CIRRELT
Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport
Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation

A Lagrangian-Based Branch-and-Bound Algorithm for the Two-Level Uncapacitated Facility Location Problem with Single-Assignment Constraints

Bernard Gendron
Paul-Virak Khuong
Frédéric Semet

April 2013

CIRRELT-2013-21

Bureaux de Montréal :
Université de Montréal
C.P. 6128, succ. Centre-ville
Montréal (Québec)
Canada H3C 3J7
Téléphone : 514 343-7575
Télécopie : 514 343-7121

Bureaux de Québec :
Université Laval
2325, de la Terrasse, bureau 2642
Québec (Québec)
Canada G1V 0A6
Téléphone : 418 656-2073
Télécopie : 418 656-2624

www.cirrelt.ca

UNIVERSITÉ LAVAL · UQÀM Université du Québec à Montréal · HEC MONTRÉAL · ÉCOLE POLYTECHNIQUE MONTRÉAL · Université de Montréal

A Lagrangian-Based Branch-and-Bound Algorithm for the Two-Level Uncapacitated Facility Location Problem with Single-Assignment Constraints

Bernard Gendron[1], Paul-Virak Khuong[1], Frédéric Semet[2]

[1] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT) and Department of Computer Science and Operations Research, Université de Montréal, P.O. Box 6128, Station Centre-Ville, Montréal, Canada H3C 3J7

[2] Laboratoire d'Automatique, Génie Informatique et Signal (LAGIS), École Centrale de Lille, Bâtiment C, bureau C343, Cité Scientifique, B.P. 48, 59651 Villeneuve D'Ascq Cedex, France

Abstract. We consider the two-level uncapacitated facility location problem with single-assignment constraints (TUFLP-S), a problem that arises in industrial applications in freight transportation and telecommunications. We present a new Lagrangian relaxation approach for the TUFLP-S, based on solving a single-level uncapacitated facility location problem (UFLP) as the Lagrangian subproblem. We also develop a Lagrangian heuristic that includes a MIP-based large neighborhood search heuristic exploring neighborhoods by solving a series of small UFLPs. The dual and primal bounds thus obtained are used within an enumerative algorithm that implements specialized branching rules. Our computational experiments on instances derived from an industrial application in freight transportation, as well as on large, hard, artificial instances confirm the efficiency of our specialized branch-and-bound algorithm.

Keywords. Two-level uncapacitated facility location, Lagrangian relaxation, Lagrangian heuristic, branch-and-bound algorithm.

Acknowledgements. Financial support for this project was provided by the Natural Sciences and Engineering Research Council of Canada (NSERC) and by the Fonds de recherche du Québec - Nature et technologies (FRQNT). This research was also partly supported by the International Campus on Safety and Intermodality in Transportation, the Nord-Pas-de-Calais Region, the European Community, the Regional Delegation for Research and Technology, the French Ministry of Higher Education and Research, and the National Center for Scientific Research. We thank Antonio Frangioni for providing his bundle code.

Results and views expressed in this publication are the sole responsibility of the authors and do not necessarily reflect those of CIRRELT.

Les résultats et opinions contenus dans cette publication ne reflètent pas nécessairement la position du CIRRELT et n'engagent pas sa responsabilité.

---

* Corresponding author: Bernard.Gendron@cirrelt.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2013

© Copyright Gendron, Khuong, Semet and CIRRELT, 2013

1 Introduction

The two-level uncapacitated facility location problem (TUFLP) (Kaufman et al. 1977) is an extension of the uncapacitated facility location problem (UFLP) (Krarup and Pruzan 1983). The UFLP consists in locating facilities from a finite set of potential sites and in assigning each customer to one of the selected facility locations in order to minimize the total costs, which include fixed costs for opening facility locations and assignment costs between customers and facility locations. In the TUFLP, the finite set of potential facility locations is replaced with two levels of such locations, depots at the upper level and satellites at the lower one. The only arcs are between depots and satellites, as well as between satellites and customers. The problem is to decide which depots and satellites to open, and to which depot-satellite pair each customer should be assigned, in order to satisfy customer demands at minimum cost (Aardal et al. 1996). This problem arises in the design and operation of hierarchical networks that take advantage of economies of scale, most notably in freight transportation (Gendron and Semet 2009), but also in telecommunications (Chardaire et al. 1999).

We are concerned with a variant of the TUFLP that forces a single-assignment property on satellites: each satellite can be linked to at most one depot, and fixed costs are imposed on the use of arcs between depots and satellites. The single-assignment constraints ensure that any solution is a forest of trees rooted at depots. This variant of the TUFLP was introduced in (Chardaire et al. 1999), motivated by an application in the design of telecommunications networks. Our study of the TUFLP with single-assignment (TUFLP-S) is motivated by an industrial application in freight transportation, related to the operation of multicommodity distribution systems over a short-term planning horizon (Gendron and Semet 2009). In that industrial problem, depot and satellite locations typically correspond to freight terminals and parking spaces, respectively, for which the locations may vary every day in response to demand fluctuations. The TUFLP-S arises as a subproblem in decomposition and heuristic methods for solving the optimization problem derived from this application.

Our contribution is three-fold. First, we compare a mixed-integer programming (MIP) formulation for the TUFLP-S with previously described formulations for variants of the TUFLP. Then, from that formulation, we derive a Lagrangian relaxation scheme that provides stronger lower bounds than the linear programming (LP) relaxation, as well as a MIP-based large neighborhood search (LNS) heuristic; these bounding algorithms are combined in a Lagrangian heuristic to compute tight lower and upper bounds. Finally, we embed the Lagrangian heuristic within a branch-and-bound algorithm that uses specialized branching schemes. On industrial instances, the tighter bounds result in the exploration of fewer nodes than by a state-of-the-art MIP solver, and, on large artificial instances, the specialized branching schemes reduce computational times by a factor of two to eight.

Section 2 introduces MIP formulations for variants of the TUFLP, and compares them with our formulation for the TUFLP-S. A description of the Lagrangian heuristic follows in Section 3: first, the Lagrangian relaxation scheme; then, a heuristic to solve the Lagrangian dual; and finally, the MIP-based LNS heuristic. Section 4 outlines the branch-and-bound algorithm and its branching schemes. Section 5 reports computational results on hard artificial instances and on large industrial instances.

2 TUFLP Formulations

A general model for two-level uncapacitated facility location problems is introduced in (Barros and Labbé 1994). In addition to transportation costs for each path from depot to satellite to customer and fixed costs on the use of depots and satellites, the model includes fixed costs for arcs from depots to satellites. The model can be stated as follows.

