Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 430243, 14 pages
http://dx.doi.org/10.1155/2014/430243

Research Article

Solving the Bilevel Facility Location Problem under Preferences by a Stackelberg-Evolutionary Algorithm

José-Fernando Camacho-Vallejo,[1] Álvaro Eduardo Cordero-Franco,[1] and Rosa G. González-Ramírez[2]

[1] *Facultad de Ciencias Físico-Matemáticas, Universidad Autónoma de Nuevo León, Avenida Universidad s/n, 66450 San Nicolás de los Garza, NL, Mexico*
[2] *Escuela de Ingeniería Industrial, Pontificia Universidad Católica de Valparaíso, Av. Brasil 2241, 2362807 Valparaiso, Chile*

Correspondence should be addressed to José-Fernando Camacho-Vallejo; jose.camachovl@uanl.edu.mx

Received 27 May 2013; Revised 30 October 2013; Accepted 26 December 2013; Published 20 February 2014

Academic Editor: Jianming Shi

This research highlights the use of game theory to solve the classical problem of the uncapacitated facility location optimization model with customer order preferences through a bilevel approach. The bilevel model provided herein consists of the classical facility location problem and an optimization of the customer preferences, which are the upper and lower level problems, respectively. Also, two reformulations of the bilevel model are presented, reducing it into a mixed-integer single-level problem. An evolutionary algorithm based on the equilibrium in a Stackelberg's game is proposed to solve the bilevel model. Numerical experimentation is performed in this study and the results are compared to benchmarks from the existing literature on the subject in order to emphasize the benefits of the proposed approach in terms of solution quality and estimation time.

1. Introduction and Literature Review

The facility location problem has the goal to determine the optimal sites to locate facilities such as plants, warehouses, and/or distribution centers. In addition, the assignment of customers being served by these facilities and how these facilities are connected with each other are interesting decisions considered within the problem. A seminal paper on this subject is the work of Weber [1] in which the problem is placing a single facility so total travel distance between the facility and a set of customers is minimized. Subsequent researches made by various authors who have worked on this problem and a wide variety of models and algorithms have been developed in the related literature within a broad range of applications. For example, [2–5] are comprehensive reviews of facility location problems (FLP), where the importance of the applications related to Supply Chain Management (SCM) and Logistics problems is remarked. Particularly, the problems that arise at the strategic long-run level involve finding the optimal sites in which facilities—such as plants, warehouses, or distribution centers—should be located, as well as the assignment of customers to be served by those facilities. Taxonomy of modern location models is provided by Daskin [6]. In [6], a classification of four groups is provided which are analytical models, continuous models, discrete models, and network models. In addition, several examples of each group, which can be found in the literature, are provided.

The uncapacitated facility location problem (UFLP) is one of the most studied problems in location theory, which assumes that the facilities to be installed are uncapacitated. The UFLP takes a great variety of forms, based on the nature of the objective function (e.g., minisum, minimax, or problems with covering constraints). Also, UFLP depends on the time horizon under consideration, static or dynamic, and the existence of hierarchical relationships between facilities. An additional aspect of modeling the UFLP is whether to use

deterministic or stochastic elements in the problem, Galvão [7].

The first UFLP model is the simplest plant location Problem proposed by Kuehn and Hamburger [8] and Balinski [9]. Subsequently, Hakimi [10, 11] introduces the p-median in this model. Next, Cornuéjols et al. [12] and Galvão and Raggi [13] propose the UFLP with a minisum objective function. This type of function is used for those cases, in which it is preferred to optimize the average performance of the system which differs from the case where the minimization of the worst case performance is aimed by using a minimax objective.

Specifically, in this paper the UFLP is considered to have a minisum objective function that incorporates preferences of the customers. Considering preferences of the customers with respect to the facilities that will serve them is an important issue due to increasing competition today, where customers demand better service levels according to their requirements and needs. The first problem that considers preference orderings of the customers was introduced by Hanjoul and Peeters [14], which extend the simple plant location problem (SPLP) to include preferences as the Simple Plant Location Problem with Order (SPLPO). Customer's orders are modeled as a set of constraints appended in the SPLP and an algorithm to solve it based on a greedy heuristic combined with a branch and bound procedure is provided. Next, Krarup and Pruzan [15] and Gorbachevskaya [16] present cases that can be solved in polynomial time, complexity results, and reductions of the uncapacitated facility location problem with user preferences (UFLPUP). Gorbachevskaya [16] also presents three reformulations of the problem.

Hansen et al. [17] introduce a new reformulation that dominates the three formulations of Gorbachevskaya [16] based on a bilevel formulation. Hansen et al. [17] propose a reduction for the minimization problem of pseudo-Boolean functions and obtain some lower bounds. In this paper, a different reformulation of the problem which reduces it to a single-level mixed integer problem is proposed. Furthermore, a novel algorithm to solve the bilevel problem based on a Stackelberg equilibrium scheme hybridized with an evolutionary algorithm is presented.

Cánovas et al. [18] consider the SPLPO introduced by Hanjoul and Peeters [14] and solve two LP relaxations: a LP relaxation for the integer problem and a strengthened LP relaxation. Secondly, relaxation was accomplished by applying some reductions and reformulations of the problem driven by the incorporation of valid inequalities to obtain a valid bound of the original problem. Contrary to the single-level model presented in Cánovas et al. [18], a bilevel formulation is presented in this research, as well as a solution methodology using a hybrid metaheuristic approach.

Ishii et al. [19] present a fuzzy modeling structure for the facility location problem with preferences for the potential sites. They propose a membership function to represent a satisfaction degree of the customer considering the distance from the customer to the facility site. They solve the model using a method that optimizes two criteria: finding the site that maximizes the minimal satisfaction degree among all demand points and maximizing the preference of the customer. The primary difference with respect to the research method proposed herein is the modeling structure. In the problem tackled in this paper preestablished preference values are considered, which differs from the member function associated with the preferences used by Ishii et al. [19].

Vasil'ev et al. [20] present new lower bounds for the UFLPUP introducing a family of new valid inequalities and show that the proposed formulation is stronger than previous formulations with respect to the linear relaxation and integrality gap. This new family of valid inequalities increases the number of constraints instead of the number of variables. Vasil'ev and Klimentova [21] consider the bilevel formulation of the UFLPUP and introduce some valid inequalities related to the preferences as a single-level integer linear programming problem. They solve, directly, a relaxation of the integer problem in order to find a lower bound and also propose a simulated annealing method to obtain upper bounds of the optimal solution used in exact methods. In addition, considering the lower and upper bounds previously found, they apply a branch and bound algorithm to solve the single-level problem. In contrast to these previous studies, a heuristic algorithm to solve the bilevel model proposed directly is presented in this research, while Vasil'ev and Klimentova [21] solved a reduction of the original problem.

Similarly, Marić et al. [22] implement three metaheuristic methods for solving the bilevel problem considered in this paper: a particle swarm optimization algorithm, a simulated annealing method, and a combination of reduced/variable neighborhood search method. In order to compare the effectiveness of their proposed algorithms, they test 28 different instances varying the size from 50 to 2000 customers and from 16 to 2000 facilities. They consider the single-level reformulation proposed in Hansen et al. [17] by using CPLEX 12.1 solver and conclude that only instances with 50 facilities and 50 customers can be optimally solved. In order to obtain the follower's best response, they rearrange the preferences matrix by sorting for each customer the index of the most preferable facilities and then search the first open facility for that specific customer.

There also have been some papers devoted to analyzing the bilevel p-median problem by considering the customers' preferences. For example, in Alekseeva and Kochetov [23] a genetic local search algorithm that provides near optimal solutions compared against an integer programming problem formulation is provided. For the experimentation they selected instances with a considerable integrality gap in order to evaluate the performance of their proposed heuristic. Also, Aksen et al. [24] supply three methods to solve the bilevel p-median problem associated with an attacker-defender problem. The methods they propose are an exhaustive algorithm, taboo search heuristic, and a sequential method considering the two levels as two separate problems. They do not address explicitly the UFLPUP and merely supply a method to solve the bilevel formulation of the UFLPUP.

Recently, J. M. Lee and Y. H. Lee [25] address a facility location problem with covering constraints and preferences

of the customers. In their work a mixed-integer programming (MIP) formulation of the problem based on customer restrictions and a heuristic procedure based on a Lagrangian Relaxation approach are provided. It is important to point out that in this paper covering constraints are not included.

