Yokohama Publishers
ISSN 1349-8169 ONLINE JOURNAL

Pacific Journal of Optimization
Vol. 1, No. 3, pp. 491–508, September 2005

IMPROVING ON BRANCH-AND-CUT ALGORITHMS FOR GENERALIZED MINIMUM SPANNING TREES

Corinne Feremans, Andrea Lodi, Paolo Toth and Andrea Tramontani

Dedicato a Toshihide Ibaraki con ammirazione
per l'uomo e il suo eccezionale contributo scientifico.

Abstract: Several variants of Generalized Minimum Spanning Tree Problems (GMSTPs) have been introduced in the literature in different papers by a number of authors. Roughly speaking, all these variants are generalizations of the classical Minimum Spanning Tree on an undirected graph $G = (V, E)$ in which the node set V is partitioned into a given set of clusters, and the minimum tree has to "span" those clusters instead of simple nodes.

In particular, in this paper we are concerned with two specific variants, the most classical one in which *Exactly* one node in each cluster has to be visited (E-GMSTP), and the less studied problem in which *at Least* one node in each cluster has to be reached (L-GMSTP).

This paper presents several effective techniques to improve on the branch-and-cut approaches for E-GMSTP and L-GMSTP proposed by Feremans, Labbé and Laporte [8] and by Feremans [6], respectively. In particular, we improved on the performances through: *i*) new effective heuristic algorithms, *ii*) updated branching strategies, and *iii*) the use of general-purpose Chvátal-Gomory cuts.

Finally, a generalization of both problems requiring some clusters to be visited exactly once and the remaining clusters at least once is presented.

Key words: *branch-and-cut, linear programming, primal heuristics, cutting planes, separation*

Mathematics Subject Classification: *90C35, 90C27, 90C57, 90C10*

1 Introduction

Several variants of Generalized Minimum Spanning Tree Problems (GMSTPs) have been introduced in the literature in different papers by a number of authors. Roughly speaking, all these variants are generalizations of the classical *Minimum Spanning Tree Problem* (MSTP, see, e.g., [12]) on an undirected graph $G = (V, E)$ in which the node set V is partitioned into a given set of clusters*, and the minimum tree has to "span" those clusters instead of simple nodes.

In particular, in this paper we are concerned with two specific variants, the most classical one in which *Exactly* one node in each cluster has to be visited (E-GMSTP), and the less studied problem in which *at Least* one node in each cluster has to be reached (L-GMSTP).

More precisely, V is partitioned into $|K|$ clusters V_k, $k \in K$. Each edge $e = \{i, j\} \in E$ has a cost $c_e \in \mathbb{R}^+$. The E-GMSTP is the problem of finding a minimum cost tree including exactly one node from each node set of the partition (see Figure 1 for a feasible solution of

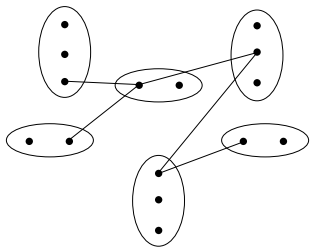

Figure 1: Feasible solutions for E-GMSTP.

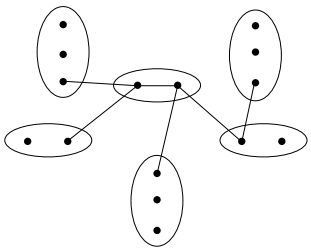

Figure 2: Feasible solutions for L-GMSTP.

E-GMSTP). This problem was introduced by Myung, Lee and Tcha [13] who have shown it is strongly $\mathcal{NP}$-hard by a reduction from the *node-cover* problem. Mathematical formulations and exact methods have been discussed by Myung, Lee and Tcha [13], Faigle, Kern, Pop and Still [5], Pop [14] and Feremans, Labbé and Laporte [7, 8]. The algorithm proposed in [8] is considered as the most effective approach for the optimal solution of E-GMSTP. A polynomial approximation algorithm has been proposed by Pop, Kern and Still [15].

In the L-GMSTP, instead, at least one node from each cluster of the partition must be included in the minimum cost tree (see Figure 2 for a feasible solution of L-GMSTP). This problem was introduced by Ihler, Reich and Widmayer [10] as a particular case of the *Generalized Steiner Tree Problem* under the name *"Class Tree Problem"*. Ihler, Reich, Widmayer [10] have shown that the decision version of the L-GMSTP is $\mathcal{NP}$-complete even if G is a tree, and that there is no constant worst-case ratio polynomial-time algorithm unless $\mathcal{P} = \mathcal{NP}$, even if G is a tree on V with edge lengths 1 and 0. Heuristic algorithms have been proposed by Ihler, Reich, Widmayer [10] and by Dror, Haouari and Chaouachi [3]. The only exact algorithm for this problem has been proposed by Feremans [6].

The L-GMSTP reduces at a first glance to the E-GMSTP when the triangle inequalities hold, but this is not true as shown by the example in Figure 3. Indeed, if the graph depicted

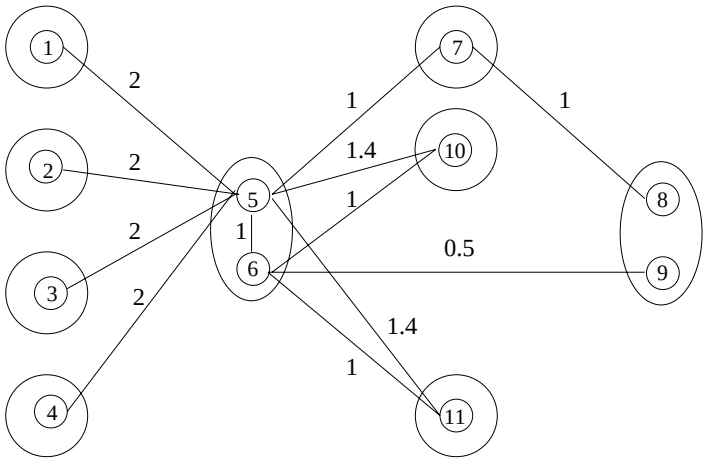

Figure 3: A graph for which E-GMSTP and L-GMSTP differ.

*A variant in which the clusters may overlap is considered in [4].

in Figure 3 is completed through shortest paths, it satisfies the triangle inequalities and the value of the optimal solution of L-GMSTP is 12.5 (using both nodes 5 and 6), while the optimal value of the E-GMSTP solution is 12.8 (only using node 5).

Applications modeled by E-GMSTP are encountered in telecommunications, where metropolitan and regional networks must be interconnected by a tree containing a gateway from each network. For this internetworking, a node has to be chosen in each local network as a hub and the hub nodes must be connected via transmission links such as optical fiber (see Myung, Lee and Tcha [13] for details).

