.60	16430.60	204272.20	84.16	30.36	60
	12	6232.3367	6569.80	20400.00	250014.00	77.37	16.60	20
	13	>7200	8218.80	19321.40	299586.40	85.78	29.58	0
	14	>7200	13880.00	13080.80	336546.40	93.16	71.25%	0
	15	>7200	16128.60	12538.00	326406.20	94.80	50.14%	0

5. A mathheuristic for MSTN

The results of the computational experiments section indicate that MSTN instances with up to less than 15 vertices can be optimally solved within the allowed time limit, but as the sizes of the instances increase the computing times become prohibitive. Below we present a mathheuristic alternative to obtain near-optimal solutions to larger MSTN instances. The main idea under the proposed algorithm is based on the observation that the problem is a biconvex problem, since fixing any of the set of variables the problem becomes an efficiently solvable optimization problem (in case x is fixed, the problem is a continuous SOCP, while if u is fixed, the problem is a standard MST problem).

The mathheuristic consists of two embedded loops. The outer loop is a multistart procedure. The input of each iteration in this loop is a spanning tree, which will be used in the initial iteration of the inner loop. The number of iterations of the outer loop is a parameter related to the initial spanning tree generation mechanism that we use, which will be explained later on.

Table 4
Average results for decomposition approach for $\mathbb{R}^3$ instances.

r	n	CPU	#SEC	#BendersCuts	#NodesB%B	%GAP$_0$	%GAP	%Solved
1	5	0.0063	0.80	0.00	0.00	2.28		100
	6	0.0125	1.60	0.60	0.00	4.53		100
	7	0.0138	2.20	1.80	9.80	9.31		100
	8	0.0445	3.60	3.40	18.80	12.49		100
	9	0.0573	6.00	5.80	28.60	9.66		100
	10	0.0883	9.60	9.20	72.60	8.28		100
	11	0.2478	30.20	17.20	162.00	17.26		100
	12	0.2455	59.00	25.00	314.20	16.00		100
	13	0.8280	87.60	85.20	1035.40	17.73		100
	14	1.1512	194.80	95.20	1535.20	11.92		100
	15	1.7121	264.00	130.20	1761.60	18.39		100
	20	8.2175	702.20	377.80	7398.60	16.68		100
2	5	0.0218	3.80	2.40	7.80	12.34		100
	6	0.0292	2.40	3.30	12.50	8.57		100
	7	0.0388	4.20	4.60	18.80	18.30		100
	8	0.2397	24.00	33.40	247.00	23.13		100
	9	0.2389	26.00	32.60	304.00	18.73		100
	10	0.2859	50.80	33.00	398.00	12.74		100
	11	0.5181	58.00	57.80	555.80	20.94		100
	12	4.8255	263.20	369.80	5574.80	28.77		100
	13	5.6111	498.60	635.80	9576.20	28.87		100
	14	11.3739	1388.00	1459.40	32630.40	27.78		100
	15	35.4121	1873.00	2982.00	67628.20	33.80		100
3	5	0.0281	2.80	2.60	10.00	16.98		100
	6	0.2437	26.80	43.80	276.40	29.17		100
	7	0.2725	39.60	42.60	348.20	38.34		100
	8	1.5945	131.40	235.40	1915.20	49.20		100
	9	3.9492	292.40	1025.80	9022.00	45.81		100
	10	2.5790	313.00	272.40	3468.00	26.01		100
	11	55.9248	689.40	1979.60	26140.60	42.86		100
	12	1258.5048	2060.40	8294.40	130089.80	47.82		100
	13	3005.2253	5083.40	10824.20	212760.60	44.99	3.81	60
	14	>7200	9029.40	15154.60	288000.20	53.37	17.53	0
	15	1751.1580	9504.80	10049.00	243900.00	40.75		100
4	5	0.0312	3.00	3.60	16.60	19.69		100
	6	0.1750	13.80	29.60	122.20	27.05		100
	7	0.6724	54.80	94.20	543.60	22.12		100
	8	1.6626	162.80	218.80	1898.40	46.48		100
	9	9.5678	326.60	916.20	8138.60	45.42		100
	10	22.7335	576.60	1450.40	17267.40	47.61		100
	11	107.0304	1037.60	3051.40	40153.60	50.56		100
	12	1005.8061	1904.80	6533.00	99639.80	50.15		100
	13	999.9207	5066.20	12905.60	211964.00	50.13		100
	14	7200.3120	9951.60	14550.00	285772.40	70.62	30.61%	0
	15	6123.5383	12659.40	12203.20	266014.20	55.75	16.35%	20

The rationale of the inner loop is to alternate in solving subproblems in the solution spaces of the two main sets of variables (x and u). We proceed iteratively, and each iteration consists of solving a pair of subproblems, one in each space of variables. When solving the subproblem in one solution space we fix the values of the variables of the other space.

Formally, let $(P_{\bar{x}u})$ and $(P_{x\bar{u}})$, respectively, denote the subproblems of the generic MSTN formulation $(\mathrm{P_{xu}})$ of Section 3, when $\bar{x}$ and $\bar{u}$ are fixed. That is,

$$\min \sum_{e\in E} u_e \bar{x}_e \qquad (\mathrm{PU_x})$$
$$\text{s.t. } u \in U$$

and
$$\min \sum_{e\in E} \bar{u}_e x_e \qquad (\mathrm{PX_u})$$
$$\text{s.t. } x \in ST_G.$$

Fig. 4 shows a flowchart of the inner loop of the mathheuristic. We start with a given spanning tree T^0 associated with a solution x^0. In the kth iteration, we compute the distances $u(x^k)$ in the current tree T^k and update the vector $\bar{u}^{k+1}$ according to $\bar{u}^k$ and $u(x^k)$. In the first iteration we use the distance lower bounds $\bar{u}^0 = \widetilde{u}$. At each iteration $k > 0$ we first solve problem $(\mathrm{PX}_{\bar{u}^k})$ and then compute the vertices distances $u(x^k)$ in its optimal tree T^k, by solving (PU_{x^k}). All components $\bar{u}_e^k$ associated with edges $e \in T^k$ are updated to the corresponding component of the distances vector $u(x^k)$. The remaining components remain unchanged. The procedure terminates when two consecutive iterations produce the same tree or a maximum number of iteration is attained.

For the sake of analyzing the quality of solutions obtained with the mathheuristic we introduce the notion of *partial optimal MSTN* adapting the notation in Wendell and Hurter (1976) for the general case of minimizing a non-separable function subject to disjoint constraints.

Definition 5.1 (Partial optimum MSTN). Let $\bar{x} \in ST_G$ and $\bar{u} \in U$. $(\bar{x}, \bar{u})$ is said a partial optimum MSTN if:

$$\sum_{e\in E} \bar{x}_e \bar{u}_e \le \sum_{e\in E} x_e \bar{u}_e \quad \textit{and} \quad \sum_{e\in E} \bar{x}_e \bar{u}_e \le \sum_{e\in E} \bar{x}_e u_e$$

for all $x \in ST_G$ and $u \in U$.

