Università di Pisa

Dipartimento di Informatica

Technical Report : TR-99-05

Fast Lower Bounds for the Capacitated Minimum Spanning Tree Problem

Antonio Frangioni, Daniele Pretolani, Maria Grazia Scutella'

March 23, 1999

ADDR: Corso Italia 40,56125 Pisa,Italy. TEL: +39-50-887111. FAX: +39-50-887226

Fast Lower Bounds for the Capacitated Minimum Spanning Tree Problem

A. Frangioni* D. Pretolani† M. G. Scutellà*

February 1999

Abstract

We propose some techniques, based on minimum cost flow computations, for efficiently computing lower bounds for the Capacitated Minimum Spanning Tree problem (CMST). With respect to other methods proposed in the literature, our approaches often provide comparable bounds, with a substantially lower computational cost. For this reason, and for other features discussed in the paper, the proposed methods may be particularly promising in the context of exact algorithms for CMST.

Keywords: *capacitated minimum spanning tree; minimum cost flow; Lagrangean relaxations.*

1 Introduction

In this paper, we consider the Capacitated Minimum Spanning Tree problem (CMST). It is a network design problem, which often appears as a subproblem in the design of local access networks. Given a directed graph G, with set of nodes $N = \{1, 2, \ldots, n\}$, assume that the nodes other than 1 correspond to terminals which have to be connected to a central site, usually a computer site, which is located at node 1. These terminals produce a given amount of information, that has to be sent to the computer site via transmission lines. Most of them use the transmission lines only a small fraction of the time; several terminals can thus be connected to the same line (named multidrop or multipoint line), in order to efficiently share its capacity. In the graph characterization, this multipoint line corresponds to a directed subtree rooted at node 1. Since each input port of the computer site can handle a fixed amount of information, say Q, the maximum flow of information along any link adjacent to the computer site is restricted to Q.

*Department of Computer Science, University of Pisa, Corso Italia 40, 56125 Pisa (Italy)

†Dipartimento di Matematica e Fisica, Università di Camerino, via Madonna delle Carceri, 62032 Camerino (Italy)

Given a cost associated with each transmission line, the network design problem consists in finding a minimum cost spanning arborescence, rooted at node 1, subject to the constraint that the maximum amount of information along any arc incident to 1 is restricted to the given value Q. In the particular case in which each terminal produces one unit of information, CMST reduces to finding a minimum cost spanning arborescence, rooted at node 1, such that any subtree rooted at 1 contains at most Q nodes or terminals.

As proved in [16], CMST is NP-hard for $2 < Q < n/2$. A significant amount of research has been done both in designing heuristics and in proposing lower bounding algorithmic approaches. For what concerns heuristics, the reader is referred to the survey of Amberg, Domschke and Voss [1], and to more recent works on the subject, such as the ones of Sharaia, Gendreau, Laporte and Osman [18] and of Ahuja, Orlin and Sharma [17].

As far as the research on lower bounding techniques is concerned, Gavish [7] presents a *flow-based* formulation, and proposes two Lagrangean relaxations, hereafter referred to as G_1 and G_2. In both cases, the flow conservation constraints are dualized; the resulting Lagrangean problem can be easily solved by inspection, in the case of G_1, while for G_2 it can be reduced to a *degree-constrained minimum arborescence* problem. Standard subgradient optimization techniques are proposed to find good Lagrangean multipliers.

An equivalent formulation, involving less constraints, is suggested by Gouveia [10]. By including different sets of valid inequalities, such as (generalized) subtour elimination constraints, Gouveia defines a hierarchy $REL_1, \ldots, REL_4$ of Lagrangean relaxations, having the flow-based Lagrangean relaxations at its bottom, i.e. $REL_1 = G_1$, whereas REL_2 coincides with a weaker version of G_2. Each relaxation in the hierarchy produces a better lower bound than the preceding one. Due to the exponential size of the considered sets of inequalities, lower bounds are computed using subgradient optimization combined with cut generation techniques. As proved by a wide computational investigation, the gap between G_2 and the bounds REL_3 and REL_4 is usually small, and quite often the improvements are obtained at the expenses of substantially increasing running times.

A quite different formulation, essentially based on generalized subtour elimination constraints, and do not involving flow variables at all, is presented by Gavish [8] for the symmetric case. Also in this case, a Lagrangean relaxation is proposed, and a constraint generation procedure is combined with standard subgradient optimization. We shall denote by G_3 the lower bound obtained by this technique.

As shown by Gouveia [9], for general instances no dominance relations can be established between G_3 and the bounds obtained from the flow-based Lagrangean relaxations, both from a theoretical and a practical point of view. For some symmetric instances, G_3 provides better lower bounds, but the gap with G_2 is usually small. On the other hand, for small values of Q, and for some hard benchmark problems (namely, Euclidean instances with "root in the corner", see [10]) G_3 can be dominated, both theoretically and practically, by some of the bounds in [10].

A tighter lower bounding scheme, with respect to G_3, is proposed by Malik and Yu[15], by using three additional sets of valid inequalities in the constraint generation procedure. However, no computational comparisons are reported by the authors.

Toth and Vigo [19] propose a specialized Lagrangean procedure for approximating the bound G_3, using a different, equivalent formulation. They also propose a new relaxation, that can be reduced to a minimum cost flow problem on a suitable auxiliary network. Finally, they combine the two techniques into an *additive* lower bounding schema, which is used within an enumerative algorithm.

More recently, pure LP based cutting plane approaches have been proposed.

The work by Hall [13] is based on polyhedral results due to Araque, Hall and Magnanti [3] and to Zhang [20]. From a theoretical point of view, Hall's bound may strictly dominate G_3. An extensive computational experimentation shows that several instances are solved to optimality. However, moderately small "root in the corner" instances are still difficult to solve, and the computed lower bounds are still far from the optimum.

Another LP based cutting plane approach is proposed by Gouveia and Martins [12], based on a generalization of the flow-based formulation, named the "hop indexed" formulation. A new set of valid inequalities, denoted "hop ordering" inequalities, are introduced, and a constraint generation procedure is proposed for adding both hop ordering inequalities and generalized subtour elimination constraints to the current linear programming formulation. From a theoretical point of view, also the "hop-indexed" bound may strictly dominate G_3; computational results show that the proposed method seems to be particularly effective on tightly capacitated "root in the corner" instances.

In order to complete the survey, we mention the *multicommodity flow* formulations proposed by Gavish [8] and Kawatra [14]. The linear relaxations provided by these formulations are theoretically tighter than those obtained from single-commodity flows; in practice however, as discussed in [8], the obtained improvements are expected to be small.

Finally, a variant of the problem that considers *flow costs* has been studied by Gouveia and Lopes [11]. In this case, in addition to the fixed costs of connecting the terminals to the computer site, a *communication cost* must be payed to send one unit of information through a transmission line. In [11] several single and multicommodity flow formulations are considered, and new, problem-specific valid inequalities are proposed.

A common feature of all the lower bounding approaches to CMST addressed in the literature sofar, which rely on flow-based formulations, is that the flow conservation constraints are either removed or dualized. No lower bounding approach thus exploits flow computations, for which very fast algorithms and codes exist. Aim of this paper is to propose new lower bounding techniques, based on minimum cost flow computations.

First, we show that the linear relaxation of the flow-based formulation can be approximated by means of a *flow* relaxation, that can be solved by a single minimum cost flow computation. Then, we obtain tighter lower bounds by combining the flow computation with the solution of (degree constrained) minimum arborescence problems. In this way, we obtain a lower bounding procedure that closely approximates the Lagrangean bound G_2. As shown by a wide computational experience, the proposed approaches provide comparable lower bounds, with respect to other methods proposed in the literature, with a substantially lower computational cost.