The L-GMSTP has been used to solve an important real life network design problem arising in desert environments and consisting in designing a minimal length irrigation network which connects at least one node from each parcel to a water source (see Dror, Haouari and Chaouachi [3] for details).

This paper presents several effective techniques to improve on the branch-and-cut approaches for E-GMSTP and L-GMSTP proposed by Feremans, Labbé and Laporte [8] and by Feremans [6] respectively. In particular, we improved on the performances through: *i*) new effective heuristic algorithms, *ii*) updated branching strategies, and *iii*) the use of general-purpose Chvátal-Gomory cuts (with and without the strengthening procedures proposed by Letchford and Lodi [11]).

Finally, a generalization of both problems requiring some clusters to be visited exactly once and the remaining clusters at least once is presented. Such a generalization is denoted as E/L-GMSTP and naturally appears when the considered network is somehow "mixed", i.e., involving clusters which may require a different behavior (fixed-charge costs).

The paper is organized as follows. In Section 2 two basic Integer Linear Programming (ILP) formulation for E-GMSTP proposed in [13] and the one discussed in [7] and tested in [8] are recalled. In Section 3 an ILP formulation for L-GMSTP is proposed and its relationship with the one for E-GMSTP is discussed, while Section 4 discusses the proposed generalized problem. Section 5 recalls the branch-and-cut method proposed in [8, 6] and presents several techniques to improve on this method. Computational experiments are reported in Section 6 showing the effectiveness of the presented techniques and preliminary results for the proposed generalized problem. Some conclusions are drawn in Section 7.

2 ILP Formulations for E-GMSTP

Myung, Lee and Tcha [13] have provided two basic formulations for the E-GMSTP using two sets of binary variables, namely $x_e, \forall e \in E$ and $y_i, \forall i \in V$. In the first formulation, called *ucut*, connectivity is ensured by *cutset constraints* of the form $x(\delta(S)) \geq y_i + y_j - 1$ ($i \in S \subset V, j \notin S$), whereas in the second, called *usub*, cycles are prevented through *subpacking constraints* of the form $x(E(S)) \leq y(S \backslash \{i\})$ ($i \in S \subset V$, $2 \leq |S| \leq |V| - 1$). As it is customary, for any $S \subseteq V$, $\delta(S)$ (resp. $E(S)$) denotes the set of edges having exactly one endpoint (resp. both endpoints) in S, and $x(\delta(S))$ (resp. $x(E(S))$) denotes the sum of the x-values on the subset $\delta(S)$ (resp. $E(S)$).

The undirected cutset formulation uses the fact that a feasible Generalized Spanning Tree (E-GST) is a connected subgraph of G containing one node per cluster and $|K| - 1$ edges:

Undirected cutset formulation (ucut)

$$\min \sum_{e \in E} c_e x_e \tag{1}$$

subject to

$$y(V_k) = 1 \qquad k \in K, \tag{2}$$

$$x(E) = |K| - 1, \tag{3}$$

$$x(\delta(S)) \geq y_i + y_j - 1 \qquad i \in S \subset V, j \notin S, \tag{4}$$

$$x_e \in \{0,1\} \qquad e \in E, \tag{5}$$

$$y_i \in \{0,1\} \qquad i \in V. \tag{6}$$

Constraints (2) guarantee that each cluster is visited exactly once, while constraint (3) forces the tree structure. As already mentioned, constraints (4) assure connectivity. Finally, constraints (5) and (6) are the integrality requirements.

An E-GST can also be defined as an acyclic subgraph of G containing one node per cluster and $|K| - 1$ edges:

Undirected subpacking formulation (usub)

$$\min \sum_{e \in E} c_e x_e$$

subject to

$$y(V_k) = 1 \qquad k \in K,$$

$$x(E) = |K| - 1,$$

$$x(E(S)) \leq y(S \backslash \{i\}) \qquad i \in S \subset V, 2 \leq |S| \leq |V| - 1, \tag{7}$$

$$x_e \in \{0,1\} \qquad e \in E,$$

$$y_i \in \{0,1\} \qquad i \in V.$$

The model is equivalent to the previous one with the only difference of the connectivity constraints (4) replaced by constraints (7).

Myung, Lee and Tcha [13] have also proved that $\mathcal{P}_{usub} \subseteq \mathcal{P}_{ucut}$, i.e., that the subpacking formulation dominates the cutset one in terms of continuous relaxation. The example depicted in Figure 4, provided by Magnanti and Wolsey [12] in the context of the Minimum Spanning Tree Problem, can also be used to show (with $|K| = 5, V_k = \{k\}\ \forall k \in K$) that this inclusion is strict. Indeed (2), (3) and (4) are satisfied while (7) is violated for $S = \{3,4,5\}$.

Several other formulations for E-GMSTP were proposed and discussed by Feremans, Labbé and Laporte [7]. Among these formulations the tightest and most compact one (in terms of linear programming relaxation and number of variables, respectively) is the undirected cluster subpacking formulation:

Undirected cluster subpacking formulation (ucsub)

$$\min \sum_{e \in E} c_e x_e$$

subject to

$$y(V_k) = 1 \qquad k \in K,$$

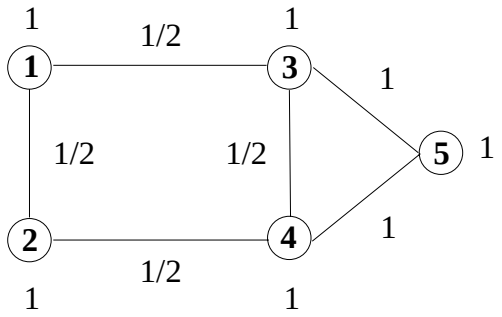

Figure 4: Example showing that $\mathcal{P}_{usub} \subset \mathcal{P}_{ucut}$.

$$\begin{aligned} x(E) &= |K| - 1, \\ x(E(S)) &\leq y(S) - 1 \qquad S \subset V, 2 \leq |S| \leq |V| - 1, \mu(S) \neq 0, \\ x_e &\in \{0,1\} \qquad e \in E, \\ y_i &\in \{0,1\} \qquad i \in V. \end{aligned} \tag{8}$$

This formulation is a strengthening of *usub* in which constraints (7) are replaced by the *cluster subpacking constraints* (8), where for any $S \subseteq V$ we define $\mu(S) = |\{k : V_k \subseteq S\}|$, i.e., the number of clusters included in S (see [7] for details and the proof that $\mathcal{P}_{ucsub} \subset \mathcal{P}_{usub}$). The undirected cluster subpacking formulation is the one used by Feremans, Labbé and Laporte [8] as a base for a branch-and-cut approach which is elaborated in Section 5.1 and tested in Section 6.1.