Observe that a partial optimum MSTN $(\bar{x}, \bar{u})$ implies that $\bar{x}$ is a MST for the weights $\bar{u}$ and that $\bar{u}$ are the optimal distances with respect to $\bar{x}$. The following result states the partial optimality of the solutions generated by the proposed mathheuristic.

Theorem 5.2. *The sequence of objective values produced at the inner loop of the mathheuristic, corresponding to a given initial solution, converges monotonically to a partial optimum MSTN.*

Proof. Let $f(x, u) = \sum_{e\in E} x_e u_e$ denote the objective function value associated with a given solution $x \in ST_G$, $u \in U$. Let also $x^1, \ldots, x^k \in ST_G$ and $u^1, \ldots, u^k \in U$ be the solutions obtained in the first k steps of the alternate convex search for a given initial solution.

Observe that in the mathheuristic, for u^j given, x^{j+1} is obtained by solving $(\mathrm{PX}_{\bar{u}})$ with weights $\bar{u} = u^j$. Hence,

$$\sum_{e\in E} x_e^{j+1}\, u_e^j \le \sum_{e\in E} x_e\, \bar{u}_e^j, \forall x \in ST_G.$$

Next, solving $(\mathrm{PU}_{\bar{x}})$ with $\bar{x} = x^{j+1}$, one obtains $u(x^{j+1})$ and then u^{j+1} with:

$$\sum_{e\in E} x_e^{j+1}\, u(x^{j+1})_e = \sum_{e\in E} x_e^{j+1}\, u_e^{j+1} \le \sum_{e\in E} x_e^{j+1}\, u_e, \forall u \in U.$$

Hence, $f(x^k, u^k) \ge f(x^k, u(x^k)) \ge f(x^{k+1}, u^{k+1})$, so the sequence $\{f(x^j, u^j)\}_{j\in\mathbb{Z}_+}$ is monotonically non-increasing. Thus, since $f(x, u) \ge 0$ for all $x \in ST_G$ and $u \in U$, the sequence of objective values converges.

Let $\Theta^* = \lim_{j\to\infty} f(x^j, u^j)$ and $x^* \in ST_G$, $u^* \in U$ such that $f(x^*, u^*) = \Theta^*$. Since ST_G and U are closed sets and f is continuous, we have that taking limits:

$$\Theta^* = \sum_{e\in E} x_e^* u_e^* \le \sum_{e\in E} x_e u_e^* \quad \textit{and} \quad \Theta^* = \sum_{e\in E} x_e^* u_e^* \le \sum_{e\in E} x_e^* u_e.$$

Thus, (x^*, u^*) is a partial optimum MSTN. □

Since only partial optimality of the solutions is assured at the end of each inner loop, it is possible that the mathheuristic gets trapped at a local optimum. Hence we have incorporated a multistart outer loop to allow escaping from local optimal. Note that the mathheuristic becomes an exact solution method if all possible spanning trees are considered as initial solutions. However, complete enumeration is prohibitive, even if the number of potential MSTs is finite (despite using varying weights). On the other hand, we have observed that (i) the mathheuristic is sensitive to the provided initial feasible solution, and; (ii) in many cases, a few changes over an initial standard MST with respect to the distances between the centers of the neighborhoods are enough to

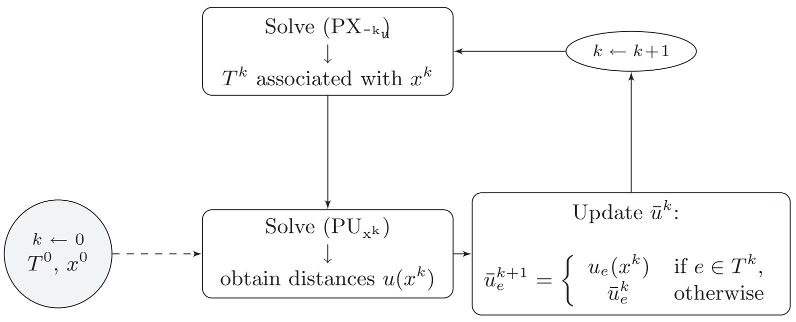
Solve (PX-$_{k}$)
T^k associated with x^k
$k \leftarrow k+1$
$k \leftarrow 0$
T^0, x^0
Solve (PU$_{x^k}$)
obtain distances $u(x^k)$
Update $\bar{u}^k$:
$\bar{u}_e^{k+1} = \begin{cases} u_e(x^k) & \text{if } e \in T^k, \\ \bar{u}_e^k & \text{otherwise} \end{cases}$

Fig. 4. Flowchart of the inner loop of the mathheuristic.

find an optimal MSTN solution. Hence, we generate the set of initial spanning trees for the multistart procedure with an adaptation of the method proposed in Sörensen and Janssens (2005), which is described in Algorithm 2. In principle, this method generates the whole set of spanning trees on a given graph (by increasing order values relative to a given weight vector). In our adaptation, we stop generating new spanning trees, when one of the following criteria is met: (1) a given number of MSTs has already been generated; or, (2) no improvement has been obtained, in the MSTNs obtained in the inner iterations, for a given number of outer iterations.

Algorithm 2: Initial solutions for the multistart procedure.

```
Initialization: $u^0_{vw} = \|v - w\|$, $\forall v, w \in V$ and $T^0$ the MST with
    respect to $u^0$, $\mathcal{T} = \{T^0\}$.
for $T \in \mathcal{T}$ do
    Let $e_1, \ldots, e_{n-1}$ be the edges of $T$.
    for $i = 1, \ldots, n-1$ do
        Construct the MST with respect to $u^0$, $T_i$, such that $e_i$
        does not belong to the tree but $e_1, \ldots, e_{i-1}$ are part of
        it. Let $c_i$ be the weight of $T_i$.
    end
    Choose $T' \in \{T_1, \ldots, T_{n-1}\}$ with $c(T') = \min_{i=1,\ldots,n} c_i$ and add it
    to $\mathcal{T}$.
end
```

A series of computational experiments have been performed to analyze the computing times and the quality of the solutions obtained with the overall heuristic. We report results based on two batteries of benchmark instances. The first one is the same that was used in our previous experiments. Here the goal is to compare the quality of the solutions obtained by the exact and the heuristic methods. The second one contains larger size instances and the goal is to explore the limit of the mathheuristic. In the experiments we do not fix limits on the number of inner iterations but we set up the maximum number of trees generated (outer iterations) to $100 \times |E|$. Table 5 summarizes the obtained numerical results. We report average values of the computing times consumed the mathheuristic (CPU) and the percentage deviation (%Dev) with respect to the optimal (or best-known) solutions obtained with the exact approaches. Observe that the quality of the solutions is extremely good, as the maximum %Dev obtained in all the experiments was 1.3086%. Furthermore, in most of the cases where the exact approaches did not prove the optimality of the best solution found, the heuristic produced a better solution. Indeed, many of the proven optimal solutions obtained with the other approaches, were also obtained with the mathheuristic. Moreover, in some cases in which our exact approaches were not able to certify optimality within the time limit, the matheuristic gives better solutions. Tables 6 and 7 show the results for the largest instances. We report, apart from the average computing times, the percentage deviations with respect to available lower (%Dev LB) and upper bounds (%Dev UB) for the optimal value of the MSTN. Lower bounds were calculated by computing the MST with respect to the original graph in which the edge lengths are given as the minimum distance between the neighborhoods that contain the vertices of each edge, i.e.:

$$\bar{u}_e = \min\{d(y_v, y_w) : y_v \in N_v, y_w \in N_w\}, \quad \text{for } e = \{v, w\} \in E.$$

Table 5
Average results for the mathheuristic.

