An Exact Solution Framework for the Minimum Cost Dominating Tree Problem

Eduardo Álvarez-Miranda [*1], Martin Luipersbeck[†2], and Markus Sinnl[‡2]

[1]Department of Industrial Engineering, Universidad de Talca, Curicó, Chile
[2]Department of Statistics and Operations Research, Faculty of Business, Economics and Statistics, University of Vienna, Vienna, Austria

Abstract

The Minimum Cost Dominating Tree Problem (MCDTP) is a recently introduced NP-hard problem, which consists of finding a tree of minimal cost in a given graph, such that for every node of the graph, the node or one of its neighbours is in the tree. We present an exact solution framework combining a primal-dual heuristic with a branch-and-cut approach based on a transformation of the problem into a Steiner Arborescence Problem with an additional constraint. The effectiveness of our approach is evaluated on testbeds proposed in literature containing instances with up to 500 nodes. Our framework manages to solve all but four instances from literature to proven optimality within three hours (most of them in a few seconds). We provide optimal solution values for 69 instances from literature for which the optimal solution was previously unknown.

1. Introduction and Motivation

Many problems concerning the design of sensor networks [9, 14], wireless networks [12], or energy efficient wireless networks [11], can be reduced to the *Minimum Cost Dominating Tree Problem* (MCDT).

Definition 1 *(Minimum Cost Dominating Tree Problem) Given an undirected graph $G = (V, E)$ and a cost function $\mathbf{c} : E \to \mathbb{R}_{\geq 0}$, the MCDT is the problem of finding a tree $T = (V_T, E_T) \subset G$ so that every node in $V \setminus V_T$ is adjacent to at least one node in V_T and $\sum_{e \in E_T} c_e$ is minimized.*

The MCDT belongs to the class of (connected) dominating set problems in graphs, which were originally proposed back in the 60s [7]. The MCDT has received considerable attention by researchers over the last years. To the best of our knowledge, the problem was first proposed in [14], in the context of routing backbone optimization in wireless networks. The authors proposed several approximation algorithms and heuristics for the case where the input graph is a disk. In [9], the authors first showed that the MCDT is NP-complete in general graphs and provided an approximation algorithm based on the connection of the problem with the Directed Steiner Tree Problem (also known as Steiner Arborescence Problem). They also provided a heuristic algorithm and a Mixed Integer Programming (MIP) formulation; the latter based on the spanning tree polytope representation based on generalized subtour elimination constraints. Numerical results showed the effectiveness of the proposed heuristic when solving instances with up to 17 nodes. Later, different bio-inspired metaheuristics were designed and implemented in [10]; using randomly generated instances with up to 500 nodes, the authors showed that their approaches were more effective than those proposed

[*]ealvarez@utalca.cl
[†]martin.luipersbeck@univie.ac.at
[‡]markus.sinnl@univie.ac.at

before in [9] and [14]. Two MIP formulations were proposed in [1]; one of the models is based on an arborescence formulation in the bi-directed counterpart of the input graph, while the second one is obtained from the generalized spanning tree polytope. Numerical results show that proposed models could be used to solve instances with up to 100 nodes by off-the-shelf solvers within hundreds of seconds. Very recently, an evolutionary algorithm with guided mutation was designed in [2]; using the instances proposed in [10], the authors showed that their approach outperformed previously proposed algorithms. Finally, in [3] a variable neighborhood search (VNS) algorithm was devised for the MCDT; considering the testbed provided in [10] and additional randomly generated instances with up to 300 nodes, the authors showed that their method was capable of computing optimal solutions for instances with up to 20 nodes (using the MIP model proposed in [9] to solve these instances to optimality), and better solutions than those found in [10] in the case of large-size instances.

Our Contribution and Paper Outline We present a solution framework combining a primal-dual heuristic with an exact branch-and-cut (B&C) approach for the MCDT based on the transformation of the problem into a Steiner Arborescence Problem. The effectiveness of our approach is evaluated on the testbeds proposed by [3] and [10]. We provide optimal solution values for 69 instances from literature, for which the optimal solution was previously unknown. Most of the instances are solved to optimality within a few seconds. The paper is organized as follows; Section 2 details our solution framework, Section 3 contains the computational results, and Section 4 gives a short conclusion.