Let I be the set of potential depot locations, J the set of potential satellite locations and K the set of customer locations, and let

$$y_i = \begin{cases} 1, & \text{if depot } i \text{ is open,} \\ 0, & \text{otherwise,} \end{cases} \qquad \forall i \in I,$$

$$z_j = \begin{cases} 1, & \text{if satellite } j \text{ is open,} \\ 0, & \text{otherwise,} \end{cases} \qquad \forall j \in J,$$

$$t_{ij} = \begin{cases} 1, & \text{if depot } i \text{ and satellite } j \text{ are operating together,} \\ 0, & \text{otherwise,} \end{cases} \qquad \forall (i,j) \in I \times J,$$

and

$$x_{ijk} = \begin{cases} 1, & \text{if customer } k \text{ is served through pair } (i,j), \\ 0, & \text{otherwise,} \end{cases} \qquad \forall\, (i,j,k) \in I \times J \times K.$$

The general TUFLP, noted TUFLP-G, can then be formulated as

$$v(\text{TUFLP-G}) = \min \sum_{i \in I} f_i y_i + \sum_{j \in J} g_j z_j + \sum_{(i,j) \in I \times J} h_{ij} t_{ij} + \sum_{(i,j,k) \in I \times J \times K} c_{ijk} x_{ijk}$$

subject to

$$
\begin{aligned}
\sum_{(i,j)\in I\times J} x_{ijk} &= 1, & \forall k \in K, && (1)\\
x_{ijk} &\le t_{ij}, & \forall (i,j,k) \in I\times J\times K, && (2)\\
\sum_{j\in J} x_{ijk} &\le y_i, & \forall (i,k) \in I\times K, && (3)\\
\sum_{i\in I} x_{ijk} &\le z_j, & \forall (j,k) \in J\times K, && (4)\\
0 \le x_{ijk} &\le 1, & \forall (i,j,k) \in I\times J\times K, &&\\
y_i &\in \{0,1\}, & \forall i \in I, &&\\
z_j &\in \{0,1\}, & \forall j \in J, &&\\
t_{ij} &\in \{0,1\}, & \forall (i,j) \in I\times J, &&
\end{aligned}
$$

where f_i, g_j and h_{ij} are the nonnegative fixed costs for, respectively, each depot $i \in I$, each satellite $j \in J$ and each pair of depot-satellite $(i,j) \in I \times J$ and where c_{ijk} is the nonnegative total transportation cost for each path from a depot i to a satellite j to a customer k (Barros and Labbé 1994).

Constraints (1) guarantee that the demand for each customer is satisfied exactly, and constraints (2)–(4) ensure that fixed costs are incurred for the use of depot-satellite pairs, depots and satellites. Note that the integrality of the x_{ijk} variables can be relaxed without affecting the optimal objective value. Thus, TUFLP-G implicitly ensures that the flow to each customer is never split.

Alternative versions of the general model also include the following constraints:

$$
\begin{aligned}
x_{ijk} &\le y_i, & \forall (i,j,k) \in I\times J\times K, && (5)\\
x_{ijk} &\le z_j, & \forall (i,j,k) \in I\times J\times K, && (6)\\
t_{ij} &\le y_i, & \forall (i,j) \in I\times J, && (7)\\
t_{ij} &\le z_j, & \forall (i,j) \in I\times J. && (8)
\end{aligned}
$$

However, (5) are dominated by (3), and (6) by (4). Constraints (7) and (8) are redundant given nonnegative fixed costs h_{ij} and constraints (2)–(4). Indeed, $h_{ij} \ge 0$ implies the existence of an optimal solution such that $t_{ij} = \max_{k\in K}\{x_{ijk}\} \equiv x_{ijk^*}$ for each $(i,j) \in I \times J$; thus, $y_i \ge \sum_{j'\in J} x_{ij'k^*} \ge x_{ijk^*} = t_{ij}$, and (7) are implied by (2) and (3). Likewise, (8) are implied by (2) and (4).

Note that an optimal solution to TUFLP-G does not necessarily satisfy the single-assignment property. Transportation costs that vary significantly depending on the depot, for a given customer, may lead to optimal solutions in which the same satellite is linked to multiple depots.

We consider a variant in which we enforce this single-assignment property, expressed with the additional constraints

$$\sum_{i \in I} t_{ij} \leq 1, \qquad \forall j \in J. \tag{9}$$

The redundant constraints (7) are also added, as they are useful in solving the model. Indeed, the binary nature of the t_{ij} variables along with constraints (7) imply $y_i \in \{0,1\}$ for each $i \in I$. When adding the redundant constraints (7), it is thus possible to relax the integrality of the y_i variables and still maintain feasibility. We have observed that state-of-the-art MIP solvers perform better when these variables are allowed to take fractional values, rather than restricting them to binary values. In addition, we also use constraints (7) to strengthen the Lagrangian relaxation described in Section 3.

Constraints (9) allow us to project out the z_j variables by using the equations

$$z_j = \sum_{i \in I} t_{ij}, \qquad \forall j \in J.$$

The fixed cost g_j on each satellite j can then be folded in the fixed cost $l_{ij} = g_j + h_{ij}$ for every arc $(i,j) \in I \times J$. Constraints (4) can be eliminated, since they are simple aggregations of constraints (2).

The resulting formulation is TUFLP-S, where the y_i variables are allowed to take fractional values, as explained above:

$$v(\text{TUFLP-S}) = \min \sum_{i \in I} f_i y_i + \sum_{(i,j) \in I \times J} l_{ij} t_{ij} + \sum_{(i,j,k) \in I \times J \times K} c_{ijk} x_{ijk}$$

subject to

$$\begin{aligned}
\sum_{(i,j) \in I \times J} x_{ijk} &= 1, & \forall k \in K,\\
\sum_{i \in I} t_{ij} &\leq 1, & \forall j \in J,\\
x_{ijk} &\leq t_{ij}, & \forall (i,j,k) \in I \times J \times K,\\
\sum_{j \in J} x_{ijk} &\leq y_i, & \forall (i,k) \in I \times K,\\
t_{ij} &\leq y_i, & \forall (i,j) \in I \times J,\\
0 \leq x_{ijk} &\leq 1, & \forall (i,j,k) \in I \times J \times K,\\
0 \leq y_i &\leq 1, & \forall i \in I,\\
t_{ij} &\in \{0,1\}, & \forall (i,j) \in I \times J.
\end{aligned}$$

Most studies on two-level uncapacitated location problems without single-assignment are concerned with a simplification of the general model TUFLP-G: fixed costs on links between depots and satellites, h_{ij}, are always zero. Two seminal papers (Kaufman et al. 1977, Ro and Tcha 1984) marked early research on this classical TUFLP: they introduced MIP formulations and specialized lower bounding methods, and exploited them in branch-and-bound algorithms. Additional side constraints were also considered in the past (Ro and Tcha 1984); we ignore such variants in this paper. More recent approaches (Aardal et al. 1996, Barros 1995, Landete and Marín 2009) are based on the MIP formulation obtained by eliminating the t_{ij} variables (since $h_{ij} = 0$ for each $(i, j) \in I \times J$) and constraints (2) from TUFLP-G; this does not affect LP relaxation bounds, as values for t_{ij} variables can be assigned easily, without affecting the total cost. We call the resulting formulation TUFLP-C:

$$v(\text{TUFLP-C}) = \min \sum_{i \in I} f_i y_i + \sum_{j \in J} g_j z_j + \sum_{(i,j,k) \in I \times J \times K} c_{ijk} x_{ijk}$$

subject to

$$\begin{aligned}
\sum_{(i,j) \in I \times J} x_{ijk} &= 1, & \forall k \in K,\\
\sum_{j \in J} x_{ijk} &\le y_i, & \forall (i,k) \in I \times K,\\
\sum_{i \in I} x_{ijk} &\le z_j, & \forall (j,k) \in J \times K,\\
0 \le x_{ijk} &\le 1, & \forall (i,j,k) \in I \times J \times K,\\
y_i &\in \{0,1\}, & \forall i \in I,\\
z_j &\in \{0,1\}, & \forall j \in J.
\end{aligned}$$

