Intl. Trans. in Op. Res. XX (20XX) 1–18
DOI: xx.xxxx/itor.xxxxx

INTERNATIONAL TRANSACTIONS IN OPERATIONAL RESEARCH

A Hybrid Adaptive Iterated Local Search Heuristic for the Maximal Covering Location Problem

Vinícius R. Máximo[a], Jean-François Cordeau[b] and Mariá C. V. Nascimento[c,*]

[a]*Instituto de Ciência e Tecnologia, Universidade Federal de São Paulo (UNIFESP)*
Av. Cesare M. G. Lattes, 1201, Eugênio de Mello, São José dos Campos-SP, CEP: 12247-014, Brazil
[b]*HEC Montréal and GERAD, 3000 chemin de la Côte-Sainte-Catherine, Montréal, H3T 2A7, Canada*
[c]*Divisão de Ciência da Computação (IEC), Instituto Tecnológico de Aeronáutica (ITA), Praça Marechal Eduardo Gomes, 50, Vila das Acácias, São José dos Campos-SP, CEP 12228-900, Brazil*
E-mail: vinymax10@gmail.com; jean-francois.cordeau@hec.ca; mariah@ita.br

Received DD MMMM YYYY; received in revised form DD MMMM YYYY; accepted DD MMMM YYYY

Abstract

Adaptive Iterated Local Search (AILS) is a recently proposed metaheuristic paradigm that focuses on adapting the diversity control of Iterated Local Search (ILS) by online learning mechanisms. It has been successfully applied to the Capacitated Vehicle Routing Problem (CVRP) and the Heterogeneous Vehicle Routing Problem. Hybridizing it with Path Relinking (PR) has further improved the intensification of the method for the CVRP, providing outstanding results. However, the potential of this metaheuristic has not yet been investigated on other combinatorial optimization problems, such as location problems. In this paper, we develop a version of AILS for the maximal covering location problem (MCLP). This problem consists of locating a number of facilities to maximize the covered customer demand, where a given facility location can meet the demand of customers located within a coverage radius. Experiments on large-scale instances of the MCLP indicate that AILS hybridized with PR, called AILS-PR, outperforms the state-of-the art metaheuristic.

Keywords: Learning in metaheurstics; Maximal Covering Location Problem (MCLP); Adaptive Iterated Local Search (AILS); Path Relinking (PR)

*Author to whom all correspondence should be addressed (e-mail: mariah@ita.br).

1. Introduction

The Maximal Covering Location Problem (MCLP) (Moore and ReVelle, 1982) is a classical facility location problem that consists of finding the ideal location for p facilities so that the sum of the demands of the customers that are within the coverage radius of these facilities is as large as possible. This problem was initially proposed by Church and ReVelle (1974) and belongs to the class of $\mathscr{NP}$-hard problems (Garey and Johnson, 1979). This problem is highly relevant and has widespread applications in various service industries.

Recent studies on the MCLP concern both exact algorithms (Cordeau et al., 2019) and heuristic methods (Máximo et al., 2017; Máximo and Nascimento, 2019). Heuristics are useful for large-scale instances, where exact methods often remain too time or memory consuming. Because metaheuristics for classical problems have been widely investigated, outperforming the current state-of-the-art heuristics is challenging. In line with this, only a metaheuristic designed to effectively balance intensification and diversification would stand a chance of achieving better performance on this problem.

Diversity control in metaheuristics is an emerging and highly relevant research topic. It is directly related to the ability of the method to adjust the degree of exploration during the search process. If the diversity degree is high, the method tends to broaden its search to unexplored regions of the solution space. However, if the exploratory feature of a metaheuristic is too sharp, it usually lacks exploitation, another desirable characteristic to refine local optima. One way to balance the exploration and exploitation features of metaheuristics is through diversity control strategies.

A challenge identified in the literature is to tackle the diversity control in the widely used metaheuristic paradigm known as Iterated Local Search (ILS) (Lourenço et al., 2003). ILS is a two-phase local search-based metaheuristic that has been used to solve a number of combinatorial optimization problems such as vehicle routing problems (Penna et al., 2013) and scheduling problems (Subramanian et al., 2014), among others. In ILS, an initial reference solution is iteratively refined by a local search followed by a perturbation phase. The perturbation phase aims at identifying solutions "close" to the reference solution but with a hopefully better local optimum within its neighborhood. Two key parameters of this metaheuristic to control its diversity are the acceptance criterion, which establishes the conditions under which a local optimum becomes the reference solution, and the perturbation degree, which is the extent of the changes applied to the reference solution to guide the search for a local optimum. To adjust these parameters by learning strategies, Máximo and Nascimento (2021) introduced the Adaptive Iterated Local Search (AILS) concept.