2. The Algorithmic Framework

The presented framework is based on a B&C algorithm for solving an exponential size formulation of a variant of the *Steiner Arborescence Problem* (SAP) [see, e.g., 13]. The B&C is initialized using a primal-dual heuristic, which constructs a starting solution and also provides an initial set of cuts (in order to avoid their potentially time-consuming separation later on).

Definition 2 *(Steiner Arborescence Problem) Given a directed graph $G = (V, A)$ with dedicated root node $r \in V$, a cost function $\mathbf{c} : A \to \mathbb{R}^{|A|}_{\geq 0}$ and a set of terminals $T \subset V$, the SAP consists of finding a subgraph $S = (V_S, A_S) \subset G$ so that every node $i \in S \setminus \{r\}$ can be reached by a directed path from $r \in S$ and $\sum_{a \in A_S} c_a$ is minimized.*

For the purpose of modeling the MCDT as SAP, a modified version of the SAP is required where the outdegree of the root node r is constrained to be one in a feasible solution. We refer to this version as *root-degree-one SAP* (RSAP).

Let $(G = (V, E), \mathbf{c})$ be an instance of the MCDT. Any instance of the MCDT can be transformed into an instance $(G' = (V', A), r, \mathbf{c}', T)$ of the RSAP as follows [see also 9, for an equivalent transformation]. The node set V' consists of V plus an additional node i' for each $i \in V$. The terminal set T consists of these nodes i', except one, which is (arbitrarily) chosen as root node r. For each $e = \{i, j\} \in E$ introduce two arcs $(i, j), (j, i)$ with $c'_{ij} = c'_{ji} = c_e$. Moreover, for each $i' \neq r$ introduce arcs $a = (k, i')$ for all $\{i, k\} \in E$ and for $k = i$, with $c_a = 0$. For $i' = r$ introduce arcs $a = (i', k)$ for all $\{i, k\} \in E$ and for $k = i$, with $c_a = 0$. In other words, the terminal set consists of a copy i' of each original node i (except the node chosen as root node), and i' is only reachable by going over i or one of its neighbors in G. Moreover, for the $i' \in V'$ chosen as root node r the arcs leaving i' go to i and all nodes which are neighbors of i in G. It is easy to see that in a feasible RSAP solution, for each $i \in V$, either i or one of its neighbors from G is contained. Furthermore, after removing the arcs adjacent to nodes i' and mapping back the remaining arcs to the underlying edges of G, the solution is a tree, and hence a feasible solution to the MCDT. (Note that without the outdegree-constraint on the root node, the backmapped solution may not be a tree.) Finally, since all the arcs to nodes in i' have cost zero, this transformation preserves the cost of feasible solutions. Hence, the MCDT can be solved as a RSAP. For ease of readability, we refer to $\mathbf{c}'$ as $\mathbf{c}$ in the remainder of the paper. As root node r, in our implementation we chose the node with the largest number of adjacent nodes in G. Moreover, preprocessing is done to reduce the size of the obtained graph G': Observe that if there are two adjacent nodes i and j, and the adjacency list of i is a subset of the adjacency list of j, only i needs to be copied to a terminal i', as any solution containing a node adjacent to i automatically contains a node adjacent to j.

2.1 Primal-Dual Heuristic

Our primal-dual heuristic is based on the dual ascent algorithm (DA) for the SAP proposed by Wong [13] [see also 8]. DA performs a greedy construction of a feasible solution to the dual of the Linear Programming (LP)-relaxation of an exponential size MIP formulation for the SAP. By building this dual solution, a subgraph G^H (called *support graph*) is obtained, which by construction contains at least one feasible solution. For the SAP and related problems [see, e.g., 8, 13], construction heuristics usually yield much better solutions when applied to G^H rather than to the original graph G. Moreover, DA also provides a valid lower bound to the optimal solution and a subset of violated inequalities from the exponential size MIP formulation. We first give a description of DA and show how to adapt it to our RSAP setting. This is followed by a description of our construction heuristic.