The primary contribution of the proposed research herein is the consideration of the bilevel formulation of the UFLPU which is reduced to a classical MIP, which can be solved directly. Reformulations found in the literature for this problem are modeled as a single-level; the unique reformulation for the bilevel problem is made in Vasil'ev et al. [20] where they exploit some properties related to the uniqueness of the lower level optimal solution. The reformulation proposed herein treats the bilevel problem without assuming uniqueness in the lower level optimal solution. Also, a new heuristic procedure is presented for solving the bilevel problem. This heuristic procedure can be easily adapted for solving different bilevel problems.

Previous work in the related literature presents heuristics procedures for solving a single-level formulation, reformulations, or reductions of the problem. Only one of them proposes the use of metaheuristics to solve the bilevel uncapacitated facility location problem under preferences. For this reason, the proposed heuristic herein is compared against the existing optimal values presented in the literature. Numerical results show that the optimality gap is always less than 1% as it can be observed in the numerical results section. For the largest-size instances, provided that the optimal solution values are not known, the best value obtained on a long-run experiment is used as a reference for the comparison. Also, the single-level MIP reformulations are tested in order to identify the maximum size of the instances that can be solved in reasonable computational time.

The proposed heuristic procedure hybridizes a metaheuristic algorithm with a game theory approach, which is an interesting way to consider the bilevel programming problems. First, Yin [26] proposes a genetic algorithm for solving Stackelberg games modeled as bilevel optimization problems. He presents two examples of transportation problems in which his genetic algorithm is efficient. Moreover, the hybridization considered in the research herein was first applied for solving multiobjective optimization problems in aerodynamics. Wang and Periaux [27] introduce the Nash genetic algorithms (N-GAs) and the Stackelberg genetic algorithms (S-GAs) as efficient algorithms for finding the corresponding equilibrium. Also, they use those algorithms to solve a multiobjective problem in order to find the optimal flap/slat position looking for the maximization of the lift of a three-element airfoil system. Also, Periaux et al. [28] develop a Nash genetic algorithm for solving a multi-objective design optimization of internal aerodynamic shape operating with transonic flow.

The remainder of the paper is organized as follows. Section 2 presents the mathematical formulation and reformulation of the bilevel UFLPUP. Section 3 describes the solution approach. Section 4 presents numerical results and finally Section 5 presents conclusions, final remarks, and recommendations for further research.

2. Problem Description

In this section the problem statement and the mathematical formulation of the uncapacitated bilevel facility location problem are described. For this problem, the assumption that the customers establish a sorted list of their preferences to be served by the facilities is considered. In the upper level, a company (leader) aims to minimize its total cost which includes the cost of opening new facilities and the distribution costs considering the preferences of the customers (followers) to be served by a specific facility. In the lower level, the customers optimize their preferences with respect to the facilities from which they will be served. Preferences of the customers are defined based on a predefined sorted list. Without loss of generality, the following assumptions are considered.

(i) The customers set their preferences with respect to each facility; these preferences are defined as an ordered list from $1, \ldots, |I|$, where 1 corresponds to the most preferable facility and $|I|$ is the least one. Observe that the minimum value of the preferences with respect to the facilities located is desired.

(ii) No capacity constraints of the facilities are considered.

According to what was mentioned above, the following questions need to be addressed: where should a facility be located? and which customers will be assigned to each facility? The problem statement can be defined as determining the location of the facilities that minimize the total cost by considering the preferences of the customers. The details of the model are further described in Section 2.1.

2.1. Mathematical Formulation of the Bilevel Uncapacitated Facility Location Problem under Preferences (BUFLPUP). An extension of the UFLP, where the customers define a sorted list of their preferences to be served by the potential facilities, is considered. Preferences of the customers were introduced by Hanjoul and Peeters [14] and the bilevel model considered in this paper was first proposed by Vasil'ev et al. [20].

Let i denote the facilities and j the customers, where $i \in I$ and $j \in J$. The parameters c_{ij} represent the costs for supplying the whole customer's demand j from the facility i. Also, define f_i as the fixed cost for opening the facility i. Each customer j has a preference p_{ij} for being served by the facility i, where a value of $p_{ij} = 1$ corresponds to the most preferable site.

The decision variables of the bilevel problem are the binary variables x_{ij} which establish whether the facility i satisfies the whole demand of the customer j; and the binary variables y_i that represent if the facility i is open or not.

These parameters and variables lead to the following mathematical formulation of the bilevel facility location problem with preferences (BFLPP):

$$\min_{y,x} \quad \sum_{i\in I}\sum_{j\in J} c_{ij}x_{ij} + \sum_{i\in I} f_i y_i, \tag{1}$$

$$\text{subject to : } y_i \in \{0,1\} \quad \forall i \in I, \tag{2}$$

$$x \in \operatorname{Argmin} \sum_{i\in I}\sum_{j\in J} p_{ij}x_{ij}, \tag{3}$$

$$\text{subject to} : \sum_{i\in I} x_{ij} = 1 \quad \forall j \in J, \tag{4}$$

$$x_{ij} \le y_i \quad \forall i \in I,\ j \in J, \tag{5}$$

$$x_{ij} \in \{0,1\} \quad \forall i \in I,\ j \in J. \tag{6}$$

Equation (1) presents the upper level objective function, where the leader minimizes the total costs represented by the sum of the costs for satisfying the demand of the customer j by the facility i and the sum of the costs for opening the facility i. The upper level constraints (2) indicate whether a facility i will be open or not. In (3) the lower level objective function is described, which accounts for the sum of the preferences of customer j for being served by the facility i. Finally, for the lower level constraints, (4) indicates that a customer must be served by a single facility. Constraints (5) state that a customer must be served by facility i only if the facility is located. Finally, constraints (6) are the binary requirements.

Ausiello et al. [29] show that this problem is NP-Hard. In the case that the user's preferences at the lower level properly correspond to the leader's transportation costs, the BUFLPUP is equivalent to the uncapacitated facility location problem. Moreover, even if there are no fixed costs for locating facilities, that is, $f_i = 0$, for all $i \in I$, this case of the BUFLPUP is still NP-Hard in the strong sense. In this paper the consideration that the customers' preferences are different than the distribution costs is made, so this problem cannot be solved in polynomial time.

Difficulty of the BUFLPUP is the motivation to propose a reformulation of the bilevel problem and a reduction of it into a single-level by using the primal-dual optimality conditions. If the y variables are fixed, the lower level problem will be an assignment problem. So, in order to determine the primal-dual relations of the problem (1)–(6), binary constraints of (6) need to be replaced by a set of nonnegativity constraints $x_{ij} \ge 0$, for all $i \in I$, $j \in J$. It can be clearly seen that these constraints are closely related to (5) and (4), and the lower level problem is naturally integer. However, after obtaining those relations, the original constraints (6) are considered again in the reformulation. Hence, the lower level's linear programming problem can be replaced by its primal-dual optimality conditions. Let α_j, for all $j \in J$, and β_{ij}, for all $i \in I$, $j \in J$, be the dual variables associated with constraints (4)-(5), respectively.

The reformulated model, called single-level uncapacitated facility location problem under preferences (SUFLPUP), is as follows:

$$\min_{y,x,\alpha,\beta} \quad \sum_{i\in I}\sum_{j\in J} c_{ij}x_{ij} + \sum_{i\in I} f_i y_i, \tag{7}$$

$$\text{subject to} : y_i \in \{0,1\} \quad \forall i \in I, \tag{8}$$

$$\sum_{i\in I} x_{ij} = 1 \quad \forall j \in J, \tag{9}$$

$$x_{ij} \le y_i \quad \forall i \in I,\ j \in J, \tag{10}$$

$$\alpha_j + \beta_{ij} \le p_{ij} \quad \forall i \in I,\ j \in J, \tag{11}$$

$$\sum_{i\in I}\sum_{j\in J} p_{ij}x_{ij} = \sum_{j\in J}\alpha_j + \sum_{i\in I}\sum_{j\in J}\beta_{ij}y_i, \tag{12}$$

$$x_{ij} \in \{0,1\} \quad \forall i \in I,\ j \in J, \tag{13}$$

$$\beta_{ij} \le 0 \quad \forall i \in I,\ j \in J. \tag{14}$$

It can be observed that the problem described in (7)–(14) is obtained by considering the leader's variables y_i as parameters of the lower level linear problem. Constraints (8)–(10) and (13) ensure primal feasibility, constraints (11) and (14) ensure dual feasibility, and constraints (12) guarantee that the follower's objective function values corresponding to its primal and dual problems are equal. As a conclusion it can be established that the SUFLPUP is equivalent to the BUFLPUP. Although SUFLPUP is a single-level problem, it is difficult to solve it due to the quadratic term of constraint (12) that appears as a result of the fact that the variable y_i is a parameter for the lower level, but in this reformulation version it is considered as a variable. Also, the problem (7)–(14) adds $|I||J| + |J|$ variables and $|I||J| + 1$ constraints, where one of them is nonlinear.