As a side-effect, the optimal dual solution of the minimum cost flow problem provides useful informations, that can be exploited in different ways. On one side, the dual solution can be used to devise *problem reduction* techniques, based on reduced costs. On the other hand, it provides a good initial set of Lagrangean multipliers for the computation of tighter

Lagrangean lower bounds.

In light of the features outlined above, the proposed lower bounding techniques seem to be particularly attractive in the context of enumerative algorithms for CMST. This will be the subject of a forthcoming paper.

Finally, we remark that all the techniques described in this work can be immediately extended to handle flow costs, as addressed by Gouveia and Lopes [11].

2 The Flow Formulation

Consider a directed graph $G = (N, A)$, where $N = \{1, 2, \ldots, n\}$ and $| A |= m$. Node 1 is assumed to be a distinguished *root node*, having no entering arcs. Each node $i \neq 1$ has an integer demand $q_i > 0$. Given a cost $c(a) = c_{ij}$ for each arc $a = (i, j) \in A$, the *Capacitated Minimum Spanning Tree* problem (CMST) consists in finding a minimum cost directed spanning tree (arborescence) T, rooted at node 1, such that the sum of the node demands in each subtree (except T itself) is not greater than a given value Q.

Since any arborescence contains exactly one arc entering each node $j \neq 1$, the cost of each arc (i, j) can be scaled by

$$\min_{h \in N, h \neq j} c_{hj}.$$

Therefore, we can make the following assumption:

Assumption 2.1 *All the arc costs are non-negative, and each backward star contains at least one zero cost arc.*

The spanning tree feasibility can be restated in terms of *flows* as follows: an arborescence T is feasible if it is possible to send through T q_i units of flow from the root to each node i, with the further restriction that no more than Q units of flow are sent through each arc leaving the root. This is the basis for flow-based formulations of CMST, which will be formally presented next. Before stating these formulations, some more definitions are required.

Let the *weight* of a path from the root to a node i in G be the sum of the demands of the nodes along the path (excluding q_i). Moreover, let $w(i)$ denote the minimum weight of a path from 1 to i for each node $i \neq 1$, whereas $w(1) = 0$.

Based on the above definitions, let us introduce the vectors $U \in \mathbb{R}^n$ and $K \in \mathbb{R}^m$ as follows:

$$\begin{array}{rcll} U_i & = & Q - w(i) & \quad 1 \leq i \leq n; \\ K_{ij} & = & U_i - q_i & \quad (i, j) \in A. \end{array}$$

In a feasible arborescence, the sum of the node demands in a subtree rooted at node i cannot be greater than U_i. Moreover, K_{ij} gives the maximum amount of flow that can be sent through arc (i, j) in any feasible arborescence. Clearly:

$$U_j = \max_{i \neq j} K_{ij}, j = 1, \ldots, n,$$

and thus $U_j \geq K_{ij}$ for each i and j.

We have now all the ingredients to mathematically formulate CMST. Let E denote the $n \times m$ incidence matrix of G, and b be the n-vector defined as follows:

$$b_i = \begin{cases} -\sum_{j=2}^{n} q_j & \text{if } i = 1 \\ q_i & \text{otherwise.} \end{cases}$$

Moreover, let $y \in \{0, \pm 1\}^m$ and $x \in N_+^m$ be two vectors of variables such that y_{ij} takes value 1 if and only if the arc (i, j) belongs to a feasible arborescence, whereas x_{ij} represents the flow along the arc (i, j). The following flow-based formulation can be stated:

Problem (P)

$$Z_P = \min \sum_{(i,j) \in A} c_{ij} y_{ij}$$

$s.t.$

$$\sum_{i \in N, i \neq j} y_{ij} = 1 \qquad j = 2, \ldots, n; \tag{1}$$

$$Ex = b \tag{2}$$

$$x_{ij} \leq K_{ij} y_{ij} \qquad \forall (i, j) \in A; \tag{3}$$

$$y_{ij} \leq x_{ij} \qquad \forall (i, j) \in A; \tag{4}$$

$$y \in \{0, 1\}^m, \; x \in N_+^m. \tag{5}$$

For unit demands, and if all the arcs outgoing from the root are present in G, it is:

$$K_{ij} = \begin{cases} Q & \text{if } i = 1 \\ Q - 1 & \text{otherwise.} \end{cases}$$

This happens, as a particular case, when G is complete. In this case, (P) is equivalent to the well known flow-based formulation due to Gavish [7].

Denote by $(\overline{P})$ the linear relaxation of (P), where the constraints (5) are replaced by:

$$0 \leq y_{ij} \leq 1, \; x_{ij} \geq 0, \qquad \forall (i, j) \in A; \tag{6}$$

let $\overline{Z}_P$ denote the cost of the optimum solution of $(\overline{P})$. Observe that constraints (4) are redundant in problem (P); however, they might be binding in the relaxation $(\overline{P})$. The same observation holds true for the following *root degree* constraint, which requires at least D arcs exiting the root in the solution:

$$\sum_{i \in N, i > 1} y_{1i} \geq D = \left\lceil \frac{\sum_{j=2}^{n} q_j}{Q} \right\rceil. \tag{7}$$

In general, D is a lower bound on the minimum number of subtrees in a feasible arborescence, and it is the actual minimum root degree for unit demands.

The above defined values K_{ij} provide the smallest coefficients which can be used in constraints (3); a value greater than K_{ij} could result in a weaker linear relaxation. In particular, if K_{ij} is replaced by a constant not less than the sum of the demands, (P) becomes a *Minimum Spanning Arborescence* problem (MSA); if, in addition, constraint (7) is included, a *Degree Constrained MSA* problem (DMSA) is obtained. Thus, both MSA and DMSA are valid relaxations of CMST; the DMSA relaxation has been used by Toth and Vigo [19] within their additive lower bound.

3 Flow Based Relaxations

In this section, we propose a new relaxation of (P), that can be reduced to a *Minimum Cost Flow* problem. That provides a first flow-based lower bounding approach to CMST. Then, we study the dual of the proposed relaxation, showing that it can provide useful informations for the CMST computation. Finally, we show how to combine the minimum cost flow computation with other known techniques, obtaining improved flow-based lower bounding methods.

3.1 The Minimum Cost Flow Lower Bound

The proposed relaxation is based on the following observations. As shown before, the sum of the demands in a subtree rooted at node i cannot exceed the value U_i. The following $(n-1)$ constraints could thus be added to (P):

$$\sum_{i \in N,\ i \neq j} x_{ij} \leq U_j \qquad j = 2, \ldots, n. \tag{8}$$

Since $U_j \geq K_{ij}$ for each arc (i,j), constraints (8) are redundant also in the linear relaxation $(\overline{P})$, indeed:

$$\sum_{i \in N,\ i \neq j} x_{ij}/U_j \leq \sum_{i \in N,\ i \neq j} x_{ij}/K_{ij} \leq \sum_{i \in N,\ i \neq j} y_{ij} = 1.$$

However, they are no longer redundant if constraints (1) are dropped (or dualized). In fact, our approach consists in replacing constraints (1) with the (weaker) constraints (8) in $(\overline{P})$. Furthermore, we drop constraints (4). The resulting problem is:

Problem (F)

$$Z_F = \min \sum_{(i,j) \in A} c_{ij} y_{ij}$$

$$s.t.\ \{(2), (3), (6), (8)\}.$$

For each arc (i,j), let us define:

$$g_{ij} = c_{ij}/K_{ij}. \tag{9}$$

Since, by Assumption 2.1, the arc cost c_{ij} can be assumed to be non-negative for each arc (i,j), w.l.o.g. we can restrict our attention to optimal solutions of (F) such that constraints (3) hold as an equality, that is:

$$y_{ij} = x_{ij}/K_{ij} \qquad \forall (i,j) \in A. \tag{10}$$

We can thus consider the following problem, say (F^*), equivalent to problem (F):

Problem (F^*)

$$Z_F = \min \sum_{(i,j)\in A} g_{ij} x_{ij}$$

$$s.t.\ \{(2),(8),x \geq 0\}.$$

Clearly, given a feasible solution x of (F^*), we can obtain a feasible solution $[y,x]$ of F with the same cost by setting $y_{ij} = x_{ij}/K_{ij}$ for each arc (i,j). Next we show that problem (F^*) can be formulated as a minimum cost flow problem in a suitably extended graph.