		2-dimensional instances		3-dimensional instances	
r	*n*	CPU	%Dev	CPU	%Dev
1	5	0.1004	0.0000	0.1594	0.0000
	6	0.2068	0.0000	0.2200	0.0000
	7	0.3368	0.0433	0.3614	0.0001
	8	0.5220	0.0000	0.7036	0.0000
	9	0.6982	0.0000	0.6792	0.0195
	10	1.2014	0.1768	1.2254	0.0000
	11	1.8868	0.2679	2.1230	0.3749
	12	2.4382	0.0000	2.3078	0.0000
	13	3.0136	0.1319	4.1954	0.1223
	14	3.9986	0.1802	4.0428	0.0527
	15	5.9238	0.3095	5.4956	0.2659
	20	15.3978	0.2068	15.4622	0.0565
2	5	0.1788	0.0000	0.2416	0.0001
	6	0.2603	0.0011	0.3098	0.0000
	7	0.3972	0.1528	0.5358	0.0000
	8	0.8566	0.0000	1.3224	0.0000
	9	0.9240	0.6322	0.9988	0.3318
	10	1.4706	0.1666	1.6722	0.0296
	11	2.0872	0.8081	2.5434	0.3964
	12	3.1428	0.0212	4.2852	0.2285
	13	3.7266	0.5755	6.3750	0.3975
	14	5.6144	0.5838	6.5618	0.0270
	15	9.1994	−0.0408	10.2092	0.3245
3	5	0.1710	0.0000	0.2370	0.0000
	6	0.2134	0.0000	0.6210	0.0000
	7	0.5969	0.1360	0.7737	0.0713
	8	0.9008	0.1571	1.3504	0.0271
	9	1.3432	1.3086	2.3226	0.7177
	10	1.8258	0.8340	2.6464	0.4596
	11	3.0670	0.1899	4.4142	1.1838
	12	4.3984	0.1122	5.2298	0.0581
	13	4.9976	0.4673	7.1142	1.2851
	14	6.7682	−0.1210	10.2342	−0.1614
	15	8.2982	−0.0949	11.2072	0.2390
4	5	0.1664	0.0000	0.2738	0.0000
	6	0.3942	0.1012	0.4942	0.5379
	7	0.7893	0.0601	0.9942	0.1123
	8	1.1640	0.0000	1.6256	0.0353
	9	1.5462	0.7477	1.8514	0.4004
	10	2.2468	1.1261	2.6576	1.3283
	11	3.2060	0.7875	3.6996	0.6159
	12	4.5152	0.2935%	4.8816	0.1611
	13	5.0992	0.7808	7.2430	1.0225
	14	6.8126	−0.1978	9.6768	0.6739
	15	8.1124	0.0105	11.6100	−0.2135

Table 6
Average results for the mathheuristic for large instances in the planar case.

r	$\|V\|$	CPU	%Dev LB	% Dev UB	% MST
1	20	14.5532	23.0877	0.1201	40.00
	25	27.1624	27.6163	0.2969	40.00
	30	54.5254	27.8230	0.3004	40.00
	35	82.7320	28.8806	0.1985	40.00
	40	122.8916	28.7590	0.3342	40.00
	45	182.2026	38.7607	0.1451	80.00
	50	255.4392	43.0832	0.0912	80.00
	60	472.9626	40.6246	0.2814	20.00
	70	724.8468	43.4054	0.1118	80.00
	80	751.3728	47.7128	0.3567	40.00
	90	1064.7958	49.2007	0.0000	100.00
	100	1480.0034	53.4484	0.1639	80.00
2	20	16.4950	62.3051	1.3996	0.00
	25	31.6210	77.3769	0.3444	20.00
	30	59.5594	77.6920	1.5311	0.00
	35	87.8010	86.6972	2.4308	0.00
	40	145.0846	87.3522	1.2426	40.00
	45	192.4576	84.7788	0.7022	60.00
	50	283.2516	91.5316	1.0501	40.00
	60	525.9362	96.1926	1.6971	0.00
	70	835.0496	96.2605	0.8858	20.00
	80	779.3946	97.2727	0.9087	40.00
	90	1122.9898	98.3883	0.5728	60.00
	100	1548.9070	99.3069	1.4232	40.00
3	20	16.0632	90.6985	2.2212	20.00
	25	32.1278	96.2322	0.7643	20.00
	30	65.7792	97.5944	1.0350	0.00
	35	90.1888	98.4009	5.9840	0.00
	40	137.5042	99.0318	2.0271	0.00
	45	198.4974	99.0682	1.0427	40.00
	50	268.2828	99.8648	2.2477	20.00
	60	502.3478	100.0000	3.2364	0.00
	70	816.0300	100.0000	2.7085	20.00
	80	756.5704	100.0000	2.3165	40.00
	90	1116.6500	100.0000	1.8877	40.00
	100	1530.6052	100.0000	1.5370	20.00
4	20	16.4998	97.9307	2.7959%	20.00
	25	33.8690	99.3203	1.6366	20.00
	30	61.1976	100.0000	2.6932	0.00
	35	89.4202	100.0000	8.7080	0.00
	40	146.1266	100.0000	3.3380	0.00
	45	213.8344	100.0000	3.0796	20.00%
	50	282.9736	100.0000	2.0663%	20.00
	60	486.8964	100.0000	4.5859	0.00
	70	763.0016	100.0000	4.2135	0.00
	80	748.1272	100.0000	4.5767	0.00
	90	1085.8690	100.0000	3.2538	20.00
	100	1668.2424	100.0000	2.7675	20.00

Table 7
Average results for the mathheuristic for large instances in the 3D case.