In order to solve this problem, (12) needs to be analyzed with the aim of linearizing it. The variables $\pi_{ij} = \beta_{ij}y_i$ are introduced assuring that when $y_i = 0$, then $\pi_{ij} = 0$; and if $y_i = 1$, then $\pi_{ij} = \beta_{ij}$. This can be done by introducing the following inequalities:

$$\begin{aligned} \pi_{ij} &\le 0 \quad \forall i \in I,\ j \in J, \\ \pi_{ij} &\ge -My_i \quad \forall i \in I,\ j \in J, \\ \pi_{ij} &\ge \beta_{ij} \quad \forall i \in I,\ j \in J, \\ \pi_{ij} + My_i &\le \beta_{ij} + \mathrm{M} \quad \forall i \in I,\ j \in J, \end{aligned} \tag{15}$$

and replacing (12) by the following equation:

$$\sum_{i\in I}\sum_{j\in J} p_{ij}x_{ij} = \sum_{j\in J}\alpha_j + \sum_{i\in I}\sum_{j\in J}\pi_{ij}. \tag{16}$$

As a result of this, a mixed integer programming model referred to as first linearized single-level uncapacitated facility location problem under preferences (1-LSUFLPUP) is generated defined by (7)–(11) and (13)–(16) that is equivalent to the BUFLPUP. This reformulation has $4|I||J| + 1$ additional constraints and $2|I||J| + |J|$ additional variables than the reformulation proposed in Hansen et al. [17]. It is worthy to mention that 1-LSUFLPUP avoids the use of $|I||J|$ subsets of preferences as in the reformulation proposed by Hansen et al.

It is worthy to mention that based on the reduction into a single-level using the primal-dual relations as an alternative reformulation, the equality of the objective functions for both problems can be substituted, which is given by (12) and by the following complementarity constraints that force complementarity slackness:

$$\begin{aligned} x_{ij}\left(\alpha_j + \beta_{ij} - p_{ij}\right) &= 0 \quad \forall i \in I,\ j \in J, \\ \beta_{ij}\left(x_{ij} - y_i\right) &= 0 \quad \forall i \in I,\ j \in J. \end{aligned} \tag{17}$$

And (17) can be linearized with the following expressions:

$$\alpha_j + \beta_{ij} - p_{ij} \geq -M\left(1 - x_{ij}\right) \quad \forall i \in I,\ j \in J,$$
$$\beta_{ij} \geq -M\left(1 + \left(x_{ij} - y_i\right)\right) \quad \forall i \in I,\ j \in J. \tag{18}$$

Then, the resulting MIP model defined by (7)–(11), (13)-(14), and (18) would be referred to as the second linearized single-level uncapacitated facility location problem under preferences (2-LSUFLPUP). This reformulation has $3|I||J|$ additional constraints and $|I||J|+|J|$ additional variables than the model proposed in Hansen et al. and it is significantly smaller than 1-LSUFLPUP.

This model can include the p-median assumption that the leader has a fixed number of p facilities that should be located. Therefore, the constraint to be included in the upper level problem is

$$\sum_{i \in I} y_i \geq p. \tag{19}$$

It is clear that this does not affect the analysis of the problem previously described that corresponds to the lower level (the primal-dual relations or the complementarity slackness) and constraint (19) is just incorporated into the upper level. Then it is enough to solve the problem either the 1-LSUFLPUP or the 2-LSUFLPUP also considering constraint (19).

3. Methodology

In this section the methodology proposed in order to solve the BUFLPUP is described. The reformulations 1-LSUFLPUP and 2-LSUFLPUP are both MIP models that can be exactly solved in reasonable time for small-size instances. Due to the difficulty for solving the reformulations, a heuristic procedure to solve the bilevel formulation of the problem is proposed. The procedure is based on the principles of the Stackelberg equilibrium in an evolutionary scheme. A set of leader's solutions as parameters of the lower level are proposed to be used to solve this problem in order to obtain the optimal response; then the leader's objective function is evaluated and solutions obtained are improved. Before describing the details of the procedure proposed, the concept of the Stackelberg equilibrium and the evolutionary algorithms main characteristics are introduced, and then the proposed methodology is described.

3.1. Stackelberg Equilibrium. The Stackelberg equilibrium for noncollaborative game theory is proposed by Stackelberg [30] and is a well-known problem in game theory. Stackelberg's game is a complete information problem, where two firms, the leader and the follower, compete in a market looking for the maximization of their profits which depends on both decisions. One of the firms, the leader, moves first and then the follower makes its own decision after knowing the leader's decision.

The Stackelberg equilibrium is obtained after using follower's decision as a reaction of leader's decision. With this reaction function, the leader firm makes a decision trying to maximize its profit (which depends only on its own decision, because the decision of follower's firm is a reaction to the leader's one). Historically bilevel optimization is closely related to the Stackelberg problem (see [31]). In a discrete bilevel optimization problem, each possible leader's solution can also be taken by the follower and this player should choose its own strategy with the aim of optimizing its profit function. These decisions can be anticipated by the leader in order to choose the solution that maximizes its profit function taking the follower's reaction into account. It is evident that finding the Stackelberg equilibrium by an exhaustive procedure that analyzes all possible leaders' solution is out of consideration for medium- and large-size problems. Hence, an evolutionary algorithm is proposed to solve the problem. The evolutionary algorithm aims to create the reaction function in order to obtain leader's solution and then Stackelberg's equilibrium (bilevel optimal solution).

The Stackelberg game can be considered as a bilevel problem and the methodology of finding its equilibrium can be used for solving different bilevel problems. In the next section this methodology is employed to find the solution of the bilevel problem BLFPP by an evolutionary heuristic technique.

3.2. Evolutionary Algorithms. Evolutionary algorithms are a population-based metaheuristic procedure that simulates the behavior of living beings. These algorithms use some mechanisms inspired by the biological evolution such as reproduction, mutation, recombination, and selection. There are several variations of this type of algorithms that differ in the implementation approaches. The general framework is as follows: there is a set of individuals (solutions) which form a population for a determined generation, and then either two individuals are selected and combined in a crossover operation or each individual is mutated. These crossover and mutations are randomly performed based on a predefined factor that guarantees diversity of the individuals. Infeasible solutions that may be generated are discarded; this procedure is referred to as an abortion. Based on a selection criterion, the strongest individuals (those with the best value of a performance metric) survive and remain for the next generation. The process is repeated until some stopping conditions are fulfilled.

The main components of an evolutionary programming algorithm are as follows.

(i) *Representation.* It characterizes the individuals of a population.

(ii) *Evaluation Function.* It is the objective function (or performance metric) of the problem that is optimized.

(iii) *Population.* It is the set of individuals. The approach used to generate the sets of individuals and the corresponding size is also defined.

(iv) *Variation Operators.* They are the variables which define how the crossover will be performed among the individuals as well as the mutations.

(v) *Selection Mechanisms.* They are the mechanisms which specify those individuals that prevalence for

the next generation, considering that the selection process must guarantee diversity of the population.

3.3. Stackelberg-Evolutionary Algorithm. In this section the components of the Stackelberg-Evolutionary (S-E) algorithm proposed to solve the bilevel uncapacitated facility location problem under preferences are introduced. The procedure is further described.

First, the way that the individuals of a population are represented is defined. An individual y_k^t consists on the kth vector that indicates the facilities that the leader has decided to locate in generation t. The evaluation function is defined by the following equation:

$$\varphi_k^t\left(y_k^t, x\left(y_k^t\right)\right) = \sum_{i\in I}\sum_{j\in J} c_{ij}x_{ij}\left(\left[y_i\right]_k^t\right) + \sum_{i\in I} f_i\left[y_i\right]_k^t, \tag{20}$$

where $x(y)$ is the optimal solution found by the follower after the low level problem defined by (3)–(6) was solved, considering the values of y as an input parameter. By solving the lower level problem for each leader's decision, the aim is to obtain the reaction function to be used in order to obtain the Stackelberg equilibrium. Furthermore, define $\varphi_k^t(y_k^t, x(y_k^t))$ as the value of the objective function for the individual k of generation t, when the leader has made a decision over the variables y_k^t and the follower has optimized its decision obtaining the values of $x(y_k^t)$. This iterative methodology, where the lower level is solved in each leader's iteration, is very common in bilevel programming (see [32–36]). The methodology proposed is further described.