It is sometimes further assumed that the transportation costs c_{ijk}, for all $(i, j, k) \in I \times J \times K$, are sums of per-arc costs $d_k c_{ij} + c_{jk}$, thus allowing the use of a more compact, but weaker, two-index formulation. Such a structure for transportation costs, in conjunction with h_{ij} costs equal to zero, make it possible to impose or to relax the single-assignment constraints without affecting the optimal value (Chardaire et al. 1999). Sets of open depots and satellites obtained from an optimal solution can be converted into a complete solution of TUFLP-S with a greedy procedure (that breaks ties consistently). Thus, if only for this large class of instances, TUFLP-C and TUFLP-S are equivalent.

Constraints (3) and (4) define facets of the feasible polytope for TUFLP-C (Barros 1995). TUFLP-S preserves nearly all the constraints of TUFLP-C, including (3); the only missing constraints are (4), which are replaced with the stronger constraints (2). Thus, when all three formulations are applicable, *TUFLP-S leads to a stronger LP relaxation than both TUFLP-G and TUFLP-C*. Moreover, the difference is obtained by strictly improving on constraints that define facets of the TUFLP-C polytope. This can only be achieved by expanding the decision variables to include t_{ij} variables and by explicitly considering single-assignment constraints.

Solution methods for TUFLP-G, TUFLP-S and TUFLP-C can be cast into three classes. Some approaches strengthen the formulation with valid inequalities, many of them facet-defining (Aardal et al. 1996, Landete and Marín 2009), leading to large-scale models. Other approaches attempt to decrease solution times by computing approximate LP bound values, via dual ascent or Lagrangian relaxations combined with subgradient methods (Barros 1995, Barros and Labbé 1994, Gao and Robinson Jr 1992). Finally, both metaheuristics (Barros and Labbé 1994) and approximation algorithms (Bumb 2001, Zhang 2006) have been developed to quickly obtain primal solutions.

The TUFLP with single-assignment has been studied in (Chardaire et al. 1999), where the TUFLP-S formulation presented here is introduced. Lower bounds are obtained with a Lagrangian relaxation scheme in which the dual is solved with a subgradient method, and upper bounds are computed with a simulated annealing method.

We have already shown that the TUFLP-S formulation leads to stronger LP relaxations bounds than the TUFLP-C formulation, when they can be compared, at the expense of increased formulation size: each constraint (4) is replaced with multiple constraints (2), and the number of binary variables grows from $|I| + |J|$ to $|I \times J|$. Rather than attempting to quickly obtain approximate LP relaxation bounds for this large-scale formulation, we will compute even stronger bounds than the LP relaxation with a Lagrangian relaxation scheme, extract upper bounds with a MIP-based LNS heuristic, and further accelerate a branch-and-bound algorithm with specialized branching schemes.

3 Lagrangian Heuristic

The TUFLP-S is similar to the (single-level) UFLP, a problem for which there exists a large body of literature and efficient solution methods. We exploit this strong foundation with a Lagrangian relaxation scheme in which the subproblems are UFLPs and with a MIP-based LNS heuristic that explores neighborhoods by solving UFLPs. These two components are further combined in a Lagrangian heuristic: the Lagrangian relaxation provides lower bounds and initial solutions, while the MIP-based LNS heuristic repairs and improves these solutions.

3.1 Lagrangian Relaxation

In (Barros and Labbé 1994), it is proposed to obtain Lagrangian bounds for TUFLP-G (or the related TUFLP-C) by relaxing constraints (1), (3) and (4). The alternative chosen in (Chardaire et al. 1999) is to bound TUFLP-S by relaxing (1), (3) and (9). In both cases, the Lagrangian bound is theoretically equal to the LP bound.

We focus on the Lagrangian relaxation obtained by dualizing only constraints (2). As we now show, the resulting Lagrangian subproblem can be converted into a significantly smaller, efficiently-solved, UFLP. Moreover, this subproblem does not exhibit the integrality property, and the relaxation thus yields stronger bounds than the LP relaxation of TUFLP-S, both in theory and in practice.

When relaxing constraints (2), the Lagrangian subproblem $S(\lambda)$ can be formulated as follows, where $\lambda \geq 0$ is the vector of Lagrange multipliers:

$$v(S(\lambda)) = \min \sum_{i \in I} f_i y_i + \sum_{(i,j) \in I \times J} \left(l_{ij} - \sum_{k \in K} \lambda_{ijk} \right) t_{ij} + \sum_{(i,j,k) \in I \times J \times K} (c_{ijk} + \lambda_{ijk}) x_{ijk}$$

subject to

$$\begin{aligned}
\sum_{(i,j) \in I \times J} x_{ijk} &= 1, & \forall k \in K,\\
\sum_{i \in I} t_{ij} &\leq 1, & \forall j \in J,\\
\sum_{j \in J} x_{ijk} &\leq y_i, & \forall (i,k) \in I \times K,\\
t_{ij} &\leq y_i, & \forall (i,j) \in I \times J,\\
0 \leq x_{ijk} &\leq 1, & \forall (i,j,k) \in I \times J \times K,\\
0 \leq t_{ij} &\leq 1, & \forall (i,j) \in I \times J,\\
y_i &\in \{0,1\}, & \forall i \in I.
\end{aligned}$$

In formulation TUFLP-S, all the variables are conceptually binary, but the x and y variables are left free to take fractional values. In the Lagrangian subproblem, all the variables are conceptually binary, but the x and t variables are free to take fractional values, simplifying the formulation. This modification is valid, since the integrality constraints on the y_i variables are redundant in TUFLP-S and can therefore be added to the Lagrangian subproblem; then, the integrality of the t_{ij} variables can be relaxed without changing the optimal value of the Lagrangian subproblem. It can be simplified further, as we now show.

A simple domination argument confirms that, for each pair of depot $i \in I$ and customer $k \in K$, all but one $x_{i \cdot k}$ variables can be eliminated. If, in a given optimal solution, there is a $j \in J$ such that $x_{ijk} = 1$, that variable can always be substituted with $x_{ij'k}$, where $j' = \arg\min_{j \in J}\{c_{ijk} + \lambda_{ijk}\}$, a variable corresponding to a least-cost (penalized) path from i to k.