AILS is an ILS-based metaheuristic that attempts to automatically define the acceptance and perturbation degree criteria based on past iteration information. It was originally designed to solve the Capac-

itated Vehicle Routing Problem (CVRP) and, hybridized with the well-known intensification strategy of Path Relinking (PR), it outperformed state-of-the-art metaheuristics. It was also applied to the Heterogeneous Fleet Vehicle Routing Problem (HFVRP) (Máximo et al., 2022) for which, without hybridization, it outperformed the existing algorithms.

Motivated by the good performance of AILS on routing problems, this paper proposes an AILS for the classical facility location problem known as the Maximal Covering Location Problem (MCLP). The literature on solution methods for the MCLP is vast and exact methods have been successfully applied to small and medium-sized instances. In spite of this, just a few solution methods are capable of handling large-scale instances of the MCLP, for which exact solutions perform poorly.

In this paper, a hybrid version of AILS with PR is introduced and computational experiments show its good performance over the current state-of-the-art metaheuristic. The rest of this paper is organized as follows. In Section 2, we present a brief literature review on metaheuristics for the MCLP, focusing on strategies specially designed for large-scale instances. Section 3 introduces the proposed hybrid metaheuristic. Section 4 shows the computation experiments carried out to assess the performance of the proposed metaheuristic. Section 5 wraps up the paper by highlighting its main contributions and by suggesting future research directions.

2. Literature review

Church and ReVelle (1974) introduced the MCLP which can be defined in terms of a set $N = \{1, 2, \ldots, n\}$ of customers and a set $M = \{1, 2, \ldots, m\}$ of potential facility locations, with $M \subseteq N$. Each customer or vertex i has a demand d_i and coordinates (x_i, y_i) that indicate the position of the vertex in the Cartesian plane. A vertex i is within the coverage radius of facility j if the distance between vertices i and j is less than or equal to r. The goal of the MCLP is to locate $p \leq m$ facilities so as to maximize the sum of the covered demands. In this paper, we assume that $N = M$, that is, the p facilities can be located at any customer location.

To our knowledge, only two exact algorithms have been designed for the MCLP. Downs and Camm (1996) provide bounds by a Lagrangean relaxation strategy to embed in a branch-and-bound method. Computational experiments were performed with several instances with up to 2241 demand points and 74 potential facility locations. Moreover, the authors tested these instances for p equal to 10, 15, 20, 25 or 30. The duality-based algorithm proved the optimality in 66% of these instances. More recently, Cordeau et al. (2019) introduced a branch-and-Benders-cut algorithm to tackle large-scale MCLP instances. Computational experiments indicate that their algorithm found on optimal solution to instances

with up to 1,500,000 customers and 100 potential facility locations within 600 seconds. However, according to the authors, when testing their algorithm on the instances from Máximo et al. (2017) with up to 7730 potential facility locations, the branch-and-Benders-cut algorithm could not obtain optimal solutions within 10 minutes of computing time. Exact algorithms thus seem to become inadequate when the size of the set M is large.

A number of heuristics have been proposed to solve the MCLP. Table 1 summarizes the main characteristics of the existing heuristic methods and highlights the size of the instances the authors tested in their experiments.

Table 1
The main solution methods proposed for the MCLP.

Reference	Solution Method	m	p
Galvão and ReVelle (1996)	Lagrangian Heuristic	55,100,150	3-8,10,12
Adenso-Díaz and Rodríguez (1997)	Tabu Search (TS)	213	16-20,22,25
Resende (1998)	Greedy Randomized Search Procedures	1000	10-900
Galvão et al. (2000)	Lagrangian and Surrogate Relaxations	55-900	3-28
Lorena and Pereira (2002)	Lagrangian and Surrogate Heuristic	100-900	1-28
ReVelle et al. (2008)	Heuristic Concentration	900	10,15,20
Zarandi et al. (2011)	GA	1800,2500	15,20,25
Máximo et al. (2017)	Intelligent-Guided Adaptive Search (IGAS)	2713-7730	17-100
Atta et al. (2018)	GA	100-2500	1-28
Máximo and Nascimento (2019)	IGAS with PR	2713-7730	50-100

Among the existing solution methods, we highlight below those intended to solve instances with over 1,000 potential facility locations.

Zarandi et al. (2011) introduced a Genetic Algorithm (GA) to solve large MCLP instances. The computational experiments were carried out on instances artificially generated using the methodology of ReVelle et al. (2008). The authors compared their results with CPLEX, which found optimal solutions for all tested instances. The optimality gaps achieved by GA were lower than 2% and the time to reach their best solution was approximately 200 seconds. Atta et al. (2018) also proposed a GA to solve the MCLP instances suggested by Zarandi et al. (2011). A comparative analysis indicated that the GA introduced by Atta et al. (2018) outperformed the GA proposed by Zarandi et al. (2011) requiring less time to achieve the best solutions.