Denote by $G^* = (N^*, A^*)$ the *expanded graph* obtained from G by duplicating the non-root nodes $2,\ldots,n$, and linking the copies of each node by a directed arc. More formally:

- $N^* = \{1^h\} \cup \{i^t, i^h :\ i = 2,\ldots,n\}$;
- $A^* = \{e_{ij} = (i^h, j^t) : (i,j) \in A\} \cup \{e_j = (j^t, j^h) :\ j = 2,\ldots,n\}$.

Assign to each $e_{ij} \in A^*$ a capacity $+\infty$ and a cost g_{ij}; each arc e_j, which is introduced to model the capacity constraint (8) relative to node j, has capacity U_j and zero cost.

Let E^* denote the $(2n-1)\times(m+n-1)$ incidence matrix of G^*, and let b^* denote the following vector:

$$b_i^* = \begin{cases} -\sum_{j=2}^n q_j & \text{if } i = 1^h \\ q_j & \text{if } i = j^h \text{ for some } j \in N \\ 0 & \text{otherwise.} \end{cases}$$

Moreover, let f be a vector of flows on the arcs of G^*, where $f(e)$ represents the flow on arc $e \in A^*$. We can then define the following minimum cost flow problem on the expanded graph G^*:

Problem (MCF)

$$Z_F = \min \sum_{e_{ij}\in A^*} g_{ij} f(e_{ij})$$

$s.t.$

$$E^* f = b^* \tag{11}$$

$$f(e_j) \leq U_j \qquad \forall e_j \in A^*; \tag{12}$$

$$f \geq 0 \tag{13}$$

Given any feasible flow f in G^*, consider the vector $x(f)$ in G where $x_{ij}(f) = f(e_{ij})$. It is easy to see that $x(f)$ is a feasible solution for (F^*), and its cost is the same as the cost of f in (MCF). Conversely, given a feasible solution x for (F^*), we can define the feasible flow $f = f(x)$ in G^* where $x(f) = x$ and:

$$f(e_j) = \sum_{i\in N, i\neq j} x_{ij} \qquad j = 2,\ldots,n;$$

clearly, the cost of $f(x)$ is the same as the one of x in (F^*). As a consequence, the following propositions hold true:

Proposition 3.1 *Problem* (F^*) *is equivalent to* (MCF).

Proposition 3.2 *The lower bound* Z_F *for CMST can be computed via a single minimum cost flow computation on the expanded graph* G^*.

3.2 The dual of the minimum cost flow relaxation

In this section, we point out some useful properties of the dual of problem (F); to this aim, we exploit the informations provided by the optimal dual solution of the equivalent problem (MCF). Let us formulate the dual (D_{MCF}) of (MCF) first; we denote by $\pi \in \mathbb{R}^{2n-1}$ and $\mu \in \mathbb{R}_+^{n-1}$ the dual variables, corresponding to the flow conservation and the capacity constraints, respectively:

- π_i^h is the potential of node i^h;
- π_i^t is the potential of node i^t (note that $i \neq 1$);
- μ_j is the dual variable associated with the capacity constraint relative to the arc e_j.

Problem (D_{MCF})

$$Z_F = \max \sum_{1 \le i \le n} \pi_i^h b_i^* - \sum_{1 < i \le n} \mu_i U_i$$

$$s.t.$$

$$-\pi_i^h + \pi_j^t \le g_{ij} \qquad \forall e_{ij} \in A^*; \tag{14}$$

$$-\pi_j^t + \pi_j^h - \mu_j \le 0 \qquad \forall e_j \in A^*; \tag{15}$$

$$\mu \ge 0$$

Denote by $[\pi^*, \mu^*]$ an optimal solution of the potential problem (D_{MCF}). For each arc $e_{ij} = (i^h, j^t) \in A^*$, let us denote by δ_{ij} the corresponding reduced cost, that is:

$$\delta_{ij} = g_{ij} + (\pi^*)_i^h - (\pi^*)_j^t.$$

Clearly, $\delta_{ij} \ge 0$ for each arc e_{ij}; moreover, $\delta_{ij} = 0$ if arc (i^h, j^t) carries flow in the optimal solution of (MCF).

Consider now the dual (D_F) of problem (F); we introduce the dual variables $u \in \mathbb{R}^n$, $w \in \mathbb{R}_+^m$ and $v \in \mathbb{R}_+^{n-1}$, corresponding to constraints (2), (3) and (8), respectively.

Problem ($\boldsymbol{D_F}$)

$$Z_F = \max \sum_{1\leq i\leq n} u_i b_i \;-\; \sum_{1<i\leq n} v_i U_i$$

$s.t.$

$$-u_i + u_j - v_j - w_{ij} \leq 0 \qquad \forall (i,j) \in A; \tag{16}$$

$$K_{ij} w_{ij} \leq c_{ij} \qquad \forall (i,j) \in A; \tag{17}$$

$$w, v \geq 0. \tag{18}$$

By suitably combining constraints (16) and (17), it is easy to see that, in any feasible solution of (D_F), the variables w_{ij} range in the following interval:

$$-u_i + u_j - v_j \leq w_{ij} \leq g_{ij}, \qquad \forall (i,j) \in A. \tag{19}$$

In order to obtain an optimal solution for (D_F), consider the optimal solution $[\pi^*, \mu^*]$ of problem (D_{MCF}). Define the vectors u^* and v^* where $u^*_j = (\pi^*)^h_j$ and $v^*_j = \mu^*_j$, for each node j of the original graph G. A technical result is required.

Lemma 3.1 *For each arc (i,j), it is:*

$$g_{ij} + u^*_i - u^*_j = \delta_{ij} - \mu^*_j.$$

Proof: In a feasible flow in the expanded graph G^*, there is at least one unit of flow along the arc $e_j = (j^t, j^h)$. By the complementarity slackness conditions, it is $-(\pi^*)^t_j + (\pi^*)^h_j - \mu^*_j = 0$. From the definition of δ_{ij}, we have:

$$\begin{aligned} g_{ij} + u^*_i - u^*_j &= g_{ij} + (\pi^*)^h_i - (\pi^*)^h_j \\ &= g_{ij} + (\pi^*)^h_i - (\pi^*)^t_j - \mu^*_j \\ &= \delta_{ij} - \mu^*_j. \end{aligned}$$

□

The following theorem shows that an optimal solution of (D_F) can be obtained from u^* and v^*.

Theorem 3.1 *Each solution $[u^*, v^*, w]$ such that, for each $(i,j) \in A$, it is*

$$g_{ij} - \delta_{ij} \leq w_{ij} \leq g_{ij},$$

is optimum for (D_F).

Proof: From Lemma 3.1, it follows that:

$$-u^*_i + u^*_j - v^*_j = g_{ij} - \delta_{ij}.$$

Since $\delta_{ij} \geq 0$, it follows from (19) that $[u^*, v^*, w]$ is feasible for (D_F). Moreover, the value of the objective function corresponding to $[u^*, v^*, w]$ is Z_F, thus $[u^*, v^*, w]$ is optimum for (D_F). □

According to Theorem (3.1), it is possible to obtain several alternative optimum solutions of (D_F). As we shall see, some of these alternative solutions may be more suitable for implementing problem reduction techniques. Let us consider two opposite situations. As a first case, consider the solution $[u^*, v^*, w^*]$ where, for each arc (i,j), we let $w^*_{ij} = g_{ij} - \delta_{ij}$. The second case is the solution $[u^*, v^*, \overline{w}]$ where, for each arc (i,j), we let $\overline{w}_{ij} = g_{ij}$.