By defining

$$\tilde{c}(\lambda)_{ik} = \min_{j \in J}\{c_{ijk} + \lambda_{ijk}\},\ \forall (i,k) \in I \times K,$$

the Lagrangian subproblem $S(\lambda)$ can then be solved as a more compact MIP, $S_c(\lambda)$, in which the number of variables scales quadratically (rather than cubically) with the instance size:

$$v(S_c(\lambda)) = \min \sum_{i\in I} f_i y_i + \sum_{(i,j)\in I\times J} \left(l_{ij} - \sum_{k\in K} \lambda_{ijk} \right) t_{ij} + \sum_{(i,k)\in I\times K} \tilde{c}(\lambda)_{ik} w_{ik}$$

subject to

$$\begin{aligned}
\sum_{i\in I} w_{ik} &= 1, & \forall k \in K,\\
\sum_{i\in I} t_{ij} &\le 1, & \forall j \in J,\\
w_{ik} &\le y_i, & \forall (i,k) \in I\times K,\\
t_{ij} &\le y_i, & \forall (i,j) \in I\times J,\\
0 \le w_{ik} &\le 1, & \forall (i,k) \in I\times K,\\
0 \le t_{ij} &\le 1, & \forall (i,j) \in I\times J,\\
y_i &\in \{0,1\}, & \forall i \in I.
\end{aligned}$$

This alternative formulation is easily seen to be equivalent to the previous one by using the domination argument above. In particular, any solution to $S_c(\lambda)$ can be converted into a solution to $S(\lambda)$, by keeping track of which c_{ijk} corresponds to each $\tilde{c}(\lambda)_{ik}$, for each pair of depot $i \in I$ and customer $k \in K$; subgradients for the Lagrangian subproblem can thus be extracted from optimal solutions to the compact formulation.

The compact Lagrangian subproblem is equivalent to the UFLP (with links only between facilities and customers). Any UFLP instance can be cast as an instance of the compact subproblem, simply by mapping facilities to depots, customers to customers and by letting the set of links between depots and satellites be empty. The compact Lagrangian subproblem itself is also easily reduced to the UFLP, by mapping depots to facilities, and satellites and customers to customers. The assignment inequalities, $\sum_{i\in I} t_{ij} \le 1, \forall j \in J$, can be turned into strict equalities by allowing every satellite to be linked to an artificial depot with zero costs. The Lagrangian subproblem, like the UFLP, therefore does not have the integrality property.

Computational experiments presented in Section 5 are based on implementations in which the compact Lagrangian subproblem is solved directly with a state-of-the-art MIP solver. Robust specialized UFLP solvers already exist (Barahona 2005, Hansen et al. 2007, Korkel 1989, Posta et al. 2012) and it might be fruitful to exploit them when tackling huge instances of the TUFLP-S.

3.2 Solving the Lagrangian Dual

The Lagrangian dual can be formulated as

$$\max_{\lambda\ge 0} v(S_c(\lambda)).$$

We solve it with a two-step process well-suited to being embedded into branch-and-bound methods. The first step computes a starting point for the Lagrangian dual, by solving the LP of the original (non-dualized) model; the second step improves the lower bound with a few iterations of the bundle method (Frangioni 1996). This approach minimizes the computational effort dedicated to improving Lagrangian multipliers that are very far from the optimum, and always yields lower bounds that are at least as strong as the LP relaxation of TUFLP-S. An alternative to the first step is to solve the Lagrangian dual with a bundle method and a continuous Lagrangian subproblem. On large instances, this can be quicker than solving the LP relaxation of TUFLP-S with a state-of-the-art simplex method. A primal optimal fractional solution can even be computed (Barahona 2000, Frangioni 2005) while solving the Lagrangian dual. However, in a branch-and-bound method, the efficiency of reoptimization is a key element, and simplex-type methods are then preferable. Rather than warm-starting the bundle method, our implementation solves the LP relaxation of TUFLP-S directly, as a linear program, and exploits the simplex reoptimization capabilities; initial Lagrangian multipliers are read as the dual values corresponding to the relaxed constraints.

The compact integer Lagrangian subproblem $S_c(\lambda)$ corresponding to this initial set of Lagrangian multipliers is then solved. Lagrangian duality, along with the fact that $S_c(\lambda)$ is solved as an integer subproblem, ensures that the resulting bound will always be at least as strong as the LP bound. Regardless of the bound value, integer solutions to $S_c(\lambda)$ are also useful to guide the primal heuristic.

The second step uses an implementation of the bundle method (Frangioni 1996) to compute Lagrange multipliers that will increase the lower bound obtained with the integer Lagrangian subproblem. The bundle method is a non-differentiable optimization method with stronger convergence properties than simpler approaches like subgradient methods; its main disadvantage is the increased computational requirement at each iteration. This trade-off seems particularly attractive given the complexity of each subproblem (an integer program) and the quality of the initial multipliers. Our computational experiments, reported in Section 5, show that this implementation of the bundle method, with default settings, leads to tighter lower bounds. Unfortunately, the method must perform several iterations before its model of the Lagrangian dual is accurate enough to yield sizable improvements in bound quality. In the branch-and-bound method described in the next section, the Lagrangian subproblem (with integer variables) is only evaluated once, with multipliers extracted from the LP relaxation. The Lagrangian dual is thus optimized with a fast heuristic.

3.3 Primal Heuristic

The primal heuristic is a MIP-based LNS approach that alternates between two large neighborhoods until no progress is observed between two consecutive iterations. Each neighborhood is explored exactly, through the solution of UFLPs.

The first neighborhood is defined by the set of solutions that preserve a given set of open depots: satellites may be opened, closed and reconnected to open depots, and each customer assigned to any open satellite. Let $I^* \subset I$ be a set of open depots; the first neighborhood, corresponding to solutions in which the depots in I^* are open and those in $I \setminus I^*$ are closed, is explored by solving the following MIP formulation:

$$\min \sum_{i \in I^*} f_i + \sum_{(i,j) \in I^* \times J} l_{ij} t_{ij} + \sum_{(i,j,k) \in I^* \times J \times K} c_{ijk} x_{ijk}$$

subject to

$$\sum_{(i,j) \in I^* \times J} x_{ijk} = 1, \qquad \forall k \in K, \tag{10}$$

$$\sum_{i \in I^*} t_{ij} \leq 1, \qquad \forall j \in J, \tag{11}$$

$$x_{ijk} \leq t_{ij}, \qquad \forall (i,j,k) \in I^* \times J \times K, \tag{12}$$

$$0 \leq x_{ijk} \leq 1, \qquad \forall (i,j,k) \in I^* \times J \times K,$$

$$t_{ij} \in \{0,1\}, \qquad \forall (i,j) \in I^* \times J.$$

This formulation corresponds to a UFLP in which the set of open facilities is subject to side conditions, the generalized upper bound (GUB) constraints (11). These side constraints can be expressed in a UFLP with the addition, for each $j \in J$, of an artificial customer k_j. The cost for linking k_j to any $t_{ij}, i \in I^*$ is very negative ($-M$), and zero for all other facilities, and the location costs l_{ij} are all increased by M. Thus, it is only profitable to open a location (i,j) if it can be linked to k_j; in that case, the M values sum to zero, and the objective function is not affected. Moreover, constraints (10) mean that each k_j is linked to exactly one t_{ij}, and thus that at most one t_{ij} is set to one, for any $j \in J$.