Máximo et al. (2017) introduced a two-phase metaheuristic called Intelligent-Guided Adaptive Search (IGAS). The first phase of IGAS is known as a construction phase and the second one, a local search phase. The construction phase employs a self-organizing map neural network to map past iteration solutions. Therefore, besides a semi-greedy construction phase, the method uses the information gathered by the neural network to build the solutions. Every solution obtained by the first phase of the method is refined by the local search phase. The authors introduced real large-scale datasets and extensive compu-

tational experiments indicate a better performances of IGAS over existing algorithms. A hybrid version of IGAS with PR was introduced by Máximo and Nascimento (2019). The method is called IGAS-PR and uses the current iteration solution and an elite solution to define the paths. The elite solution is chosen considering a probability distribution that benefits the best quality solutions. The results indicate that IGAS-PR yields significantly better results than IGAS.

3. AILS-PR

AILS is a hybrid meta-heuristic that has achieved outstanding results in solving vehicle routing problems (Máximo and Nascimento, 2021; Máximo et al., 2022). The main feature of this method is its ability of self-adapting to control the solution diversity during the search. AILS-PR is a hybridization of AILS with Path Relinking (PR).

An iteration of AILS has two main stages: the perturbation and the local search. These steps are iteratively applied to a reference solution until a stopping criterion is met. At each iteration, the solution produced by the local search is evaluated by considering the acceptance criterion in order to update the reference solution. AILS controls its diversity through the degree of perturbation and the acceptance criterion.

Algorithm 1 describes AILS-PR applied to the MCLP. This algorithm has virtually the same steps as the one proposed by Máximo and Nascimento (2021). However, this version of AILS-PR does not require a feasibility step. Initially, the construction of an initial solution s is performed by a strategy that randomly chooses p facilities. Then, the local search improves the quality of s, and the reference solution s^r and the best solution found so far, s^*, are updated. The first step of the main loop refers to the choice of the perturbation heuristic, which is applied to the reference solution s^r. This step is described in Section 3.1. The solution resulting from the perturbation goes to the local search procedure. More details on this algorithm are presented in Section 3.2. After the local search, the updating of the perturbation degree occurs as explained in Section 3.3. The method keeps the elite set $\mathscr{E}$ with the best solutions found during the search process, as described in Section 3.4. The acceptance criterion then decides whether the current solution s will replace the reference solution s^r. This criterion is part of the diversity control of the method. More details on the acceptance criterion can be found in Section 3.5. The role of PR is to intensify the search process by exploring the region between two solutions. The employed PR is described in Section 3.6.

Dynamic updating requires less computational effort for the implementation of metaheuristics. For this reason, we use some variables that are kept updated throughout the search process, which makes

Algorithm 1: AILS-PR

Data: Instance data
Result: The best solution found s^*
$s \leftarrow$ Construct an initial solution
s^r, $s^* \leftarrow$ Local Search(s)
repeat
  Choose at random between removal heuristics $\mathscr{R}_1$ and $\mathscr{R}_2$ and assign to $\mathscr{R}_k$
  $s \leftarrow$ Perturbation (s^r, $\mathscr{R}_k$)
  $s \leftarrow$ Local Search(s)
  Update the diversity control parameter $\omega_{\mathscr{R}_k}$ considering the distance between s and s^r
  Update the elite set $\mathscr{E}$
  $s^r \leftarrow$ Apply acceptation criterion to s
  Update the acceptance criterion
  $s^b \leftarrow$ Path Relinking (s)
  Update the best solution s^* with solution s or s^b if applicable
until stopping criterion is met;

it easier to consult them whenever necessary. The main variables used in this paper are the following, where s is a feasible MCLP solution:

- t_i^s: indicates the number of facilities that cover vertex i;
- b_i^s: indicates the gain that would result from inserting a facility at vertex i and is calculated in the following way: $b_i^s = \sum_{j \in S_i \mid t_j^s = 0} d_j$, where S_i is the set that contains all vertices within the coverage radius of vertex i, $\forall\, i \in N$;
- f^s: indicates the objective function value of solution s;
- P^s: is the set of all vertices that are facilities in solution s, i.e., $|P^s| = p$;
- C^s: is the set of all non-facility vertices according to solution s.

The dynamic updating of these values occurs during the addition and removal of facilities in the solution.

3.1. Perturbation

The perturbation step changes the reference solution to produce a potentially different solution s. In our case, the strategy iteratively removes a facility and replaces it by a different one. This process repeats ω_k times, i.e., until ω_k have been modified from the current solution. The facilities to be removed are decided by two removal heuristics, $\mathscr{R}_1$ and $\mathscr{R}_2$, described next.

