European Journal of Operational Research 210 (2011) 27–38

Contents lists available at ScienceDirect

European Journal of Operational Research

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

Discrete Optimization

The discrete facility location problem with balanced allocation of customers

Alfredo Marín *

Departamento de Estadística e Investigación Operativa, Facultad de Matemáticas, Universidad de Murcia, 30100 Murcia, Spain

ARTICLE INFO

Article history:
Received 4 January 2010
Accepted 9 October 2010
Available online 19 October 2010

Keywords:
Allocation
Facility location
Integer Programming
Branch-and-cut

ABSTRACT

We consider a discrete facility location problem where the difference between the maximum and minimum number of customers allocated to every plant has to be balanced. Two different Integer Programming formulations are built, and several families of valid inequalities for these formulations are developed. Preprocessing techniques which allow to reduce the size of the largest formulation, based on the upper bound obtained by means of an ad hoc heuristic solution, are also incorporated. Since the number of available valid inequalities for this formulation is exponential, a branch-and-cut algorithm is designed where the most violated inequalities are separated at every node of the branching tree. Both formulations, with and without the improvements, are tested in a computational framework in order to discriminate the most promising solution methods. Difficult instances with up to 50 potential plants and 100 customers, and largest easy instances, can be solved in one CPU hour.

© 2010 Elsevier B.V. All rights reserved.

1. Introduction

During the last two decades different models which capture more features of real location problems have been introduced in the literature. Among these features the issue of equity in cost or distance-to-travel distribution, i.e., equity with respect to the set of customers, has attracted considerable attention. Thus for instance Erkut [1] proposed a general framework for quantifying inequality and presented some axioms for the appropriateness of the inequality measures. Berman and Kaplan [2] addressed the equity question using taxes while Drezner et al. [3] dealt with the minimization of the Gini index when a facility is located. The interested reader can consult Marsh and Schilling [4] and Eiselt and Laporte [5], two reviews of the literature on equity measurement in Location Theory. General equitable objective functions are studied in the recent work of Marín et al. [6], where the formulation developed in Marín et al. [7] for the Discrete Ordered Median Problem is extended and improved. A different approach, in which a discrete facility location problem with a form of equity criterion called customers' envy is introduced, can be consulted in Espejo et al. [8]. Another related paper is Berger and Bechwati [9], where the authors maximize customer equity when organizing promotion budget allocation.

The papers cited above study models that look for equitable solutions with respect to the distance traveled by the customers. Just a handful of papers take into account equity from the point of view of the providers. One of these papers is Berman et al. [10]. These authors adopt the point of view of the providers, in the sense that an equitable solution is such that the amount of load assigned to each provider is balanced. Concretely, they consider the problem of locating p facilities to serve a set of customers with given demands, such that the maximum demand attracted to each facility is minimized. Another paper that deals with providers equity is Kalcsics et al. [11]. These authors consider several ordered location problems. Among them, the *ordered capacitated facility location problem from the supplier point of view*, where the supplier cost of a facility is the sum of the transportation costs of shipments from the facility to the customers, is studied. The objective is built sorting the supplier costs and multiplying them by a weights vector (plus a term associated to the setup costs). Adequately choosing these weights, different equitable objectives can be managed.

To the best of our knowledge, there is not any other equitable –with respect to the providers point of view – discrete location model in the literature. To partially fill this gap, this paper examines a discrete location problem that consists of establishing a fixed number of p plants to cover n demand points in such a way that every customer is allocated to its closest plant and the number of customers allocated to each plant is balanced. Instead of considering a *minimax* objective, we try to increase the balance among the suppliers using a *range* objective, i.e., minimizing the difference between the supplier with maximum number of assigned customers and the supplier with

* Tel.: +34 968 363627.
E-mail address: amarin@um.es

0377-2217/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.ejor.2010.10.012

minimum number of assigned customers. The reader should note that without the constraint of closest allocation the problem lacks interest since any set of p locations gives lots of optimal solutions.

Some applications of this model can be found in the field of *Territory Design* (see [12]), where small geographic areas must be grouped into larger geographic clusters according to some planning criteria (for instance, political districting, solid waste collection, school districting). Another application is location of antennas for mobile phones. In general, our model will be useful when absolute distances between customers and suppliers can be neglected but either customers will not accept other provider than the closest one – schools, voting stations – or customers must be allocated to its closest provider for technical reasons – the case of antennas –. That is to say, only relative distances are meaningful in the model. The reader may note that using ordinal ranking in location problems is not new and there is an important body of literature in competitive location that deals with this issue (see e.g. [13–16]).

The paper is organized as follows. In Section 2 we formally state the problem and give some notation. The problem is modeled as an Integer Programming Problem in Section 3, where two different formulations are introduced. Valid inequalities for both formulations are derived in Section 4. These inequalities, together with preprocessing strategies, formulations strengthening and heuristic solutions make up the improvement phase to which Section 5 is devoted. The resulting solution algorithms are detailed in Section 6 and tested in Section 7. Finally, some conclusions and future research lines are presented in Section 8.

2. Problem description

Let $A = \{1,\dots,n\}$ be a set used to represent *customers* and let $B = \{1,\dots,m\}$ be a set used to represent *potential plants*. Let $C = (c_{ij})_{i\in A, j\in B}$ be a matrix of *costs* (transportation costs, distances). We do not impose any additional constraint on the matrix C (costs can be negative, do not need to satisfy triangle inequality nor symmetry). A solution to the *Balanced Location Problem* (LOBA) is given by a set of plants $X \subset B$, $|X| = p$, such that each customer is allocated to its closest plant, i.e., given a solution X, we assume that each customer $i \in A$ will be supplied from a plant $j \in X$ such that $c_{ij} = \min_{k\in X}\{c_{ik}\}$. In case of tie, the customer can be assigned to any of his/her closest plants. The objective is to minimize the difference between the maximum and the minimum number of customers allocated to any plant in X.

Example 1. Consider $\{0,3,4,10\}$ four points in the real line corresponding with both the set of customers and the set of potential plants. Let c_{ij} be the Euclidean distance between customer i and potential plant j. Let $p = 2$ be the number of plants to be located.

In this example there are six possible choices for X, represented in the six lines of Table 1. Five of the six choices result in three customers allocated to one of the plants and one customer allocated to the other plant, giving an objective value of $3 - 1 = 2$. The remaining choice, $(2,3)$, gives the optimal value 0, since both plants receive two customers and the solution is perfectly balanced.

Example 1 gives an objective value of 0 because a solution exists where the same amount of customers is allocated to each plant. We give now another example where such a perfect allocation does not exist.

Example 2. Consider the case where $n = m = 3$ and the cost matrix is

$$C = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}.$$

Regardless the number of plants p, all customers will be allocated to the same plant, the one with the minimum index. If $p \geqslant 2$, one plant will receive all the customers and the rest of plants will not receive any customer. Therefore, the optimal value to LOBA will be 3.