The second neighborhood is defined by the set of solutions that preserve a given assignment of customers to satellites: depots can be closed or open, and satellites reconnected to different open depots. For each satellite j, let $K^*(j)$ be the set of customers linked to that satellite in a given solution. The second neighborhood for that solution may be directly explored as a UFLP, in which facilities correspond to depots and customers to satellites (for which $K^*(j) \neq \emptyset$). Let $J^* \subset J$ be the set of depots for which $K^*(j) \neq \emptyset$, and let

$$d_{ij} = l_{ij} + \sum_{k \in K^*(j)} c_{ijk}, \qquad \forall (i,j) \in I \times J^*.$$

We then have the following MIP model for the second neighborhood:

$$\min \sum_{i \in I} f_i y_i + \sum_{(i,j) \in I \times J^*} d_{ij} t_{ij}$$

subject to

$$\begin{aligned}
\sum_{i \in I} t_{ij} &= 1, && \forall j \in J^*, \\
t_{ij} &\leq y_i, && \forall (i,j) \in I \times J^*, \\
0 \leq t_{ij} &\leq 1, && \forall (i,j) \in I \times J^*, \\
y_i &\in \{0,1\}, && \forall i \in I.
\end{aligned}$$

The primal heuristic must be initialized with a feasible solution. Rather than constructing one, it is possible to repair solutions from the integer Lagrangian subproblem $S_c(\lambda)$. The first neighborhood only requires a set of open depots; such a set can be extracted from any solution to $S_c(\lambda)$.

Tight lower and upper bounds are thus obtained in three steps:

1. Solve the LP relaxation of TUFLP-S;
2. Solve $S_c(\lambda)$, with Lagrange multipliers λ extracted from the previous step;
3. Perform the primal heuristic, starting with the set of open depots in an optimal solution to $S_c(\lambda)$.

4 A Specialized Branch-and-Bound Method

The bounding procedure described in the previous section depends, in part, on the full solution of the LP relaxation. It could be directly embedded within a standard branch-and-bound method, to only improve bound values. We found it more efficient to augment an LP-based branch-and-bound with the specialized lower and upper bounds, and with branching schemes guided by complementary slackness violations in solutions to the Lagrangian subproblem $S_c(\lambda)$.

The method follows LP-based branch-and-bound approaches: variables are fixed according to reduced costs, nodes are explored in a best-first order (but with respect to Lagrangian lower bounds), and the primal heuristic is executed at each node that is not fathomed. Evaluating each node is computationally heavy, and we attempt to minimize the size of the search tree by using reliability branching (Achterberg et al. 2005) to determine which variable each node should branch on.

Reliability branching generalizes pseudocost branching and combines it with strong branching. As usual, variables are considered for branching when they take fractional values in the LP relaxation, but history-based pseudocosts are used only when they are based on sufficiently many evaluations (more than the reliability parameter $\eta \in \mathbb{N}$). Remaining candidate branches (for which too little history is available) are ordered according to their potentially unreliable pseudocosts, and partially evaluated with a small number of dual simplex iterations γ. This is repeated until the incumbent branching choice has not changed for $\lambda \in \mathbb{N}$ (the lookahead factor) iterations. The chosen branching choice is then processed as usual, i.e., by evaluating the children completely and adjoining them to the search queue.

We implemented this method, with $\eta = 8$, $\lambda = 4$, as suggested in (Achterberg et al. 2005), and $\gamma = 200$ (rather than adaptively). Once a branch is chosen, children are fully evaluated by performing the simplex until convergence and then evaluating the Lagrangian integer subproblem once; children are then adjoined to the best-first search queue. Finally, solutions to the Lagrangian subproblems are used to initialize the primal heuristic. To make sure a diverse range of initial solutions are provided to the primal heuristic, any feasible solution corresponding to a neighborhood that has already been explored is rejected when the MIP corresponding to each Lagrangian subproblem $S_c(\lambda)$ is solved.

We implemented two branching schemes that exploit the GUB constraints (9): the GUB branching scheme and the polytomic branching scheme.

The GUB scheme branches on sums of variables $\sum_{i \in I} t_{ij}$, for some satellite $j \in J$, forcing them to be equal to 0 or 1, i.e., converting GUB constraints (9) to equalities. The LP relaxation is exploited to consider only those $j \in J$ for which the sum is fractional. Note that these branches do not correspond directly to the reliability branching framework described above; instead, average improvements are computed for each satellite $j \in J$ and for each right-hand side (whether the equality is fixed to 0 or 1).

The polytomic scheme branches on multiple variables at once. It can be used alone, or in conjunction with the GUB branching scheme; it is then also applicable to GUB constraints that were converted to strict equalities by earlier branching steps. Rather than turning GUB constraints into either of two equalities, a child is spawned for each variable in the sum, fixing that variable to one (and the others to zero), and one more in which the sum is set to zero, when the constraint is an inequality. Again, the LP relaxation is exploited to consider only those $j \in J$ such that at least one t_{ij}, for some $i \in I$, is fractional. Reliability branching was adapted to the polytomic branches obtained with this branching scheme by separately tracking increases in bound value for each variable when it is set to one, and when they are all set to zero (equivalently, when the GUB constraint for satellite j becomes an equality with 0 on the right-hand side).

The polytomic branching scheme, used alone, ensures the convergence of the branch-and-bound method. However, on many instances, the number of nodes evaluated and the solution time are significantly reduced by initially branching with the GUB scheme and resorting to the polytomic scheme only when necessary, i.e., the node is not fathomed and all the $\sum_{i \in I} t_{ij}$ are integral.

In both cases, scores for candidate branching choices are aggregated by taking the geometric average of the estimated bound increase for all the children (with a small minimum value $\epsilon > 0$). Additionally, while history-based estimates seem accurate for relative ordering purposes, they provide a poor basis for comparison with actually evaluated decisions; the initial incumbent branching choice, if any, is partially evaluated, like choices with too little history.

Finally, the branch-and-bound method explores relatively few nodes, and the instances tend to be large enough that even a single full strong branching step seems unreasonable. Rather than attempting to initialize pseudocosts, branching choices with no historic data are sorted with respect to complementary slackness violations in the Lagrangian subproblem. For each satellite j,

$$\sum_{(i,k) \in I \times K} \lambda_{ijk} |x_{ijk} - t_{ij}|$$

is computed, and satellites corresponding to larger values are considered first; search nodes for both branching schemes are defined by the satellite to which they relate. In preliminary experiments, this heuristic proved to reduce the number of nodes compared to summing reduced costs, even on instances for which the integer Lagrangian subproblem does not improve the LP bound value.

5 Computational Experiments

We compare the strength of the formulations and bounding methods by reporting gaps at the root node. We also compare the performance of our branch-and-bound method with that of the state-of-the-art MIP solver CPLEX, by reporting the runtime and number of search nodes until optimality is proven (within 0.1%). The tests were performed on three sets of instances.

All the computations were performed in single-threaded mode on a 2.8Ghz Intel X5660 with 24 GB RAM (Hyperthreading and TurboBoost were disabled), the solvers were compiled with G++ 4.6, with optimizations, and CPLEX 12.1 was used to solve LP and MIP models. The only exceptions are values copied from (Landete and Marín 2009), which were computed on an older platform: 2.6 GHz AMD Opteron CPUs with 4 GB RAM and CPLEX 9.1.