3 ILP Formulation for L-GMSTP

The L-GMSTP can be formulated as an integer linear program as follows.

$$\min \sum_{e \in E} c_e x_e$$

s.t.

$$y(V_k) \geq 1 \qquad k \in K, \tag{9}$$

$$x(E) = y(V) - 1, \tag{10}$$

$$\begin{aligned} x(\delta(S)) &\geq y_i + y_j - 1 \qquad i \in S \subset V, j \notin S, \\ x_e &\in \{0,1\} \qquad e \in E, \\ y_i &\in \{0,1\} \qquad i \in V, \end{aligned}$$

where constraints (9) and (10) replace constraints (2) and (3), respectively.

Notice that constraints

$$x(E(S)) \leq y(S) - 1 \qquad S \subset V, 2 \leq |S| \leq |V| - 1,\ \mu(S) \neq 0,$$

i.e., constraints (8), valid for the E-GMSTP remain valid for the L-GMSTP. However, they do not dominate all the constraints (4) unlike for the E-GMSTP polytope. Indeed, consider the following example (see Figure 5) where all the constraints (8) are satisfied but at least one of the constraints (4) is violated. Let $V = \{1, 2, \ldots, 6\}, V_1 = \{1, 2, 3\}, V_2 = \{4, 5\}, V_3 = \{6\}$ and the graph is complete. If $y_i = 1$, $\forall i \in V, x_{12} = x_{14} = x_{24} = x_{35} = x_{56} = 1, x_e = 0$ otherwise, then the constraint (4) for $S = \{1, 2, 4\}, i = 1, j = 3$ is violated.

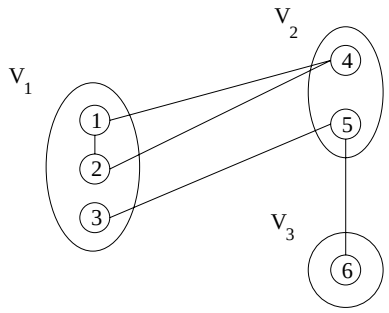

Figure 5: Constraints (4) are not all implied by constraints (8) for L-GMSTP.

The above linear formulation for the L-GMSTP is not strong with respect to its linear relaxation. We can justify this claim in exploiting the same argument as the one used in Magnanti and Wolsey [12] for the Minimum Spanning Tree problem. Indeed, if each cluster is reduced to a single node, the L-GMSTP boils down to the classical MSTP. However, this formulation is particularly effective in a branch-and-cut context since the so-called cut-constraints (4) are easy to separate using max-flow algorithms. Such a formulation is the one used by Feremans [6] as a base for a branch-and-cut approach and such an approach is elaborated in Section 5.2 and tested in Section 6.2.

3.1 From L-GMSTP to E-GMSTP

One way to solve the L-GMSTP consists of seeing it as a variant of the E-GMSTP. For this purpose, the following transformed graph has to be defined.

Let $G = (V, E)$, with V partitioned into clusters $V_1, V_2, \ldots, V_{|K|}$, be the graph of the L-GMSTP instance. The transformed graph $\widetilde{G} = (\widetilde{V}, \widetilde{E})$ is defined as follows:

- $\widetilde{V}$ is equal to V, and the partition into clusters remains the same,
- for each pair of nodes $i, j \in \widetilde{V}$ belonging to different clusters and such that there exists a path between i and j in G, there is an edge $\{i, j\} \in \widetilde{E}$ with cost equal to the value of the shortest path from i to j in the original graph G.

The transformed graph $\widetilde{G}$ is $|K|$-partite complete if G is connected on V.

Proposition 1 *The optimal solution of the E-GMSTP solved on $\widetilde{G}$ can be transformed into a feasible solution of L-GMSTP on G. It gives then an upper bound on the value of the optimal solution of L-GMSTP.*

Proof. An edge in an E-GMSTP solution in $\widetilde{G}$ corresponds to a path in G. Removing the repeated edges and the cycles (in deleting the edge with highest cost in each cycle, one cycle at a time), we obtain a feasible solution to L-GMSTP with value less or equal to the corresponding solution in $\widetilde{G}$. □

It is not difficult to see that repeated edges and cycles can occur from the transformation.

It does not always exist an optimal solution of E-GMSTP on $\widetilde{G}$ such that in removing repeated edges and cycles, we get an optimal solution for L-GMSTP on G. It means that solving the E-GMSTP on $\widetilde{G}$ can only provide an upper bound to the L-GMSTP on G. To see this, it suffices to consider again the graph in Figure 3. The optimal solution of L-GMSTP

on G is 12.5, while the optimal value of the E-GMSTP solution on $\widetilde{G}$ is 12.8 and corresponds to a feasible solution of L-GMSTP in G of value 12.8.

In [6], such a transformation of G into $\widetilde{G}$ with edges in $\widetilde{G}$ corresponding to the shortest paths with the maximum number of edges has been tested as a heuristic. This is done, in order to get a solution in $\widetilde{G}$ that is transformed into a solution in G with as many as possible repeated edges and cycles. To obtain shortest paths with maximum number of edges, the following scaling is performed on G. The cost c_e is replaced by $100c_e - 1$. Three heuristics based on the solution of E-GMSTP have been tested in [6]: *1)* the transformation using the shortest paths, *2)* the transformation using the shortest paths with maximum number of edges, and *3)* L-GMSTP solved directly as E-GMSTP. (Such a third case clearly provides an upper bound for L-GMSTP since a feasible solution of E-GMSTP is also feasible for L-GMSTP.)

4 A generalization: the E/L-GMSTP

The E/L-GMSTP can be formulated as an integer linear program as follows.

$$\min \sum_{e \in E} c_e x_e$$

s.t.

$$\begin{aligned}
y(V_k) &= 1 && k \in K_E, && (11)\\
y(V_k) &\geq 1 && k \in K_L, && (12)\\
x(E) &= y(V) - 1, &&&&\\
x(\delta(S)) &\geq y_i + y_j - 1 && i \in S \subset V, j \notin S, &&\\
x_e &\in \{0,1\} && e \in E, &&\\
y_i &\in \{0,1\} && i \in V, &&
\end{aligned}$$

where $K = K_E \cup K_L$ is partitioned in two sets K_E and K_L such that the clusters in K_E (resp. K_L) must be visited exactly (resp. at least) once.

This problem is clearly a generalization of both E-GMSTP and L-GMSTP since both sets K_E and K_L can reduce to the empty set. Obviously, in case K_L is the empty set the model can be tightened to become the undirected cluster subpacking formulation described in Section 2.

5 Improving on GMSTP

In this section we concentrate on several techniques to improve on the branch-and-cut framework for Generalized Minimum Spanning Tree problems developed by Feremans, Labbé and Laporte [8] and Feremans [6]. This framework has been modified in the current paper to obtain better results through new branching rules, primal-heuristic algorithms and the use of general-purpose Chvátal-Gomory cuts.