Dual Ascent Algorithm For solving the RSAP with the proposed DA, we use a directed cut-set MIP formulation to model it [see, e.g., 5, 6, 8, for similar models used in DAs applied to related problems]. Let $\mathbf{x} \in \{0,1\}^{|A|}$ be a vector of binary variables so that $x_a = 1$ iff arc $a \in A$ is in the solution. Additionally, consider the following notation. For a given set $H \subset A$, let $\mathbf{x}(H) = \sum_{a \in H} x_a$. Complementary, for a given set $S \subset V'$, let $\delta^-(S) = \{a : (i,j) \in A \mid j \in S,\ i \in V' \setminus S\}$ and $\delta^+(S) = \{a : (i,j) \in A \mid i \in S,\ j \in V' \setminus S\}$. For a terminal $i \in T$, let $\mathcal{W}_i = \{W \subset V : W \cap T = \{i\}, r \notin W\}$ be the set of nodes inducing the so-called Steiner connectivity cuts [see, e.g., 8], and let $\mathcal{W} = \bigcup_{i \in T} \mathcal{W}_i$. The RSAP is formulated as

$$\min \sum_{a \in A} c_a x_a \tag{1}$$

$$\mathbf{x}(\delta^-(W)) \geq 1 \qquad (\beta_W) \qquad \forall W \in \mathcal{W} \tag{2}$$

$$-\mathbf{x}(\delta^+(r)) \geq -1 \qquad (\lambda) \tag{3}$$

$$\mathbf{x} \in \{0,1\}^{|A|} \tag{4}$$

where β_W and λ are dual variables associated with the corresponding constraints. Connectivity cuts (2) ensure that in a feasible solution there exists a directed path from the root node r to each terminal $i \in T$. Note that due to the fact that (2) are exponential in number, non-trivial instances require on-the-fly separation, as in a B&C scheme. Constraint (3) ensures that there is only one outgoing arc from the root node. Observe that (1), (2) and (4) are a formulation for the SAP. The dual of the LP-relaxation of (1)-(4) is given as

$$\max \sum_{W \in \mathcal{W}} \beta_W - \lambda \tag{5}$$

$$\sum_{W \in \mathcal{W}:(i,j) \in W} \beta_W \leq c_{ij} \qquad \forall (i,j) \in A, i \neq r \tag{6}$$

$$\sum_{W \in \mathcal{W}:(r,j) \in W} \beta_W - \lambda \leq 0 \qquad \forall (r,i) \in A \tag{7}$$

$$\lambda \geq 0 \text{ and } \beta_W \geq 0,\ \forall W \in \mathcal{W}. \tag{8}$$

Due to the presence of the root-outgoing constraint (3), in contrast to the dual for the SAP, the dual for the RSAP contains the dual variable λ, which occurs in dual constraints (7) associated with arcs leaving the root node. We deal with this by fixing the dual variable λ to a large value M before starting the DA. It is easy to see that this is equivalent to deal with constraints (3) in a Lagrangian fashion by imposing a penalty M for violation. For λ fixed to M, constraints (7) become similar to (6) (with $c_{0j} = M$) and the dual problem

```
Data: SA instance $(G = (V, A), c, T)$
Result: Lower bound $LB$, support graph $G^H$, subset of cuts (2)
1 $LB \leftarrow 0$; $\tilde{c}_{ij} \leftarrow c_{ij} \forall (i, j) \in A$, $T_a \leftarrow T$, $G^H \leftarrow \emptyset$
2 while $T_a \neq \emptyset$ do
3     $k \leftarrow$ getActiveTerminal($T_a$)
      /* Let $W_k$ be the set of nodes reachable by a reverse breadth-first-search in $G^H$
         */
4     $\Delta \leftarrow \min_{(i,j)\in\delta^-(W_k)} \tilde{c}_{ij}$
      /* Store cut $\delta^-(W_k)$ for use in the branch-and-cut                        */
5     $\tilde{c}_{ij} \leftarrow \tilde{c}_{ij} - \Delta \qquad \forall (i, j) \in \delta^-(W_k)$
6     $G^H \leftarrow \{(i, j) : \tilde{c}_{ij} = 0\}$
7     $LB \leftarrow LB + \Delta$
8     $T_a \leftarrow$ updateActiveTerminals($G^H$)
```

Algorithm 1: Dual Ascent Algorithm

is similar to the dual for the SAP. In the following, we assume λ fixed to M, i.e., that all constraints in the dual are of type (6). In our implementation, we set M to the sum of the two largest costs of edges adjacent to the node chosen as root.