r	$\|V\|$	CPU	%Dev LB	% Dev UB	% MST
1	20	14.6272	10.3986	0.0467	80.00
	25	40.6772	13.0944	0.0378	60.00
	30	69.8356	10.6289	0.0006	80.00
	35	106.5134	11.2375	0.1286	60.00
	40	175.6634	11.0897	0.1831	40.00
	45	262.2358	15.1212	0.0322	80.00
	50	370.5236	17.9594	0.2323	60.00
	60	631.6412	14.9262	0.0000	100.00
	70	1071.5590	18.0318	0.1747	60.00
	80	1071.1360	17.2028	0.1713	60.00
	90	1570.6312	17.1973	0.0046	80.00
	100	2256.3462	20.5805	0.1206	60.00
2	20	24.0912	34.4738	0.9106	20.00
	25	49.7172	47.0066	0.4466	20.00
	30	81.0262	40.1495	1.3887	20.00
	35	123.2108	45.9130	0.4637	60.00
	40	211.2694	48.8337	0.9941	20.00
	45	295.5366	52.4260	0.2171	60.00
	50	401.4358	55.8653	0.5822	60.00
	60	743.1540	61.8838	0.2815	60.00
	70	1139.6448	68.2234	0.7040	40.00
	80	1145.8188	69.4113	0.4693	40.00
	90	1835.7320	71.7928	0.5406	40.00
	100	2456.1402	77.1601	0.1699	60.00
3	20	24.9052	66.9737	2.4841	20.00
	25	51.9204	76.8203	2.8566	0.00
	30	83.2864	75.1517	3.4033	20.00
	35	136.2574	83.2923	0.8824	40.00
	40	207.1532	82.1425	3.4419	0.00
	45	293.3924	85.7698	1.1218	20.00
	50	431.9292	91.9528	1.6269	40.00
	60	741.9330	96.3082	2.9933	20.00
	70	1163.3446	97.8903	2.2103	0.00
	80	1231.5932	97.5674	0.9325	40.00
	90	1770.6206	98.3531	1.5740	20.00
	100	2357.2434	98.5889	2.7997	20.00
4	20	24.4860	90.5059	4.3812	0.00
	25	50.6444	93.6932	2.9003	0.00
	30	84.3946	96.3750	4.9004	20.00
	35	134.4824	97.3869	2.5519	20.00
	40	213.2442	98.0207	5.4728	0.00
	45	304.6368	99.5034	1.9230	0.00
	50	415.3388	99.3344	3.2609	0.00
	60	721.3308	99.9964	2.7762	20.00
	70	1189.9664	100.0000	3.0113	0.00
	80	1233.2842	100.0000	2.1201	20.00
	90	1922.6220	100.0000	2.3436	0.00
	100	2412.5672	100.0000	2.9934	20.00

Upper bounds are computed as the optimal value of ($PU_{\bar{x}}$), when $\bar{x}$ is the standard MST. Finally, column % MST in Tables 6 and 7 reports the percentage of instances (out of 5) in which the solution of the matheuristic coincides with the upper bound (i.e. the underlined MSTN equals the MST). As expected, the deviations with respect to the lower and upper bounds increases as the radii of the neighborhoods do. The same happens with the number of instances in which the solutions of the MSTN coincide with those of MST. In scenario 4, the instances with largest radii, the lower bounds are close to zero in most of the cases since almost all pairs of neighborhoods intersect, and several 100% deviations were obtained. The reader may observe that deviation with respect to lower bounds are few significative since these bounds are always rather far from the actual optimal solution. We would also like to emphasize that computing times for the 3-dimensional instances are slightly larger than those obtained for the planar instances. This behaviour is caused by the higher number of variables of the problems ($PU_{\bar{x}}$) that must be iteratively solved in the inner loop of the algorithm. However, the times do not seem to largely depend of the size of the neighborhoods.

6. Concluding remarks

We analyzed the problem of finding Minimum Spanning Trees with neighborhoods, where the neighborhoods are defined as SOC-representable objects and the lengths of the arcs in the graph are induced by a ℓ_q norm. Two MINLP formulations are provided whose differences come from the representation of the subtour elimination constraints. We propose a decomposition-based methodology to solve the problem based on the efficiency of solving SOCP problems. Furthermore, a new mathheuristic procedure is applied to solve the problem exploiting not only the SOC-representability of the neighborhoods but also that the MST problems are easily solvable. The results of an extensive computational experience are reported to compare all formulations and procedures provided throughout this paper. In this paper, the results of the experiments for Euclidean distances and ℓ_2-based neighborhoods are reported. We have also performed the same experiments for ℓ_1-norm based distances and rectangular neighborhoods. They are shown in Tables A.9–A.15 in the Appendix.

In addition, we have performed some experiments in order to compare our mathematical programming approaches against

Table 8
Comparisons of brute-force enumeration with respect to our approach for small-size instances.

			ℓ_2		ℓ_1	
n	$\#ST_G$	List ST_G	BF	(SEC-MSTN)	BF	(SEC-MSTN)
5	125	0.040	0.11	0.0250	0.03	0.0076
6	1296	0.032	1.39	0.0334	0.27	0.0143
7	16,807	0.072	20.77	0.0456	3.73	0.0161
8	262,144	0.163	359.54	0.0677	60.17	0.0231
9	4,782,969	3.230	7616.38	0.0826	1187.91	0.0382

brute-force enumeration of the spanning trees and their cost evaluation by solving problem ($P_{\bar{x}u}$) for each of them. In Table 8 we report the times for solving the MSTN of the planar instances used in our computational experiments, both with a brute force strategy (BF), and with our formulation (SEC-MSTN). We show average results for both for Euclidean (ℓ_2) norm with disk-neighborhoods, and ℓ_1-norm with rectangular neighborhoods, both for the scenario $r = 1$. The enumeration of the spanning trees ($\#\ ST_G$) for a given undirected graph was performed by using the algorithm provided in Shioura, Tamura, and Uno (1997) whose complexity is $O(|V| + |E|)$ and their computation times (in seconds) are also reported in the third column of the table (List ST_G).In view of the results, one could estimate that for a complete graph with 15 vertices and (optimistically) assuming that, once a spanning tree is provided, each problem ($P_{\bar{x}u}$) is solved in 10^{-5} seconds, the overall problem would be solved in $19,461,950,684$ seconds (roughly 625 years), plus the time for listing all the spanning trees of the complete graph. Our formulations solve these instances, in average, in less than 7.5 minutes using (SEC-MSTN), and less than 46 seconds applying our decomposition scheme (Algorithm 1).

Acknowledgments

The first and third authors were partially supported by the projects MTM2013-46962-C2-1-R and MTM2016-74983-C2-1-R (MINECO, Spain). The second author was partially supported by the project MTM2015-63779-R (MINECO, Spain). Thanks are due to the anonymous referees for their constructive comments.

Appendix A. Computational experiments for ℓ_1-norm based distances and rectangular neighborhoods

Table A9
Results of MSTN–MTZ and MSTN–SEC for planar instances with ℓ_1 norm and rectangular neighborhoods.