For each arc $(i,j) \in A$, let us denote by g'_{ij} and by c'_{ij} the reduced costs of the variables x_{ij} and y_{ij}, respectively, with respect to an optimum solution $[u, v, w]$ of problem (D_F):

$$\begin{aligned} g'_{ij} &= 0 - (-u_i + u_j - v_j - w_{ij}), \\ c'_{ij} &= c_{ij} - K_{ij} w_{ij}. \end{aligned}$$

The optimum solution $[u^*, v^*, w^*]$ yields the following reduced costs:

$$\begin{aligned} g'_{ij} &= u^*_i - u^*_j + v^*_j + w^*_{ij} = \delta_{ij} - g_{ij} + w^*_{ij} = 0; \\ c'_{ij} &= c_{ij} - K_{ij} g_{ij} + K_{ij}\delta_{ij} = K_{ij}\delta_{ij}. \end{aligned}$$

On the other hand, the optimum solution $[u^*, v^*, \overline{w}]$ yields the following reduced costs:

$$\begin{aligned} g'_{ij} &= u^*_i - u^*_j + v^*_j + \overline{w}_{ij} = \delta_{ij} - g_{ij} + g_{ij} = \delta_{ij}; \\ c'_{ij} &= c_{ij} - K_{ij} g_{ij} = 0. \end{aligned}$$

Note that the solutions $[u^*, v^*, \overline{w}]$ and $[u^*, v^*, w^*]$ assign zero reduced costs to the y variables and to the x variables, respectively. Moreover, for an arc $(i,j) \in A$ such that $\delta_{ij} = 0$, we have $w^*_{ij} = \bar{w}_{ij} = g_{ij}$, and $c'_{ij} = g'_{ij} = 0$.

For several difficult combinatorial optimization problems, *problem reduction techniques* have been introduced in the context of exact solution approaches like branch and bound. These techniques allow one to fix integer variables at specific nodes of the branch and bound tree. If a linear programming (or Lagrangean) lower bound is available, it is often possible to exploit the corresponding reduced costs for problem reduction. An example from the literature concerning CMST can be found in [15].

As proved before, the optimum solution of the minimum cost flow problem (MCF) provides the lower bound Z_F; moreover, the dual problem (D_{MCF}) allows one to obtain alternative optimum solutions of problem (D_F). This gives alternative reduced cost vectors, g' and c', for the variable vectors x and y, respectively. Assume that an upper bound, say UB, is available for the CMST optimum solution. Then the following reduction techniques, depending on g' and c', can be suggested:

Proposition 3.3 *If $Z_F + c'_{ij} \geq UB$, then it is possible to fix $y_{ij} = 0$, i.e. to delete the arc (i,j) from the current graph.*

Similarly, if $Z_F + g'_{ij} \geq UB$, x_{ij}, and consequently y_{ij}, can be fixed to value 0.

Observe that, by the results presented before, in the case of an arc (i,j) having a *positive* reduced cost δ_{ij}, the choice $w_{ij} = w^*_{ij} = g_{ij} - \delta_{ij}$ could be preferable, being $c'_{ij} = K_{ij}\delta_{ij} \geq \delta_{ij} > 0$. This might be crucial in deleting the arc (i,j) from the graph.

Clearly, such reduction techniques can be particularly significant in practice, since the arc deletion process may greatly improve the computational efficiency of enumerative methods, by limiting the depth of the branch and bound tree.

3.3 Combined Lower Bounds

The *additive lower bound* technique (see [4]) allows to combine different relaxations of the same problem into an improved lower bounding procedure. In particular, this technique is well suited for "complementary" relaxations, obtained from different sets of constraints. In our case, the flow-based relaxation may be successfully combined with the "arborescence-based" MSA and DMSA relaxations. For the latter problems, quite efficient algorithms are known [4, 19]; moreover, the optimal reduced costs can be obtained, at the end of the computation, at substantially no extra cost.

We can obtain additive lower bounds for CMST as follows: first, solve an MSA (or DMSA) problem, and then solve problem (F) (via problem (MCF)) using the optimum reduced costs provided by the previous computation, instead of the original arc costs. The sum of the optimal values of the two problems is a valid lower bound for (P). We denote by *DMSA+MCF* and by *MSA+MCF* the additive lower bounds obtained solving a DMSA problem and an MSA problem, respectively.

In principle the opposite approach, namely, solving a (D)MSA problem with the reduced costs of problem (F), might be pursued too. Unfortunately, a direct application of this approach does not improve the bound provided by (F). To see why this is true, consider an arborescence, say T_F, only containing arcs (i, j) whose corresponding arcs $(i^h, j^t) \in G^*$ carry a positive flow in the optimum solution of problem (MCF). Such an arborescence exists by the flow decomposition theorem (see e.g. [2]). For each arc (i, j) in T_F, both the reduced costs c'_{ij} and g'_{ij} are zero. Moreover, observe that at least $D = \left\lceil \frac{\sum_{j=2}^{n} q_j}{Q} \right\rceil$ arcs must carry flow out of the root in the optimal solution of MCF; therefore, at least D arcs exiting the root have zero reduced costs. This suffices to conclude that the cost of the optimal arborescences for MSA and DMSA is zero.

Due to this negative result, a slightly more involved approach has been followed. In practice, instead of computing reduced costs for (F), and using them as the arc costs in an MSA (or DMSA) problem, we use the optimal variables u^* of problem (D_F) as the Lagrangean multipliers in a suitable Lagrangean relaxation of problem (P). This amounts to solving a DMSA (or MSA) problem where we use Lagrangean reduced costs, instead of Linear Programming reduced costs.

Consider the formulation obtained from (P) by replacing constraints (1) and (5) with the (implicit) constraints $y \in \mathcal{A}$, where $\mathcal{A}$ is the family of y vectors defining an arborescence in G. A Lagrangean relaxation of (P) can be obtained by dualizing the flow conservation constraints (2) via multipliers $\lambda \in \mathbb{R}^n$:

$$\varphi(\lambda) = \lambda^T b + \min \sum_{(i,j)\in A} c_{ij} y_{ij} + (\lambda_i - \lambda_j) x_{ij}$$

$$s.t.\ \{(3), (4), y \in \mathcal{A}\}.$$

It is easy to see that, for a given multiplier vector λ, in any optimal solution of $\varphi(\lambda)$ it is:

$$x_{ij} = \begin{cases} y_{ij} K_{ij} & \text{if} \quad \lambda_i - \lambda_j < 0; \\ y_{ij} & \text{if} \quad \lambda_i - \lambda_j \geq 0. \end{cases}$$

Therefore, it is possible to "project" the cost of the x-variables onto the one of the y-variables:

$$c_{ij}^{\lambda} = \begin{cases} c_{ij} + K_{ij}(\lambda_i - \lambda_j) & \text{if} \quad \lambda_i - \lambda_j < 0; \\ c_{ij} + (\lambda_i - \lambda_j) & \text{if} \quad \lambda_i - \lambda_j \geq 0; \end{cases}$$

so obtaining the following equivalent formulation:

$$\varphi(\lambda) = \lambda^T b + \min \sum_{(i,j)\in A} c_{ij}^{\lambda} y_{ij}$$

$$s.t. \ \{y \in \mathcal{A}\}.$$

Actually, $\varphi(\lambda)$ corresponds to the Lagrangean relaxation REL_2 in the hierarchy defined by Gouveia [10]. The (slightly) stronger relaxation G_2, defined by Gavish [7], also includes the constraints (7); we denote the corresponding Lagrangean problem by $\varphi_D(\lambda)$. Recall that $\varphi_D(\lambda)$ reduces to a DMSA problem.

Consider now the setting $\lambda = u^*$, and the corresponding Lagrangean costs c_{ij}^*. Observe that, if $u_i^* - u_j^* < 0$, then c_{ij}^* might be negative; however, if $u_i^* - u_j^* \geq 0$, then $c_{ij}^* \geq 0$, by Assumption 2.1. The following technical results are required.