The instances in the first set were generated as subproblems in a Lagrangian relaxation used to solve the industrial location problem described in (Gendron and Semet 2009). These instances cannot be solved as TUFLP-C, and are relatively large, but exhibit low gaps at the root node; they highlight the effectiveness of the Lagrangian heuristic to compute tight upper and lower bounds, and the significant reduction in search nodes when using the specialized branch-and-bound method.

The second set contains Gap instances: artificial, small and difficult, TUFLP (without single-assignment) instances constructed from the single-level UFLP Gap instances (Kochetov and Ivanenko 2005). These instances can be formulated as both TUFLP-C and TUFLP-S, since their transportation costs are sums of per-arc costs. Even on such difficult instances, our specialized methods perform reasonably well, with runtimes in the same order of magnitude as CPLEX.

The instances in the third and final set are large-size TUFLP-S instances obtained with Gap instances generated with the procedures described in (Kochetov and Ivanenko 2005). The conversion procedure was also modified to force the explicit consideration of the single-assignment constraints. These large, difficult, instances allow us to show the improved scaling properties of the specialized branch-and-bound method with respect to instance size, when compared to CPLEX.

5.1 Industrial Instances

Our interest in the TUFLP-S stems from its appearance as a subproblem in a Lagrangian relaxation method for an application in freight transportation. This subsection reports performance values for 80 TUFLP-S instances derived from that industrial application, 20 on each of four networks: tiny, small, medium and full (Gendron and Semet 2009). Full instances comprise 93 depots, 320 satellites, and 701 customers. Medium instances are about 3/4 as large at each level, small ones 1/2 and tiny ones 1/4.

5.1.1 Bounds at the Root

The industrial instances exhibit a cost structure that cannot be expressed with formulation TUFLP-C. Thus, only formulations and relaxations derived from TUFLP-S are compared. From left to right, the columns in Table 1 report the average gaps (with respect to the optimal value) and CPU times for:

- "LP-S", the LP relaxation of the TUFLP-S formulation;
- "Lagrangian", the Lagrangian relaxation with a single execution of the integer Lagrangian subproblem following the solution of the LP relaxation;
- "Lagrangian/300", which improves the initial Lagrange multipliers with up to 300 iterations of the bundle method;
- "Heuristic", the primal Lagrangian heuristic.

The gaps on these instances seem representative of industrial location problems: across all instances, it is lower than 1%, even for the LP relaxation. Solving the Lagrangian subproblem once, with Lagrange multipliers extracted from the LP relaxation, suffices to roughly halve the gap at the root, and the bundle method further reduces the gap, but significantly increases the computational times.

Table 1: Runtimes and gaps at the root for industrial instances

Instances		LP-S	Lagrangian	Lagrangian/300	Heuristic
Tiny	Gap (%)	0.00	0.00	0.00	0.57
	Time (s)	0.03	0.06	0.15	0.01
Small	Gap (%)	0.36	0.28	0.18	0.59
	Time (s)	0.51	1.21	17.95	0.01
Medium	Gap (%)	0.13	0.08	0.04	0.20
	Time (s)	3.18	6.26	51.19	0.01
Full	Gap (%)	0.16	0.05	0.03	0.24
	Time (s)	19.75	32.51	649.91	0.01

The Lagrangian heuristic performs well on these instances, with solutions that are nearly always less than 1% away from the optimum. For these industrial instances, the Lagrangian heuristic produces good primal solutions with sufficiently tight lower bounds, so that branching could be avoided.

5.1.2 Performance of Enumerative Methods

Recall that industrial instances cannot be solved as TUFLP-C. Thus, three enumerative methods are compared: "MIP-S" corresponds to the TUFLP-S formulation solved with CPLEX, "Lag/GUB" to the specialized branch-and-bound combining the GUB and polytomic branching schemes, and "Lag" to the specialized branch-and-bound with only polytomic branching. Table 2 displays the CPU times and the average number of nodes for each enumerative method. The geometric average of their runtimes, relative to CPLEX, are also reported ($\frac{\text{Lag/GUB}}{\text{MIP}}$ and $\frac{\text{Lag}}{\text{MIP}}$). Five of the twenty instances in the "Small" class were not solved by Lag/GUB within two hours; the corresponding cells are marked with n/a.

For tiny and medium instances, the specialized branch-and-bound method with polytomic branching proves optimality in much fewer nodes than CPLEX on model "MIP-S". However, CPLEX nevertheless executes faster on average. Overall, it does not seem useful, for these instances, to enable the GUB branching strategy.

The small instances seem more difficult: the other specialized method with both branching schemes, "Lag/GUB", reaches the time limit on many instances. The specialized branch-and-bound with only polytomic branching, "Lag", always terminates in the allotted time, but is significantly slower than "MIP-S" on average, while generally exploring less nodes.

On the largest ("Full") instances, the specialized branch-and-bound method explores significantly fewer nodes than CPLEX on "MIP-S", in comparable time. In fact, it proves optimality faster than "MIP-S" for 9 of the 20 full instances; the average slowdown compared to "MIP-S" seems to be caused by the high variance of

Table 2: Runtimes and node counts for enumerative methods on industrial instances

Instances		MIP-S	Lag/GUB	Lag	$\frac{\text{Lag/GUB}}{\text{MIP}}$ (%)	$\frac{\text{Lag}}{\text{MIP}}$ (%)
Tiny	Time (s)	0.06	0.10	0.13	124.9	127.4
	Nodes	0.00	0.30	0.30	n/a*	n/a*
Small	Time (s)	4.37	n/a	118.67	n/a	321.9
	Nodes	52.55	n/a	26.80	n/a	70.7
Medium	Time (s)	27.83	784.83	94.27	210.6	149.5
	Nodes	35.70	35.55	1.60	80.1	39.7
Full	Time (s)	243.53	758.32	795.98	121.2	132.0
	Nodes	83.60	9.90	5.15	16.0	13.4

* all but a few instances are solved without any branching.

runtimes for the specialized branch-and-bound method.

5.2 Gap Instances

(Landete and Marín 2009) describe a simple procedure to construct TUFLP-C instances from small and hard UFLP instances (Kochetov and Ivanenko 2005). We used the same procedure, on the same input, to obtain the same set of 90 instances with 50 depots, 50 satellites and 50 customers. These instance have transportation costs that are sums of per-arc costs. Therefore, they can also be modeled as TUFLP-S. This allows us to compare the bounds obtained with our methods with those in (Landete and Marín 2009). The only difference is that we consider the UFLP instances as sparse graphs, while (Landete and Marín 2009) directly sums "big-M" costs. Slightly less than half the instances are rendered infeasible, leaving 20 instances derived from GapA, 12 from GapB and 23 from GapC.