Initialization (t = 0). First, the size n of the initial population P^t is determined. Then, n feasible facility location vectors are randomly generated, y_k^t, $k = 1, 2, \ldots, n$, where each component of the vector has a value of 0 or 1 representing whether or not the facility is located. After that, for all y_k^t, the follower's objective function (the customer's preferences) is optimized obtaining the assignments $x(y_k^t)$. Finally, the fitness function is the leader's objective function $\varphi_k^t(y_k^t, x(y_k^t))$ which is calculated in order to minimize the locating and distribution costs.

Crossover. A standard crossover operator is implemented. First, a pair of solutions y_k^t and y_l^t and $k \neq l$ is randomly matched, that will be the parents. Then, a crossover point is randomly selected. This crossover point cannot be in the initial or final position of the chromosome. That is to say, if the crossover point is selected in one extreme of the chromosome, then the offspring will be exactly as one of their parents. So, an offspring that differs from its respective parents is generated. As a result of this crossover there will be two new offspring; the first one inherits the first characteristics from parent 1 and the second characteristics from parent 2 with respect to the crossover point; the second offspring will be analogously generated.

Mutation. For each vector y_k^t one evolutionary mechanism is randomly applied, considering three alternatives based on a random number ($0 < \text{rand} \leq 1$).

(i) If rand $\leq \sigma$, the number of facilities located is reduced. For a randomly selected located facility its value is set to zero.

(ii) If $\sigma < \text{rand} < 1 - \sigma$, the number of facilities located is maintained; this is to say the value of zero to one facility that is not currently located will be randomly set, as well as the value of one to zero to another currently located facility.

(iii) If rand $\geq 1 - \sigma$, the number of facilities located is increased. For this, a randomly selected closed facility turns its value from zero to one.

With the previous mechanisms, m individuals are created that will be referred to as P_{new}^t. These individuals are included in the initial population set, P^t, so as to define the augmented population set, $P_{\text{aug}}^t = \{P^t, P_{\text{new}}^t\}$, which is compounded by $n+m$ individuals. Then, for each new individual y_k^t, $k = n+1, n+2, \ldots, m$, the objective function of the follower is minimized to obtain the assignment values $x(y_k^t)$. Finally, the objective function of the leader is computed, $\varphi_k^t(y_k^t, x(y_k^t))$, $k = n+1, n+2, \ldots, m$.

Selection. For each vector y_k^t, $k = 1, 2, \ldots, n+m$, another y_l^t, $k \neq l$ is randomly selected, and then the values of the corresponding objective functions are compared by a tournament selection. If $\varphi_k^t \leq \varphi_l^t$, then it is established that φ_k^t has won the tournament and the victory is registered. The tournament is performed during a predetermined number of times (Max_Tourn), so as each individual $y_k^t \in P_{\text{aug}}^t$ will accumulate victories through the tournaments. Once all the Max_Tourn tournaments are performed, the n individuals associated with the highest number of victories will be part of the new population P_k^{t+1}. In this part, the crossover and mutation operators can produce a solution that already exists in the population. In order to avoid the duplication of individuals in the current population, an identifier value obtained with a hash function is associated with using the following formula:

$$H\left(y_k^t\right) = \sum_{i\in I} y_i 2^i. \tag{21}$$

If a solution that is candidate to be part of the new population is already on it, this solution is discarded. This process guarantees that all the solutions in the population will be different from each other. Then, the *crossover* or *mutation* step is performed until the number of generations (Max_Gen) is obtained. This means that the iterations will be repeated as long as $t \leq Max\_Gen$.

It is important to notice that in the proposed algorithm, the selection of the evolutionary mechanism is randomly chosen. This means that only an individual will enter into the crossover or into the mutation operator. An individual cannot enter into both of those operators. In the proposed algorithm the chances to enter into any of those operators are the same.

The decision of equally allowing the individuals to enter into the crossover or mutation operators is based on the preliminary tests that showed that if crossover is considered,

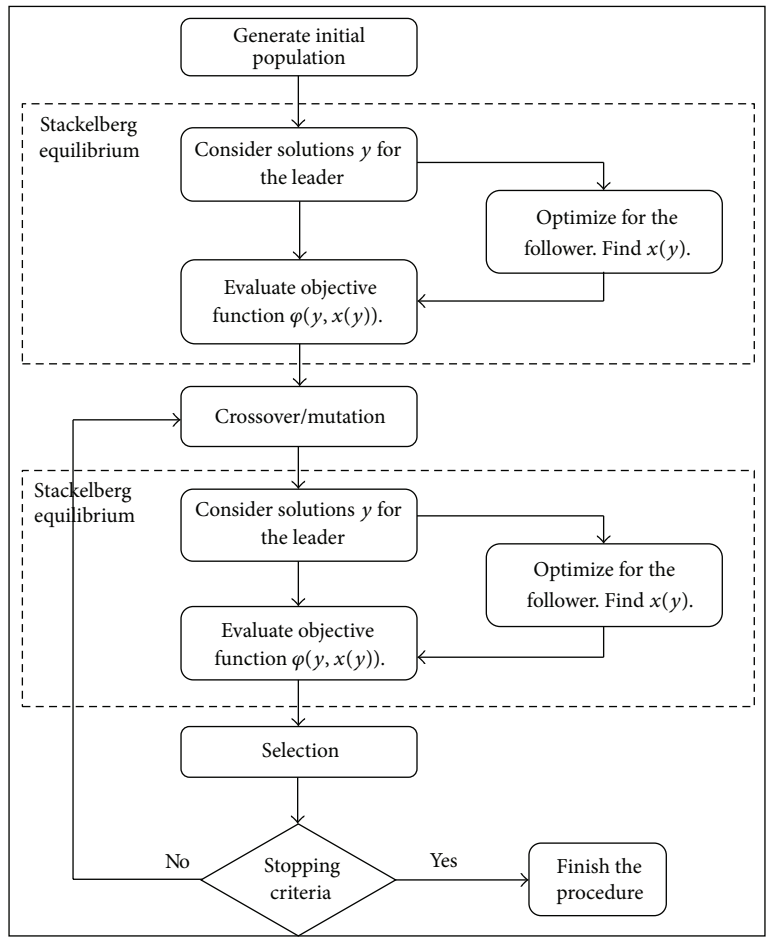

Figure 1: Stackelberg-evolutionary algorithm description.

the algorithm would lead to very diverse results when the algorithm is reaching a good solution. This was expected because when a leader's solution reaches the cardinality that provides good objective function values, it is better to explore the neighborhood around that solution. Based on this fact and considering that in Fogel and Atmar [37] it is justified that the crossover is not strictly necessary or superior to the mutation, both operators are considered in the same proportion for this particular problem.

In Figure 1 the overall framework of the algorithm proposed that summarizes the steps previously described is illustrated.

Details of the predetermined values such as the size of the population n, the number of generations Max_Gen, the number of tournaments Max_Tourn, the probability ρ of entering into the crossover or mutation phase, and the value of σ for the mutation step are described in the next section.

4. Numerical Experimentation

In this section results of the numerical experimentation are presented. To test the performance of the algorithm a particular set of benchmark instances is considered and another set of larger size instances is generated. All the experimentation was conducted on a 3.00 GHz Pentium Dual-Core Processor with 2.00 GB RAM running under Windows 7 Professional operative system. The reformulations 1-LSUFLPUP and 2-LSUFLPUP were solved directly with CPLEX 12.1. The Stackelberg-evolutionary algorithm proposed to solve the bilevel formulation (BUFLPUP) was implemented in C++. In order to find the $x(y)$ solutions, at each iteration this algorithm solves exactly the lower level problem.

For solving the lower level problem, the approach considered in Marić et al. [22] is employed, where an ordered preferences matrix was considered. For each customer they sort the index of the most preferred locations. Then, they search for the opened facility most preferred by a particular customer and assign it to that location. By following such procedure it is guaranteed that the optimal follower's response will be the same as the one obtained by solving the integer problem associated with the lower level.

The main reason, which led to the decision of solving the lower level using this approach, is the computational time. The required time for CPLEX 12.1 to solve the lower level problem is on average 0.0225 seconds (for the 50×50 instances) plus the time that is necessary for creating and

Table 1: Instance sizes.

Instance	Locations	Clients
Small_1	50	50
Small_2	50	75
Medium_1	75	100
Medium_2	100	1000
Large_1	300	1000
Large_2	500	1000

initializing the CPLEX environment. Let us suppose that in the best-case scenario all the 100 individuals perform the mutation operation over the 150 generations; then 100 offspring will result and their optimal lower level solution will need to be computed. Hence, neglecting the required time to create the environment, the computation associated with this case is $0.0225(100)(150) = 337.5$ seconds. This time is considerably high for the algorithm proposed herein to be competitive in terms of computational time.