Example 2 can be easily extended to any number of customers and potential plants to show that the optimal solution to LOBA can be extremely unbalanced.

3. Integer Programming formulations

In the following, we give two Integer Programming formulations for LOBA. This first one is called *standard* since it comes from the standard formulation for the p-median problem adding two variables (maximum and minimum number of customers allocated to any plant) and the closest allocation constraints. The second formulation is called *ordered* because it shares the main structure with some formulations recently developed for the Discrete Ordered Median Problem, among others (see [6]).

Table 1
Example of LOBA.

X	Allocation				# Allocated	Objective
(1,2)	1	2	2	2	(1,3)	2
(1,3)	1	3	3	3	(1,3)	2
(1,4)	1	1	1	4	(3,1)	2
(2,3)	2	2	3	3	(2,2)	0
(2,4)	2	2	2	4	(3,1)	2
(3,4)	3	3	3	4	(3,1)	2

3.1. Standard formulation

This formulation uses variables u and ℓ, the maximum and minimum number of customers allocated to any plant respectively, plus the two usual sets of location/allocation variables:

$$y_j = \begin{cases} 1 & \text{if a plant is located at } j, \\ 0 & \text{otherwise,} \end{cases} \quad \forall j \in B,$$

$$x_{ij} = \begin{cases} 1 & \text{if customer } \ i \text{ is allocated to plant } j, \\ 0 & \text{otherwise,} \end{cases} \quad \forall i \in A,\ j \in B.$$

The *standard* Integer Programming formulation for LOBA is the following:

$$(\text{SF}) \quad \min \quad u - \ell \tag{1}$$

$$\text{s.t.} \quad \sum_{j=1}^{m} y_j = p, \tag{2}$$

$$x_{ij} \leqslant y_j \quad \forall i \in A,\ j \in B, \tag{3}$$

$$\sum_{j=1}^{m} x_{ij} = 1 \quad \forall i \in A, \tag{4}$$

$$u \geqslant \sum_{i=1}^{n} x_{ij} \quad \forall j \in B, \tag{5}$$

$$\ell \leqslant \sum_{i=1}^{n} x_{ij} + n(1 - y_j) \quad \forall j \in B, \tag{6}$$

$$\sum_{a \in B} c_{ia} x_{ia} + (M_i - c_{ij}) y_j \leqslant M_i, \quad \forall i \in A,\ j \in B, \tag{7}$$

$$y_j \in \{0, 1\} \quad \forall j \in B, \tag{8}$$

$$x_{ij} \in \{0, 1\}, \quad \forall i \in A,\ j \in B, \tag{9}$$

where $M_i := \max_{j \in B}\{c_{ij}\}\ \forall i \in A$.

To ensure that p plants will be used and each customer will be allocated to some plant, the usual p-median problem constraints (2)–(4) are added to the formulation. Constraints (5) and (6), in conjunction with the objective function (1), give to u and ℓtheir desired values. In our problem, Constraints (2)–(6) do not guarantee allocation of each customer to its closest plant. To impose this condition, we add a new set of constraints (7) (an adaptation of the constraints used in Berman et al. [10]) to the formulation. By (7), each customer is allocated to the closest plant.

3.2. Ordered formulation

Another formulation is used in order to get in a different way the maximum and minimum numbers of customers allocated to any plant. Here $r := \lceil n/p \rceil$, $s := \lfloor n/p \rfloor$ and $t := r - s$.

This formulation uses five sets of variables. The two first sets are again the usual sets of location/allocation variables x and y. The third set of variables contains auxiliary binary variables defined as

$$w_{ij} = \begin{cases} 1 & \text{if the number of customers allocated} \\ & \text{to plant } \ j \text{ is at least } \ i, \\ 0 & \text{otherwise,} \end{cases} \quad \forall i \in A,\ j \in B.$$

The last two sets of variables contains auxiliary binary variables defined as

$$u_i = \begin{cases} 1 & \text{if the maximum number of customers} \\ & \text{allocated to any plant is at least } \ i, \\ 0 & \text{otherwise,} \end{cases} \quad \forall i = r+1, \ldots, n,$$

$$\ell_i = \begin{cases} 1 & \text{if the minimum number of customers} \\ & \text{allocated to any plant is less than } \ i, \\ 0 & \text{otherwise,} \end{cases} \quad \forall i = 1, \ldots, s.$$

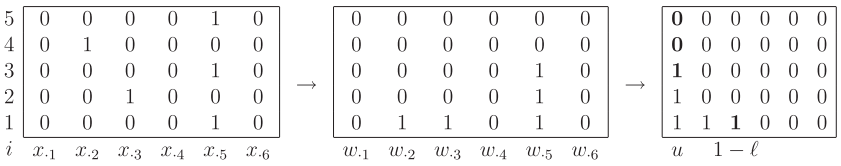

Fig. 1. Example of how formulation (OF) acts.

The *ordered* Integer Programming formulation for LOBA is the following:

$$\text{(OF)}\quad t + \min \quad \sum_{i=1}^{s} \ell_i + \sum_{i=r+1}^{n} u_i \tag{10}$$

$$\text{s.t.}\ \ (2), (3), (4), (7), (8),\ (9)$$

$$\sum_{i\in A} x_{ij} = \sum_{i\in A} w_{ij} \quad \forall j \in B, \tag{11}$$

$$w_{ij} \leqslant w_{i-1,j} \quad \forall i \geqslant 2,\ j \in B, \tag{12}$$

$$u_i \geqslant w_{ij} \quad \forall i \geqslant r+1,\ j \in B, \tag{13}$$

$$\sum_{j\in B} w_{ij} + p\ell_i \geqslant p \quad \forall i \leqslant s, \tag{14}$$

$$u_i \in \{0,1\} \quad \forall i \geqslant r+1, \tag{15}$$

$$\ell_i \in \{0,1\} \quad \forall i \leqslant s, \tag{16}$$

$$w_{ij} \in \{0,1\} \quad \forall i \in A, j \in B. \tag{17}$$

Constraints (2), (3), (4) and (7) guarantee correct allocation and number of open plants. To get the desired values for the w-variables from the values of the x-variables, constraints (11) and (12) are added.

For a given value of i, constraints (13) force u_i to take value 1 as soon as one w-variable in this row takes value 1. Regarding ℓ_i-variables, Constraints (14) force them to take value 1 if there are not p $w_{i\cdot}$-variables taking value 1. These constraints and the shape of the objective function force u_i and ℓ_i to adopt the desired values.

In Fig. 1 we show an example with $n = 5$ customers, $m = 6$ potential plants and $p = 3$. Therefore $r = 2$, $s = 1$ and variables ℓ_1, u_3, u_4 and u_5 are defined. Note that u_1 and u_2 are replaced by constant 2 since we know that at least two customers will be allocated to a common plant (there are 5 customers and 3 plants). Analogously, $\ell_2, \ldots, \ell_5$ are not defined (replaced by 0) because we know beforehand that there will be at least one plant with at most 1 allocated customer (if not, 3 plants with 2 allocated customers each would give more than 5 customers).