- **Concentric removal, $\mathscr{R}_1$:** A facility $x \in M$ is chosen at random. Then, the closest $\omega_{\mathscr{R}_1}$ facilities to x are removed. The removal process follows the order of closeness to facility x.
- **Random removal, $\mathscr{R}_2$:** This heuristic consists of randomly removing $\omega_{\mathscr{R}_2}$ facilities.

The insertion of the new facilities occurs according to two insertion heuristics, $\mathscr{A}_1$ and $\mathscr{A}_2$, as follows:

- **Concentric insertion, $\mathscr{A}_1$:** This heuristic randomly picks a facility within the coverage radius of the removed facility.
- **Random insertion, $\mathscr{A}_2$:** In this heuristic, the facilities to be inserted are chosen at random.

3.2. Local search

The local search is an improvement strategy that enhances the quality of a given solution to attain a local optimum. Algorithm 2 shows a pseudo-code of the employed strategy.

The neighborhood investigated in this paper consists of changing the position of a given facility. The neighbor strategy is an intermediate between the best and first improvement methods. On the one hand, the choice of the facilities to be removed is random. On the other hand, the nodes with the best benefit $b^s_{\hat{j}}$ are selected to be the new facilities. The main loop of the local search routine ensures that the search continues until it reaches a local optimum. The second loop evaluates if a given facility i in solution s is better being replaced by the facility $\hat{j}$ that would provide the best benefit, i.e. $b^s_{\hat{j}} > b^s_i$.

3.3. Perturbation degree updating

Controlling the perturbation degree is an important step in perturbation-based metaheuristics. In the proposed AILS, it indicates the number of facilities to be removed from the solution in the perturbation process. Small perturbation degree values promote a more intense search around the reference solutions. However, if it is too small, the method may get stuck a local optimum. High perturbation degree values promote a more exploratory search around the solutions. Nevertheless, being too large may randomly drive the method to arbitrary regions of the search space, thus failing in their adequate exploitation.

In the proposed AILS, the tuning of this parameter depends on the removal heuristic being used. Therefore, the perturbation degree of a given heuristic $\mathscr{R}_k$ is here referred to as $\omega_{\mathscr{R}_k}$, where $k = 1$ or 2. First, let $d_{\mathscr{R}_k}$ be the average distance between the reference solution s^r and the solution obtained by the local search applied to s^r modified by the removal heuristic $\mathscr{R}_k$. The method for adjusting $\omega_{\mathscr{R}_k}$ consists in relating $d_{\mathscr{R}_k}$ and d_β. Thus, at every γ iterations of $\mathscr{R}_k$, we set $\omega_{\mathscr{R}_k} = \min\{p, \max\{1, \lfloor \omega_{\mathscr{R}_k} d_\beta / d_{\mathscr{R}_k} \rfloor\}\}$.

Algorithm 2: Local Search

```
Data: Solution s
Result: The obtained local optimum s
repeat
    improved ← false
    foreach i ∈ M do
        Remove from s the facility i
        ĵ ← arg max_{j∈C^s} b^s_j
        if b^s_ĵ > b^s_i then
            Add the facility ĵ to solution s
            improved ← true
        end
        else
            Add the facility i to solution s
        end
    end
until improved;
return s
```

This strategy is the same as the one suggested by Máximo and Nascimento (2021).

3.4. Updating of the elite set $\mathscr{E}$

Here, $\mathscr{E}$ stores the best solutions found during the search process. The PR strategy uses this set, which must have at most σ solutions. The minimum pairwise distance between solutions from $\mathscr{E}$ is d_β. This requirement is to guarantee a certain diversity among the solutions. Thus, updating this set aims to guarantee this minimum distance. The procedure is very similar to the one presented by Máximo and Nascimento (2021). For a solution s to be part of the elite set $\mathscr{E}$, it must meet the two requirements $\mathscr{Q}_1$ and $\mathscr{Q}_2$ described as follows:

- **Requirement of Quality $\mathscr{Q}_1$:** A solution s will be added to the elite set $\mathscr{E}$ if $f^s > f^{s'}\ \forall\ s' \in \mathscr{E}$ such that $d(s, s') < d_\beta$.
- **Improvement requirement $\mathscr{Q}_2$:** If $|\mathscr{E}| = \sigma$, the quality of solution s must be higher than or equal to the quality of the worst solution in $\mathscr{E}$, i.e., $f^s \geq \min_{s' \in \mathscr{E}} f^{s'}$.