Note that although there exists an exponential number of variables β_W, this is no concern for the DA procedure, as their values are not tracked explicitly. Rather, they are tracked implicitly based on constraints (6) and their associated *reduced costs* $\tilde{c}_{ij} = c_{ij} - \sum_{W\in\mathcal{W}:(i,j)\in W} \beta_W$, i.e., $\tilde{c}_{ij}$ is the slack of constraint (6) for arc $(i, j) \in A$. The support graph G^H is defined as the graph induced by arcs with $\tilde{c}_{ij} = 0$ at termination of DA.

Algorithm 1 shows a pseudocode of the DA procedure. Note that since a feasible solution to (5)-(8) is computed, also a lower bound LB to the optimal solution is obtained. The algorithm starts with the dual feasible solution where all β_W are set to zero, hence initially $\tilde{c}_{ij} = c_{ij}$. Subsequently, a variable β_W is increased in each iteration, thus decreasing $\tilde{\mathbf{c}}$ in such a way that $\tilde{c}_{ij} \geq 0, \forall (i, j) \in A$; hence the solution stays dual feasible. This process is performed with the help of a list of *active terminals* T_a, which at a given iteration consists of all terminals $t \in T$ with no path from r to t in G^H. The method `updateActiveTerminals`(G^H) removes from T_a terminals that have become inactive during the current iteration, while method `getActiveTerminal`(T_a) returns a terminal from T_a. Note that the returned terminal influences the performance of the DA, as the cuts discovered by it are driven by the returned terminals. In our implementation the method uses a score s_t, $t \in T_a$ and returns the terminal with the smallest score. The score is calculated in two steps: First, for each node $i \in V'$, a score s'_i is calculated as the number of terminals from which i is reachable by a reverse breadth-first-search (rbfs) in G^H. For $t \in T_a$, let $C(t)$ be the set of nodes reachable by a rbfs in G^H. We calculate $s_t = \sum_{i\in C(t)} s'_i$. Using this score turned out to be computationally advantageous compared to other schemes used in the DA for SAP, e.g., using $s_t = |C(t)|$. This could be explained by the fact that the transformation of an MCDT instance gives a quite specifically structured SAP instance. For more details about DA for SAP and related problems, we refer to [5, 6, 8].

Construction Heuristic The given procedure is applied to G^H as computed by DA. Let L be a list of terminals ordered by their degree and Sol be the (partial) solution constructed. Sol is initialized with a node i adjacent to r (the heuristic is repeated for all of them) and all terminals adjacent to i are removed from L (as these are covered by Sol). The following steps are executed iteratively until $L = \emptyset$, which implies that all terminals have been covered: Let t be the next entry of L, the shortest path from Sol to t in G^H is calculated and its nodes and arcs are added to Sol. L is updated w.r.t. Sol (i.e., all terminals adjacent to some node in Sol are removed). As soon as $L = \emptyset$, a *pruning step* is performed which iteratively removes unnecessary leaf-nodes from Sol (as terminals may be covered by more than one node in Sol) [see also 2]. Finally, a simple local search phase using *node-deletion*, *node-insertion* and *2-node-swap* operators is performed [see, e.g., 3].

2.2 Branch-and-Cut Approach

The exact B&C scheme is based on the MIP formulation (1)-(4). Its main components are described below.

Connectivity Cut Separation Let $\tilde{\mathbf{x}}$ be the LP solution at a given node of the B&C tree, and $\tilde{G} = (V, A, \tilde{\mathbf{x}})$ be a capacitated graph with $\tilde{\mathbf{x}}$ as capacities. Violated connectivity cuts of type (2) are identified by solving maximum flow problems on $\tilde{G}$ from r to the nodes in T [see, e.g., 4, for further details].

Branching Priorities We introduce binary node variables $\mathbf{y} \in \{0,1\}^{|V'|}$ with $y_i = 1$ iff node $i \in V'$ is in the solution (y_t for $t \in T$ are fixed to one at initialization). Node variables are linked to arc variables by $\sum_{(ji)\in\delta^-(i)} x_{ji} = y_i,\ \forall i \in V' \setminus \{r\},$. During the B&C, variables $\mathbf{y}$ are given higher branching priority, i.e., branching on nodes is preferred, as this strategy results in a more balanced search-tree.