		MTZ				SEC				
r	n	CPU	#Nodes	GAP	%Solved	CPU	#SECs	#Nodes	GAP	%Solved
1	5	0.0122	2.00	0	100	0.0035	3.40	0.00	0	100
	6	0.0099	0.00	0	100	0.0076	8.00	8.20	0	100
	7	0.0173	13.60	0	100	0.0143	8.00	0.00	0	100
	8	0.0270	35.20	0	100	0.0161	9.80	1.00	0	100
	9	0.0302	2.40	0	100	0.0231	13.40	7.80	0	100
	10	0.0620	59.80	0	100	0.0382	19.60	16.80	0	100
	11	0.0619	36.60	0	100	0.0419	22.60	56.80	0	100
	12	0.1014	105.80	0	100	0.0642	30.40	42.20	0	100
	13	0.1556	271.20	0	100	0.1025	128.00	349.40	0	100
	14	0.1388	221.80	0	100	0.0840	52.00	146.00	0	100
	15	0.3307	1682.80	0	100	0.3354	437.20	1617.60	0	100
	20	1.1922	3835.00	0	100	1.3981	999.60	3234.80	0	100
2	5	0.0140	6.20	0	100	0.0085	4.60	4.60	0	100
	6	0.0155	1.00	0	100	0.0098	8.17	21.67	0	100
	7	0.0207	29.20	0	100	0.0171	12.20	11.60	0	100
	8	0.0379	55.60	0	100	0.0233	14.60	28.80	0	100
	9	0.0557	14.60	0	100	0.0310	17.00	19.60	0	100
	10	0.0607	75.40	0	100	0.0446	19.80	16.20	0	100
	11	0.0958	175.80	0	100	0.0618	40.00	94.60	0	100
	12	0.1958	481.20	0	100	0.2248	285.80	1657.80	0	100
	13	0.3594	1777.60	0	100	0.5427	602.80	2569.60	0	100
	14	0.5830	2236.80	0	100	0.3942	339.80	1620.80	0	100
	15	2.4404	14591.80	0	100	1.9606	825.80	3918.00	0	100
3	5	0.0146	8.00	0	100	0.0082	4.20	7.00	0	100
	6	0.0124	0.00	0	100	0.0073	4.25	5.25	0	100
	7	0.0342	50.40	0	100	0.0194	21.60	69.40	0	100
	8	0.0859	386.20	0	100	0.0379	95.60	417.60	0	100
	9	0.1039	160.40	0	100	0.1383	372.00	1645.60	0	100
	10	0.1108	282.00	0	100	0.0678	125.20	553.20	0	100
	11	0.5803	3795.60	0	100	0.6030	831.80	4209.00	0	100
	12	0.9778	5493.40	0	100	1.2976	823.00	5939.40	0	100
	13	1.8502	12233.60	0	100	1.7234	1022.00	5384.00	0	100
	14	3.2169	58069.80	0	100	2.5580	17408.40	65716.80	0	100
	15	13.9309	56326.40	0	100	11.0653	3167.60	24217.00	0	100
4	5	0.0097	5.00	0	100	0.0060	5.40	3.60	0	100
	6	0.0255	28.40	0	100	0.0137	20.20	70.80	0	100
	7	0.0369	33.60	0	100	0.0220	21.80	78.60	0	100
	8	0.0853	343.80	0	100	0.0285	53.00	252.60	0	100
	9	0.1047	159.40	0	100	0.1021	236.20	1140.60	0	100
	10	0.2518	1612.00	0	100	0.2835	301.00	2065.80	0	100
	11	0.7276	5166.00	0	100	0.9480	965.40	4795.80	0	100
	12	0.9112	4334.20	0	100	1.3085	845.00	7471.80	0	100
	13	2.2704	13849.00	0	100	3.3051	1675.20	10064.80	0	100
	14	2606.0142	155972.80	3.14	80	2255.5929	31126.80	173781.40	1.01	80
	15	1959.7415	160724.00	0.68	80	1950.8311	8133.20	96718.60	0	100

Table A10
Results of MSTN–MTZ and MSTN–SEC for 3D instances with ℓ_1 norm and rectangular neighborhoods.

		MTZ				SEC				
r	n	CPU	#Nodes	GAP	%Solved	CPU	#SECs	#Nodes	GAP	%Solved
1	5	0.0045	0.00	0	100	0.0033	4.20	0.00	0	100
	6	0.0199	6.80	0	100	0.0077	2.60	1.00	0	100
	7	0.0243	4.60	0	100	0.0133	6.40	0.60	0	100
	8	0.0290	5.80	0	100	0.0172	6.40	0.20	0	100
	9	0.0393	68.20	0	100	0.0211	13.20	10.00	0	100
	10	0.0313	0.00	0	100	0.0251	7.20	0.00	0	100
	11	0.0502	9.40	0	100	0.0317	13.00	11.40	0	100
	12	0.0734	0.60	0	100	0.0619	18.20	23.60	0	100
	13	0.1346	56.20	0	100	0.0859	79.40	162.40	0	100
	14	0.1456	75.00	0	100	0.0920	35.60	57.60	0	100
	15	0.2278	74.00	0	100	0.1386	93.40	228.60	0	100
	20	0.5472	303.40	0	100	0.3229	43.80	72.20	0	100
2	5	0.0070	0.00	0	100	0.0063	3.40	0.00	0	100
	6	0.0178	8.40	0	100	0.0114	3.00	0.00	0	100
	7	0.0243	4.80	0	100	0.0138	7.60	5.60	0	100
	8	0.0568	13.80	0	100	0.0275	9.00	7.80	0	100
	9	0.0436	109.00	0	100	0.0265	13.00	20.00	0	100
	10	0.0424	1.00	0	100	0.0355	12.00	1.20	0	100
	11	0.0866	79.40	0	100	0.0418	22.60	55.40	0	100
	12	0.1383	124.80	0	100	0.1129	94.60	444.20	0	100
	13	0.1817	223.40	0	100	0.1414	118.60	393.40	0	100
	14	0.2568	225.00	0	100	0.2107	208.40	573.40	0	100
	15	0.3568	350.60	0	100	0.5568	414.40	1265.40	0	100
3	5	0.0072	0.00	0	100	0.0047	2.40	0.00	0	100
	6	0.0190	11.80	0	100	0.0093	2.80	0.40	0	100
	7	0.0459	18.60	0	100	0.0195	13.20	26.00	0	100
	8	0.0702	59.00	0	100	0.0301	22.80	79.20	0	100
	9	0.0980	205.60	0	100	0.0467	41.60	137.40	0	100
	10	0.0860	28.60	0	100	0.0567	15.60	26.00	0	100
	11	0.1564	227.00	0	100	0.1056	101.40	555.80	0	100
	12	0.3007	663.20	0	100	0.4866	499.80	2588.40	0	100
	13	0.3899	1567.20	0	100	0.4506	520.60	2378.80	0	100
	14	0.7133	3031.40	0	100	1.4542	1269.40	5288.20	0	100
	15	0.4470	1010.00	0	100	1.0931	678.60	2556.40	0	100
4	5	0.0083	0.00	0	100	0.0081	5.20	1.40	0	100
	6	0.0275	15.20	0	100	0.0162	4.60	13.40	0	100
	7	0.0399	18.60	0	100	0.0231	13.60	24.60	0	100
	8	0.0739	32.00	0	100	0.0304	15.40	43.60	0	100
	9	0.0823	230.20	0	100	0.0477	35.60	157.00	0	100
	10	0.1263	91.40	0	100	0.1009	150.80	660.80	0	100
	11	0.2241	485.80	0	100	0.1713	164.00	1134.40	0	100
	12	0.2180	213.00	0	100	0.2612	257.80	1517.20	0	100
	13	0.6875	2034.20	0	100	0.7246	628.80	3289.80	0	100
	14	1.2201	4354.80	0	100	1.9160	1214.80	4778.80	0	100
	15	1.0402	2616.20	0	100	1.6548	910.00	2315.60	0	100