If solution s meets the requirements $\mathscr{Q}_1$ and $\mathscr{Q}_2$, then it will be part of the elite set. Consequently, every solution in W^s will be removed from $\mathscr{E}$, where $W^s = \{s' \in \mathscr{E} \mid f(s') < f(s) \text{ and } d(s, s') < d_\beta\}$. If

$W^s = \emptyset$ and $|\mathscr{E}| = \sigma$, then the worst solution from the elite set will be removed.

3.5. Acceptance criterion

The acceptance criterion establishes whether the solution obtained in a given iteration will replace the current reference solution. On the one hand, if the acceptance criterion is too soft, the method tends to be excessively diverse. On the other hand, if the acceptance criterion is too strict, the algorithm may have difficulties traversing the search space. Therefore, the adequate control of the acceptance criterion allows the algorithm to have a better performance.

A convergent criterion is a good option for those interested in being more exploratory in the first iterations of the search process to then exploit more promising regions in later iterations. In line with this, the strategy we employ here starts with a more relaxed criterion to accept new reference solutions. Then, as the algorithm runs, this criterion gradually becomes more restrictive.

This criterion consists of calculating a threshold $\bar{b}$ that will not accept solutions with a quality lower than this threshold. The threshold $\bar{b}$ is calculated by $\bar{b} = \underline{f} + (1 - \eta)(\bar{f} - \underline{f})$, where:

- $\underline{f}$: represents the average quality of the solutions found by the local search. This average is dynamically calculated and each new solution value has a weight of $1/\gamma$;
- $\bar{f}$: represents the quality of the best solution obtained in the last γ iterations of the algorithm;
- η: auxiliary parameter to adjust the value of $\bar{b} \in [\underline{f}, \bar{f}]$. If $\eta = 0$, then $\bar{b} = \bar{f}$, that is, a very restrictive criterion. If $\eta = 1$, then a more relaxed criterion is employed, and all solutions that have a quality higher than $\underline{f}$ will be accepted.

To employ the convergent behavior we initialize the value of η to 1 and at each iteration of the algorithm we decrease the value of $\eta \leftarrow \eta\alpha$. The value of $\alpha \in\ ]0, 1[$ is adjusted so that the value of η starts at 1 and ends at a minimum value $\underline{\eta}$. The value of α is adjusted every γ iterations of the algorithm as $\alpha \leftarrow \underline{\eta}^{t_a/it\ t_f}$ where:

- t_a: represents the CPU-time of the algorithm;
- t_f: represents the total CPU time of the method. This value is fixed at the beginning of the search process;
- it: represents the number of iterations performed so far.

The value of $\underline{\eta}$ is a parameter of AILS.

3.6. Path Relinking

The Path Relinking (PR) strategy, proposed by Glover (1997), consists in searching the neighborhood between a pair of reference solutions called initial and guide solutions to find better quality intermediate solutions. PR virtually consists of promoting changes in an initial solution so that it approaches a guide solution. The solutions found in the path to reach the guide solution are evaluated and those presenting a higher quality than the reference solutions are stored. Algorithm 3 describes the PR used in this paper, which receives as input the current solution s and the elite set $\mathscr{E}$. The initial solution is referred to as i, and the guide solution as g. In the proposed strategy, there are three possible ways to define i and g:

- $i = s$ and $g \in \mathscr{E}$;
- $i \in \mathscr{E}$ and $g = s$;
- $i, g \in \mathscr{E}$.

Every elite solution is randomly selected from the elite set. After defining the solutions i and g, the algorithm initializes the current solution c and the best solution s^b found by PR. Then, the algorithm updates the following sets:

- $O^c = P^i \setminus P^g$, which contains the facilities in solution i but not present in g. The elements in O^c are the facilities that will leave the current solution c;
- $E^c = P^g \setminus P^i$ is the set of the facilities that are in g but not in i. These facilities are the ones that will be added to solution c;
- T is the set that contains the adjacency movements represented by the pairs of facilities $(o, \hat{e})$ where $o \in O^c$ and $\hat{e} = \arg\max_{e \in E^c} \Delta^c_{(o,e)}$. $\Delta^c_{(o,e)}$ represents the cost of removing facility o and adding facility e to solution c and is calculated as follows:

$$\Delta^c_{(o,e)} = b^c_e - \sum_{j \in S_o \mid r^c_j = 1 \text{ and } j \notin S_e} d_j.$$

Then, the steps to browse the neighborhood between the initial and guide solutions are performed. The best moves to reach the guide solution are chosen and applied to the current solution c. If c has a better quality than solution g and s^b, then solution s^b is updated. At the end of the path, s^b will join the elite set.