5.2.1 Bounds at the Root

Table 3 reports the average gap (with respect to the optimal value) and CPU times at the root for various bounding methods on Gap A, B and C instances. These instances are derived from hard UFLP instances and are expected to exhibit huge integrality gaps on all practical formulations. In order, the columns are:

- "Landete", the formulation with specialized facet-defining constraints developed in (Landete and Marín 2009). The result tables in (Landete and Marín 2009) do not report solution times at the root, and we are unable to compare them with those reported here;

- "LP-C", the LP relaxation of the TUFLP-C formulation;

Table 3: Runtimes and gaps at the root for Gap instances

Instances		Landete	LP-C	LP-S	Lagrangian	Lagrangian/300	Heuristic
GapA	Gap (%)	10.52	11.39	10.84	10.84	8.09	9.82
	Time (s)		0.05	0.12	0.21	182.88	0.01
GapB	Gap (%)	7.63	8.12	7.92	7.91	5.46	5.10
	Time (s)		0.04	0.10	0.19	171.93	0.01
GapC	Gap (%)	12.03	13.22	12.58	12.58	9.58	9.15
	Time (s)		0.06	0.14	0.29	263.92	0.01

- "LP-S", the LP relaxation of the TUFLP-S formulation;
- "Lagrangian", the integer Lagrangian subproblem solved only once, with Lagrange multipliers extracted from TUFLP-S;
- "Lagrangian/300", the Lagrangian relaxation with up to 300 iterations of the bundle method;
- "Heuristic", the primal Lagrangian heuristic.

For each instance, "LP-S" provides a better bound than "LP-C", and "Lagrangian/300" a better bound than "LP-S". However, "Lagrangian" almost always obtains the exact same bound as "LP-S": for Gap instances, forcing the t_{ij} variables to take integer values only improves the lower bound when the Lagrangian multipliers are optimized with a bundle method. Overall, the valid inequalities introduced in (Landete and Marín 2009) improve "LP-C" and yield bounds that are comparable with (and generally better than) those derived with "LP-S"; they too are weaker than the bounds obtained with "Lagrangian/300".

The "LP-S" formulation takes more time to solve than the more compact "LP-C", by a factor of 2 to 3. Solving one Lagrangian subproblem adds reasonable overhead, roughly doubling the runtime compared to "LP-S", but rarely improves the bound value for Gap instances. Optimizing the Lagrange multipliers with up to 300 iterations of the bundle method improves the lower bound significantly, but requires too much CPU time to be practical.

The "Heuristic" column shows the optimality gap after one execution of the primal heuristic, from a solution to the Lagrangian relaxation. The execution of the primal heuristic took negligible time compared to the Lagrangian subproblem, but the average gaps are modest, between 5% to 10% depending on the instance sets. However, even on these hard instances for which the relaxations are poor approximations of the integer problem, the primal heuristic converged to near-optimal solutions (with gaps in the order of 0.1%) in one fifth of the instances.

Table 4: Runtimes and node counts for enumerative methods on Gap instances

Instances		Landete	MIP-C	MIP-S	Lag/GUB	Lag
GapA	Time (s)	101.00	1.75	5.74	9.10	17.03
	Nodes	92.68	154.08	385.54	18.15	29.27
GapB	Time (s)	74.17	1.22	3.03	5.93	10.22
	Nodes	96.08	123.92	210.58	13.17	16.83
GapC	Time (s)	153.11	3.13	19.29	16.52	45.93
	Nodes	191.47	238.48	1094.56	33.07	66.41

5.2.2 Performance of Enumerative Methods

In Table 4, columns “Landete”, “MIP-C” and “MIP-S” correspond, respectively, to the formulation with specialized valid inequalities in (Landete and Marín 2009), the TUFLP-C formulation and the TUFLP-S formulation, solved with CPLEX. Column “Lag/GUB” reports values for the specialized branch-and-bound method combining the two branching schemes, GUB and polytomic, while column “Lag” reports runtimes and node counts when only polytomic branching is used.

The difference in runtimes between our results and those of (Landete and Marín 2009) is so wide – a factor of 50 – that the averages are of limited usefulness. The main reason for the difference seems to be their use of “big-M” values, instead of a sparse formulation that implicitly filters forbidden paths. Improvements in CPU performance and in CPLEX, from version 9 to 12, are other possible explanations for such a huge difference in CPU times.

Formulation TUFLP-C (without single-assignment constraints) achieves the lowest runtimes. It seems likely that a more sophisticated implementation of the formulation with additional facet-defining inequalities described by (Landete and Marín 2009) would achieve slightly lower runtimes: the formulation size is similar, and the facet-defining inequalities help decrease the number of nodes explored. TUFLP-S comprises more variables and constraints than TUFLP-C for the same instance, and this is reflected in increases in CPU time and number of nodes.

The specialized branching schemes (“Lag” and “Lag/GUB”) significantly reduce the node count, compared to all the other formulations. This is even more marked for “Lag/GUB”, which exploits the structure of GUB constraints to convert such constraints to equalities, before executing polytomic branching. However, runtimes remain comparable to or longer than those for the MIP formulations.

5.3 Large Gap Instances

We reimplemented the generators in (Kochetov and Ivanenko 2005) to produce 10 UFLP instances each of the A, B and C classes, of size 150×150, and converted

Table 5: Runtimes and gaps at the root for large Gap instances

Instances		LP-S	Lagrangian	Lagrangian/300	Heuristic
LargeA	Gap (%)	16.62	16.62	15.57	12.25
	Time (s)	1.31	2.81	2645.40	0.01
LargeB	Gap (%)	20.14	20.14	19.73	17.66
	Time (s)	1.55	2.75	2766.17	0.01
LargeC	Gap (%)	18.38	18.38	17.90	8.54
	Time (s)	1.99	4.10	2819.93	0.01

them to TUFLP-S instances of size $75 \times 75 \times 75$ ("MediumA", "MediumB" and "MediumC"). We used the procedure described in (Landete and Marín 2009) with one difference: each transportation cost was incremented by $(7i+3j+k) \bmod 10$, where i is the rank of the depot, j that of the satellite and k that of the customer. These modifications are necessary to consider the single-assignment constraints explicitly.

5.3.1 Bounds at the Root

Table 5 reports the average gaps (with respect to the optimal values) and the average CPU times for the three classes of instances, and for the same methods as Table 1. It paints a picture very similar to that of Table 3. The Lagrangian subproblem does not improve bounds overly, but does not take too much additional time either, and the heuristic finds solutions with gaps that often exceed 10%.

5.3.2 Performance of Enumerative Methods

The TUFLP-C formulation is not equivalent to TUFLP-S on these instances, so we only consider "MIP-S" solved with CPLEX and "Lag/GUB", which was already shown to be preferable to "Lag" for instances in the Gap family. Indeed, "Lag", the Lagrangian branch-and-bound with only polytomic branching, failed to solve all but a few of the large Gap instances.

Table 6 reports average CPU times and node counts until completion when solving these large artificial instances (75x75x75) of the TUFLP-S, with a time limit of two hours. "MIP-S" failed to provably solve a few instances (one MediumA, two MediumB and two MediumC). The averages consider the runtimes for these instances as two hours and count the number of nodes explored until the time limit. On average, across all instances, the Lagrangian branch-and-bound method explores one fifth as many nodes and uses two thirds as much time as CPLEX on "MIP-S".