Table A11
Average results for the decomposition approach for planar instances for ℓ_1-norm and rectangular neighborhoods.

r	n	CPU	#SEC	#BendersCuts	#NodesB%B	%GAP$_0$	%GAP	%Solved
1	5	0.0022	1.00	0.60	0.00	1.67	0	100
	6	0.0083	2.60	2.60	2.20	8.64	0	100
	7	0.0141	3.40	5.00	12.40	7.46	0	100
	8	0.0111	1.80	4.40	7.20	7.80	0	100
	9	0.0165	8.00	6.00	25.40	9.33	0	100
	10	0.0334	9.40	19.40	83.20	8.96	0	100
	11	0.0723	32.20	47.60	401.60	19.63	0	100
	12	0.0873	24.80	51.80	416.20	25.25	0	100
	13	0.1843	61.40	104.60	1081.60	36.44	0	100
	14	0.2099	47.00	112.20	1035.80	23.29	0	100
	15	3.0139	325.00	663.00	9696.80	40.27	0	100
	20	14.5891	1295.80	1858.40	44855.00	37.53	0	100
2	5	0.0156	2.20	5.60	12.20	26.63	0	100
	6	0.0228	4.20	13.40	46.00	24.10	0	100
	7	0.0207	5.20	9.00	27.40	22.27	0	100
	8	0.0472	13.40	35.20	198.60	34.17	0	100
	9	0.0514	15.40	30.60	192.20	21.48	0	100
	10	0.0562	18.80	32.00	239.20	20.67	0	100
	11	0.3067	91.00	184.00	1910.40	30.67	0	100
	12	8.6071	413.00	1409.60	20791.80	51.64	0	100
	13	10.4075	526.80	1709.00	27168.40	45.33	0	100
	14	15.8339	1409.60	2572.20	52010.00	38.77	0	100
	15	>7200	4089.20	9677.00	190166.60	62.90	5.86	0
3	5	0.0153	2.40	5.60	13.40	20.63	0	100
	6	0.0126	4.00	5.40	15.40	22.90	0	100
	7	0.0652	17.80	55.80	329.60	44.07	0	100
	8	0.3336	68.80	289.00	2136.20	54.70	0	100
	9	3.4215	272.60	829.80	8834.80	50.00	0	100
	10	639.8128	111.80	400.00	4172.60	45.38	0	100
	11	5631.1871	1566.20	4282.60	63423.80	58.17	16.83	40
	12	>7200	1287.80	4083.00	60204.40	67.52	31.36	0
	13	>7200	1874.20	4292.60	67538.00	67.39	28.09	0
	14	>7200	3006.60	2987.40	77310.60	81.56	53.91	0
	15	>7200	2592.20	3354.20	79375.40	71.18	36.70	0
4	5	0.0109	2.20	4.00	6.40	19.46	0	100
	6	0.0562	15.80	54.80	262.60	34.65	0	100
	7	0.0568	17.80	47.00	292.00	41.84	0	100
	8	0.2792	49.80	248.40	1874.80	43.57	0	100
	9	643.8900	259.80	949.80	10455.60	62.19	0	100
	10	1445.3231	518.80	1791.00	20737.40	65.94	5.47	80
	11	5766.7558	1432.40	4164.40	55404.80	68.46	21.01	20
	12	>7200	1899.60	4578.40	66754.40	74.97	27.17	0
	13	>7200	1963.00	3672.20	62163.60	75.84	39.24	0
	14	>7200	3333.40	3249.80	75414.40	87.12	70.36	0
	15	>7200	3300.60	3328.60	74878.60	88.81	51.00	0

Table A12
Average results for the decomposition approach for 3D instances for ℓ_1-norm and rectangular neighborhoods.

r	n	CPU	#SEC	#BendersCuts	#NodesB%B	%GAP$_0$	%GAP	%Solved
1	5	0.0010	0.60	0.20	0.00	0.13	0	100
	6	0.0054	1.60	2.00	2.20	6.70	0	100
	7	0.0073	1.60	2.60	2.00	5.82	0	100
	8	0.0078	1.60	2.60	3.40	2.47	0	100
	9	0.0101	2.80	4.20	10.00	2.25	0	100
	10	0.0084	4.00	3.00	4.60	2.03	0	100
	11	0.0166	6.80	6.20	18.80	2.95	0	100
	12	0.0323	11.60	19.60	109.60	4.92	0	100
	13	0.0560	15.00	32.00	158.80	8.43	0	100
	14	0.0354	15.00	18.40	72.80	3.69	0	100
	15	0.0962	35.40	39.00	347.40	9.93	0	100
	20	0.2478	52.00	72.80	700.60	7.35	0	100
2	5	0.0048	1.40	1.40	0.80	3.64	0	100
	6	0.0111	2.00	5.00	6.20	5.25	0	100
	7	0.0123	3.00	3.40	6.80	14.15	0	100
	8	0.0227	5.00	10.80	28.60	11.01	0	100
	9	0.0232	6.80	14.20	51.80	7.40	0	100
	10	0.0149	5.40	5.60	17.60	4.77	0	100
	11	0.0434	11.80	27.60	122.60	10.46	0	100
	12	0.2303	24.80	119.80	1071.40	9.04	0	100
	13	0.5307	63.20	232.80	2001.20	14.03	0	100
	14	0.3709	68.40	147.40	1588.80	14.76	0	100
	15	1.8171	586.60	867.80	15792.20	22.30	0	100
3	5	0.0034	1.40	0.80	0.00	6.90	0	100
	6	0.0042	1.00	1.40	0.00	2.05	0	100
	7	0.0275	6.83	18.50	69.50	19.52	0	100
	8	0.0551	15.00	41.60	224.60	18.47	0	100
	9	0.1651	35.60	140.60	811.60	25.35	0	100
	10	0.0668	22.00	44.80	292.20	10.55	0	100
	11	1.3428	123.40	430.20	4317.40	19.60	0	100
	12	12.1502	308.60	1689.80	21923.80	27.30	0	100
	13	76.6432	378.20	1755.20	23285.20	20.49	0	100
	14	5833.2085	1166.60	4502.80	60496.80	30.83	7.53	20
	15	4246.5358	1197.00	3429.20	55744.20	25.23	1.14	60
4	5	0.0083	2.00	2.40	1.60	10.39	0	100
	6	0.0244	4.40	15.00	38.20	16.50	0	100
	7	0.0394	7.00	31.75	111.00	27.56	0	100
	8	0.0491	11.00	35.40	174.80	14.57	0	100
	9	0.1451	36.00	107.80	682.60	20.13	0	100
	10	0.4102	104.00	260.00	2422.60	26.68	0	100
	11	4.1641	241.80	770.20	8311.40	30.48	0	100
	12	8.8121	171.00	1087.60	10670.40	23.66	0	100
	13	4321.3553	715.20	3512.20	44623.80	30.02	3.89	40
	14	>7200	1535.00	4640.60	58166.00	42.04	17.11	0
	15	>7200	2041.60	4399.80	71291.40	35.42	10.54	0

Table A13
Average results for the mathheuristic for ℓ_1-norm and rectangular neighborhoods.