Algorithm 3: Path-Relinking

Data: Solution s and elite set $\mathscr{E}$
Result: The best solution found s^b

Let i be the initial solution and g be the guide solution.
$c, s^b \leftarrow i$
$O^c \leftarrow P^i \setminus P^g$
$E^c \leftarrow P^g \setminus P^i$
$T \leftarrow \emptyset$
foreach $o \in O^c$ **do**
    $T \leftarrow T \cup \{(o, \hat{e})\}$, where $\hat{e} = \arg\max_{e \in E^c} \Delta^c_{(o,e)}$
    $E^c \leftarrow E^c \setminus \{\hat{e}\}$
end
while $|T| > 0$ **do**
    Find $\widehat{(o,e)} = \arg\max_{(o,e) \in T} \Delta^c_{(o,e)}$
    Remove facility o and add facility e to solution c
    $T \leftarrow T \setminus \{\widehat{(o,e)}\}$
    if $f^c > f^g$ *and* $f^c > f^{s^b}$ **then**
        $s^b \leftarrow c$
    end
end
Update the set $\mathscr{E}$ with solution s^b
return s^b

4. Computational Experiments

Experiments were carried out to evaluate the performance of AILS-PR. Also, a comparative analysis between AILS-PR and IGAS-PR (Máximo and Nascimento, 2019) was performed. IGAS-PR was shown to outperform the existing metaheuristics and has been considered the best MCLP metaheuristic to tackle large-scale instances so far. The parameters used in IGAS-PR are the same as those suggested in Máximo and Nascimento (2019). Moreover, we report the results of experiments where we try to find the optimal solution using CPLEX 12.6 considering the formulation proposed by Church and ReVelle (1974). Both IGAS-PR and AILS-PR were programmed in Java. All experiments were run on a cluster with 104 nodes, each node with 2 Intel Xeon E5-2680v2 processors running at 2.8 GHz, 10 cores and 128 GB DDR3 1866 MHz of RAM.

The solution methods were tested on the set of instances compiled by Máximo et al. (2017). The dataset consists of information regarding American population demographic density according to the 2010 census. There are data concerning the Manhattan, Bronx, San Francisco and Kings counties. Each

vertex of the data corresponds to the geographic location of a block of streets of the given county. Moreover, the demand of each point refers to the number of citizens living in the corresponding region. For each county, Máximo et al. (2017) suggested 6 different values of p totaling 24 test cases. Table 2 summarizes the characteristics of the instances tested in the experiments.

Table 2
The main characteristics of the tested instances.

County	m	r	p
Manhattan	2713	400	50, 60, 70, 80, 90, 100
Bronx	3839	600	50, 60, 70, 80, 90, 100
San Francisco	5137	600	50, 60, 70, 80, 90, 100
Kings	7730	800	50, 60, 70, 80, 90, 100

4.1. Parameter Tuning

In Section 3.4, the parameter d_β was introduced to define the minimum pairwise distance between solutions in the elite set. This and other parameters have to be fine tuned for a better performance of the proposed algorithm. The parameters to be adjusted are:

- d_β: parameter used to indicated both the minimum pairwise distance between solutions from the elite set and in the perturbation degree control updating;
- $\underline{\eta}$: lower bound on η, as described in Section 3.5;
- γ: parameter that indicates the periodicity of updating the diversity control degree, as presented in Section 3.5;
- σ: indicates the maximum cardinality of the elite set $\mathscr{E}$.

Let gap be calculated as in Equation (1), where Avg is the average solution of a given number of runs and BKS is the objective function value of the best known solution for the corresponding instance:

$$gap = 100(BKS - Avg)/BKS. \tag{1}$$

The design of the experiments for the parameter configuration followed a greedy iterative procedure. The Avg is calculated considering 30 independent executions. The mean Avg on the 24 instances is evaluated and referred to as $\overline{Avg}$. The employed stopping criterion was a time limit of 300 seconds.

Step 1: Define the current parameter configuration with the following values: $d_\beta = 10$, $\underline{\eta} = 0.01$, $\gamma = 30$ and $\sigma = 40$. These values were chosen empirically and by relying on the literature.

Step 2: Run AILS-PR with the current configuration and vary $d_\beta \in \{4, 7, 10, 13, 16, 19, 22\}$. Update d_β if the value that provides the best $\overline{Avg}$ is different from the current parameter value.

Step 3: Run AILS-PR with the current parameter configuration and vary $\underline{\eta} \in \{0.01, 0.1, 0.2, 0.3, 0., 0.5, 0.6\}$. Update $\underline{\eta}$ if the value that provides the best $\overline{Avg}$ is different from the current parameter value.

Step 4: Run AILS-PR with the current configuration and vary $\gamma \in \{10, 20, 30, 40, 50\}$. Update γ if the value that provides the best $\overline{Avg}$ is different from the current parameter value.

Step 5: Run AILS-PR with the current parameter configuration and vary $\sigma \in \{4, 7, 10, 13, 16, 19, 22\}$. Update σ if the value that provides the best $\overline{Avg}$ is different from the current parameter value.