MIP Initialization Along with constraint (3), the MIP model is initialized using the cuts selected by the DA procedure (Algorithm 1, Step 4). Moreover, the constraints $x_{ij} + x_{ji} \leq y_i,\ \forall (i,j) \in G^H$, a special case of (2), are added to the MIP model.

Primal Heuristic Given the LP solution $\tilde{\mathbf{x}}$ at a given node of the B&C tree, a feasible solution is constructed by using the procedure outlined in §2.1 on G' using modified arc weights $c_a(1 - \tilde{x}_a), a \in A'$.

3. Computational Results

Experiments have been performed on a single core of an Intel E5-2670v2 with 2.5GHz and 3GB RAM. All algorithms have been implemented in C++, with CPLEX 12.6 employed as MIP solver. The CPLEX parameters (except branching priorities, as detailed in the previous Section) have been set to their default values. Each test run is given a timelimit of three hours.

Benchmark Instances Two sets of instances from the MCDT literature are considered: `dtp` (proposed in [3][1]) and `range` (proposed in [2, 10][2]).

- `dtp`: The set contains instance graphs with $|V| \in \{10, 15, 20, 100, 200, 300\}$ and $|E| \in \{15, 20, 30, 50, 150, 200, 400, 600, 1000\}$.Both G and $\mathbf{c}$ are generated randomly, with $c_e \in [1, 10], e \in E$. Per $|V|$, three seeds are used to obtain three different instances. In total, 33 instance have been generated. Instances are denoted by "$|V|$-$|E|$-*id*", with $id \in \{1, 2, 3\}$.
- `range`: The set's instance graphs correspond to disk graphs, where the disks indicate the transmission range R of a node. Each graph is constructed as follows: For each node, a point is chosen at random in a 500×500 plane. An edge exists between two nodes if they are within transmission range. The edge weight is the squared Euclidean distance (rounded to two decimals). Three graphs have been constructed for each combination of $|V| \in \{50, 100, 200, 300, 400, 500\}$ and $R \in \{100, 125, 150\}$, leading to 54 instances. Instances are denoted by "R-$|V|$-*id*", with $id \in \{1, 2, 3\}$.

Instance set `dtp` has been approached with a VNS in [3] and also with a MIP (using the formulation proposed in [9]) for the instances with $|V| \in \{10, 15, 20\}$. As this MIP approach already struggled for these small instances, leaving `20-50-1` unsolved and taking up to 4700 seconds for the other small instances, the authors did not test the MIP on the larger instances (i.e., for these instances, the optimal solution value is not known).

Instance set `range` has been approached with a VNS in [3] (for $R = 100$), with an Ant Colony Optimization and Bee Colony Optimization in [10] (for $R = 100$), and with an evolutionary algorithm (EA) in [2] (all

[1]available at `http://poincare.matf.bg.ac.rs/~zdrazic/dtp`
[2]available at `http://scis.uohyd.ac.in/~alokcs`

R). Both, [3] and [2] conclude that their approaches outperform the approaches presented in [10]. For all instances from set `range`, the optimal solution value is not known.

Results Table 1 shows results on instance set `dtp`. We also provide the best solution value ($z^{[3]}$) and runtime ($t^{[3]}[s]$) obtained by the VNS of [3] (runtime is averaged over 20 VNS runs for an instance; the experiments in [3] were performed on a Intel Core I7-4702MQ with 2.2 GHz and 4GB RAM.) From the reported results, we can see that for instances 10-15-0 to 100-200-1 our approach computes the same solutions reported by [3], but much more efficiently. For larger instances, 200-400-0 to 300-1000-2, our approach provides considerably better solutions than those reported by [3] and, in almost all cases, more quickly. Note that even the purely primal-dual heuristic solutions, whose primal and dual bounds reported in columns z^H and LB^H, respectively, are typically better than those found in [3]. In bold we highlight those cases in which we outperform the best-known primal bounds (12 cases), and with an asterisk we mark those cases in which optimality proof is, for the first time, provided (16 cases).