Lemma 3.2

$$u_i^* - u_j^* < 0 \implies c_{ij}^* = c_{ij} + K_{ij}(u_i^* - u_j^*) = K_{ij}(g_{ij} + u_i^* - u_j^*) = K_{ij}(\delta_{ij} - \mu_j^*).$$

Proof: It follows from (9) and from Lemma (3.1). □

Lemma 3.3

$$\delta_{ij} = 0 \implies c_{ij}^* = K_{ij}(-\mu_j^*).$$

Proof: By Lemma (3.1), it is $u_i^* - u_j^* = -g_{ij} - \mu_j^* \leq 0$. If $u_i^* - u_j^* = 0$, then it must be $g_{ij} = -\mu_j^* = 0$, which implies $c_{ij} = 0$ and thus $c_{ij}^* = 0$. Otherwise, that is if $u_i^* - u_j^* < 0$, it is $c_{ij}^* = K_{ij}(-\mu_j^*)$, by Lemma (3.2). □

Lemma 3.4

$$c_{ij}^* \geq U_j(-\mu_j^*) \qquad\qquad \forall (i,j) \in A.$$

Proof: Recall that $c_{ij} \geq 0$, $\delta_{ij} \geq 0$, $\mu_j^* \geq 0$, and $K_{ij} \leq U_j$ for each (i,j). From Lemma 3.2, if $u_i^* - u_j^* < 0$ then $c_{ij}^* = K_{ij}(\delta_{ij} - \mu_j^*) \geq -U_j\mu_j^*$. On the other hand, if $u_i^* - u_j^* \geq 0$ then $c_{ij}^* \geq 0 \geq -U_j\mu_j^*$. □

By the above characterization of the Lagrangean costs c^*, the following relationship can be established between the proposed combined bounds and the lower bound Z_F:

Theorem 3.2

$$\varphi_D(u^*) = \varphi(u^*) \geq Z_F.$$

Proof: Let $\overline{y}$ denote any spanning arborescence for G, and let $c^*(\overline{y})$ denote its cost w.r.t the costs c^*; from Lemma 3.4 it is:

$$c^*(\overline{y}) = (u^*)^T b + \sum_{(i,j)\in A} c^*_{ij}\overline{y}_{ij} \geq (u^*)^T b + \sum_{j\in N, j>1} U_j(-\mu^*_j) = Z_F.$$

This suffices to conclude that $\varphi_D(u^*) \geq \varphi(u^*) \geq Z_F$. In order to prove that $\varphi_D(u^*) = \varphi(u^*)$, recall that in the optimal solution of (MCF) there exist at least D arcs carrying flow out of the root. Consider one such arc $(1,j)$: since $\delta_{1j} = 0$, by Lemma 3.3 it is $c^*_{1j} = K_{1j}(-\mu^*_j)$; moreover, it is $K_{1j} = U_j = Q$. From Lemma 3.4, we conclude that c^*_{1j} is the minimum cost of an arc entering node j. As a consequence, we can assume that $(1,j)$ belongs to the optimal solution for $\varphi(u^*)$, and the thesis follows. □

In light of Theorem 3.2, it is computationally preferable to solve $\varphi(u^*)$, i.e. an MSA problem, instead of $\varphi_D(u^*)$. We shall denote by *MCF+MSA* the resulting combined bound, which can be tighter than the lower bound based on a single minimum cost flow computation. In fact, we can give an upper bound for the obtained improvement, i.e. $\varphi(u^*) - Z_F$.

To this aim, consider again an arborescence T_F containing arcs with positive flow. From Lemma 3.3, the cost $c^*(T_F)$ of T_F w.r.t. the costs c^* is given by

$$c^*(T_F) = \sum_{(i,j)\in T_F} -K_{ij}\mu^*_j.$$

For each node $j > 1$, define:

$$\overline{U}_j = \min_i K_{ij}.$$

Then, being the maximum gap between Z_F and $\varphi(u^*)$ not greater than the difference between Z_F and the cost of the arborescence T_F plus $(u^*)^T b$, it is:

$$\varphi(u^*) - Z_F \leq (u^*)^T b + c^*(T_F) - Z_F = \sum_{(i,j)\in T_F} -K_{ij}\mu^*_j - \sum_{j\in N, j>1} -U_j\mu^*_j \leq \sum_{j\in N, j>1} (U_j - \overline{U}_j)\mu^*_j.$$

In particular, in the case of complete graphs and unit demands, being $K_{1j} = U_j = Q$ and $K_{ij} = U_j - 1$ for $i > 1$, the maximum gap is bounded from above by $\sum_{j\in N, j>1} \mu^*_j$.

4 Experimental results

In this section we report the results of a wide computational experimentation, aimed to investigate the practical efficiency of the proposed flow-based lower bounding approaches.

All the algorithms have been implemented in C++ language, and run on a workstation HP 9000-712/80. Minimum spanning arborescences were solved by an implementation of the algorithm of Fischetti and Toth [5], whereas degree-constrained MSA problems were solved adapting the method proposed by Toth and Vigo [19]. Minimum cost flow problems were computed via a C++ implementation of the algorithm RELAX IV by Bertsekas; unlike the original Fortran code, the C++ code can handle non integer flow costs.

In our experimentation, three groups of (symmetric) instances have been considered.

The first group contains the unit demand Euclidean instances TC and TE, introduced in [10]. In these instances, non-root nodes are points in a 100×100 grid, and arc costs are the Euclidean distances (rounded up). The two classes differ for the position of the root node, which is placed in the center of the grid for TC instances, and in the corner for TE. For each class, we considered two sets of instances, with 40 and 80 non-root nodes, respectively. We let Q vary in $\{3, 5, 10\}$ for instances with 40 nodes, and in $\{5, 10, 20\}$ for instances with 80 nodes.

The second group contains non-unit demands *random* instances with 50, 100 and 200 nodes, named $cm50$, $cm100$ and $cm200$, respectively. For these instances, costs and demands are randomly generated integers in the interval $[1, 100]$. For each class, Q varies in $\{200, 400, 800\}$.

The above instances are part of the *OR Library* maintained at Imperial College, and can be retrieved by anonymous `ftp` to `mscmga.ms.ic.ac.uk`.

The third group consists of 16 non-unit demand, Euclidean instances obtained from test sets for the *Capacitated Vehicle Routing* problem. These sets can be retrieved by anonymous `ftp` to `mscmga.ms.ic.ac.uk` and to `elib.zib-berlin.de`; the solved problems (denoted by VRP) are the ones listed in [13], Table VI. In Table 8 we report for each instance the number of nodes n, the capacity Q and the interval $[d_{\min}, d_{\max}]$ of node demands.

We report three different groups of computational results.

Aim of the first one (see Section 4.1) is to compare the behaviour of our flow-based lower bounds, namely MCF (i.e. the value Z_F), MSA+MCF, $DMSA$+MCF and MCF+MSA. We also compare our bounds to the optimum value of the Lagrangean dual:

$$L = \max_{\lambda} \varphi_D,$$

which is the best bound obtainable from relaxation G_2 [7]. In practice, we observed that relaxation φ could be used instead, since the corresponding optimum value (i.e. the best bound obtainable from REL_2) is about the same as L. We do not report the CPU times relative to such a computation, since our only aim is to show that value L can be closely approximated via efficient flow-based procedures.

The Lagrangean dual optima have been computed via a *bundle* algorithm. Bundle algorithms are variants of cutting plane algorithms for nondifferentiable optimization problems, which are carachterized by the use of some "stabilizing device" to improve the numerical stability and efficiency. Unlike the implementations of the subgradient algorithm commonly used for approaching Lagrangean duals, bundle methods guarantee finite termination and certify the epsilon-optimality of the obtained solution. The particular bundle implementation used to compute value L is described in [6].