First of all we briefly recall the branch-and-cut method of [8, 6]:

1. **Initialization**: Insert the linear program:

$$\{\min \sum_{e \in E} c_e x_e : (2), (3), x_e, y_i \geq 0, \quad \forall i \in V,\ \forall e \in E\}$$

for E-GMSTP or

$$\{\min \sum_{e \in E} c_e x_e : (9), (10), x_e, y_i \in [0,1], \quad \forall i \in V,\ \forall e \in E\}$$

for L-GMSTP, respectively, in a problem list $\mathcal{L}$. Initialize the incumbent solution $\bar{z}$ to infinity.

2. **Termination**: If $\mathcal{L}$ is empty, STOP. Otherwise, extract one subproblem from $\mathcal{L}$ according to a best-first rule.

3. **LP solution**: Solve the subproblem using an LP-solver and let x^* be its optimal solution and z^* its value. If $z^* \geq \bar{z}$, go to STEP 2. Otherwise[†], if the solution is integer and feasible update the incumbent solution $\bar{z}$ and go to STEP 2.

4. **Separation I**: Separate the special cases of (8) and (7):

$$x(\delta(i)) \geq y_i \quad i \in V \tag{13}$$

and

$$x(E(\{i\} : V_k)) \leq y_i \quad i \in V \setminus (W \cup V_k),\ \forall k \in K \tag{14}$$

for E-GMSTP (where $E(\{i\} : V_k)$ denotes the set of edges having exactly one endpoint in i and the other in V_k, and $W = \{i \in V : i \in V_k, |V_k| = 1\}$ is the set of nodes belonging to a cluster which is a singleton) or

$$x_e \leq y_i \ \text{ and } \ x_e \leq y_j \quad \forall\ e = \{i,j\} \in E \tag{15}$$

for L-GMSTP, respectively[‡].
If violated inequalities are found, add them to the current subproblem and go to STEP 3.

5. **Separation II**: Separate constraints (8).
If violated inequalities are found, add them to the current subproblem and go to STEP 3.

6. **Separation III**: Separate

odd-cycle inequalities and *odd-hole* inequalities[§]

for E-GMSTP or

cutset constraints (4)

for L-GMSTP, respectively[¶].
If violated inequalities are found, add them to the current subproblem and go to STEP 3.

7. **Branching**: Create two new subproblems by branching on a constraint (2) for E-GMSTP, or on the edge whose value is closest to 0.5 and with maximum cost for L-GMSTP, respectively.
Add the subproblems to the list $\mathcal{L}$ and go to STEP 2.

[†]Both local improvement and rounding procedures are applied in the E-GMSTP context (see [8] for details).

[‡]Constraints (13)-(14) are special cases of (8), while constraints (15) are special cases of (7). For the proof of validity and separation details of (13)-(14) and (15) see [8] and [6], respectively.

[§]See [8] for proof of validity and separation details.

[¶]Note that the constraints for E-GMSTP and L-GMSTP in STEP 6 play a rather different role. Indeed, constraints (4) are necessary for the correctness of the branch-and-cut in the L-GMSTP context, while odd-cycle and odd-hole inequalities are redundant constraints for E-GMSTP.

5.1 Improving on E-GMSTP

The branch-and-cut method outlined in the previous section has been modified as follows.

First, we modified the separation procedure: preliminary tests have shown that odd-cycle inequalities and odd-hole inequalities separation routines are rather time-consuming, and these constraints are seldom generated. Instead of these constraints we used the general-purpose Chvátal-Gomory cuts, with and without the two strengthening procedures proposed by Letchford and Lodi [11]. The use of Chvátal-Gomory cuts has been recently re-discovered by several authors (see, e.g., Balas, Ceria, Cornuéjols and Natraj [2]) , and a number of ways of using them have been proposed. In our computational experiments we found very useful to separate a single round of Chvátal-Gomory cuts at the root node of our branch-and-cut tree when no other cuts have been identified and before resorting to branching. After the addition of such a first round, we restart from STEP 3 of the algorithm above.

Second, we modified the branching strategy by branching on nodes, i.e., on the y_i variable closest to 0.5. As it is customary, two subproblems are created having $y_i = 0$ and $y_i = 1$, respectively.

Computational results comparing the algorithm in [8] with the modified one are reported in Section 6.1.

5.2 Improving on L-GMSTP

In addition to the two modifications already described in the previous section for E-GMSTP, and also used for L-GMSTP (i.e., the use of general-purpose Chvátal-Gomory cuts and an effective branching strategy) in the context of L-GMSTP we performed two new modifications.

In the original version of the branch-and-cut method presented in [6], no heuristic algorithm was used in STEP 3. We propose a greedy heuristic based on the classical algorithm of Prim for MSTP. This heuristic is divided in two phases: the first is a rounding procedure which starts from a fractional solution while the second one is a simple local improvement. More precisely:

Phase 1): a maximum-priority spanning tree is computed using Prim's algorithm (a priority proportional to x_e^* is associated with each edge e and, among edges of the same priority, the inter-cluster edges are considered first). This phase stops when a feasible solution $T = (V', E')$ for L-GMSTP is reached.

Phase 2): the solution is improved by removing all the redundant nodes of degree 1 (a node i is redundant if $|(V' \setminus \{i\}) \cap V_k| \geq 1,\ k : i \in V_k$). The corresponding adjacent edges are removed from T one at a time according to nonincreasing edge cost while the remaining edges still form a feasible solution.

Moreover, a more prudent separation policy has been implemented for constraints (4). Indeed, we decided to apply the separation routine based on max-flow computation in case the solution is integer (to detect if it is infeasible), while in case of a fractional solution such that constraints (8), their special cases (13) and constraints (15) are not violated we resort to branching, i.e., we execute STEP 7.

Computational results comparing the algorithm in [6] with the modified one are reported in Section 6.2.

6 Computational Results

In this section we concentrate on the computation of provably optimal solutions for E-GMSTP, L-GMSTP and E/L-GMSTP. This is done by using the same branch-and-cut

framework presented in the previous section, which is specified so as to take into account the main differences among the three problems.

All the codes have been implemented in $C++$ by using the branch-and-cut framework ABACUS [1] version 2.4 (alpha release) and ILOG-Cplex 9.0 as LP solver. All tests were run on a Pentium M 1.6 Ghz notebook with 512 MByte of main memory.

6.1 E-GMSTP Results

Computational results comparing the algorithm in [8] with the modified one on both random-generated Euclidean instances from [8] and the set of instances proposed by Dror, Haouari, and Chaouachi [3] (and denoted in the following as "DHC" instances) are presented in Tables 1-2 and Table 3-4, respectively.

We compare four algorithms, namely, the original algorithm presented in [8], and three versions (v. 0,1,2) of the modified algorithm depending on the type of Chvátal-Gomory cuts used, i.e., classical ones (v.0), strengthened of type 1 (v.1), and strengthened of type 2 (v.2) (see, Letchford and Lodi [11] for details).