Table 1: Performance of our exact and heuristic approach on the instance set `dtp` compared to the VNS presented in [3].

inst	z^*	LB	$t[s]$	z^H	LB^H	$t^H[s]$	$z^{[3]}$	$t^{[3]}[s]$
10-15-0	5.89188	5.89188	1	5.89188	5.89188	0	5.89	0
10-15-1	14.42328	14.42328	1	14.42328	14.42328	0	14.42	0
10-15-2	14.35039	14.35039	1	14.35039	14.35039	0	14.35	0
15-20-0	18.87450	18.87450	0	18.87450	18.87450	0	18.87	0
15-20-1	23.02953	23.02953	1	23.02953	23.02953	0	23.03	0
15-20-2	24.94884	24.94884	1	24.94884	24.94884	0	24.95	0
15-30-0	18.19640	18.19640	1	18.19640	18.19640	0	18.20	0
15-30-1	8.31711	8.31711	1	8.31711	8.23169	0	8.32	0
15-30-2	18.06508	18.06508	1	18.27780	16.70500	0	18.07	0
20-30-0	33.80565	33.80565	1	33.80565	33.80565	0	33.81	0
20-30-1	36.03494	36.03494	2	36.03494	36.03494	0	36.03	0
20-30-2	43.49886	43.49886	2	43.49886	43.49886	0	43.50	0
20-50-0	9.81012	9.81012	1	9.81012	8.91638	0	9.81	0
20-50-1	12.18698	12.18698 *	2	12.18698	11.79562	0	12.19	0
20-50-2	17.42320	17.42320	2	17.42320	17.42320	0	17.42	0
100-150-0	152.57259	152.57259 *	2	152.57259	149.94661	1	152.57	295
100-150-1	192.20663	192.20663 *	2	192.20663	188.88742	0	192.21	286
100-150-2	146.34474	146.34474 *	1	146.34474	145.91871	0	146.34	246
100-200-0	135.04003	135.04003 *	3	137.75252	126.95836	1	135.04	334
100-200-1	91.88264	91.88264 *	3	91.97222	91.61818	1	91.88	133
100-200-2	115.93330	115.93330 *	2	116.42103	109.94687	1	115.93	372
200-400-0	**257.09155**	257.09155 *	15	260.41598	238.46089	4	306.06	565
200-400-1	**258.77261**	258.77261 *	5	261.13322	249.52902	2	303.53	559
200-400-2	**238.27034**	238.27034 *	4	238.27034	232.68769	1	274.37	550
200-600-0	**121.62120**	121.62120 *	67	125.60597	107.54256	3	132.49	554
200-600-1	**135.08124**	135.08124 *	6	141.31769	129.82916	3	162.92	557
200-600-2	**123.30669**	123.30669 *	243	132.38667	106.26319	2	139.08	521
300-600-0	**348.02734**	348.02734 *	9	367.50394	333.38925	4	471.69	539
300-600-1	**413.93416**	413.93416 *	9	418.08229	404.31348	2	494.91	544
300-600-2	**352.14821**	352.14821 *	5	355.63689	347.06032	3	500.72	534
300-1000-0	**147.16534**	144.16129	TL	158.87691	125.18579	4	257.72	575
300-1000-1	**165.32331**	155.21621	TL	171.21897	140.05051	5	242.79	531
300-1000-2	**154.58602**	150.58588	TL	166.89772	132.83734	5	223.18	483

Table 2 shows he result on instance set `range`. We also provide the best solution value obtained by the EA of [2] in column $z^{[2]}$ (the authors performed 20 runs of the EA per instance). Note that [2] only report the runtime of the EA in a plot and do not give it explicitly by instance. From their plot, an average runtime of 50 to 100 seconds can be estimated for the largest instances in the set (the experiments in [2] were performed on an Intel Core2Duo with 3.0 GHz and 2GB RAM). In bold we report the primal bounds, computed by our approach, that outperform those computed in [2]; this occurs in 15 cases. From the table, we can see that the proposed algorithmic framework provides optimality proof for 53 out of 54 instances.