Assume now that allocation is done as shown in the x-matrix of Fig. 1. Then each column in the w-matrix will contain as many 1s as the same column in the x-matrix but the 1s will cumulate at the bottom rows. Then, to see how the values of the u- and ℓ-variables are obtained, we sort the rows of the w-matrix moving the 1s to the left hand side. Now, the first column in the third matrix contains the u-variables whereas the third column ($p = 3$) contains the values of $1 - \ell$.

4. Valid inequalities

In order to solve LOBA, we used several families of valid inequalities.

Consider the sets Q_{ij} of plants whose allocation cost is, with respect to customer i, less than the allocation cost of plant j, i.e.,

$$Q_{ij} := \{a : c_{ia} < c_{ij}\} \quad \forall i \in A,\ j \in B$$

and the constants

$$q_{ij} := |Q_{ij}| \quad \forall i \in A,\ j \in B.$$

Consider any $i \in A$, $j \in B$: $q_{ij} > p$. Then, the following is a valid inequality for LOBA:

$$p \sum_{a\notin Q_{ij}} x_{ia} + \sum_{a\in Q_{ij}} y_a \leqslant p. \tag{18}$$

If a plant exists whose allocation cost for i is less than c_{ij}, i is not allocated to any plant if the allocation cost is greater than or equal to c_{ij}. On the contrary, if $y_a = 0$ for all $a \in Q_{ij}$, then x_{ia} with $a \notin Q_{ij}$ can take any value. In any case, the left hand side of (18) is bounded by p.

In the case $1 \leqslant q_{ij} \leqslant p$, consider instead the inequality

$$q_{ij} \sum_{a\notin Q_{ij}} x_{ia} + \sum_{a\in Q_{ij}, a\neq\alpha(i,i',j)} x_{i'a} + \sum_{a\in Q_{ij}} y_a \leqslant q_{ij}, \tag{19}$$

where $i' \neq i$ and

$$\alpha(i,i',j) := \arg\min_{a\in Q_{ij}} \{c_{i'a}\}.$$

If x_{ia} takes value 1 for some $a \notin Q_{ij}$, all plants in Q_{ij} must be closed and then $y_a = x_{i'a} = 0$ for all $a \in Q_{ij}$. Furthermore, if x_{ia} takes value 0 for all $a \notin Q_{ij}$, in case the sum $\sum_{a\in Q_{ij}} y_a$ would take value q_{ij}, the allocation plant for customer i' would be either $\alpha(i,i',j)$ or a plant out of the set Q_{ij} and then $\sum_{a\in Q_{ij}, a\neq\alpha(i,i',j)} x_{i'a} = 0$.

Another set of valid inequalities we consider was given in Wagner and Falkson [17]. Let R_{ij} be the set of plants whose allocation cost is, with respect to customer i, greater than the allocation cost of plant j, i.e.,

$$R_{ij} := \{a : c_{ia} > c_{ij}\} \quad \forall i \in A,\ j \in B.$$

Then, inequalities

$$\sum_{a \in R_{ij}} x_{ia} + y_j \leqslant 1 \quad \forall i \in A,\ j \in B \tag{20}$$

are valid since once a plant j is opened, customer i cannot be allocated to plants with an associated cost greater than c_{ij}.

Valid inequalities (18)–(20) do not dominate one another and can be used to improve formulation (SF) as well as (OF). We now introduce a family of inequalities which can be used to improve formulation (OF) only:

$$\sum_{i=1}^{s} w_{ij} \geqslant \sum_{i \in S} x_{ij}, \quad \forall s \in A,\ S \subset A : |S| = s, \quad j \in B. \tag{21}$$

Inequalities (21) were used in [6]. Columns of the w-matrix contain as many 1s as their corresponding columns in the x-matrix, but the ones of the w-matrix are in the first positions of the columns. Consequently, if there are s ones in some part of the w-column, the sum of the s first values of the w-column is at least equal to s, leading to (21). Note that there is an exponential number of constraints in this family.

Step 0
Choose at random a set P of p plants from B.
For all $i \in A$ get $j_i := \arg\min\{c_{ij} :\ j \in P\}$.
For all $j \in P$ get $n_j := |\{i \in A :\ j_i = j\}|$.
Get $JM := \arg\max\{n_j :\ j \in P\}$, $MAX := n_{JM}$,
$jm := \arg\min\{n_j :\ j \in P\}$, $MIN := n_{jm}$ and
$z_{UB} := MAX - MIN$.
Repeat
Step 1. Repeat for all $j \notin P$:
Step 1.1. Set $P' := P \backslash \{JM\} \cup \{j\}$
Step 1.2. For all $i \in A$ get $j'_i := \arg\min\{c_{ij} :\ j \in P'\}$
Step 1.3. For all $j \in P'$ get $n'_j := |\{i \in A :\ j'_i = j\}|$
Step 1.4. Get $JM' := \arg\max\{n'_j :\ j \in P'\}$, $MAX' := n'_{JM'}$
$jm' := \arg\min\{n'_j :\ j \in P'\}$, $MIN' := n'_{jm'}$ and
$z := MAX' - MIN'$
Step 1.5. If $z < z_{UB}$, then replace P by P', z_{UB} by z
JM by JM' and jm by jm'
until no further improvement in Step 1.5 is obtained
Step 2. Repeat for all $j \notin P$:
Step 2.1. Set $P' := P \backslash \{jm\} \cup \{j\}$.
Step 2.2. For all $i \in A$ get $j'_i := \arg\min\{c_{ij} :\ j \in P'\}$
Step 2.3. For all $j \in P'$ get $n'_j := |\{i \in A :\ j'_i = j\}|$
Step 2.4. Get $JM' := \arg\max\{n'_j :\ j \in P'\}$, $MAX' := n'_{JM'}$
$jm' := \arg\min\{n'_j :\ j \in P'\}$, $MIN' := n'_{jm'}$ and
$z := MAX' - MIN'$
Step 2.5. If $z < z_{UB}$, then replace P by P', z_{UB} by z
JM by JM' and jm by jm',
until no further improvement in Step 2.5 is obtained
Step 3. Repeat for all $j_1, j_2 \notin P$, $j_1 \neq j_2$**:**
Step 3.1. Set $P' := P \backslash \{JM, jm\} \cup \{j_1, j_2\}$.
Step 3.2. For all $i \in A$ get $j'_i := \arg\min\{c_{ij} :\ j \in P'\}$
Step 3.3. For all $j \in P'$ get $n'_j := |\{i \in A :\ j'_i = j\}|$
Step 3.4. Get $JM' := \arg\max\{n'_j :\ j \in P'\}$, $MAX' := n'_{JM'}$
$jm' := \arg\min\{n'_j :\ j \in P'\}$, $MIN' := n'_{jm'}$ and
$z := MAX' - MIN'$
Step 3.5. If $z < z_{UB}$, then replace P by P', z_{UB} by z
JM by JM' and jm by jm',
until no further improvement in Step 3.5 is obtained
until no further improvement in Steps 1.5, 2.5 and 3.5 is obtained

Fig. 2. Heuristic solution algorithm.