Table 1 is organized as follows. Each line refers to a set of five random Euclidean instances generated as in [8]. First, $|V|$, $|K|$ and $|E|$ indicate the number of nodes, clusters and edges, respectively, of the corresponding five instances. Then, we report the number of instances solved to optimality within the time limit of 3,600 CPU seconds (Succ), the average number of nodes of the branch-and-cut tree (Nodes), the average gap of the lower bound computed at the root node with respect to either the optimal solution value or the best know one (LB%), the average separation time expressed in seconds (sepT) and, finally, the average overall time expressed in seconds[‖] (TT). The last row of Table 1 reports for each algorithm the total number of solved instances and the average value over all the instances of the other entries.

Table 2 reports the disaggregated results for the three sets of randomly-generated instances for which not all the instances were solved to optimality (hard instances).

Tables 1 and 2 show that the modified version v.0 obtains the best results for what concerns the number of solved instances and the computing time. As for the lower bound at the root node, the use of the Chvátal-Gomory cuts generally leads to better values, mainly if the strengthening procedures proposed in [11] are applied.

Table 3 has the same structure as Table 1, but the first column indicates the identifier of the single instance (ID) and no column indicating the number of successes is present because the all set of 20 instances is solved to optimality by the four algorithms. In addition, the fifth column indicates the optimal value for the instance (Opt).

Table 4 reports the incremental effects of our modifications on the most significative "DHC" instances. First of all we introduced general-purpose Chvátal-Gomory cuts (only version v.0 is reported) instead of odd-cycle inequalities and odd-hole inequalities, then we changed the branching strategy. The table has the same structure as Table 3, but in addition we report the number of solved linear problems (Lp). Table 4 shows that each modification leads to a reduction of the corresponding average computing times.

Finally, we tested our branch-and-cut algorithm v.0 with respect to that proposed in [8] on generalized Traveling Salesman instances generated according to Fischetti, Salazar and Toth [9]. As also reported in [8], the branch-and-cut algorithms solve all the problems with up to 226 nodes at the root node by using only constraints (8) (i.e., without additional cuts) but instance `ts225`. Thus, the two algorithms perform in the same way. On `ts225`, we

[‖]An instance which is not solved to optimality in the time limit has a computing time of 3,600 CPU seconds for computation of the average time.

Table 1: E-GMSTP: Euclidean Instances Comparison.

			Branch-and-cut [8]					Branch-and-cut, v.0					Branch-and-cut, v.1					Branch-and-cut, v.2				
$\|V\|$	$\|K\|$	$\|E\|$	Succ	Nodes	LB%	sepT	TT	Succ	Nodes	LB%	sepT	TT	Succ	Nodes	LB%	sepT	TT	Succ	Nodes	LB%	sepT	TT
120	20	7140	5	12.20	97.79	6.90	131.60	5	17.40	98.09	6.23	135.49	5	16.20	98.09	5.85	128.47	5	14.20	98.09	5.97	131.45
120	30	7140	5	13.40	98.69	12.98	104.02	5	5.80	98.69	5.09	70.75	5	7.40	98.79	5.18	73.76	5	8.20	98.90	6.18	85.81
120	40	7140	5	30.60	97.70	36.64	237.17	5	33.40	97.62	13.34	197.45	5	55.40	97.70	19.63	323.30	5	49.80	97.78	18.23	284.86
150	15	11175	5	1.40	99.75	1.82	60.87	5	1.80	99.75	2.04	84.93	5	1.80	99.75	2.24	90.71	5	2.20	99.75	2.25	82.69
150	25	11175	5	21.00	98.29	24.04	456.68	5	23.40	98.29	13.98	443.77	5	31.40	98.29	16.30	530.15	5	38.20	98.29	19.03	607.99
150	30	11175	5	25.80	97.75	33.59	429.71	5	25.00	97.85	16.57	394.15	5	23.40	97.85	16.20	393.55	5	23.00	97.96	16.59	405.27
160	20	12720	5	15.00	97.53	20.15	498.30	5	9.40	97.90	14.64	471.61	5	11.40	98.07	13.81	491.34	5	11.80	98.07	15.30	523.99
160	40	12720	3	81.80	95.59	228.94	2124.00	4	109.40	95.63	86.75	2226.96	3	115.40	95.45	95.67	2256.79	2	125.40	95.68	101.33	2402.99
180	15	16110	5	1.80	100.00	4.49	224.44	5	1.80	99.70	3.88	252.71	5	1.80	99.70	4.02	261.15	5	1.80	99.70	4.12	270.94
180	30	16110	5	13.80	98.55	45.57	846.88	5	15.80	98.55	28.23	774.31	5	14.20	98.55	26.16	715.95	5	12.60	98.55	26.49	707.36
200	20	19900	5	11.00	98.45	36.00	1075.30	5	9.40	98.65	20.71	939.69	5	7.40	98.65	21.27	940.80	5	7.40	98.65	22.50	956.61
200	25	19900	4	22.20	96.48	80.56	2483.33	4	17.00	96.54	58.54	2197.82	4	17.00	96.58	56.69	2187.61	3	16.60	96.62	56.61	2329.57
200	40	19900	1	41.00	91.70	212.13	3509.48	2	43.00	91.75	105.84	3208.37	1	40.20	91.78	100.81	3143.80	2	44.20	91.79	98.83	3053.37
			58	22.38	97.56	57.22	937.06	60	24.05	97.62	28.91	876.77	58	26.38	97.63	29.53	887.49	57	27.34	97.68	30.26	910.99

Table 2: E-GMSTP: Euclidean Hard Instances Comparison.

			Branch-and-cut [8]			Branch-and-cut, v.0			Branch-and-cut, v.1			Branch-and-cut, v.2		
$\|V\|$	$\|K\|$	$\|E\|$	Nodes	UB	TT	Nodes	UB	TT	Nodes	UB	TT	Nodes	UB	TT
160	40	12720	123	249	> 3600.00	135	249	3436.41	111	249	2620.05	165	260	> 3600.00
			21	226	502.65	17	226	387.13	7	226	291.22	19	226	375.77
			145	254	> 3600.00	185	252	> 3600.00	171	248	> 3600.00	169	251	> 3600.00
			93	239	1977.30	183	239	2888.31	247	244	> 3600.00	251	241	> 3600.00
			27	247	940.04	27	247	822.94	41	247	1172.67	23	247	839.17
200	25	19900	9	128	2536.06	7	128	2010.21	15	128	2644.38	7	128	1989.08
			15	145	> 3600.00	21	145	> 3600.00	21	154	> 3600.00	21	151	> 3600.00
			7	121	427.56	7	121	572.13	5	121	509.41	3	121	499.64
			57	136	3563.77	39	136	3167.65	33	136	2665.40	39	136	> 3600.00
			23	133	2289.26	11	133	1639.12	11	133	1518.86	13	133	1959.15
200	40	19900	25	264	> 3600.00	31	267	> 3600.00	27	286	> 3600.00	33	249	> 3600.00
			27	244	> 3600.00	47	239	> 3600.00	39	223	> 3600.00	57	224	> 3600.00
			43	208	3147.41	19	208	1874.64	11	208	1319.01	9	208	1245.63
			61	219	> 3600.00	67	216	3367.23	67	222	> 3600.00	47	216	3221.22
			49	251	> 3600.00	71	249	> 3600.00	57	243	> 3600.00	75	230	> 3600.00

Table 3: E-GMSTP: "DHC" Instances Comparison.