4.1. Instances. For the numerical experimentation two main sets of instances are considered. The first set is obtained from Cánovas et al. [18]. For these instances the authors use Beasley's OR-Library and adapt them to incorporate preferences. It is worthy to note that these instances were also used to test the performance of the proposed work in Vasil'ev and Klimentova [21]. The second set of instances is obtained in the same way as the first one. The uncapacitated warehouse location instances from Beasley's OR-Library are considered and selected, the *capa, capb,* and *capc* files. The size of these three instances is 100 locations and 1000 customers. In order to establish the preferences, the procedure described in Cánovas et al. [18] is implemented, where the preferences are generated throughout a triangular distribution taking into account that the customers that are near to a location have bigger probability of preferring that specific location. For the larger size instances values between the corresponding ranges for the 100×1000 files are randomly generated. Six different instance sizes are considered as described in Table 1.

For the first set of instances, three replicates of each instance size and four different values of the vector of the preferences are considered. This results to a total of $3 \cdot 3 \cdot 4 = 36$ instances. For the Small_1 instances, the three replicates referred to as 132, 133, and 134 differ from each other on the values of the distribution and fixed costs (c_{ij}, f_i). Also, the instance considers four different matrices of preferences. Hence, an instance 132_1 refers to an instance of 50 clients, 50 locations, replicate 132, and preference matrix 1. The same structure is considered for the other two instance sizes, but the replicates for both instance sizes are referred to as A, B, and C. For example, A75_50_1 consists of 75 clients, 50 locations, replicate A, and preference vector 1. The nomenclature is similar to the 100 clients and 75 locations (A100_75_1 for example). Therefore, for the second set of instances a similar procedure was done resulting in 36 other instances, giving a total of 72 instances to measure the performance of the Stackelberg-evolutionary algorithm.

4.2. Reformulation Scheme. As previously mentioned, the proposed reformulations 1-LSUFLPUP and 2-LSUFLPUP are implemented in C++ by using the CPLEX 12.1 optimizer in order to solve the problem. Unfortunately, only the small size instances, Small_1 and Small_2, can be optimally solved. Hence, as mentioned in Section 3, the use of a heuristic procedure is justified because of the complexity of the problem.

Also, in the same way as in Marić et al. [22] the fourth reformulation presented in Hansen et al. [17] (here referred to as 4-UFLPUP) was implemented in the same computational environment in order to compare the efficiency of the reformulations.

In Table 2 the required time (in seconds) for optimally solving the tested problems is presented. 10 instances from the Small_1 and Small_2 pools are randomly selected. The "*Size*" column indicates the number of possible locations and the total customers to be served. The "*Instance*" column represents the label for the instance. The 4-UFLPUP column corresponds to the fourth reformulation presented in Hansen et al. [17] and the remaining two columns show the time required by the reformulations.

In Table 2 the "$*$" mark followed by a percentage represents that the optimal value was not found due to the limitations of the computational environment and such percentage indicates the optimality gap. In the cases, where the mark "$*$" appears, it implicates that the computer ran out of memory and the optimizer stops with the best found value until that moment and the optimality gap is presented between parenthesis. For the rest of the cases, the optimal value was found.

From Table 2, it can be noticed that the reformulation proposed in Hansen et al. [17] optimally solved both subsets of instances; the 50×50 instances were solved in a short time but for the 50×75 instances the time increases considerably. This significant increment of time may be caused by the computation of all the subsets related to the ordered preferences.

Also, it can be noticed that CPLEX 12.1 is able to optimally solve the models 1-LSUFLPUP and 2-LSUFLPUP for the 50×50 instances. On the other hand, only the 2-LSUFLPUP reformulation reaches the optimal value for the 50×75 instances. Hence, it can be concluded that the reformulation scheme proposed is valid. As a conclusion it may be established that the 2-LSUFLPUP reformulation needs more than 40 minutes for optimally solving the 50×50 instances, while the 1-LSUFLPUP needs more than 80 minutes.

With respect to the 50×75 instances, the 1-LSUFLPUP reformulation cannot be solved because the computer ran out of memory giving small optimality gaps. The 2-LSUFLPUP model requires less time than the 4-UFLPUP in four of the five instances. However, the computational times are not very acceptable and due to this fact the heuristic algorithm described in this paper is proposed.

It is important to notice that the reformulation presented in Hansen et al. [17] only solved 30×30 instances to optimality. In Marić et al. [22] it is indicated that the maximum instances sizes that can be solved by that reformulation are 50×50 instances. Since different instances are used, we cannot

Table 2: Required time for the reformulations (in seconds).

Size	Instance	4-UFLPUP	2-LSUFLPUP	1-LSUFLPUP
50 × 50	132_1	184.41	3,723.45	8,723.45
50 × 50	132_2	204.98	5,368.27	11,368.27
50 × 50	133_1	464.88	2,461.96	4,878.99
50 × 50	133_3	394.31	3,686.08	6,669.94
50 × 50	134_1	200.30	3,721.31	6,208.53
50 × 75	a75_50_1	34,408.00	28,701.58	(13.46%)*
50 × 75	a75_50_3	28,039.10	30,165.06	(13.09%)*
50 × 75	b75_50_1	16,755.00	9,691.56	(10.58%)*
50 × 75	b75_50_4	12,370.00	8,601.30	(8.11%)*
50 × 75	c75_50_1	34,951.90	10,599.17	(11.33%)*

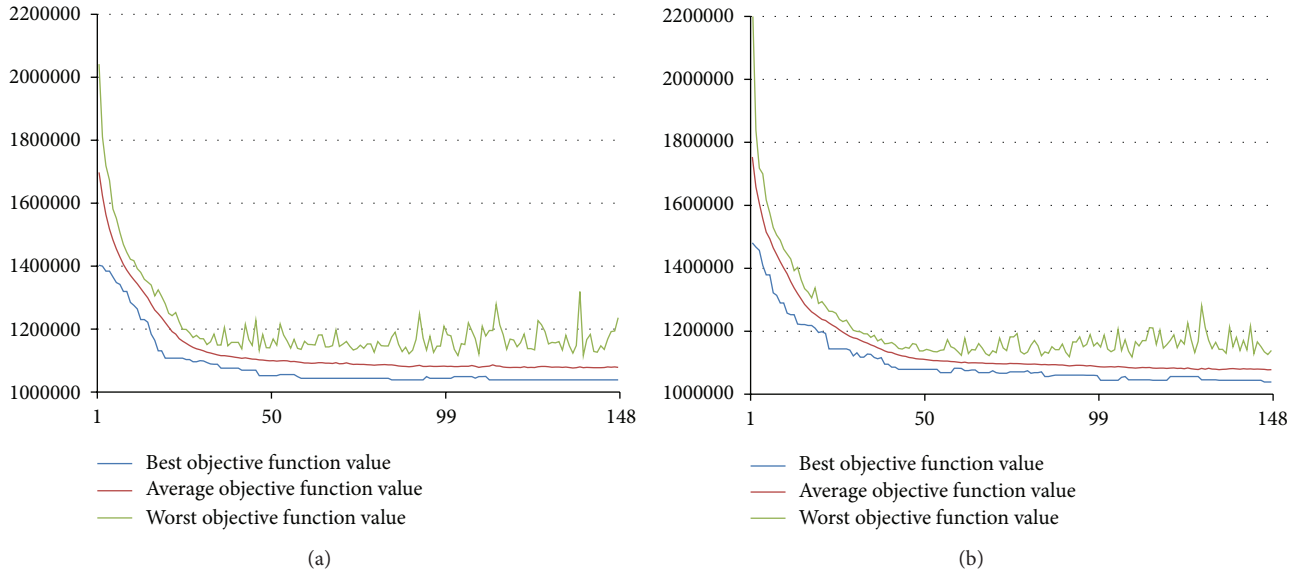

Figure 2: Plot of an example of the preliminary tests.

generalize the fact that any of the reformulations considered is able to solve just a particular size of instances.

4.3. Stopping Rules and Control Parameters Definition. Throughout the computational testing, the S-E procedure is stopped when the maximum number of generations is reached. The preliminary testing points out that the main control parameters correspond to the size of the population n, the number of generations to be created *Max_Gen* and the number of tournaments to be performed *Max_Tourn*. The latter parameter is defined through a preliminary experimentation, where it was observed that as long as the number of tournaments is increased, diversity is lost. Hence, based on the results and also considering commonly used values in this type of algorithm, its value was set to five.