The second group of results concerns *fast* Lagrangean bounds, where we limit the time spent in the Lagrangean dual optimization process computing value L. In particular, we considered the cases in which the number of iterations of the bundle method was limited to 50 and 100. In this situation, the choice of the initial Lagrangean multipliers can be crucial. Indeed, we investigated a flow based *warm start* strategy, where the initial multipliers are given by the optimum potentials of the dual problem (D_F). This strategy is compared to the standard choice of zero initial multipliers. As shown in Section 4.2, the obtained results seem to confirm the effectiveness of the flow-based "warm start" strategy.

Finally, we compare our flow-based bounds to the results of Hall [13] and of Gouveia and Martins [12], who obtained the current best lower bounds proposed in the literature. The comparison is limited to a small subset of the above instances. The results show that the flow-based bounds are reasonably close to the bounds in [13, 12], taken into account that they are obtained at a significantly lower computational cost.

In the following tables, "Time" denotes the CPU time, in seconds, whereas "Gap" refers to $(UB-LB) \div LB \times 100$, where LB is the generated lower bound and UB is the best known upper bound for the instance under consideration. Upper bounds are taken from several sources [1, 13, 18, 12]; as far as we know, the same upper bounds were used by Gouveia and Martins [12] and by Hall [13] to extimate relative errors. For Euclidean and random problems, each entry is the average over five instances, while for the VRP problems each entry refers to a single instance.

4.1 Flow based bounds

Tables 1, 2 and 3 show the results of our flow-based bounds on Euclidean, random and VRP instances respectively. The gap corresponding to the Lagrangean Dual optimum L is reported.

		L	**MCF**		**MSA+MCF**		**DMSA+MCF**		**MCF+MSA**	
Type	Q	Gap	Time	Gap	Time	Gap	Time	Gap	Time	Gap
TC40	3	3.46	0.01	6.37	0.01	14.21	0.03	12.84	0.02	6.18
TC40	5	5.15	0.01	7.36	0.01	10.31	0.03	8.12	0.01	7.32
TC40	10	1.71	0.00	6.07	0.01	2.80	0.02	2.41	0.01	6.07
TE40	3	2.85	0.02	6.46	0.03	15.24	0.05	23.53	0.02	5.54
TE40	5	6.05	0.02	8.89	0.03	15.75	0.05	18.38	0.03	8.12
TE40	10	4.73	0.02	7.36	0.02	8.55	0.03	8.13	0.02	7.00
TC80	5	4.38	0.03	6.48	0.05	7.61	0.08	9.62	0.04	5.90
TC80	10	1.59	0.03	2.31	0.05	2.11	0.09	2.73	0.04	2.26
TC80	20	0.28	0.03	1.29	0.04	0.32	0.07	0.37	0.04	1.29
TE80	5	5.93	0.21	8.82	0.28	22.39	0.35	27.62	0.19	8.10
TE80	10	7.25	0.17	10.21	0.18	17.15	0.28	21.38	0.17	9.78
TE80	20	2.87	0.14	7.92	0.12	6.19	0.17	6.82	0.15	7.82

Table 1: TC and TE instances: Time and Gap

First observe that $MCF + MSA$ improves MCF at a negligible computational cost[1]. However, the obtained improvements are in some cases quite small, in particulare for random and VRP instances.

By the above results, no clear dominance relationship can be observed between $MSA + MCF$ and $DMSA + MCF$. Moreover, there is no dominance relationship between $MCF +$

[1]Due to the approximations introduced by Unix time accounting procedures, time for $MCF + MSA$ is smaller than time for MCF in some cases.

		L	MCF		MSA+MCF		DMSA+MCF		MCF+MSA	
Type	Q	Gap	Time	Gap	Time	Gap	Time	Gap	Time	Gap
cm50	200	15.06	0.04	17.92	0.06	34.49	0.08	43.81	0.04	17.75
cm50	400	13.43	0.05	20.48	0.05	24.26	0.07	25.81	0.05	20.36
cm50	800	10.94	0.04	25.06	0.03	13.81	0.05	14.40	0.05	25.01
cm100	200	22.85	0.10	23.91	0.08	27.03	0.23	27.11	0.10	23.73
cm100	400	8.67	0.10	12.37	0.08	11.12	0.17	10.74	0.12	12.37
cm100	800	2.97	0.10	9.83	0.08	3.82	0.18	4.03	0.12	9.83
cm200	200	30.98	0.56	31.49	0.62	31.68	1.35	34.70	0.55	31.32
cm200	400	11.52	0.26	12.29	0.32	11.79	1.18	15.23	0.33	12.29
cm200	800	3.65	0.35	5.26	0.43	3.73	0.89	6.13	0.45	5.26

Table 2: cm instances: Time and Gap

MSA, on one side, and the best value provided by $MSA + MCF$ and $DMSA + MCF$ on the other side. However, a clear trend can be observed in Euclidean and random instances: for a given set of graphs, $MCF + MSA$ gives the best results when the capacity constraints are tight, but can be outperformed if the capacity constraint is loose.

As follows from the above observations, none of the flow-based bounds emerges as a clear winner. However, the *maximum* of the flow-based bounds is quite close to L for most of the instances, and is always greater than $9L/10$. Due to this property, and being all these bounds computable in a very fast way, an *integrated* approach, which computes two (or more) bounds, and returns their maximum, can be suggested. According to this idea, we shall denote by BF our *best flow-based* bound, where:

$$BF = \max\{MSA + MCF,\ MCF + MSA\}.$$

Clearly, the CPU time for computing BF is the sum of the CPU times for $MSA + MCF$ and $MCF + MSA$. The choice of $MSA + MCF$ instead of $DMSA + MCF$ is somehow arbitrary, and is due to the fact that in most cases the maximum bound is provided either by $MSA + MCF$ or by $MCF + MSA$.

4.2 Lagrangean Bounds

The aim of the second group of results is twofold: on one side, we compare fast Lagrangean bounds to flow-based bounds, on the other side, we investigate the effectiveness of the "warm start" strategy. In practice, the "warm start" approach consists in computing the bound $MCF + MSA$ first, and then applying a limited number of iterations of the bundle method, possibly improving the bound. In other words, $MCF + MSA$ can be seen as an extreme case of "warm start" where we limit the number of iterations to one.

The results of the Lagrangean computations for Euclidean, random and VRP instances are reported in tables 4, 5 and 6 respectively. The entries "ws" refer to the "warm start" strategy, while entries "nows" refer to zero initial multipliers; for both cases, we considered a limit of 50 and of 100 iterations. Note that the entry "Time" for "warm start" gives the

	L	MCF		MSA+MCF		DMSA+MCF		MCF+MSA	
Type	Gap	Time	Gap	Time	Gap	Time	Gap	Time	Gap
Ei122	4.68	0.00	12.33	0.01	6.49	0.01	6.49	0.01	12.33
Ei123	5.05	0.00	13.18	0.00	6.18	0.00	6.18	0.01	13.18
Ei130	5.41	0.01	13.45	0.01	9.47	0.01	9.47	0.01	13.45
Ei133	10.79	0.02	16.17	0.02	33.07	0.03	30.65	0.02	16.17
Ei151	2.70	0.02	6.15	0.02	3.26	0.04	3.83	0.03	6.15
Eilb76	8.29	0.04	11.92	0.05	12.97	0.07	15.13	0.05	11.71
Eila76	5.36	0.04	8.81	0.05	7.71	0.09	9.03	0.05	8.81
Eilc76	2.90	0.04	6.65	0.04	4.19	0.09	5.30	0.06	6.65
Eild76	2.12	0.04	6.40	0.05	2.99	0.09	3.21	0.05	6.40
Eila101	2.15	0.08	5.95	0.10	3.64	0.19	3.64	0.08	5.95
Eilb101	8.00	0.09	10.47	0.09	12.31	0.21	12.13	0.10	10.29
Att48	4.06	0.03	13.31	0.03	4.49	0.03	4.49	0.03	13.31
Vrpnc4	5.12	0.22	16.93	0.30	7.97	0.52	6.95	0.23	16.93
Vrpnc5	6.75	0.49	17.13	0.74	13.66	1.38	12.40	0.55	17.13
Vrpnc11	9.84	0.66	13.70	0.37	27.92	0.62	28.72	0.67	13.49
Vrpnc12	7.79	0.08	13.27	0.07	21.04	0.19	10.82	0.09	13.27

Table 3: VRP instances: Time and Gap

total time spent in the bundle iterations and in the flow problem. We also report time and gap values for the bound BF.