Initialization

For all $i \in A$ sort plants $(j_1^i, \ldots, j_m^i)$ in such a way that $c_{ij_k^i} \leq c_{ij_{k+1}^i}$ $\forall 1 \leq k \leq m-1$.

For all $i \in A$ compute $M_i := c_{ij_{m+p-1}^i}$, the $(m-p+1)$-th allocation cost for customer i.

Obtain the heuristic solution.

Modify formulation (OF) by:

fixing w-, u- and x-variables to zero;

using the above M_i values.

Add constraints (18), (19) and (20) to formulation (OF) as indicated in Section (5.1).

At each node k of the branching tree repeat

Solve the linear relaxation (LP_k) of the subproblem (P_k) corresponding with the node.

Obtain the optimal solution $(x_k^*, y_k^*, w_k^*, u_k^*, \ell_k^*)$.

For all $j \in B$, $s = 1, \ldots, m$:

Look for $(S^*, s^*) = \arg\max\{\sum_{i \in S} x_{ij}^* - \sum_{i=1}^{s} w_{ij}^* : \quad s \in A, \quad S \subset A, \quad |S| = s\}$.

If $\sum_{i \in S^*} x_{ij}^* - \sum_{i=1}^{s^*} w_{ij}^* > 0$, then add constraint $\sum_{i=1}^{s^*} w_{ij} \geq \sum_{i \in S^*} x_{ij}$ to (P_k) and descendent nodes.

until

no further violated constraints are obtained or

$5n$ constraints have been added to (P_k).

Fig. 3. Branch-and-cut algorithm.

5. Improving formulations

In this section we will improve, from different points of view, both formulations.

5.1. Improving formulation (SF)

We have considered an improved version of formulation (SF). In this version we have added all valid inequalities in families (18) and (20). Regarding constraints (19), for every $i \in A$ and $j \in B$ we have randomly chosen an index $i' \neq i$ and added the corresponding constraint to the formulation.

We have also modified constraints (7). In the original version of these constraints, all coefficients M_i were indicated to be *large amounts M*. In the improved version we take into account that a customer will never be allocated to the $p-1$ (here we are ignoring possible ties) most costly plants and then M_i can be replaced by the cost associated to the $(m-p+1)$th allocation cost with respect to customer i. For the same reason, x-variables representing these allocations will be fixed to zero.

5.2. Improving formulation (OF)

We have also considered an improved version of formulation (OF), where the improvements of (SF) referred above have been also applied.

Furthermore, since (OF) is a large formulation with many useless variables, we start fixing some of them to zero (specifically some w- and u-variables). To this end, an upper bound on the optimal value of LOBA is needed. To get this upper bound, the interchange heuristic shown in Fig. 2 has been implemented. This heuristic is run $\min\{25, [n/2]\}$ times if $n < 100$ or 5 times if $n \geqslant 100$. The idea behind the heuristic is quite simple. A subset P of p plants is updated by choosing a plant in P with maximum (resp. minimum) number of allocated customers and a plant out of P which, in case of begin interchanged with the former, would reduce the objective function. In step 3, both couples are interchanged at the same time.

Now let z_{UB} be the upper bound got by this procedure, and consider a w-variable w_{ij} with $i > z_{UB} + \left\lfloor \frac{n - z_{UB}}{p} \right\rfloor$. Assuming $w_{ij} = 1$, it would follow from formulation (OF) and the given upper bound that, for any optimal solution, (i) $w_{aj} = 1$ $\forall a \leqslant i$, (ii) $w_{ab} = 1$ $\forall b \in X \backslash \{j\}$ (where X is the set of p plants in the solution) $\forall a \leqslant i - z_{UB}$. The total number of w-variables taking value 1 would be at least

$$i + (p-1)(i - z_{UB}) > z_{UB} + \left\lfloor \frac{n - z_{UB}}{p} \right\rfloor + (p-1)\left(z_{UB} + \left(\left\lfloor \frac{n - z_{UB}}{p} \right\rfloor\right) - z_{UB}\right) = z_{UB} + p\left\lfloor \frac{n - z_{UB}}{p} \right\rfloor \geqslant z_{UB} + n - z_{UB} = n.$$

Since

$$\sum_{i \in A} \sum_{j \in B} w_{ij} = \sum_{i \in A} \sum_{j \in B} x_{ij} = \sum_{i \in A} 1 = n,$$

we conclude that w_{ij} cannot take value one and therefore it can be fixed to zero.

As a consequence, variables u_i with $i > z_{UB} + \left\lfloor \frac{n - z_{UB}}{p} \right\rfloor$ can also be fixed to zero.

6. Branch-and-cut algorithm based on (OF)

Putting together heuristic, preprocessing phase, improved formulation (OF) and valid inequalities, we designed a branch-and-cut algorithm (B& C) which is summarized in Fig. 3. The details are given in the following.

Table 2
Standard formulation (SF) and ordered formulation (OF). Computational results.