					Branch-and-cut [8]				Branch-and-cut, v.0				Branch-and-cut, v.1				Branch-and-cut, v.2			
ID	$\|V\|$	$\|K\|$	$\|E\|$	Opt	Nodes	LB%	sepT	TT	Nodes	LB%	sepT	TT	Nodes	LB%	sepT	TT	Nodes	LB%	sepT	TT
1	25	4	50	23	1	100.00	0.00	0.20	1	100.00	0.00	0.14	1	100.00	0.00	0.15	1	100.00	0.00	0.11
2	25	8	100	45	1	100.00	0.00	0.07	1	100.00	0.00	0.07	1	100.00	0.00	0.07	1	100.00	0.00	0.06
3	25	10	150	37	1	100.00	0.00	0.07	1	100.00	0.00	0.07	1	100.00	0.00	0.08	1	100.00	0.00	0.08
4	50	5	150	18	1	100.00	0.00	0.10	1	100.00	0.00	0.07	1	100.00	0.00	0.07	1	100.00	0.00	0.08
5	50	10	300	27	1	100.00	0.00	0.09	1	100.00	0.00	0.09	1	100.00	0.00	0.08	1	100.00	0.00	0.09
6	75	8	200	55	1	100.00	0.00	0.11	1	100.00	0.00	0.11	1	100.00	0.00	0.11	1	100.00	0.00	0.11
7	75	10	300	67	1	100.00	0.01	0.17	1	100.00	0.00	0.17	1	100.00	0.00	0.18	1	100.00	0.00	0.17
8	75	15	400	58	1	100.00	0.01	0.13	1	100.00	0.00	0.15	1	100.00	0.00	0.12	1	100.00	0.00	0.13
9	100	7	300	37	1	100.00	0.00	0.10	1	100.00	0.00	0.11	1	100.00	0.00	0.10	1	100.00	0.00	0.10
10	100	10	500	49	1	100.00	0.00	0.20	1	100.00	0.02	0.20	1	100.00	0.00	0.19	1	100.00	0.00	0.20
11	150	8	300	65	1	100.00	0.02	0.33	1	100.00	0.02	0.34	1	100.00	0.03	0.33	1	100.00	0.00	0.34
12	150	12	500	79	1	100.00	0.31	3.26	1	100.00	0.55	3.27	1	100.00	0.34	3.48	1	100.00	0.51	3.74
13	200	10	500	65	9	95.38	0.46	7.67	9	96.92	0.40	6.37	5	96.92	0.40	5.23	5	96.92	0.34	4.52
14	200	20	1000	53	1	100.00	0.04	0.47	1	100.00	0.01	0.46	1	100.00	0.04	0.45	1	100.00	0.04	0.46
15	250	10	500	60	1	100.00	0.01	0.43	1	100.00	0.03	0.39	1	100.00	0.02	0.45	1	100.00	0.01	0.42
16	250	25	1000	123	1	100.00	0.39	2.91	1	100.00	0.27	2.87	1	100.00	0.36	2.92	1	100.00	0.31	2.94
17	300	20	1000	95	1	100.00	0.10	1.42	1	100.00	0.13	1.43	1	100.00	0.10	1.43	1	100.00	0.05	1.43
18	300	30	2000	85	1	100.00	0.25	3.07	1	100.00	0.24	3.10	1	100.00	0.29	3.08	1	100.00	0.26	3.12
19	300	40	3000	88	5	100.00	3.50	12.12	7	98.86	1.20	8.60	7	98.86	1.29	8.72	3	100.00	1.08	7.55
20	500	50	5000	109	85	95.41	343.88	1813.08	51	95.41	138.21	1180.59	35	95.41	143.96	1160.76	51	95.41	234.33	2265.11
					5.80	99.54	17.45	92.30	4.20	99.56	7.05	60.43	3.20	99.56	7.34	59.40	3.80	99.62	11.85	114.54

Table 4: E-GMSTP: "DHC" Instances Detailed Results.

	Branch-and-cut [8]				Chvátal-Gomory cuts, v.0				Branch-and-cut, v.0			
ID	Nodes	Lp	LB%	TT	Nodes	Lp	LB%	TT	Nodes	Lp	LB%	TT
13	9	430	95.38	7.67	7	358	96.92	6.34	9	357	96.92	6.37
19	5	337	100.00	12.12	5	187	98.86	5.19	7	306	98.86	8.60
20	85	6047	95.41	1813.08	117	5684	95.41	1594.15	51	3764	95.41	1180.59
	33.00	2271.33	96.93	610.96	43.00	2076.33	97.06	535.23	22.33	1475.67	97.06	398.52

obtained a speedup of 1.12 (computed as the ratio of the computing times of the original algorithm and of the modified one), a slightly reduced number of branching nodes (15 instead of 23) and a better lower bound at the root node due to the use of Chvátal-Gomory cuts (2 units improvement on a absolute gap of 22 units).

6.2 L-GMSTP Results

Computational results comparing the algorithm in [6] with the modified one on the "DHC" instances are presented in Table 5-6.

Table 5 has the same structure as Table 3 but the first algorithm is now the branch-and-cut approach presented in [6]. Moreover, we report as column six the number of selected edges in the optimal solution ($|E'|$).

The table shows that the first eighteen instances require very short computing time. As for instances 19 and 20 the modified version v.0 obtains the best computing times, while the best lower bound values are obtained by the strengthening procedures proposed in [11].

Table 6 reports the incremental effects of our modifications on the most significative "DHC" instances. First of all we implemented a more prudent separation for constraints (4) (i.e., constraints (4) are separated only in case the solution is integer), then we changed the branching strategy, introduced the heuristic and finally we introduced general-purpose Chvátal-Gomory cuts (only version v.0 is reported). The table has the same structure as Table 5, but in addition we report the number of solved linear problems (Lp) and the time spent for the heuristic expressed in seconds (HeurT).

Table 6 shows that separating constraints (4) only in case the solution is integer leads, in some instances, to worse values of the LB%. Anyway, such a negative effect is completely removed by the use of Chvátal-Gomory cuts, which perform better if constraints (4) are not separated in case the solution is fractional. Table 6 also shows that each modification leads to a reduction of the corresponding average computing times.