		2-dimensional instances		3-dimensional instances	
r	n	CPU	%Dev	CPU	%Dev
1	5	0.0501	0.0000	0.0656	0.0000
	6	0.0933	0.0000	0.1074	0.0000
	7	0.1351	0.0000	0.1603	0.0000
	8	0.3132	0.0000	0.7855	0.0037
	9	0.2488	0.0000	1.3627	0.0538
	10	0.3393	0.1246	0.5364	0.0000
	11	1.4827	0.0446	0.5580	0.0000
	12	2.4810	0.0268	1.1630	0.0987
	13	1.2527	0.0000	0.8056	0.1088
	14	1.4183	0.1192	0.9533	0.0671
	15	2.8917	0.0423	1.7673	0.1199
	20	2.4849	0.3632	2.7856	0.0488
2	5	0.0556	0.0000	0.0579	0.0000
	6	0.0963	0.0000	0.1093	0.0000
	7	0.1300	0.1610	0.1604	0.0000
	8	0.1994	0.0367	0.2102	0.0000
	9	1.1819	0.0000	0.3124	0.0394
	10	1.6446	0.2628	0.4307	0.0042
	11	1.6624	0.2226	0.5281	0.0809
	12	0.6514	0.2499	0.7326	0.1795
	13	0.7195	0.1416	1.4565	0.2698
	14	0.8685	0.5442	0.9676	0.2398
	15	1.5918	0.5998	1.7437	0.2687
3	5	0.0694	0.0000	0.3834	0.0000
	6	0.0994	0.0000	0.1085	0.0000
	7	0.1389	0.0000	0.4271	0.0000
	8	0.2166	0.0000	0.2414	0.0000
	9	0.3109	0.7205	0.3182	0.0307
	10	0.3593	0.2427	0.4385	0.3942
	11	0.5479	0.5686	0.7874	0.1480
	12	0.7328	0.5994	0.9501	1.0149
	13	0.7832	0.6054	0.9950	0.6702
	14	1.0458	0.7584	1.3970	0.5432
	15	1.4019	0.5245	1.4386	0.2215
4	5	0.0511	0.0000	0.0592	0.0000
	6	0.0999	0.0000	0.1109	0.0000
	7	0.1361	0.0745	0.1561	0.0000
	8	0.2094	0.0000	0.9138	0.0000
	9	0.3650	0.9745	1.1039	0.3631
	10	0.4236	0.8142	0.4621	0.3154
	11	0.6061	0.1490	0.6606	0.2416
	12	0.6831	1.3134	0.8185	0.6113
	13	0.7899	1.1115	1.2024	0.9530
	14	1.0270	1.6971	1.2350	0.9403
	15	1.3304	1.0020	1.4597	0.9309

Table A14
Average results for the mathheuristic for large instances in the 3D case for ℓ_1-norms and rectangular neighborhoods.

r	$\|V\|$	CPU	%Dev LB	%Dev UB	%MST
1	20	3.2718	17.8734	1.7621	0
	25	6.6355	22.6283	1.7051	40
	30	9.3052	22.4836	1.8052	0
	35	15.8989	23.8553	1.6075	20
	40	20.5503	23.2102	1.6839	0
	45	31.5300	32.0624	1.1210	20
	50	44.3083	34.8768	1.6471	0
	60	78.6957	32.8190	1.4473	0
	70	113.8717	35.3224	1.6023	20
	80	74.2517	39.5795	1.6678	40
	90	102.9631	41.3705	0.3986	80
	100	186.1598	43.7887	2.3123	20
2	20	3.7265	53.1616	3.9797	0
	25	5.2124	66.5402	5.0290	0
	30	9.9285	68.6913	5.5394	0
	35	16.9959	79.0108	6.9387	0
	40	25.0273	78.9813	5.8777	0
	45	38.2699	77.7870	5.3047	0
	50	44.5787	83.0475	6.5810	0
	60	92.1195	88.8487	7.2299	0
	70	138.7432	91.0040	5.8910	0
	80	88.7842	93.7916	5.2288	0
	90	151.0290	96.1807	4.7431	20
	100	213.5375	97.4415	8.0734	0
3	20	6.4307	83.4390	10.3132	0
	25	6.9695	89.8750	5.9345	0
	30	9.7627	92.5665	7.8432	0
	35	17.2009	96.1816	9.4858	0
	40	29.1791	96.4066	9.6613	0
	45	34.8684	97.6215	5.8366	0
	50	53.4088	98.8003	10.9474	0
	60	84.6073	99.9880	12.8502	0
	70	138.2981	99.5556	11.6571	0
	80	91.4367	99.5121	15.1611	0
	90	138.3870	99.9182	17.7213	0
	100	195.6471	99.9487	19.1954	0
4	20	4.1700	94.9978	11.8213	0
	25	8.0141	98.3129	12.0262	0
	30	12.6966	99.4911	13.6889	0
	35	19.1026	100	18.0065	0
	40	24.6893	99.6579	14.8827	0
	45	34.4745	100	19.3029	0
	50	43.2740	100	18.7120	0
	60	80.2812	100	29.2938	0
	70	117.1115	100	32.2512	0
	80	87.2616	100	27.3434	0
	90	128.2288	100	34.2494	0
	100	160.3780	100	37.2976	0

Table A15
Average results for the mathheuristic for large instances in the 3D case for ℓ_1-norms and rectangular neighborhoods.