Instance	OPT	SF		Improved SF		OF		B&C		
		Time	Nodes	Time	Nodes	Time	Nodes	Time	Cuts	Nodes
20-20-3-10	15	13	5300	5	889	39	20000	2	110	275
20-20-3-50	3	13	5100	4	557	8	5000	6	101	854
20-20-6-10	12	7	6600	4	723	11	8000	11	101	2000
20-20-6-50	6	24	13000	6	1100	12	8000	8	108	1200
30-30-3-10	25	159	22000	40	2900	77	80000	12	195	1300
30-30-3-30	22	153	22000	38	1900	513	95000	13	177	900
30-30-3-100	9	117	17000	36	2400	284	58000	147	165	10000
30-30-4-10	24	154	25000	74	6300	327	86000	93	157	9000
30-30-4-30	17	159	22000	53	3300	394	91000	14	157	1200
30-30-4-100	9	173	27000	74	4000	288	65000	11	152	767
30-50-3-10	45	375	30000	85	3000	1282	146000	25	288	850
30-50-3-50	34	197	14000	89	1800	919	127000	34	437	1200
30-50-3-100	27	179	14000	100	2500	891	124000	27	415	1000
30-50-3-200	13	215	16000	91	2600	571	80000	22	254	675
30-50-6-10	45	501	71000	129	8300	749	108000	34	280	1000
30-50-6-50	34	469	47000	193	5800	779	109000	46	265	1300
30-50-6-100	19	503	47000	170	3900	779	115000	34	275	1000
30-50-6-200	10	813	77000	340	8700	909	134000	39	283	1100
30-50-10-10	45	194	34000	104	3300	308	53000	21	275	483
30-50-10-50	27	273	41000	92	4300	549	85000	30	261	623
30-50-10-100	18	964	41000	415	1700	392	63000	33	265	657
30-50-10-200	5	1029	130000	211	5600	299	49000	383	253	13000
30-100-3-10	94	1800	56000	472	3400	**	**	113	774	2000
30-100-3-50	85	870	25000	435	2300	**	**	194	660	6000
30-100-3-100	75	1350	46000	482	2400	**	**	182	702	4000
30-100-3-200	63	1367	45000	494	5500	**	**	151	794	3000
30-100-6-10	92	2050	85000	883	12000	**	**	91	521	609
30-100-6-50	84	2130	55000	788	6000	**	**	144	516	1800
30-100-6-100	64	1010	36000	1234	9000	**	**	198	556	2000
30-100-6-200	48	2400	78000	3314	30000	**	**	174	570	1300
30-100-10-10	96	1100	96000	821	6000	1803	95000	111	547	571
30-100-10-50	77	622	36000	268	1600	1780	111000	94	511	557
30-100-10-100	63	1700	80000	190	3100	1805	90000	95	516	591
30-100-10-200	41	3500	119000	955	3400	1452	83000	151	564	900
50-50-3-10	46	2400	73000	631	9000	**	**	165	428	3000
50-50-3-50	39	**	**	698	8000	**	**	134	343	3000
50-50-3-100	32	**	**	1055	9000	**	**	206	328	4000
50-50-3-400	11	2576	83000	982	10000	**	**	211	334	3000
50-50-6-10	43	**	**	2813	29000	**	**	711	284	16000
50-50-6-50	33	**	**	969	8000	**	**	**	**	**
50-50-6-100	24	**	**	2647	15000	**	**	480	318	9000
50-50-6-400	3	**	**	**	**	**	**	560	296	7000
50-50-10-10	43	**	**	2894	49000	**	**	238	253	5000
50-50-10-50	30	**	**	**	**	**	**	203	252	3000
50-50-10-100	20	**	**	**	**	**	**	670	250	8000
50-50-10-400	2	**	**	**	**	**	**	1290	268	23000
50-100-3-10	96	**	**	**	**	**	**	950	506	9000
50-100-3-50	87	**	**	**	**	**	**	884	619	10000
50-100-3-100	82	**	**	**	**	**	**	876	512	10000
50-100-3-400	48	**	**	**	**	**	**	1007	1096	4000
50-100-6-10	95	**	**	**	**	**	**	**	**	**
50-100-6-50	83	**	**	**	**	**	**	**	**	**
50-100-6-100	70	**	**	**	**	**	**	**	**	**
50-100-6-400	35	**	**	**	**	**	**	**	**	**
50-100-10-10	96	**	**	**	**	**	**	1195	504	6000
50-100-10–50	83	**	**	**	**	**	**	635	500	2000
50-100-10-100	69	**	**	**	**	**	**	508	518	2000
50-100-10-400	18	**	**	**	**	**	**	**	**	**

In the initialization phase we compute tight values for the bounds M_i in constraints (7). M_i will be the $(m - p + 1)$th allocation cost of customer i as explained in Section 5.1. Then the heuristic solution as explained in Section 5.2 is obtained, the preprocessing phase is applied and valid inequalities in families (18)–(20) are included in (OF) as indicated above.

The difficulty with constraints (21) is the exponential number of them to be considered. Fortunately, they can be efficiently separated as follows. Given an optimal solution $(x_k^*, y_k^*, w_k^*, u_k^*, \ell_k^*)$ associated with any subproblem generated in the branching tree, for all $j \in B$ we sort the vector x_j^* and compare the sum of the s largest values of the vector with the sum of the s first w_j^*-values (for all $s = 1, \ldots, m$). If the first sum is greater than the second sum, a violated inequality in family (21) is violated and we add it to the corresponding subproblem. When a limit of $5n$ inequalities (in the whole branching tree) is reached, we turn off the separation routine.

The branching scheme was the default one used by the commercial solver.

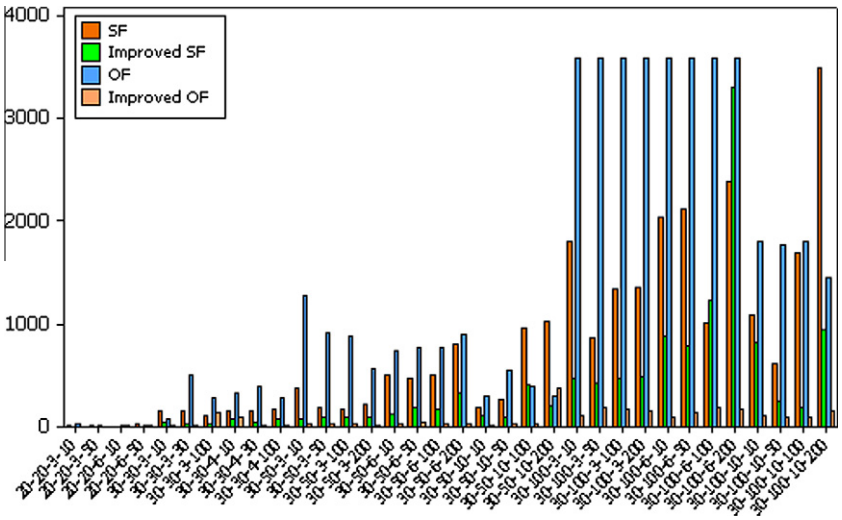

Fig. 4. Computational times for both formulations without preprocessing. Small instances.

7. Computational results

A computational experiment was carried out in order to check the performance of our solution methods. Since, to the best of our knowledge, this is the first paper in studying the LOBA, we cannot compare our results with others, nor using any data set from the literature specifically designed for our problem. Furthermore, we will show below that instances with allocation costs based on Euclidean distances between points use to have (almost) perfectly balanced solutions (with optimal values equal to 0 if $n = \dot{p}$ and equal to 1 if $n \neq \dot{p}$) and turn out to be easy to solve, like instances whose costs were totally generated at random. For these reasons we have generated our own instances for the experiment using a slightly more complicated method.

First we consider an *extreme* instance, similar to the one given in Example 2, where $c_{ij} = j \ \forall i \in A, j \in B$. Then we fix the parameter *pl*, the *perturbation level*, chose at random *pl* pairs in $A \times B$ (i_1, j_1) and (i_1, j_2), and swap $c_{i_1 j_1}$ and $c_{i_1 j_2}$. Low values of *pl* produce complicated instances with high optimal values whereas the larger *pl* is, the smaller optimal values are obtained. We denote an instance in this set by '*m*-*n*-*p*-*pl*'. For example, 30-50-3-10 means 30 potential plants, 50 customers, $p = 3$ and 10 pairs of allocation costs have been interchanged in the extreme instance.