Table 2: Performance of our exact and heuristic approach on the instance set `range` compared to the EA presented in [2].

inst	z^*	LB	$t[s]$	z^H	LB^H	$t^H[s]$	$z^{[2]}$
100-050-1	1204.41	1204.41	1	1204.41	1195.97	0	1204.41
100-050-2	1340.44	1340.44	1	1353.25	1318.48	0	1340.44
100-050-3	1316.39	1316.39	1	1316.39	1296.74	0	1316.39
100-100-1	1217.47	1217.47	4	1221.11	1130.54	1	1217.47
100-100-2	1128.40	1128.40	4	1128.40	1112.83	2	1128.40
100-100-3	1252.99	1252.99	4	1277.09	1199.37	2	1253.49
100-200-1	1206.79	1206.79	16	1206.79	1129.79	3	1206.79
100-200-2	**1213.24**	1213.24	50	1242.95	1129.17	3	1216.41
100-200-3	**1247.25**	1247.25	17	1307.94	1195.61	4	1247.63
100-300-1	**1215.48**	1215.48	258	1244.39	1126.55	6	1225.22
100-300-2	1170.85	1170.85	88	1199.25	1083.85	7	1170.85
100-300-3	**1247.51**	1247.51	238	1282.85	1167.04	5	1252.14
100-400-1	**1211.33**	1211.33	3768	1262.91	1115.94	8	1211.72
100-400-2	**1197.66**	1197.66	1609	1244.08	1102.94	10	1199.92
100-400-3	**1245.25**	1245.25	3454	1287.08	1130.63	11	1248.29
100-500-1	**1201.31**	1195.82	TL	1229.99	1072.96	14	1206.07
100-500-2	**1220.47**	1220.47	8147	1296.38	1093.37	12	1226.78
100-500-3	**1231.81**	1231.81	5227	1259.07	1129.90	15	1232.15
125-50-1	802.95	802.95	2	825.73	771.92	0	802.95
125-50-2	1055.10	1055.10	2	1055.10	1055.10	0	1055.10
125-50-3	877.77	877.77	1	918.37	877.77	0	877.77
125-100-1	943.01	943.01	2	943.01	905.28	1	943.01
125-100-2	**917.00**	917.00	4	917.95	888.35	2	917.95
125-100-3	998.18	998.18	4	1013.91	964.72	1	998.18
125-200-1	910.17	910.17	13	913.96	889.50	4	910.17
125-200-2	921.76	921.76	32	949.82	871.43	4	921.76
125-200-3	939.58	939.58	19	974.04	911.54	2	939.58
125-300-1	977.65	977.65	74	979.69	958.79	8	977.65
125-300-2	913.01	913.01	99	925.92	881.75	7	913.01
125-300-3	**974.78**	974.78	60	985.18	950.43	9	974.85
125-400-1	965.99	965.99	179	974.99	936.22	12	965.99
125-400-2	**934.17**	934.17	663	951.27	892.61	13	941.02
125-400-3	**1002.61**	1002.61	797	1016.10	974.96	11	1002.97
125-500-1	963.89	963.89	2100	1012.42	923.83	20	963.89
125-500-2	948.57	948.57	1513	969.55	919.16	16	948.57
125-500-3	980.67	980.67	509	1005.34	957.32	24	980.67
150-50-1	647.75	647.75	2	647.75	642.32	0	647.75
150-50-2	863.69	863.69	2	886.40	843.32	0	863.69
150-50-3	743.94	743.94	1	743.94	719.58	0	743.94
150-100-1	876.69	876.69	6	885.97	860.02	3	876.69
150-100-2	657.35	657.35	5	657.35	657.35	3	657.35
150-100-3	722.87	722.87	5	722.87	712.81	2	722.87
150-200-1	809.90	809.90	24	814.30	779.87	6	809.90
150-200-2	736.23	736.23	19	765.99	714.67	3	736.23
150-200-3	792.71	792.71	29	805.71	761.86	4	792.71
150-300-1	796.15	796.15	89	796.70	778.04	7	796.15
150-300-2	**741.02**	741.02	281	759.33	697.70	10	741.92
150-300-3	819.76	819.76	298	825.56	787.41	6	819.76
150-400-1	795.53	795.53	557	799.30	774.18	10	795.53
150-400-2	779.63	779.63	1171	785.15	755.13	12	779.63
150-400-3	814.14	814.14	1591	824.60	780.65	11	814.14
150-500-1	792.21	792.21	1634	796.57	777.82	20	792.21
150-500-2	**779.35**	779.35	2754	794.31	752.28	26	799.35
150-500-3	**808.37**	808.37	4649	814.11	775.08	25	810.27