Once the value of the number of tournaments is defined, a preliminary experimentation was performed to set the value of the two other parameters. For such purpose, the values of $n = 100, 150$ and the values of *Max_Gen* $= 100, 150$ are considered for all the instances. Each instance is run ten times and the average value found is reported. As a result of the experimentation, n is set to the value of 100 and *Max_Gen* to 100.

A full factorial 2^2 experiment is developed by using, as factors, the size of the population n and the number of generations *Max_Gen* at the minimum and maximum levels 100 and 150 for n and the same for *Max_Gen* considering the different sizes of problems as blocks. The experiment's main objective is to determine the optimal levels of each parameter by analyzing the effect of each factor in the performance of the experiment measured in two variables: execution time and the gap between the heuristic and optimal values of each instance solved, which is defined by the following equation:

$$\%\text{Gap} = \left(\frac{\varphi_{\text{SE}} - \varphi_{\text{opt}}}{\varphi_{\text{opt}}} \right) \times 100\%, \tag{22}$$

where φ_{SE} is the value of the solution obtained by the S-E procedure and φ_{opt} is the value of the optimal solution (when it is known, and in the cases when this is not possible, the strategy described in the next subsection is utilized).

It can be concluded that the factor *Max_Gen* significantly affects the gap, while the factor n does not affect this response

Table 3: Results for the small-size instances.

Instance size	Instance ID	Optimal value	Average value	Best value	Worst value	Gap (%)	Time (sec)	Number of opt
50 × 50	132_1	1,122,750	1,129,632.20	1,122,750	1,156,712	0.613	1.4458	9
50 × 50	132_2	1,157,722	1,166,628.87	1,157,722	1,193,100	0.769	1.4325	9
50 × 50	132_3	1,146,301	1,157,273.67	1,146,301	1,283,510	0.957	1.4391	11
50 × 50	132_4	1,036,779	1,040,920.87	1,036,779	1,056,677	0.399	1.4704	9
50 × 50	133_1	1,103,272	1,104,166.80	1,103,272	1,110,614	0.081	1.472	12
50 × 50	133_2	0,135,443	1,043,974.87	0,135,443	1,099,432	0.824	1.4681	13
50 × 50	133_3	1,171,331	1,179,617.13	1,171,331	1,192,603	0.707	1.4576	6
50 × 50	133_4	1,083,636	1,083,955.53	1,083,636	1,087,769	0.029	1.4703	12
50 × 50	134_1	1,179,639	1,185,867.73	1,179,639	1,219,335	0.528	1.4673	11
50 × 50	134_2	1,121,633	1,132,109.40	1,121,633	1,149,761	0.934	1.4701	8
50 × 50	134_3	1,171,409	1,183,031.40	1,171,409	1,198,847	0.992	1.4624	6
50 × 50	134_4	1,210,465	1,222,462.13	1,210,465	1,236,753	0.991	1.4604	6
50 × 75	a75_50_1	1,661,269	1,664,342.60	1,661,269	1,673,507	0.185	1.9681	6
50 × 75	a75_50_2	1,632,907	1,648,883.00	1,632,907	1,682,822	0.978	1.9632	8
50 × 75	a75_50_3	1,632,213	1,632,698.13	1,632,213	1,633,426	0.029	1.9820	8
50 × 75	a75_50_4	1,585,028	1,585,428.40	1,585,028	1,591,034	0.025	1.9557	14
50 × 75	b75_50_1	1,252,804	1,260,789.53	1,252,804	1,304,636	0.637	1.9883	12
50 × 75	b75_50_2	1,337,446	1,341,283.27	1,337,446	1,357,102	0.287	1.9729	5
50 × 75	b75_50_3	1,249,750	1,257,445.27	1,249,750	1,282,987	0.616	1.9696	7
50 × 75	b75_50_4	1,217,508	1,228,422.53	1,217,508	1,235,590	0.896	1.9687	3
50 × 75	c75_50_1	1,310,193	1,318,412.47	1,310,193	1,332,980	0.627	1.9745	2
50 × 75	c75_50_2	1,244,255	1,252,745.53	1,244,255	1,283,134	0.682	1.9746	1
50 × 75	c75_50_3	1,201,706	1,203,925.27	1,201,706	1,232,129	0.185	1.9737	12
50 × 75	c75_50_4	1,334,782	1,341,951.47	1,334,782	1,370,764	0.537	1.9686	7

Table 4: Results for the medium-size instances.

Instance size	Instance ID	Optimal value	Average value	Best value	Worst value	Gap (%)	Time (sec)	Number of opt
75 × 100	a100_75_1	2,286,397	2,305,392.47	2,286,397	2,407,972	0.831	3.4230	11
75 × 100	a100_75_2	2,463,187	2,466,300.40	2,463,187	2,509,888	0.126	3.3775	14
75 × 100	a100_75_3	2,415,836	2,439,790.27	2,415,836	2,481,304	0.992	3.3760	6
75 × 100	a100_75_4	2,380,150	2,386,191.33	2,380,150	2,398,158	0.254	3.4679	7
75 × 100	b100_75_1	1,950,231	1,968,184.47	1,950,231	1,999,198	0.921	3.5263	8
75 × 100	b100_75_2	2,023,097	2,043,233.40	2,023,097	2,067,532	0.995	3.5099	6
75 × 100	b100_75_3	2,062,595	2,082,063.53	2,062,595	2,142,439	0.944	3.5200	2
75 × 100	b100_75_4	1,865,323	1,883,910.00	1,965,323	1,953,993	0.996	3.4973	3
75 × 100	c100_75_1	1,843,620	1,851,536.07	1,843,620	1,875,764	0.429	3.5425	7
75 × 100	c100_75_2	1,808,867	1,826,892.60	1,808,867	1,885,459	0.997	3.5293	8
75 × 100	c100_75_3	1,820,587	1,829,550.67	1,820,587	1,869,527	0.492	3.4979	10
75 × 100	c100_75_4	1,839,007	1,856,116.80	1,839,007	1,896,699	0.930	3.5146	4
100 × 1000	cap_a_1	27,490,608*	27,669,164.8	27,490,608	27,848,872	0.649	49.939	5
100 × 1000	cap_a_2	27,434,572*	27,620,332.0	27,434,572	27,909,206	0.677	49.662	5
100 × 1000	cap_a_3	27,435,430*	27,639,741.2	27,435,430	27,792,660	0.745	49.241	5
100 × 1000	cap_a_4	27,804,896*	27,813,746.1	27,804,896	27,937,648	0.032	51.115	14
100 × 1000	cap_b_1	24,449,922*	24,449,922.0	24,449,922	24,449,922	0.000	51.631	15
100 × 1000	cap_b_2	24,532,388*	24,556,056.3	24,532,388	24,708,592	0.096	50.304	12
100 × 1000	cap_b_3	24,318,664*	24,331,881.6	24,318,664	24,351,708	0.054	51.039	9
100 × 1000	cap_b_4	24,526,456*	24,535,120.8	24,526,456	24,591,442	0.035	50.444	13
100 × 1000	cap_c_1	23,095,054*	23,321,130.5	23,095,054	23,740,388	0.979	50.557	4
100 × 1000	cap_c_2	23,368,530*	23,447,282.0	23,368,530	23,579,496	0.337	51.621	4
100 × 1000	cap_c_3	23,237,760*	23,366,553.3	23,237,760	23,502,456	0.554	52.952	6
100 × 1000	cap_c_4	23,192,496*	23,257,851.5	23,192,496	23,461,264	0.282	52.449	5

Table 5: Results for the large-size instances.