Throughout all the testing, we used a Pentium IV with 2.5 GigaHertz and 3 GigaBytes of RAM running linux. The IP solver was FICO Xpress v3.0.0. It is worth to note that, for the sake of cleanly comparing the formulations and algorithms, all computational results (unless the contrary is said) were obtained without using the preprocessing and cutting algorithms used by default by Xpress. A limit of 1 hour was fixed for all the combinations of formulation/algorithm and instance.

Our first results correspond to formulation (SF) with and without improvement. Fifty-eight different instances were generated combining different values of m (20,30,50), n (20,30,50,100), p (3,4,6,10) and the perturbation level (10,30,50,100,200,400). In some cases p divides n and in other cases it does not. On the left hand side of Table 2 optimal values (**OPT**), computational times in seconds (**Time**) and (possibly rounded) number of nodes of the branching tree (**Nodes**) are given when formulation (SF) is used to solve the instances, with and without improvement. When the time limit of 1 h was reached, ** is shown. Small instances ($m \leqslant 30$) were solved by both formulations. The average time was 143 s for (SF) and 104 s for the improved version. The difference in the average number of nodes is of one order of magnitude, 43,647 reduced to 4764, showing the effectivity of the valid inequalities for this formulation. In the case of large instances ($m = 50$), (SF) could solve only two of them, whereas the improved version was able to solve eight within the time limit.

On the right hand side of Table 2, we solve the same instances using formulation (OF) without preprocessing. We show again optimal values (**OPT**), computational times in seconds (**Time**) and number of nodes of the branching tree (**Nodes**) for pure formulation (OF). It can be seen in the table that (OF) performs worse than (SF), needing larger times, giving larger number of nodes, and being unable to solve most of the instances with $n = 100$. In the same table we show the results for the B&C algorithm. Here we need to activate the separation procedure, since a subset of valid inequalities (21) are generated on the fly as seen in Section 6. The total number of valid inequalities added in any node of the branching tree is indicated in column **Cuts**. In the case of small instances the average time is 80.7 and the average number of nodes is 2168, much less than the figures obtained for the improved version of (SF). Moreover, most of the large instances (all but 7) could be solved in less than 20 minutes. A graphical comparison of the computational times of the small instances for (SF) and (OF), where the limit time (3600) is used in the case of unsolved instances, is drawn in Fig. 4. Note that in the rest of the figures, 'Improved OF' refers to the branch and cut algorithm based on formulation (OF).

We added the preprocessing phase to both pure (OF) and the B&C algorithm to generate the results shown in Table 3 where again ** is shown if a time limit of 1 h is reached. Here **HEUR** refers to the heuristic solution (**SOL** is the objective value) which, in most of the cases, obtained the optimal value in small time. The percentage of w-variables fixed to zero during the preprocessing phase is indicated in column '**%pre**'. The larger the perturbation level is, the larger percentage of variables can be fixed, ranging from about 5% to about 80%. Note that the preprocessing phase is the same for (OF) and the B&C algorithm. Still after running the preprocessing, the results for (OF) are not completely satisfactory. The running times are less than those obtained with the unpreprocessed version of the formulation, and some instances which could not be solved with (OF) can be solved after preprocessing, but some small instances remain unsolved and, in a portion of cases, the computational times are larger than those obtained using (SF). The B&C algorithm based on (OF) gathers the

Table 3
Preprocessed ordered formulation (OF). Computational results.

Instance	OPT	HEUR		OF		Both	B&C			XP_presolve		
		SOL	Time	Time	Nodes	% pre	Time	Cuts	Nodes	Time	Cuts	Nodes
20-20-3-10	15	15	0	16	10000	20	27	103	6000	0	0	1
20-20-3-50	3	3	1	4	3000	60	1	101	87	1	11	5
20-20-6-10	12	12	0	5	5000	35	19	100	4000	0	0	1
20-20-6-50	6	6	0	5	6000	60	17	100	5000	0	5	5
30-30-3-10	25	25	3	384	96000	13	1080	221	86000	2	54	25
30-30-3-30	22	22	3	364	70000	20	11	154	700	2	147	53
30-30-3-100	9	10	3	174	45000	47	1305	216	101000	737	154	178000
30-30-4-10	24	24	3	603	146000	17	7	161	591	0	0	1
30-30-4-30	17	17	3	310	80000	33	13	184	2000	6	150	1100
30-30-4-100	9	10	3	138	43000	50	320	177	33000	680	156	188000
30-50-3-10	45	45	4	1021	125000	8	24	252	605	2	0	5
30-50-3-50	34	24	6	927	129000	22	17	378	389	6	266	157
30-50-3-100	27	17	6	607	97000	32	19	453	401	11	255	231
30-50-3-200	13	13	5	577	85000	50	16	333	401	14	272	365
30-50-6-10	45	46	4	238	47000	8	25	261	765	1	1	1
30-50-6-50	34	34	5	445	76000	28	23	255	775	71	250	29000
30-50-6-100	19	19	4	530	102000	52	23	282	931	1	0	1
30-50-6-200	10	10	5	263	52000	66	23	295	695	6	250	257
30-50-10-10	45	46	3	29	16000	8	86	253	1200	1	0	0
30-50-10-50	27	27	4	86	14000	42	85	255	1300	1	10	9
30-50-10-100	18	20	3	69	18000	54	68	256	1100	313	253	128000
30-50-10-200	5	5	3	48	12000	82	57	262	1000	2	101	23
30-100-3-10	94	94	9	**	**	4	92	750	320	36	85	41
30-100-3-50	85	85	9	**	**	10	90	790	351	37	502	187
30-100-3-100	75	75	14	3343	160000	17	89	625	513	46	619	636
30-100-3-200	63	63	9	**	**	25	88	738	421	76	960	765
30-100-6-10	92	92	10	1710	430000	7	85	514	751	4	0	1
30-100-6-50	84	84	8	1188	84000	14	96	544	817	157	508	16000
30-100-6-100	64	64	8	1802	133000	30	160	540	1500	**	**	**
30-100-6-200	48	48	8	1372	113000	44	148	509	1500	21	569	141
30-100-10-10	96	96	5	717	60000	4	651	522	2000	4	0	1
30-100-10-50	77	77	5	541	30000	21	510	502	2000	4	5	1
30-100-10-100	63	63	8	430	26000	34	666	512	2000	4	37	4
30-100-10-200	41	41	8	440	36000	54	492	532	2000	16	517	399
50-50-3-10	46	46	34	**	**	6	232	516	3000	4	2	9
50-50-3-50	39	39	45	**	**	16	123	385	1700	44	331	841
50-50-3-100	32	32	35	**	**	24	158	405	2000	**	**	**
50-50-3-400	11	11	43	**	**	52	202	404	3000	94	352	2000
50–50-6-10	43	43	38	**	**	12	421	259	12000	2	0	1
50-50-6-50	33	33	37	**	**	30	264	280	7000	11	251	65
50–50-6-100	24	24	36	**	**	41	301	257	8000	**	**	**
50-50-6-400	3	3	37	**	**	80	194	262	4000	33	254	741
50-50-10-10	43	45	25	2356	247000	10	412	263	3000	7	6	14
50-50-10-50	30	30	30	2215	226000	36	383	253	3000	4	36	23
50-50-10-100	20	23	33	2754	273000	50	406	252	4000	642	251	113000
50-50-10-400	2	2	28	1562	171000	88	539	268	10000	44	261	1300
50-100-3-10	96	96	62	**	**	3	1085	585	4000	124	42	74
50-100-3-50	87	87	61	**	**	9	1107	633	4000	138	670	703
50-100-3-100	79	79	72	**	**	14	1613	553	4000	206	584	948
50-100-3-400	48	48	89	**	**	35	636	634	2000	353	624	1300
50-100-6-10	95	95	95	**	**	5	**	**	**	27	0	3
50-100-6-50	83	83	51	**	**	15	**	**	**	129	500	4000
50-100-6-100	70	76	73	**	**	30	**	**	**	93	524	557
50-100-6-400	35	35	80	**	**	55	1773	576	10000	245	616	2000
50-100-10-10	96	96	51	**	**	4	899	505	5000	16	0	3
50-100-10-50	83	83	51	**	**	16	672	504	3000	224	504	12000
50-100-10-100	69	72	53	**	**	26	1044	509	5000	43	175	22
50-100-10-400	18	18	71	**	**	74	3110	500	10000	147	516	1000