Before ending this section it has to be noted that Duin, Volgenant and Voß [4] report computational results on the exact solution of L-GMSTP computed through a transformation to the *Steiner Tree Problem* (STP). In particular, a code for the STP is used to compute optimal solutions of the L-GMSTP and then assert the quality of heuristic algorithms for L-GMSTP proposed by the same authors. Incidentally, the computing times on the "DHC" instances are rather short suggesting that this transformation is a competitive way of solving L-GMSTP.

6.3 E/L-GMSTP Results

Preliminary results on the introduced generalization of E-GMSTP and L-GMSTP are presented in this section by naturally adapting the branch-and-cut framework described in the previous section and using a subset of the "DHC" instances for which the optimal solutions computed in the previous sections for E-GMSTP and L-GMSTP were different. These results are reported in Table 7. The table presents the results for the version of the algorithm, among the three versions considered in the previous sections, which has on average the best results, i.e., v.0. The structure of the table is once again as the one of the previous ones, but for the number of clusters for which we specify $|K_E|$ (resp. $|K_L|$), i.e., the number of clusters for which exactly (resp. at least) one node has to be reached.

The table shows these instances as well can be effectively solved through the proposed branch-and-cut algorithm.

Table 5: L-GMSTP: "DHC" Instances Comparison.

						Branch-and-cut [6]				Branch-and-cut, v.0				Branch-and-cut, v.1				Branch-and-cut, v.2			
ID	$\|V\|$	$\|K\|$	$\|E\|$	Opt	$\|E'\|$	Nodes	LB%	sepT	TT	Nodes	LB%	sepT	TT	Nodes	LB%	sepT	TT	Nodes	LB%	sepT	TT
1	25	4	50	23	3	1	100.00	0.00	0.07	1	100.00	0.00	0.03	1	100.00	0.00	0.07	1	100.00	0.00	0.07
2	25	8	100	41	9	1	100.00	0.00	0.08	1	100.00	0.00	0.04	1	100.00	0.00	0.13	1	100.00	0.00	0.08
3	25	10	150	36	10	3	100.00	0.00	0.09	3	100.00	0.00	0.04	3	100.00	0.00	0.10	3	100.00	0.00	0.11
4	50	5	150	18	4	1	100.00	0.01	0.12	1	100.00	0.00	0.06	1	100.00	0.00	0.10	1	100.00	0.00	0.09
5	50	10	300	27	9	1	100.00	0.00	0.10	1	100.00	0.00	0.06	1	100.00	0.01	0.09	1	100.00	0.00	0.10
6	75	8	200	55	7	1	100.00	0.00	0.15	1	100.00	0.00	0.11	1	100.00	0.01	0.14	1	100.00	0.00	0.14
7	75	10	300	67	9	19	95.52	0.13	0.76	19	95.52	0.06	0.63	13	95.52	0.05	0.63	19	95.52	0.08	0.76
8	75	15	400	53	16	39	94.34	0.19	0.92	15	94.34	0.04	0.37	15	94.34	0.03	0.40	11	94.34	0.02	0.37
9	100	7	300	37	6	1	100.00	0.00	0.15	1	100.00	0.00	0.12	1	100.00	0.01	0.16	1	100.00	0.01	0.16
10	100	10	500	48	10	11	97.92	0.07	0.72	3	100.00	0.09	0.54	3	100.00	0.03	0.47	3	100.00	0.09	0.63
11	150	8	300	50	8	5	92.00	0.01	0.33	3	98.00	0.02	0.37	1	100.00	0.03	0.30	1	100.00	0.04	0.29
12	150	12	500	68	14	5	98.53	0.13	0.97	3	98.53	0.17	1.03	1	100.00	0.17	0.89	1	100.00	0.21	0.99
13	200	10	500	44	12	23	86.36	0.14	1.53	7	90.91	0.08	0.81	7	93.18	0.07	0.94	5	93.18	0.06	0.97
14	200	20	1000	50	20	1	100.00	0.00	0.38	1	100.00	0.00	0.38	1	100.00	0.03	0.41	1	100.00	0.01	0.45
15	250	10	500	60	9	3	96.67	0.00	0.72	3	98.33	0.03	0.67	3	98.33	0.04	0.70	1	100.00	0.01	0.60
16	250	25	1000	117	26	3	100.00	0.66	4.02	1	100.00	0.25	2.21	1	100.00	0.20	2.20	1	100.00	0.28	2.90
17	300	20	1000	88	23	27	95.45	0.97	4.95	27	95.45	0.44	5.18	23	95.45	0.45	4.21	19	95.45	0.65	4.51
18	300	30	2000	85	29	7	97.65	0.95	4.87	3	97.65	0.63	4.15	11	97.65	0.55	5.81	17	97.65	0.72	6.85
19	300	40	3000	88	39	67	98.86	11.48	30.08	23	98.86	1.19	8.62	11	98.86	1.39	8.78	19	98.86	1.35	8.44
20	500	50	5000	105	54	71	97.14	313.09	480.36	27	97.14	14.69	94.17	55	98.10	13.95	125.37	51	98.10	16.31	135.58
						14.50	97.52	16.39	26.57	7.20	98.24	0.88	5.98	7.70	98.57	0.85	7.60	7.90	98.66	0.99	8.20

Table 6: L-GMSTP: "DHC" Instances Detailed Results.

	Branch-and-cut [6]				Constraints (4) separated only with integer solutions				Branching on nodes			
ID	Nodes	Lp	LB%	TT	Nodes	Lp	LB%	TT	Nodes	Lp	LB%	TT
3	3	23	100.00	0.09	13	26	97.22	0.05	7	26	97.22	0.05
7	19	122	95.52	0.76	33	140	95.52	0.69	25	132	95.52	0.66
8	39	133	94.34	0.92	275	618	90.57	3.00	71	211	90.57	0.98
13	23	260	86.36	1.53	9	177	86.36	0.77	11	234	86.36	0.97
16	3	290	100.00	4.02	3	290	100.00	4.04	3	290	100.00	4.05
17	27	311	95.45	4.95	37	428	94.32	6.84	29	294	94.32	4.14
18	7	267	97.65	4.87	17	301	97.65	5.92	13	284	97.65	5.52
19	67	557	98.86	30.08	107	734	98.86	30.43	15	356	98.86	8.79
20	71	1015	97.14	480.36	101	1295	97.14	262.21	99	1432	97.14	272.39
	28.78	330.89	96.15	58.62	66.11	445.44	95.29	34.88	30.33	362.11	95.29	33.06