Step 6: Repeat Steps 2 to 5 until no parameter updating happens.

Figure 1 shows the last iteration of the parameter configuration. It is possible to observe that d_β and $\underline{\eta}$ are the most sensitive parameters, since the AILS-PR performance has a significant change with the different tested values. Table 3 shows the parameter values obtained with this tuning procedure.

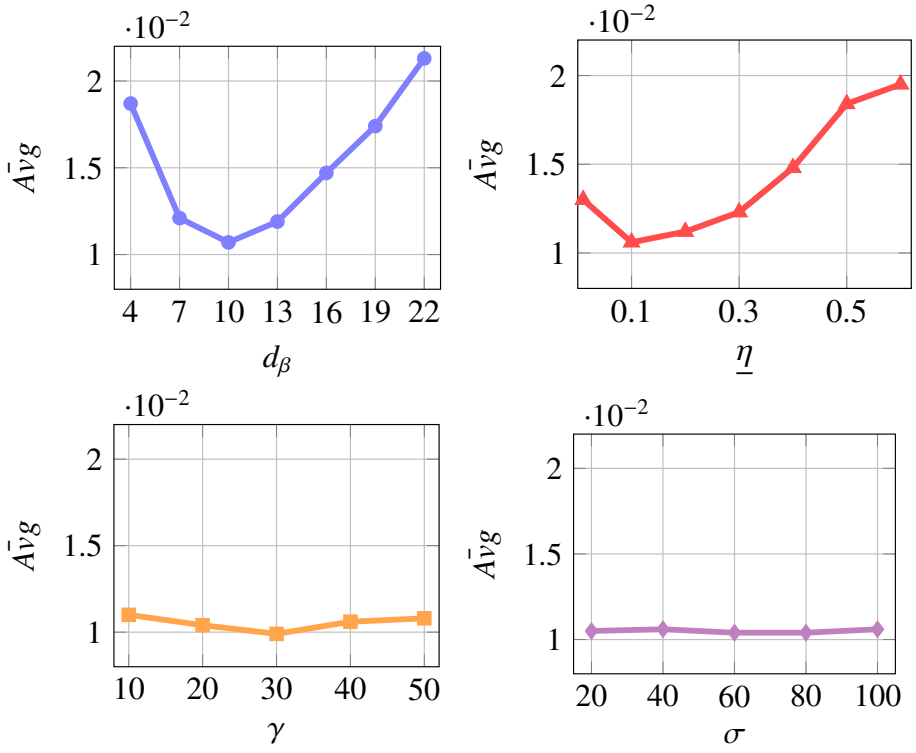

Fig. 1. Final result obtained in the configuration of the parameters.

Table 3
Values employed in AILS-PR after the calibration of parameters.

Parameter	Type	Interval	Value
d_β	Integer	[4, 22]	10
$\underline{\eta}$	Real	[0.01,0.6]	0.1
σ	Integer	[20, 100]	40
γ	Integer	[10, 50]	30

4.2. Experiment I

This section presents the results of the comparative analysis between IGAS-PR, AILS and AILS-PR. The parameter tuning of AILS also followed the procedure described in the earlier section and the best configuration was $d_\beta = 10$, $\underline{\eta} = 0.01$ and $\gamma = 30$.

Table 4 shows a summary of the results, indicating the *BKS* of each instance as well as the performance of IGAS-PR and AILS-PR by displaying the average solution values (*Avg*), the best of the 30 independent executions (Best) and average time in seconds that the algorithm took to find the final solution in each run. The stopping criterion of both IGAS-PR, AILS and AILS-PR was a time of 300 seconds. If a *BKS* is highlighted in bold, it means that the reported value corresponds to an optimal solution value. If it is marked with an asterisk, it means that AILS-PR improved this *BKS*.

Table 4
Results of the computational experiments of the metaheuristics on the tested instances.