preprocessing and cutting phases. It can solve almost all the instances, most of them in less time than any other formulation. In a couple of cases (30-30-3-10 and 30-30-3-100) the results are amazingly bad. For example, Instance 30-30-3-10 can be solved in 12 seconds by B&C algorithm without preprocessing and needs more than 1000 s if we apply the preprocessing. Since the inner operation method of Xpress is unknown for us, we cannot satisfactorily explain this fact.

Once the empirical comparison of all methods has been carried out, we allowed Xpress to presolve the instances its way. In this manner, we can check the actual computational times needed to obtain optimal solutions using the best method (B&C with preprocessing). The last three columns of Table 3 show the times, number of cuts of type (21) and nodes of the branching tree needed when using this method. We can observe that, as expected, the computational times are largely reduced in most of – but not all – the cases. Three instances could not be solved in one hour by B&C, whereas they could be solved by XP-presolve in a reasonable time; however, B&C was finally able to close three

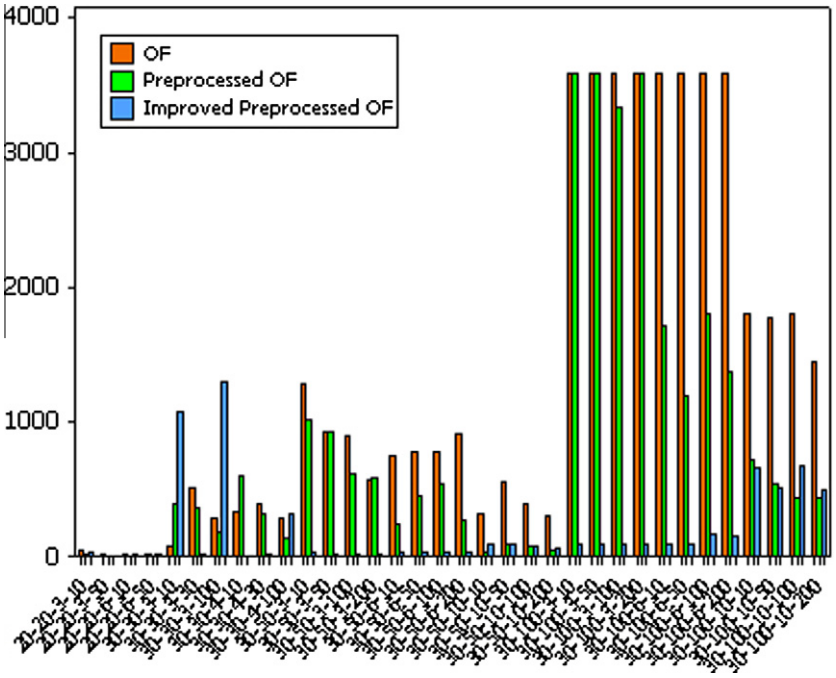

Fig. 5. Computational times for (OF) with and without preprocessing. Small instances.

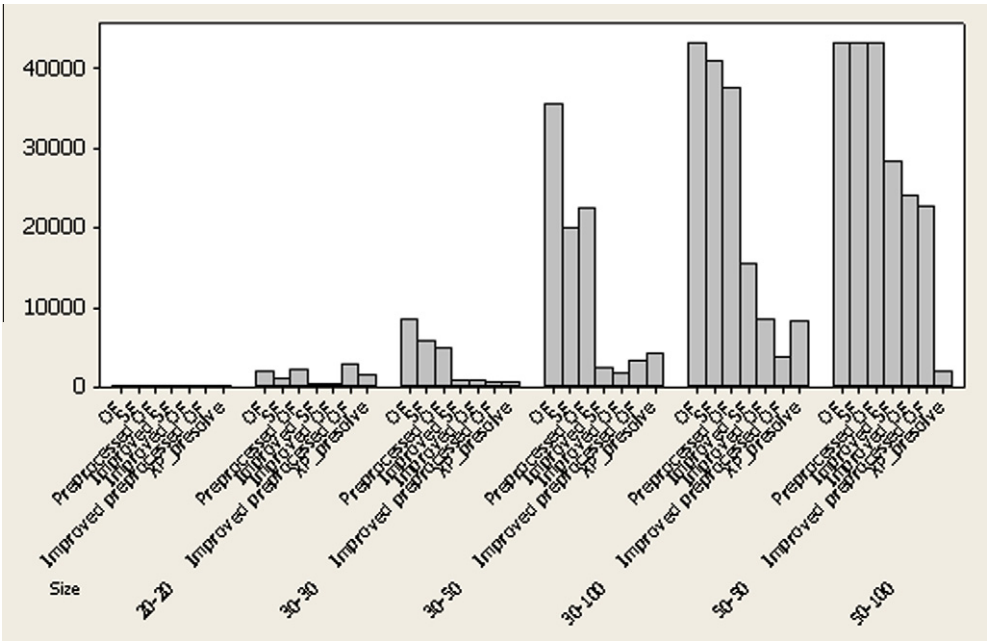

Fig. 6. All computational times gathered by size.

instances still unsolved by the previous methods. All solved instances took at most 13 minutes, and the number of nodes was also small in general. Less cuts of family (21) were needed – also in general – to solve the instances. A graphical comparison of the computational times of the small instances for all the variants of (OF) is drawn in Fig. 5.