As a preliminary remark, we notice that the "warm start" strategy does not necessarily give the best bound, althogh it gives a better bound more often than not. This is not surprising, however: a different starting point imply a different trajectory followed by the optimization process, and, even if the initial point is better, there is no guarantee that, after a fixed number of iterations, the final point will be better too.

More in detail, we must distinguish the case of Euclidean instances from the case of random and VRP instances.

For Euclidean instances, the Lagrangean bounds clearly improve on the flow-based ones. The improvement is in some cases relevant, and the obtained results closely approximate the theoretical optimum L; in fact, in some cases the bundle method converges before the fixed limit of iterations. However, computational times are at least one order of magnitude higher than those required by flow-based bounds. For these instances, "warm start" and zero initial multipliers are substantially equivalent, both in terms of bound and computation times.

On the other hand, for random and VRP instances flow-based bounds dominate fast Lagrangean bounds. Except for two cases (VRP problems Att48 and Vrpnc12) the bound BF is tighter, and the difference in the gap values can be relevant. Moreover, BF is at least one order of magnitude faster. For these instances, the "warm start" approach does not improve the bound $MCF + MSA$: to this aim, more than 100 bundle iterations would be necessary. However, the behaviour of the Lagrangean bound without "warm start" is even worse: its results are reasonably better in favourable cases, but they are much worse in some

		nows50		nows100		ws50		ws100		BF	
Type	Q	Time	Gap	Time	Gap	Time	Gap	Time	Gap	Time	Gap
TC40	3	0.29	3.46	0.36	3.46	0.25	3.46	0.34	3.46	0.03	6.18
TC40	5	0.53	5.33	0.84	5.15	0.48	5.30	0.81	5.15	0.02	7.32
TC40	10	0.60	2.00	1.25	1.71	0.50	2.12	0.96	1.71	002	2.80
TE40	3	0.28	3.08	0.45	2.85	0.26	2.85	0.31	2.85	0.05	5.54
TE40	5	0.41	6.27	0.81	6.02	0.41	6.08	0.78	6.02	0.05	8.12
TE40	10	0.49	5.18	1.05	4.80	0.36	5.18	0.75	4.77	0.04	7.00
TC80	5	1.88	4.60	4.07	4.40	1.89	4.54	4.15	4.40	0.09	5.90
TC80	10	2.32	1.89	5.10	1.61	1.92	2.02	3.99	1.68	0.09	2.11
TC80	20	1.68	0.44	3.63	0.35	1.26	0.71	2.85	0.33	0.08	0.32
TE80	5	1.69	6.42	3.49	6.00	1.87	6.01	3.67	5.95	0.47	8.10
TE80	10	2.25	8.27	4.80	7.42	2.03	7.82	4.34	7.34	0.35	9.78
TE80	20	2.10	3.78	4.67	3.13	1.92	3.67	4.01	3.09	0.27	6.19

Table 4: Lagrangean bounds: TC and TE instances

bad cases.

From the above results, we may conclude that the "warm start" strategy, though not always optimal, shows a more stable behaviour. More precisely, an approach that combines flow-based computations and Lagrangean computations seems to be preferable to a purely Lagrangean approach. It must also be remarked that a combined approach shows more degrees of flexibility. For example, by carefully choosing the number of bundle iterations after the "warm start" we can find a tradeoff between the quality of the bound and its computational cost.

Observe that the "warm start" technique is very general indeed, and the obtained results may suggest to try it within different formulations, or different lower bounding approaches.

4.3 Comparison to Other Bounds

For a subset of the instances considered above, we can compare our flow-based bounds, in particular BF, to the best lower bounds proposed in the literature.

We consider the results of the cutting plane method proposed by Hall [13], as well as two lower bounds by Gouveia and Martins [12], referred to as Lag and as $RHSCF+HO+GSE$ by the authors: Lag is the best lower bound given by the Lagrangean approaches proposed in [10], whereas $RHSCF + HO + GSE$ is a cutting plane method described in [12]. In the following tables, we shall denote by H the results reported by Hall, and by $GM1$ and $GM2$ the results of Lag and $RHSCF + HO + GSE$ respectively.

The comparison with the three bounds above is limited to the Euclidean problems with 80 nodes, and is reported in Table 7. The symbol "-" indicates that the computation has not been completed due to memory limitations. In this table, as far as Gouveia and Martins' results are concerned, the entry "Gap " indicates the ratio $(UB - LB) \div UB \times 100$, as reported by the authors. In Table 8 we compare our results for the VRP instances to the

		nows50		nows100		ws50		ws100		BF	
Type	Q	Time	Gap	Time	Gap	Time	Gap	Time	Gap	Time	Gap
cm50	200	0.38	67.78	0.86	57.13	0.35	17.75	0.87	17.75	011	17.75
cm50	400	0.36	36.29	0.84	22.06	0.36	20.36	0.93	20.36	0.10	20.36
cm50	800	0.26	17.52	0.66	17.52	0.34	25.01	0.80	25.01	0.08	13.81
cm100	200	1.35	30.37	2.66	30.37	1.44	23.73	2.71	23.60	0.18	23.73
cm100	400	0.98	12.89	2.08	12.89	1.35	12.37	2.97	12.37	0.20	11.12
cm100	800	0.81	5.36	1.74	5.36	1.28	9.83	2.58	9.83	0.21	3.82
cm200	200	5.65	37.18	10.91	37.18	6.19	31.32	10.77	31.32	1.17	31.32
cm200	400	4.24	17.30	9.35	17.30	5.09	12.29	11.12	12.29	0.66	11.79
cm200	800	3.75	7.56	7.20	7.56	5.55	5.26	10.37	5.26	0.88	3.73

Table 5: Lagrangean bounds: cm instances

ones given by Hall (see Table VI in [13]).

In the case of Hall's results, "Time" refers to CPU seconds on a Silicon Graphics Indy R4000 SC workstation, whereas Gouveia and Martins' results refer to a Sun Sparc station 10-52. In order to compare the running times on these different machines, in Table 9 we report the corresponding values of SPECfp92 and of SPECint92. Our machine is thus about twice faster than the Indy R4000, whereas the Indy R4000 and the Sun Sparc station are more comparable in speed.

It must remarked that we are comparing our fast lower bound BF to the results of rather sophisticated cutting plane algorithms, that is, to lower bounds obtained at high computational cost. Clearly, we did not expect our flow based bounds to be competitive in terms of gap values; indeed, the cutting plane lower bounds always dominate BF. Nevertheless, the obtained results are encouraging, and allow us to draw some interesting conclusions.

Indeed, in most cases the results of BF can be considered comparable to those of the best known bounds. Here, "comparable" means that the cutting plane bounds often "close" only a fraction of the gap obtained by BF, moreover, this improvement is obtained at a high cost, both in terms of execution times and memory requirements.

Only for a few VRP instances the bound H turns out to be significantly tighter than BF, and reasonably fast; actually, for some of these instances the cutting plane method is quite effective, and can found the optimum solution in a couple of cases.