Instance size	Instance ID	Optimal value	Average value	Best value	Worst value	Gap (%)	Time (sec)	Number of opt
300 × 1000	big_a_1	22,255,196*	22,463,397.9	22,255,196	23,011,264	0.935	137.294	7
300 × 1000	big_a_2	22,189,452*	22,203,899.7	22,189,452	22,406,168	0.065	136.519	14
300 × 1000	big_a_3	22,765,286*	22,930,434.2	22,765,286	23,615,044	0.725	136.765	9
300 × 1000	big_a_4	22,799,540*	22,888,083.0	22,799,540	23,186,908	0.388	136.618	6
300 × 1000	big_b_1	22,475,276*	22,693,778.4	22,475,276	23,597,092	0.972	136.842	4
300 × 1000	big_b_2	21,988,192*	22,368,965.6	21,988,192	24,321,834	1.732	136.854	7
300 × 1000	big_b_3	22,919,924*	23,276,402.6	22,919,924	24,159,568	1.555	136.714	6
300 × 1000	big_b_4	22,442,388*	22,528,429.8	22,442,388	22,660,422	0.383	137.477	8
300 × 1000	big_c_1	21,207,902*	21,213,366.2	21,207,902	21,215,708	0.026	138.299	5
300 × 1000	big_c_2	21,199,978*	21,411,178.8	21,199,978	22,758,784	0.996	137.756	12
300 × 1000	big_c_3	21,404,592*	21,439,839.4	21,404,592	21,748,904	0.165	137.719	4
300 × 1000	big_c_4	21,896,128*	22,000,610.0	21,896,128	22,228,388	0.477	137.965	8
500 × 1000	large_a_1	38,635,228*	44,263,374.8	38,635,228	47,614,256	14.567	246.448	2
500 × 1000	large_a_2	34,642,732*	37,575,132.3	34,642,732	54,083,736	8.465	245.541	4
500 × 1000	large_a_3	37,365,008*	52,054,726.0	37,365,008	53,931,292	12.551	247.318	2
500 × 1000	large_a_4	36,443,660*	41,851,030.5	36,443,660	48,943,700	14.838	247.371	4
500 × 1000	large_b_1	30,678,758*	34,006,518.8	30,678,758	38,201,136	10.847	247.373	4
500 × 1000	large_b_2	30,304,860*	32,972,009.2	30,304,860	36,488,528	8.801	248.489	3
500 × 1000	large_b_3	30,870,146*	34,027,882.8	30,870,146	38,467,564	10.229	258.161	1
500 × 1000	large_b_4	30,330,226*	31,763,682.9	30,330,226	36,202,968	4.726	248.756	3
500 × 1000	large_c_1	30,354,964*	31,914,530.3	30,354,964	36,746,900	5.138	250.254	3
500 × 1000	large_c_2	30,341,740*	32,659,784.3	30,341,740	36,388,612	7.639	248.719	1
500 × 1000	large_c_3	30,281,492*	31,677,372.4	30,281,492	34,613,144	4.609	249.228	1
500 × 1000	large_c_4	24,334,680*	30,592,168.1	24,334,680	34,946,604	15.714	248.388	2

variable. Regarding the time of execution response variable, both variables affect it negatively, n being the parameter with higher effect. Hence, the factor *Max_Gen* is set to the value of 150 and the factor n to the value of 100.

For example, in Figure 2 the leader's objective function values found are plotted in two different runs of the 132_4 instance with the parameter setting as it is described above. In the left plot it can be seen that the optimal value was found around the 110 generation, but on the right plot it was not found until the 148 generation. If the *Max_Gen* value is set to 100 in both cases, the optimum would not be reached.

The parameters ρ and σ are selected based on the preliminary testing which showed that the crossover does not lead to better solutions in comparison with the mutation operator for this particular problem, so it was decided to set $\rho = 0.5$. Finally, σ is selected in such a way that assures that any version of the mutation is equally possible (adding, deleting, or interchanging locations).

It is important to mention that for the Small_1, Small_2, and Medium_1 instances the optimal value is known *a priori*. Hence, parameter setting is performed by considering this value in the preliminary testing. For the Medium_2, Large_1, and Large_2 instances before the preliminary testing, 10 long-run experiments are done setting *Max_Gen* = 1000 and n = 500 with the aim at getting the best leader's objective function value that will be considered as the optimum. After that, the size of the population is set to 100 and 200 generations. The *Max_Tourn*, ρ, and σ parameters remain as described above.

4.4. Numerical Results. This subsection presents the numerical results. Tables 3, 4, and 5 show a summary of the results obtained from the computational testing for the S-E algorithm. It is important to point out that the presented results correspond to all the instance sizes considered and that 15 runs of the algorithm for each of the benchmark instances are performed. The average value of the results obtained and the number of the times, where the optimal solution is obtained by the algorithm, are presented. As previously mentioned, an average gap between the heuristic and optimal values of each instance solved is computed, as defined in (22). A positive gap means that, in average, the optimal solution is not obtained by the algorithm.

These tables also show the average of the leader's objective function value that corresponds to the solutions found by the algorithm (over the 15 runs) for each instance. The first two columns refer to the identification of the instance. The next column presents the value of the optimal solution, the "$*$" means that the optimum is unknown, and the best value obtained in the long-run experiments is presented. The next four columns present the results of the Stackelberg-evolutionary algorithm. The values corresponding to the average value obtained during the 15 runs are listed in the "*Average value*" column. The column "*Best value*" (*Worst value*) presents the minimum (maximum) value obtained in the experimentations by the algorithm over the 15 runs performed. The column "*Gap (%)*" presents the value of the gap obtained with respect to the optimal solution, as

indicated by (22), where φ_{SE} is the value that corresponds to the "*Average value*" column. The next column presents the average computational time measured in seconds of the S-E algorithm. Finally, the *"# opt"* column indicates the number of times that the algorithm reaches the optimal value over the 15 runs.

It can be observed from Table 3 that in average the optimality gap is less than 1%, which demonstrates that the proposed algorithm has a very consistent performance. In addition, more than half (around 54.7%) of the instances solved reached the optimal solution. Computational time is on average small (less than 2 seconds) which are very good values for a strategic problem of this size.

As it can be observed in Table 4, the average of the optimality gap is less than 1% for the 75 × 100 instances and for the 100 × 1000 instances; it is also less than 1% with respect to the best known value obtained by the long-run experimentation. Furthermore, the S-E algorithm always finds the optimal solution. Moreover, in more than 50% of the experiments, the algorithm reaches the optimal value. Computational time increases with respect to the small-size instances, but this is expected because of the size of the problem and the extra generations computed; no exponential increase is observed.

In Table 5 the results for the large-size instances are presented. It can be appreciated that the required time for solving the 300 × 1000 instances is on average less than 2.5 minutes; in the same way, it is on average less than 4.2 minutes for the 500 × 1000 problems. The optimality gaps corresponding to the 300 × 1000 instances are on average less than 1.75% reaching the best value known in half of the runs.

On the other hand, for the 500 × 1000 instances the average of the optimality gaps is relatively large when compared to the other instances (from 4.6% to 15.7%). The algorithm only reaches the best value in around 16.7% of the times. The main reason of this is that due to the number of the possible locations, it is not possible to compute the hash function given by (21) for the solutions belonging to the current population. This issue clearly affects the algorithm because it cannot discard the duplicated individuals of the population affecting in this way the diversity of the solutions pool. This is the reason why the S-E algorithm does not reach the best value very often for the large-size instances. One possible solution for this issue is to extend the size of the population and increase the number of generations in order to have more probability of getting a good value, but it is evident that this decision will increase the total time.

5. Conclusions and Further Research

In this paper the bilevel facility location problem under preferences (BFLPUP) of the customers with respect to the most desired locations is addressed. Two reformulations of the problem to reduce the bilevel programming problem to a single-level mixed integer program (MIP) are proposed. The reformulations are solved directly by CPLEX for a number of small-size instances. In order to measure the effectiveness of the reformulations, computational tests are done, comparing the running time against the time required for the fourth reformulation that was presented in Hansen et al. [17] validating in this way that the proposed reformulations find the optimal solution.

Motivated by the difficulty of the problem and due to the fact that only the smaller size instances could be solved directly by the reformulations proposed, a heuristic algorithm for solving the bilevel formulation of the problem is presented. It is important to point out that only the three procedures presented in Marić et al. [22] solve the bilevel formulation of the problem. Most of the existing techniques are designed to solve a reduction or a reformulation of the original problem. Furthermore, Li and Fang [38] also propose an evolutionary algorithm based on the duality conditions for a general bilevel problem obtaining efficient results. This led to the conclusion that evolutionary algorithms may be a good option for solving bilevel programming problems.

Numerical results for different size instances are presented showing a good performance of the S-E algorithm. In more than half of the times that the S-E was run, the optimal value was found and for the rest of the times gaps obtained were relatively small.

Low computational time is observed for the S-E procedure with a maximum time of 2, 53, and 250 seconds for the small, medium, and large-size instances, respectively. The computational time obtained can be regarded as a very efficient value for a strategic problem of the dimensions considered in this paper. Furthermore, the presented times consist of the total time for solving the problem without considering any initial solution as a seed for the algorithm. For instance, in Vasil'ev and Klimentova [21] lower bounds are obtained with a cutting plane approach and then with a heuristic based on simulated annealing. By using these bounds, Vasil'ev and Klimentova, [21] preprocessed a branch and price procedure to solve the problem, reporting computational time of the branch and price method. Hence, in order to compare computational time, the lower and upper bounds to preprocess the S-E procedure should be considered.