r	$\|V\|$	CPU	%Dev LB	%Dev UB	%MST
1	20	3.9244	4.2397	0.2847	40
	25	7.3535	6.3234	0.2888	40
	30	11.1005	5.7059	0.0560	80
	35	15.1186	5.0536	0.1201	60
	40	23.9661	6.0740	0.2721	40
	45	34.8230	8.0205	0.4048	40
	50	50.1349	9.3624	0.4327	60
	60	84.6652	7.8607	0.3417	60
	70	146.8459	9.7645	0.3315	40
	80	87.6570	9.5274	0.1653	40
	90	119.0164	9.0978	0.0985	80
	100	184.5800	12.0535	16.0131	40
2	20	3.3065	17.0344	1.2530	20
	25	7.7484	25.0740	2.3043	20
	30	12.6865	20.3597	2.1098	20
	35	18.7399	24.3101	1.0825	20
	40	28.0525	26.5215	2.1389	20
	45	41.1231	28.8200	2.9969	0
	50	56.9099	32.7247	0.7609	40
	60	92.0581	35.2961	1.5897	20
	70	151.3542	41.4838	2.4979	20
	80	108.1210	43.5643	2.6506	0
	90	159.6481	45.0713	2.4967	20
	100	209.1270	49.6948	1.9244	0
3	20	4.7462	38.9261	5.1346	0
	25	8.0449	50.8300	5.8753	0
	30	12.8401	45.0230	5.4193	0
	35	19.4514	51.1207	4.2940	0
	40	30.7106	54.7966	3.7982	0
	45	50.0257	59.1045	5.6015	0
	50	70.5336	68.0738	3.7970	20
	60	110.6772	72.5850	4.1774	0
	70	155.4668	79.3740	3.9421	0
	80	113.2362	80.4278	4.2382	0
	90	157.1447	81.1355	3.5754	0
	100	234.6815	85.4162	5.2324	0
4	20	4.6605	62.3675	7.4389	0
	25	7.6516	72.6303	5.2245	0
	30	12.9930	72.3479	4.8505	0
	35	19.5038	78.3774	6.3135	0
	40	28.6141	81.5346	7.7731	0
	45	40.1882	87.0697	6.8866	0
	50	54.2520	90.8204	7.4710	0
	60	113.4294	95.4042	8.5400	0
	70	189.2968	96.9925	7.1464	0
	80	135.5644	97.7376	7.3322	0
	90	204.3473	97.9449	8.9383	0
	100	219.4976	98.2737	7.5345	0

References

Arkin, E. M., & Hassin, R. (1994). Approximation algorithms for the geometric covering salesman problem. *Discrete Applied Mathematics, 55*, 197–218.

Arkin, E. M., & Hassin, R. (2000). Minimum diameter covering problems. *Networks, 36*, 147–155.

Benders, J. F. (1962). Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik, 4*(3), 238–252.

Bertsimas, D., & Howell, L. H. (1993). Further results on the probabilistic traveling salesman problem. *European Journal of Operational Research, 65*(1), 68–95.

Blanco, V., Puerto, J., & El-Haj Ben-Ali, S. (2014). Revisiting several problems and algorithms in continuous location with l_τ norms. *Computational Optimization and Applications, 58*(3), 563–595.

Brimberg, J., & Wesolowsky, G. O. (2002). Locating facilities by minimax relative to closest points of demand areas. *Computers & Operations Research, 29*(6), 625–636.

Cooper, J. (1978). Bounds on the weber problem solution under conditions of uncertainty. *Journal of Regional Science, 18*(1), 87–92.

Disser, Y., Mihalák, M., Montanari, S., & Widmayer, P. (2014). Rectilinear shortest path and rectilinear minimum spanning tree with neighborhoods. In *Proceedings of the third international symposium on combinatorial optimization (ISCO)* (pp. 208–220).

Dorrigiv, R., Fraser, R., He, M., Kamali, S., Kawamura, A., López-Ortiz, A., & Seco, D. (2013). On minimum-and maximum-weight minimum spanning trees with neighborhoods. In *Proceedings of the tenth international workshop on approximation and online algorithms (WAOA)* (pp. 93–106).

Dror, M., Efrat, A., Lubiw, A., & Mitchell, J. S. B. (2003). Touring a sequence of polygons. In *Proceedings of the thirty-fifth symposium on theory of computing* (pp. 473–482). ACM Press.

Dufour, S. W. (1973). *Intersections of random convex regions*. Department of Statistics, Standford University. Ph.D. thesis

Edmonds, J. (1970). Submodular functions, matroids, and certain polyhedra. In *Proceedings of the calgary international conference on combinatorial structures and their applications* (pp. 69–87).

Fernández, E., Pozo, M. A., Puerto, J., & Scozzari, A. (2016). Ordered weighted average optimization in multiobjective spanning tree problems. *European Journal of Operational Research, 17*(1), 886–903.

Fischetti, M., Ljubic, I., & Sinnl, M. (2016a). Benders decomposition without separability: A computational study for capacitated facility location problems. *European Journal of Operational Research, 253*, 557–569.

Fischetti, M., Ljubic, I., & Sinnl, M. (2016b). Redesigning benders decomposition for large scale facility location. *Management Science*. Articles in Advance. http://dx.doi.org/10.1287/mnsc.2016.2461.

Frank, H. (1969). Shortest paths in probabilistic graphs. *Operations Research, 17*(4), 583–599.

Gentilini, I., Margot, F., & Shimada, K. (2013). The travelling salesman problem with neighborhoods: MINLP solution. *Optimization Methods and Software, 28*(2), 364–378.

Geoffrion, A. M. (1972). Generalized benders decomposition. *Journal of Optimization Theory and Applications, 10*(4), 237–260.

Gorski, J., Pfeuffer, F., & Klamroth, K. (2007). Biconvex sets and optimization with biconvex functions: A survey and extensions. *Mathematical Methods of Operations Research, 66*(3), 373–407.

Gurobi Optimization Inc. (2015). Gurobi optimizer reference manual. http://www.gurobi.com.

Juel, H. (1981). Bounds in the generalized weber problem under locational uncertainty. *Operations Research, 29*(6), 1219–1227.

Landete, M., & Marín, A. (2014). Looking for edge-equitable spanning trees. *Computers & Operations Research, 41*, 44–52.

Lobo, M., Vandenberghe, L., Boyd, S., & Lebret, H. (1998). Applications of second-order cone programming. *Linear Algebra and its Applications, 284*, 193–228.

Löffler, M., & van Kreveld, M. (2010). Largest and smallest convex hulls for imprecise points. *Algorithmica, 56*, 235–269.

Martin, R. (1991). Using separation algorithms to generate mixed integer model reformulations. *Operations Research Letters, 10*(3), 119–128.

McCormick, G. P. (1976). Computability of global solutions to factorable nonconvex programs: Part I. Convex underestimating problems. *Mathematical Programming, 10*, 147–175.

Miller, C. E., Tucker, A. W., & Zemlin, R. A. (1960). Integer programming formulation of traveling salesman problems. *Journal of the ACM, 7*(4), 326–329.

Nickel, S., Puerto, J., & Rodríguez-Chía, A. M. (2003). An approach to location models involving sets as existing facilities. *Mathematics of Operations Research, 28*(4), 693–715.

Shioura, A., Tamura, A., & Uno, T. (1997). An optimal algorithm for scanning all spanning trees of undirected graphs. *SIAM Journal on Computing, 26*, 678–692.

Slater, M. (1950). Lagrange multipliers revisited: a contribution to nonlinear programming. Cowles Commission Discussion Paper, Math. 403.

Sörensen, K., & Janssens, G. K. (2005). An algorithm to generate all spanning trees of a graph in order of increasing cost. *Pesquisa Operacional, 25*(2), 219–229.

Wendell, R. E., & Hurter, A. P., Jr. (1976). Minimization of a non-separable objective function subject to disjoint constraints. *Operations Research, 24*(4), 643–657.

Yang, Y., Lin, M., Xu, J., & Xie, Y. (2007). Minimum spanning tree with neighborhoods. In *Proceedings of the third international conference on algorithmic aspects in information and management (AAIM 2007)* (pp. 306–316).