4. Conclusions

In this paper we address the *Minimum Cost Dominating Tree Problem* (MCDT), a recently introduced network design problem. We design an exact algorithmic framework combining a primal-dual heuristic and an exact branch-and-cut algorithm. The framework is based on a MIP formulation which results from a suitable transformation of the problem into a related Steiner Arborescence Problem. A computational study on benchmark instances from literature shows that our framework outperforms the previously proposed algorithms for the MCDT. In particular, our framework manages to solve all but four instances from literature to proven optimality within three hours, most of them in a few seconds. In total, we provide optimal solution values for 69 instances from literature, for which the optimal solution was previously unknown.

Acknowledgements

E.A.-M. acknowledges the support of the Chilean Council of Scientic and Technological Research, CONICYT, through the grant FONDECYT N.11140060 and through the Complex Engineering Systems Institute (ICM-FIC:P-05-004-F, CONICYT:FB0816). The research of M.S. was supported by the Austrian Research Fund (FWF, Project P 26755-N19). M.L. acknowledges the support of the University of Vienna through the uni:docs fellowship programme.

References

[1] P. Adasme, R. Andrade, J. Leung, and A. Lisser. Models for minimum cost dominating trees. *Electronic Notes in Discrete Mathematics*, 52:101–107, 2016.

[2] S. Chaurasia and A. Singh. A hybrid heuristic for dominating tree problem. *Soft Computing*, 20(1): 377–397, 2016.

[3] Z. Dražić, M. Čangalović, and V. Kovačević-Vujčić. A metaheuristic approach to the dominating tree problem. *Optimization Letters*, pages 1–13, 2016.

[4] M. Fischetti, M. Leitner, I. Ljubić, M. Luipersbeck, M. Monaci, M. Resch, D. Salvagnin, and M. Sinnl. Thinning out Steiner trees: a node-based model for uniform edge costs. *Mathematical Programming Computation*, accepted, 2016.

[5] M. Leitner, I. Ljubić, M. Luipersbeck, and M. Sinnl. A dual-ascent-based branch-and-bound framework for the prize-collecting Steiner tree and related problems. *Technical Report*, 2016.

[6] M. Leitner, I. Ljubić, J.-J. Salazar-González, and M. Sinnl. An algorithmic framework for the exact solution of tree-star problems. *Technical Report*, 2016.

[7] O. Ore. *Theory of Graphs*, volume 38 of *Colloquium Publications*. American Mathematical Society, 1st edition, 1962.

[8] T. Polzin and S. Daneshmand. Improved algorithms for the Steiner problem in networks. *Discrete Applied Mathematics*, 112(1):263–300, 2001.

[9] I. Shin, Y. Shen, and M. Thai. On approximation of dominating tree in wireless sensor networks. *Optimization Letters*, 4(3):393–403, 2010.

[10] S. Sundar and A. Singh. New heuristic approaches for the dominating tree problem. *Applied Soft Computing*, 13(12):4695–4703, 2013.

[11] M. Thai, F. Wang, D. Liu, S. Zhu, and D. Du. Connected dominating sets in wireless networks with different transmission ranges. *IEEE Transactions on Mobile Computing*, 6(7):721–730, 2007.

[12] M. Thai, R. Tiwari, and D. Du. On construction of virtual backbone in wireless ad hoc networks with unidirectional links. *IEEE Transactions on Mobile Computing*, 7(9):1098–1109, 2008.

[13] R Wong. A dual ascent approach for Steiner tree problems on a directed graph. *Mathematical Programming*, 28(3):271–287, 1984.

[14] N. Zhang, I. Shin, B. Li, C. Boyaci, R. Tiwari, and M. Thai. New approximation for minimum-weight routing backbone in wireless sensor network. In Y. Li, D. Huynh, S. Das, and D. Du, editors, *Proceedings of WASA 2008*, volume 5258 of *LNCS*, pages 96–108. Springer, 2008.