For further research, the design and implementation of a Stackelberg-Scatter Search algorithm are proposed, with the aim of obtaining better results of this approach because of the combination and improvement steps which are not randomly performed as it is done in the S-E procedure. A solution methodology to solve the reformulations of the problem may also be proposed, in order to be able to solve larger size instances. Also, an extension of the problem to the capacitated version maintaining the preferences of the customers may be addressed for further research. Also, the lower bound that has been proposed in Vasil'ev and Klimentova [21] can be included in the procedure in order to improve the S-E algorithm.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This research has been partially supported by the Secretariat of Public Education (SEP by its acronym in Spanish) as a part of the Teachers Improvement Program and the Project named PROMEP/103.5/10/3889 and the Academic Groups research Project PROMEP/103.5/12/4953 and the State University of Nuevo León (UANL) within the Support Program for Scientific Research and Technology (PAICYT) with the Project CE960-11. The authors would also like to thank Dr. Ivan Guardiola, Assistant Professor from the Missouri University of Science & Technology, for several useful comments that improved this paper as well as anonymous reviewers for their constructive input and suggestions.

References

[1] A. Weber, *Theory of the Location of Industries*, University of Chicago Press, Chicago, Ill, USA, Translated by C.J. Friedrich, 1929.

[2] Z. Drezner, *Facility Location: A Survey of Applications and Methods*, Springer, New York, NY, USA, 1995.

[3] M. S. Daskin, *Network and Discrete Location: Models, Algorithms and Applications*, John Wiley & Sons, New York, NY, USA, 1995.

[4] Z. Drezner and H. Hamacher, *Facility Location: Applications and Theory*, Springer, Berlin, Germany, 2002.

[5] D. Simchi-Levi, P. Kaminsky, and E. Simchi-Levi, *Designing and Managing the Supply Chain: Concepts Strategies and Case Studies*, McGraw-Hill, Boston, Mass, USA, 2nd edition, 2003.

[6] M. S. Daskin, "What you should know about location modeling," *Naval Research Logistics*, vol. 55, no. 4, pp. 283–294, 2008.

[7] R. D. Galvão, "Uncapacitated facility location problems: contributions," *Pesquisa Operacional*, vol. 24, no. 1, pp. 7–38, 2004.

[8] A. A. Kuehn and M. J. Hamburger, "A heuristic program for locating warehouses," *Management Science*, vol. 9, no. 4, pp. 643–666, 1963.

[9] M. L. Balinski, "Integer programming: methods, uses, computation," *Management Science*, vol. 12, no. 3, pp. 253–313, 1965.

[10] S. L. Hakimi, "Optimum locations of switching centers and the absolute centers and medians of a graph," *Operations Research*, vol. 12, no. 3, pp. 450–459, 1964.

[11] S. L. Hakimi, "Optimum distribution of switching centers in a communication network and some related graph theoretic problems," *Operations Research*, vol. 13, no. 3, pp. 462–475, 1965.

[12] G. Cornuéjols, M. L. Fisher, and G. L. Nemhauser, "On the uncapacitated location problem," *Annals of Discrete Mathematics*, vol. 1, pp. 163–177, 1977.

[13] R. D. Galvão and L. A. Raggi, "A method for solving to optimality uncapacitated location problems," *Annals of Operations Research*, vol. 18, no. 1–4, pp. 225–244, 1989.

[14] P. Hanjoul and D. Peeters, "A facility location problem with clients' preference orderings," *Regional Science and Urban Economics*, vol. 17, no. 3, pp. 451–473, 1987.

[15] J. Krarup and P. M. Pruzan, "The simple plant location problem: survey and synthesis," *European Journal of Operational Research*, vol. 12, no. 1, pp. 36–81, 1983.

[16] L. E. Gorbachevskaya, *Polynomially solvable and NP-hard standardization problems [Ph.D. thesis]*, Department of Mathematics and Physics, IM SO RAN, Novosibirsk, Russia, 1998.

[17] P. Hansen, Y. Kochetov, and N. Mladenovic, "Lower bounds for the uncapacitated facility location problem with user preferences," Tech. Rep. G-2004-24, GERAD-HEC, Montreal, Canada, 2004.

[18] L. Cánovas, S. García, M. Labbé, and A. Marín, "A strengthened formulation for the simple plant location problem with order," *Operations Research Letters*, vol. 35, no. 2, pp. 141–150, 2007.

[19] H. Ishii, Y. L. Lee, and K. Y. Yeh, "Fuzzy facility location problem with preference of candidate sites," *Fuzzy Sets and Systems*, vol. 158, no. 17, pp. 1922–1930, 2007.

[20] I. L. Vasil'ev, K. B. Klimentova, and Yu. A. Kochetov, "New lower bounds for the facility location problem with clients' preferences," *Computational Mathematics and Mathematical Physics*, vol. 49, no. 6, pp. 1010–1020, 2009.

[21] I. L. Vasil'ev and K. B. Klimentova, "The branch and cut method for the location problem with client's preferences," *Journal of Applied and Industrial Mathematics*, vol. 4, no. 3, pp. 441–454, 2010.

[22] M. Marić, Z. Stanimirović, and N. Milenković, "Metaheuristic methods for solving the bilevel uncapacitated facility location problem with clients' preferences," *Electronic Notes in Discrete Mathematics*, vol. 39, pp. 43–50, 2012.

[23] E. V. Alekseeva and T. A. Kochetov, "Genetic local search for the p-median problem with client's preferences," *Diskretnyĭ Analiz i Issledovanie Operatsiĭ*, vol. 14, no. 1, pp. 3–31, 2007.

[24] D. Aksen, N. Aras, and N. Piyade, "A bilevel p-median model for the planning and protection of critical facilities," *Journal of Heuristics*, vol. 19, no. 2, pp. 373–398, 2013.

[25] J. M. Lee and Y. H. Lee, "Facility location and scale decision problem with customer preference," *Computers & Industrial Engineering*, vol. 63, no. 1, pp. 184–191, 2012.

[26] Y. Yin, "Genetic-algorithms-based approach for bilevel programming models," *Journal of Transportation Engineering*, vol. 126, no. 2, pp. 115–120, 2000.

[27] J. F. Wang and J. Periaux, "Multi-point optimization using GAs and Nash/Stackelberg games for high lift multi-airfoil design in aerodynamics," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '01)*, vol. 1, pp. 552–559, Seoul, Republic of Korea, May 2001.

[28] J. Periaux, H. Q. Chen, B. Mantel, M. Sefrioui, and H. T. Sui, "Combining game theory and genetic algorithms with application to DDM-nozzle optimization problems," *Finite Elements in Analysis and Design*, vol. 37, no. 5, pp. 417–429, 2001.

[29] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi, *Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties*, Springer, Berlin, Germany, 1999.

[30] H. V. Stackelberg, *The Theory of the Market Economy*, Springer, Berlin, Germany, 1934, Translated by Oxford University Press, 1952.

[31] B. Colson, P. Marcotte, and G. Savard, "An overview of bilevel optimization," *Annals of Operations Research*, vol. 153, no. 1, pp. 235–256, 2007.

[32] S. Fang, P. Guo, M. Li, and L. Zhang, "Bilevel multiobjective programming applied to water resources allocation," *Mathematical Problems in Engineering*, vol. 2013, Article ID 837919, 9 pages, 2013.

[33] H. I. Calvete, C. Galé, and M.-J. Oliveros, "Bilevel model for productiondistribution planning solved by using ant colony optimization," *Computers & Operations Research*, vol. 38, no. 1, pp. 320–327, 2011.

[34] V. Kalashnikov, F. Camacho, R. Askin, and N. Kalashnykova, "Comparison of algorithms for solving a bi-level toll setting problem," *International Journal of Innovative Computing, Information and Control*, vol. 6, no. 8, pp. 3529–3549, 2010.

[35] F. Legillon, A. Liefooghe, and E. G. Talbi, "CoBRA: a coevolutionary meta-heuristic for bi-level optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '12)*, pp. 1–8, Queensland, Australia, 2012.

[36] D. Aksen and N. Aras, "A bilevel fixed charge location model for facilities under imminent attack," *Computers & Operations Research*, vol. 39, no. 7, pp. 1364–1381, 2012.

[37] D. B. Fogel and J. W. Atmar, "Comparing genetic operators with Gaussian mutations in simulated evolutionary processes using linear systems," *Biological Cybernetics*, vol. 63, no. 2, pp. 111–114, 1990.

[38] H. Li and L. Fang, "An evolutionary algorithm for solving bilevel programming problems using duality conditions," *Mathematical Problems in Engineering*, vol. 2012, Article ID 471952, 14 pages, 2012.