	Heuristic						Branch-and-cut, v.0					
ID	Nodes	Lp	LB%	UB%	HeurT	TT	Nodes	Lp	LB%	UB%	HeurT	TT
3	7	22	97.22	111.11	0.01	0.05	3	17	100.00	111.11	0.00	0.03
7	17	115	95.52	100.00	0.00	0.58	19	115	95.52	100.00	0.02	0.63
8	19	91	90.57	143.40	0.02	0.42	15	83	94.34	107.55	0.01	0.37
13	7	191	86.36	100.00	0.02	0.72	7	175	90.91	100.00	0.03	0.81
16	1	228	100.00	100.00	0.08	2.40	1	226	100.00	100.00	0.12	2.21
17	17	249	94.32	102.27	0.11	3.59	27	364	95.45	102.27	0.18	5.18
18	13	280	97.65	101.18	0.23	5.62	3	280	97.65	101.18	0.15	4.15
19	15	356	98.86	106.82	0.52	9.08	23	252	98.86	109.09	0.39	8.62
20	57	997	97.14	105.71	2.08	201.19	27	675	97.14	102.86	1.45	94.17
	17.00	281.00	95.29	107.83	0.34	24.85	13.89	243.00	96.65	103.78	0.26	12.91

Table 7: E/L-GMSTP: Modified "DHC" Instances.

							Branch-and-cut v.0			
ID	$\|V\|$	$\|K_E\|$	$\|K_L\|$	$\|E\|$	Opt	$\|E'\|$	Nodes	LB%	sepT	TT
2	25	1	7	100	45	7	1	100.00	0.01	0.08
3	25	1	9	150	37	9	1	100.00	0.00	0.04
8	75	1	14	400	58	14	7	98.28	0.03	0.27
10	100	1	9	500	49	9	1	100.00	0.01	0.29
11	150	1	7	300	61	8	15	78.69	0.05	0.96
12	150	1	11	500	75	12	13	94.67	0.34	2.62
13	200	1	9	500	46	12	19	86.96	0.09	1.39
14	200	1	19	1000	53	19	1	100.00	0.00	0.54
16	250	1	24	1000	119	26	7	98.32	0.71	4.60
17	300	1	19	1000	91	20	31	92.31	0.72	7.13
20	500	3	47	5000	108	53	475	94.44	52.04	1085.49

7 Conclusion

We have considered three $\mathcal{NP}$-hard generalizations of the classical MSTP which often arise in practical applications, e.g., in the telecommunication and agricultural settings.

The relationships among these problems, and in particular with respect to their ILP formulations, have been discussed and branch-and-cut approaches have been extensively tested. More precisely, we improved on existing branch-and-cut algorithms from the literature [8, 6] by using new effective primal heuristics, more powerful branching strategies, and general-purpose Chvátal-Gomory cuts.

These modifications have been proved to be effective through computational results and the branch-and-cut approaches seem to be a flexible and powerful tool to handle such problems.

Acknowledgements

Part of this work was carried out when the first author was visiting the University of Bologna exploiting the FNRS grant from the Belgian government. Work partially supported by MIUR and CNR, Italy. Thanks are due to anonymous referees for useful comments and suggestions.

References

[1] ABACUS. *A Branch-And-CUt System*, Version 2.4, alpha release. `http://www.informatik.uni-koeln.de/abacus/`.

[2] E. Balas, S. Ceria and G. Cornuéjols, N. Natraj, Gomory cuts revisited, *Oper. Res. Lett.* 19 (1996) 1–9.

[3] M. Dror, M. Haouari and J. Chaouachi, Generalized Spanning Trees, *European J. Oper. Res.* 120 (2000) 583–592.

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

[5] U. Faigle, W. Kern, P.C. Pop and G. Still, The Generalized Minimum Spanning Tree Problem, *Working Paper, Department of Operations Research and Mathe matical Programming, University of Twente*, `http://www.math.utwente.nl/dos/ormp/preprints.htm`, 2000.

[6] C. Feremans, Generalized Spanning Trees and Extensions, *Ph.D. Dissertation, Université Libre de Bruxelles*, 2001.

[7] C. Feremans, M. Labbé and G. Laporte, A Comparative Analysis of Several Formulations for the Generalized Minimum Spanning Tree Problem, *Networks* 39 (2002) 29–34.

[8] C. Feremans, M. Labbé and G. Laporte, The Generalized Minimum Spanning Tree Problem: Polyhedral Analysis and Branch-and-Cut Algorithm, *Networks* 43 (2004) 71–86.

[9] M. Fischetti, J.J. Salazar and P. Toth, A Branch-and-Cut Algorithm for the Symmetric Generalized Traveling Salesman Problem, *Oper. Res.* 45 (1997) 378–394.

[10] E. Ihler, G. Reich and P. Widmayer, Class Steiner Trees and VLSI-design, *Discrete Appl. Math.* 90 (1999) 173–194.

[11] A.N. Letchford and A. Lodi, Strengthening Chvátal-Gomory cuts and Fractional cuts, *Oper. Res. Lett.* 30 (2002) 74–82.

[12] T.L. Magnanti, L.A. Wolsey and Optimal Trees, in *Handbooks in Operations Research & Management Science*, M.O. Ball, T.L. Magnanti, C.L. Monma and G.L. Nemhauser (eds), Elsevier Science, Amsterdam, Vol. 7, 1995, pp. 503–615.

[13] Y.S. Myung, C.H. Lee and D.W. Tcha, On the Generalized Minimum Spanning Tree Problem, *Networks* 26 (1995) 231–241.

[14] P.C. Pop, New Models of the Generalized Minimum Spanning Tree Problem, *Journal of Mathematical Modelling and Algorithms* 3 (2004) 153–166.

[15] P.C. Pop, W. Kern and G.J. Still, An Approximation Algorithm for the Generalized Minimum Spanning Tree Problem with Bounded Cluster Size, *Working Paper, Department of Operations Research and Mathematical Programming, University of Twente* `http://www.math.utwente.nl/dos/ormp/preprints.htm`, 2001.

Manuscript received 25 January 2005
revised 30 March 2005
accepted for publication 18 April 2005

Corinne Feremans
Institut de Statistique et de Recherche Opérationnelle, Service d'Optimisation,
Université Libre de Bruxelles, Belgium
E-mail address: `cferemans@yahoo.fr`

Andrea Lodi
D.E.I.S., University of Bologna, Viale Risorgimento 2, 40136 Bologna, Italy,
IBM, T.J. Watson Research Center, Yorktown Heights, NY 10598, USA
E-mail address: `alodi@us.ibm.com`

Paolo Toth
D.E.I.S., University of Bologna, Viale Risorgimento 2, 40136 Bologna, Italy
E-mail address: ptoth@deis.unibo.it

Andrea Tramontani
D.E.I.S., University of Bologna, Viale Risorgimento 2, 40136 Bologna, Italy
E-mail address: atramontani@deis.unibo.it