We have also cumulated the computational times of each set of instances with the same size (m and n), associated with all the solution methods, and represented them in Fig. 6.

Finally, we adapted some instances from the literature and solved them with the B&C algorithm to show their ease. We used file *pmed10* from the well-known OR-Library (http://people.brunel.ac.uk/~mastjjb/jeb/info.html), containing the coordinates of 100 points in the plane, whose rounded Euclidean distances are used as allocation costs. From these 100 points we chose subsets of 20, 30, ..., 100 points which represent both customers and potential plants. Values of p of 3, 5 and 7 were used. In Table 4 we present the results. Here an instance with $n = m$ customers and potential plants and p plants to be opened is represented by '$n - p$'. The reader can check that all instances in this set

Table 4
Branch and cut algorithm, *p*-median instances.

Instance	OPT	HEUR		B&C			
		SOL	Time	Time	Nodes	% pre	Cuts
20-3	1	1	0	0	0	100	0
20-5	0	0	0	0	0	100	0
20-7	1	2	0	0	18	80	5
30-3	0	0	0	0	0	100	0
30-5	0	0	0	0	0	100	0
30-7	1	1	1	1	0	100	0
40-3	1	1	0	0	0	100	0
40-5	0	0	4	4	0	100	0
40-7	1	1	7	7	0	100	0
50-3	1	1	0	0	0	100	0
50-5	0	2	40	56	593	78	309
50-7	1	2	40	48	411	82	292
60-3	0	0	0	0	0	100	0
60-5	0	0	13	13	0	100	0
60-7	1	1	13	13	0	100	0
70-3	1	1	10	10	0	100	0
70-5	0	0	103	103	0	100	0
70-7	0	2	123	1005	10000	84	439
80-3	1	1	5	5	0	100	0
80-5	0	0	62	62	0	100	0
80-7	1	2	233	254	1151	84	400
90-3	0	0	1	1	0	100	0
90-5	0	2	303	1475	1450	79	500
90-7	1	1	39	39	0	100	0
100-3	1	1	9	9	0	100	0
100-5	0	2	111	921	613	79	852
100-7	1	3	141	1490	700	84	742

have optimal values 0 or 1 depending on if p divides n or not. The heuristic algorithm obtains almost always this solution and stops the calculations. Otherwise, most of the w-variables are fixed to 0 in the preprocessing phase and the number of nodes is very small when compared with the instances of similar sizes previously generated. In this manner, instances with up to 100 customers and potential plants could be solved in less than half an hour.

The conclusion of the computational study is that enforcement we considered to improve the proposed formulations – preprocessing, separation of valid inequalities – proved to be helpful in solving the problem, especially when applied to the ordered formulation (OF).

8. Conclusions

In this paper we have proposed a new discrete location problem where the number of customers allocated to every plant has to be balanced. We have formulated it in two ways, enforced the formulation, considered preprocessing techniques and valid inequalities for this problem, separating some of them within a branch-and-cut algorithm.

LOBA is a hard problem, with a highly combinatorial flavor, enforced by the fact that only relative values of the costs are needed to obtain the objective function. We were able to solve medium sized instances using the proposed approach, based on a non-standard location-oriented formulation.

As a matter of future research, better formulations could be investigated, in an attempt of optimally solve larger instances. More realistic situations, including different kinds of costs or other characteristics, could be also approached. Balanced network location problems ([18]) and balanced bundled location problems ([19]) are also being considered by the author.

Acknowledgments

This research has been partially supported by Spanish Ministry of Education and Science Grant No. MTM2009-14039-C06-04, RDEF funds and Fundación Séneca, Grant No. 08716/PI/08.

References

[1] E. Erkut, Inequality measures for location problems, Location Science 1 (1993) 199–217.
[2] O. Berman, E.H. Kaplan, Equity maximizing facility location schemes, Transportation Science 24 (1990) 137–144.
[3] T. Drezner, Z. Drezner, J. Guyse, Equitable service by a facility: Minimizing the Gini coefficient, Computers and Operations Research 36 (2009) 3240–3246.
[4] M.T. Marsh, D.A. Schilling, Equity measurement in facility location analysis: A review and framework, European Journal of Operations Research 74 (1994) 1–17.
[5] H.A. Eiselt, G. Laporte, Objectives in Location Problems, in: Z. Drezner (Ed.), Facility Location: A Survey of Applications and Methods, Springer-Verlag, 1995.
[6] A. Marín, S. Nickel, S. Velten, An extended covering model for flexible discrete and equity location problems, Mathematical Methods of Operations Research 71 (2010) 125–163.
[7] A. Marín, S. Nickel, J. Puerto, S. Velten, A flexible model and efficient solution strategies for discrete location problems, Discrete Applied Mathematics 157 (2009) 1128–1145.
[8] I. Espejo, A. Marín, J. Puerto, A. Rodríguez-Chía, A comparison of formulations and solution methods for the minimum-envy location problem, Computers and Operations Research 36 (2009) 1966–1981.
[9] P.D. Berger, N.N. Bechwati, The allocation of promotion budget to maximize customer equity, Omega 29 (2001) 49–61.
[10] O. Berman, Z. Drezner, A. Tamir, G.O. Wesolowsky, Optimal location with equitable loads, Annals of Operations Research 167 (2009) 307–325.
[11] J. Kalcsics, S. Nickel, P. Puerto, A. Rodríguez-Chía, The ordered capacitated facility location problem. Top 18 (2010) 203–222.

[12] J. Kalcsics, S. Nickel, M. Schröder, Towards a unified territory design approach – Applications, algorithms and GIS integration, Top 13 (2005) 1–56.
[13] P. Hansen, J.F. Thisse, Outcomes of voting and planning, Journal of Public Economics 16 (1981) 1–15.
[14] E. Carrizosa, E. Conde, M. Muñoz-Márquez, J. Puerto, Simpson points in planar problems with locational constraints. The polyhedral-gauge case, Mathematics of Operations Research 22 (1997) 291–300.
[15] S.L. Hakimi, Locations with spatial interactions: Competitive locations and games, in: P.B. Mirchandani, R.L. Francis (Eds.), Discrete Location Theory, Wiley, 1990.
[16] M. Labbé, Outcomes of voting and planning in single facility location problems, European Journal of Operations Research 20 (1985) 299–313.
[17] J.L. Wagner, L.M. Falkson, The optimal nodal location of public facilities with price-sensitive demand, Geographical Analysis 7 (1975) 69–83.
[18] I. Contreras, E. Fernández, A. Marín, Lagrangian bounds for the optimum communication spanning tree problem, Top 18 (2010) 140–157.
[19] A. Marín, Discrete location for bundled demand points, Top 18 (2010) 242–256.