Table 6: Runtimes and node counts for enumerative methods on large Gap instances

Instances		MIP-S	Lag/GUB	$\frac{\text{Lag/GUB}}{\text{MIP}}$ (%)
LargeA	Time (s)	2996	1326	66.8
	Nodes	24982	701	19.7
LargeB	Time (s)	4631	2044	70.6
	Nodes	33764	972	21.4
LargeC	Time (s)	4419	1829	64.8
	Nodes	26027	693	19.5

6 Conclusion

We addressed the two-level uncapacitated facility location problem with single-assignment constraints (TUFLP-S), a problem that arises in industrial applications in freight transportation (Gendron and Semet 2009) and in telecommunications (Chardaire et al. 1999). The problem can also be used to model two-level uncapacitated facility location problems without single-assignment constraints, where transportation costs are sums of per-arc costs, and there are no assignment cost between depots and satellites (TUFLP-C). We show that the LP relaxation of TUFLP-S is stronger than the LP relaxation of the usual MIP model used in this case, at the expense of increasing the size of the formulation.

We presented a Lagrangian relaxation approach for which the Lagrangian subproblem reduces to a single-level uncapacitated facility location problem (UFLP). The Lagrangian dual is solved with a fast two-step heuristic; in the first step, the LP relaxation is solved, while the second step solves a single Lagrangian subproblem with Lagrange multipliers initialized with the LP-optimal dual solution.

We also developed a Lagrangian heuristic that includes a MIP-based LNS heuristic that solves a series of small UFLPs. The dual and primal bounds thus obtained were embedded within a specialized branch-and-bound method that implements two branching strategies: the GUB branching strategy and the polytomic branching strategy. The latter can be used alone or combined with the first strategy.

We presented and analyzed computational results on three sets of instances. On instances derived from a freight transportation application (Gendron and Semet 2009), the Lagrangian heuristic, without any branching, provides lower and upper bounds that are within 1% of optimality on average. On these instances, the Lagrangian lower bound improves the (already strong) LP bound. The specialized branch-and-bound method reduces significantly the number of nodes compared to CPLEX, but its CPU times are nevertheless higher. On difficult instances, the combined polytomic/GUB branching strategy performs well: compared to CPLEX on the TUFLP-S formulation, the number of nodes is significantly reduced, and the CPU times comparable, if not shorter. Since these instances can be cast as TUFLP-C, our experiments showed that a weaker, but smaller, formulation for the problem is solved more efficiently by CPLEX than both CPLEX on the TUFLP-S model and our specialized branch-and-bound method. Finally, on large, difficult, artificial instances that cannot be cast as TUFLP-C instances, our specialized branch-and-bound method outperforms CPLEX.

The performance of the specialized branch-and-bound method could be improved by adaptive parameter tuning and branching strategy choice, along with further refinements. In particular, it is likely that a specialized UFLP solver would be beneficial, as nearly half the runtime of the specialized branch-and-bound methods is used to solve such problems. Nevertheless, our specialized branch-and-bound method exhibits better scaling properties to large and difficult instances than CPLEX: it is competitive with CPLEX when solving large industrial instances, and markedly more efficient on large, difficult, artificial instances.

Acknowledgments

Financial support for this project was provided by the Natural Sciences and Engineering Research Council (NSERC) of Canada and by the Fonds de recherche du Québec en nature et technologies (FRQNT). This research was also partly supported by the International Campus on Safety and Intermodality in Transportation, the Nord-Pas-de-Calais Region, the European Community, the Regional Delegation for Research and Technology, the French Ministry of Higher Education and Research, and the National Center for Scientific Research. We thank Antonio Frangioni for providing his bundle code.

References

Aardal, K., M. Labbé, J.M.Y. Leung, M. Queyranne. 1996. On the two-level uncapacitated facility location problem. *INFORMS Journal on Computing* **8**(3) 289–301.

Achterberg, T., T. Koch, A. Martin. 2005. Branching rules revisited. *Operations Research Letters* **33**(1) 42–54.

Barahona, F. 2000. The volume algorithm: producing primal solutions with a subgradient method. *Mathematical Programming A* **87**(3) 385–399.

Barahona, F. 2005. Near-optimal solutions to large-scale facility location problems. *Discrete Optimization* **2**(1) 35–50.

Barros, A.I. 1995. Discrete and Fractional Programming Techniques for Location Models. Ph.D. thesis, Erasmus University, Rotterdam.

Barros, A.I., M. Labbé. 1994. A general model for the uncapacitated facility and depot location problem. *Location Science* **2**(3) 173–191.

Bumb, A. 2001. An approximation algorithm for the maximization version of the two level uncapacitated facility location problem. *Operations Research Letters* **29**(4) 155–161.

Chardaire, P., J.L. Lutton, A. Sutter. 1999. Upper and lower bounds for the two-level simple plant location problem. *Annals of Operations Research* **86**(0) 117–140.

Frangioni, A. 1996. Solving semidefinite quadratic problems within nonsmooth optimization algorithms. *Computers and Operations Research* **23**(11) 1099–1118.

Frangioni, A. 2005. About lagrangian methods in integer optimization. *Annals of Operations Research* **139**(1) 163–193.

Gao, L.L., E.P. Robinson Jr. 1992. A dual-based optimization procedure for the two-echelon uncapacitated facility location problem. *Naval Research Logistics* **39**(2) 191–212.

Gendron, B., F. Semet. 2009. Formulations and relaxations for a multi-echelon capacitated location-distribution problem. *Computers and Operations Research* **36**(5) 1335–1355.

Hansen, P., J. Brimberg, D. Urosevic, N. Mladenovic. 2007. Primal-dual variable neighborhood search for the simple plant-location problem. *INFORMS Journal on Computing* **19**(4) 552–564.

Kaufman, L., M.V. Eede, P. Hansen. 1977. A plant and warehouse location problem. *Operational Research Quarterly* **28**(3) 547–554.

Kochetov, Yuri, Dmitry Ivanenko. 2005. Computationally difficult instances for the uncapacitated facility location problem. *Metaheuristics: Progress as Real Problem Solvers*, *Operations Research/Computer Science Interfaces Series*, vol. 32. Springer US, 351–367.

Korkel, M. 1989. On the exact solution of large-scale simple plant location problems. *European Journal of Operational Research* **39**(2) 157–173.

Krarup, J., P.M. Pruzan. 1983. The simple plant location problem: Survey and synthesis. *European Journal of Operational Research* **12**(1) 36–81.

Landete, M., A. Marín. 2009. New facets for the two-stage uncapacitated facility location polytope. *Computational Optimization and Applications* **44**(3) 487–519.

Posta, M., J.A. Ferland, P. Michelon. 2012. An Exact Cooperative Method for the Simple Plant Location Problem. *Mathematical Programming Computation* Forthcoming.

Ro, H.-B., D.-W. Tcha. 1984. A branch and bound algorithm for the two-level uncapacitated facility location problem with some side constraints. *European Journal of Operational Research* **18**(3) 349–358.

Zhang, J. 2006. Approximating the two-level facility location problem via a quasi-greedy approach. *Mathematical Programming A* **108**(1) 159–176.