In light of the above results, it is conceivable that flow based lower bounds may be successfully applied within (exact or approximate) enumerative methods for CMST. Recall that the flow computation provides useful informations for problem reduction techniques. A branch and bound algorithm exploiting fast flow bounds is currently under investigation, and will be the subject of a forthcoming paper.

	nows50		nows100		ws50		ws100		BF	
Type	Time	Gap	Time	Gap	Time	Gap	Time	Gap	Time	Gap
Ei122	0.08	7.89	0.19	7.89	0.09	12.33	0.22	10.31	0.02	6.49
Ei123	0.08	9.42	0.20	9.42	0.08	13.18	0.21	13.18	0.01	6.18
Ei130	0.12	13.87	0.34	13.87	0.12	13.45	0.38	13.45	0.02	9.47
Ei133	0.20	39.72	0.49	37.43	0.22	16.17	0.52	16.17	0.04	16.17
Ei151	0.34	4.97	0.88	4.97	0.39	6.15	0.83	6.15	0.05	3.26
Eilb76	0.66	19.25	1.76	19.25	0.74	11.71	1.67	11.71	0.10	11.71
Eila76	0.94	11.55	2.29	9.94	0.69	8.81	1.65	8.81	0.10	7.71
Eilc76	0.81	6.65	2.30	6.65	0.82	6.65	1.66	6.65	0.10	4.19
Eild76	0.54	4.33	1.85	4.33	1.24	5.47	2.45	3.66	0.10	2.99
Eila101	2.11	4.97	4.72	3.83	1.71	5.95	3.28	5.95	0.18	3.64
Eilb101	1.57	15.19	2.94	15.19	1.71	10.29	3.87	10.29	0.19	10.29
Att48	0.25	4.72	0.54	4.26	0.23	5.44	0.51	4.78	0.06	4.49
Vrpnc4	2.53	9.37	9.17	9.37	3.87	16.93	8.20	16.93	0.53	7.97
Vrpnc5	4.76	16.28	13.96	16.28	6.91	17.13	14.74	17.13	1.29	13.66
Vrpnc11	2.49	37.05	4.26	14.55	3.34	13.49	5.99	13.49	1.04	13.49
Vrpnc12	1.60	11.79	3.76	11.79	1.85	13.27	5.28	13.27	0.16	13.27

Table 6: Lagrangean bounds: VRP instances

References

[1] S. Voß A. Amberg, W. Domschke. Capacitated minimum spanning trees: Algorithms using intelligent search. *COTP*, 1:9–40, 1996.

[2] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. Networks flows. In G. L. Nemhauser, A. H. G. Rinnooy-Kan, and M. J. Todd, editors, *Optimization*, volume 1 of *Handbooks Oper. Res. Management Sci.*, pages 211–369. North-Holland, Amsterdam, 1989.

[3] J. R. Araque and L. A. Hall T. L. Magnanti. Capacitated trees, capacitated routing, and associated polyhedra. Technical report, Princeton University, Princeton NJ, 1990.

[4] M. Fischetti and P. Toth. An additive bounding procedure for combinatorial optimization problems. *Operations Research*, 37(2):319–328, 1989.

[5] M. Fischetti and P. Toth. An efficient algorithm for the min-sum arborescence problem on complete digraphs. *ORSA Journal on Computing*, 5(4):426–434, 1993.

[6] A. Frangioni. *Dual-Ascent methods and Multicommodity flow problems.* PhD thesis, TD 5/97, Dipartimento di Informatica, Università di Pisa, Pisa, Italy, 1997.

[7] B. Gavish. Formulations and algorithms for the capacitated minimal directed tree problem. *J. Assoc. Comput. Mach.*, 30:118–132, 1983.

[8] B. Gavish. Augmented lagrangean based algorithms for centralized network design. *IEEE Transactions on Communications*, 33(12):1247–1257, 1985.

[9] L. Gouveia. A comparison of directed formulations for the capacitated minimal spanning tree problem. *telecommunications Systems*, 1:51–66, 1993.

[10] L. Gouveia. A $2n$ constraint formulation for the capacitated minimal spanning tree problem. *Oper. Res.*, 43(1):130–141, January–February 1995.

[11] L. Gouveia and M. J. Lopes. Valid inequalities for non-unit demand capacitated spanning tree problems with flow costs. Unpublished manuscript, July 1998.

[12] L. Gouveia and P. Martins. The capacitated minimum spanning tree problem: an experiment with a hop-indexed model. (to appear in *Ann. Oper. Res.*), 1997.

[13] L. Hall. Experience with a cutting plane algorithm for the capacitated spanning tree problem. *INFORMS Journal on Computing*, 8(3):219–234, Summer 1996.

[14] R. Kawatra. A multicommodity network flow application for the capacitated minimal spanning tree problem. *Opsearch*, 31(4):296–308, 1994.

[15] K. Malik and G. Yu. A branch and bound algorithm for the capacitated minimum spanning tree problem. *Networks*, 23:525–532, 1993.

[16] C. Papadimitriou. The complexity of the capacitated tree problem. *Networks*, 8:217–230, 1978.

[17] D. Sharma R. Ahuja, J.B. Orlin. New neighborhood search structures for the capacitated minimum spanning tree problem. working paper, 1998.

[18] Y. M. Sharaiha, M. Gendreau, G. Laporte, and I. H. Osman. A tabu search algorithm for the capacitated shortest spanning tree problem. *Networks*, 29:161–171, 1997.

[19] P. Toth and D. Vigo. An exact algorithm for the capacitated shortest spanning arborescence. *Ann. Oper. Res.*, 61:121–141, 1995.

[20] N. Zhang. Facet–defining inequalities for minimum spanning trees. Master's thesis, Princeton University, Princeton NJ, 1993.

		H		GM1		GM2		BF	
Type	Q	Time	Gap	Time	Gap	Time	Gap	Time	Gap
TC80	5	303.61	2.903	574	3.440	3095	0.865	0.09	5.90
TC80	10	81.86	1.182	12256	1.752	10620	0.997	0.09	2.11
TC80	20	54.72	0.00	9415	0.163	-	-	0.08	0.32
TE80	5	4733.89	5.211	601	3.661	8858	0.713	0.47	8.10
TE80	10	648.83	4.704	13699	5.067	11141	1.227	0.35	9.78
TE80	20	145.75	1.823	14148	2.181	-	-	0.27	6.19

Table 7: Comparison to other bounds: TC and TE instances

					H		BF	
Type	n	Q	$d_{\min}$	$d_{\max}$	Time	Gap	Time	Gap
Ei122	22	6000	100	2100	0.54	0.0	0.02	6.49
Ei123	23	4500	60	4100	0.41	0.64	0.01	6.18
Ei130	30	4500	100	3100	0.70	0.0	0.02	9.47
Ei133	33	8000	40	4000	3.72	1.39	0.04	16.17
Ei151	51	160	3	41	6.29	1.06	0.05	3.26
Eilb76	76	100	1	37	127.42	5.68	0.10	11.71
Eila76	76	140	1	37	66.09	3.28	0.10	7.71
Eilc76	76	180	1	37	52.80	1.28	0.10	4.19
Eild76	76	220	1	37	32.83	0.38	0.10	2.99
Eila101	101	200	1	41	82.09	0.20	0.18	3.64
Eilb101	101	112	1	41	187.75	5.49	0.19	10.29
Att48	48	15	1	1	4.02	1.74	0.06	4.49
Vrpnc4	151	200	1	41	842.33	3.00	0.53	7.97
Vrpnc5	200	200	1	41	6336.11	5.06	1.29	13.66
Vrpnc11	121	200	2	35	1904.51	5.29	1.04	13.49
Vrpnc12	101	200	10	50	44.12	0.39	0.16	13.27

Table 8: Comparison to Hall's bound: VRP instances

Machine	**SPECfp92**	**SPECint92**
Indy R4000	63.0	57.5
HP9000 712/80	122.3	93.0
SPARCStation 10/51	83.0	65.2

Table 9: Comparison among machines