			IGAS-PR			AILS			AILS-PR		
Instance	*p*	*BKS*	*Avg* (*gap*)	Best	T (s)	*Avg* (*gap*)	Best	T (s)	*Avg* (*gap*)	Best	T (s)
Manhattan	50	**1153640**	**1153640.00 (0.0000)**	**1153640**	4.3	**1153640.00 (0.0000)**	**1153640**	70.9	**1153640.00 (0.0000)**	**1153640**	**2.4**
	60	**1288174**	1288108.60 (0.0051)	**1288174**	85.8	**1288174.00 (0.0000)**	**1288174**	182.0	**1288174.00 (0.0000)**	**1288174**	**22.1**
	70	**1396333**	**1396333.00 (0.0000)**	**1396333**	27.2	**1396333.00 (0.0000)**	**1396333**	185.5	**1396333.00 (0.0000)**	**1396333**	**12.0**
	80	**1481434**	1481392.13 (0.0028)	**1481434**	120.3	1481433.90 (0.0000)	**1481434**	195.8	**1481434.00 (0.0000)**	**1481434**	**49.4**
	90	**1539992**	1539899.13 (0.0060)	**1539992**	108.5	1539843.23 (0.0097)	**1539992**	196.3	**1539989.97 (0.0001)**	**1539992**	**81.8**
	100	1574256	1573536.40 (0.0457)	1574142	**156.8**	1573854.80 (0.0255)	1574225	211.5	**1574142.10 (0.0072)**	**1574229**	219.8
Bronx	50	**1205051**	1205019.27 (0.0026)	**1205051**	57.5	**1205051.00 (0.0000)**	**1205051**	121.1	**1205051.00 (0.0000)**	**1205051**	**48.7**
	60	**1290635**	1290393.77 (0.0187)	**1290635**	134.4	1290451.90 (0.0142)	**1290635**	160.9	**1290616.80 (0.0014)**	**1290635**	**102.5**
	70	1348232	1348031.20 (0.0149)	**1348232**	134.4	1348177.87 (0.0040)	**1348232**	231.0	**1348231.73 (0.0000)**	**1348232**	**77.6**
	80	1376061	1375514.33 (0.0397)	1375729	**167.5**	1375395.03 (0.0484)	**1376061**	239.2	**1375669.07 (0.0285)**	**1376061**	196.1
	90	1384376*	1384165.77 (0.0152)	1384309	**184.0**	1384158.17 (0.0157)	1384314	247.9	**1384268.43 (0.0078)**	**1384364**	228.8
	100	1385099*	1385074.30 (0.0018)	**1385091**	**106.4**	1385074.33 (0.0018)	**1385091**	227.6	**1385075.73 (0.0017)**	**1385091**	252.1
San Francisco	50	**617592**	**617592.00 (0.0000)**	**617592**	40.6	617589.20 (0.0005)	**617592**	89.9	**617592.00 (0.0000)**	**617592**	**11.5**
	60	683139	682947.63 (0.0280)	**683139**	137.1	683073.10 (0.0096)	**683139**	129.8	**683139.00 (0.0000)**	**683139**	**72.4**
	70	733772	733512.53 (0.0354)	733727	188.6	733699.53 (0.0099)	**733772**	214.0	**733769.00 (0.0004)**	**733772**	**145.8**
	80	769781	769394.67 (0.0502)	**769781**	161.9	769540.13 (0.0313)	**769781**	225.5	**769757.77 (0.0030)**	**769781**	**154.2**
	90	791562	790529.93 (0.1304)	791279	**171.3**	791085.60 (0.0602)	**791562**	230.8	**791276.37 (0.0361)**	**791562**	175.6
	100	802307*	801448.57 (0.1070)	801977	**200.0**	801733.57 (0.0715)	802107	249.6	**801845.93 (0.0575)**	**802211**	260.0
Kings	50	2022077	2021854.60 (0.0110)	**2022077**	127.7	2021953.73 (0.0061)	**2022077**	195.8	**2022077.00 (0.0000)**	**2022077**	**99.5**
	60	2227025	2225965.07 (0.0476)	**2227025**	**153.2**	2226516.20 (0.0228)	**2227025**	203.0	**2226850.97 (0.0078)**	**2227025**	163.2
	70	2380660	2378909.13 (0.0735)	**2380660**	192.3	2380016.87 (0.0270)	**2380660**	178.0	**2380502.47 (0.0066)**	**2380660**	**173.0**
	80	2467701	2465524.33 (0.0882)	2467488	202.8	2466653.67 (0.0424)	2467555	230.9	**2467311.57 (0.0158)**	**2467701**	**166.1**
	90	2502669*	2500467.33 (0.0880)	2501706	**199.1**	2501248.70 (0.0568)	2502098	263.0	**2501635.90 (0.0413)**	**2502222**	230.6
	100	**2504700**	2504688.00 (0.0005)	**2504700**	**163.9**	2504690.40 (0.0004)	**2504700**	262.2	**2504692.47 (0.0003)**	**2504700**	202.7
Average			0.0338		134.4	0.0191		197.6	0.0090		**131.2**

It is possible to observe in Table 4 that AILS-PR outperformed IGAS-PR on all instances. Moreover,

its average solution was the same as the BKS on nine instances and the highest gap was 0.0413 on Kings instances with $p = 90$. On the other hand, on three instances the average solutions obtained by IGAS-PR were the same as the BKS. Still, considering all runs, AILS-PR obtained the BKS in 20 out of the 24 instances, whereas IGAS-PR achieved the BKS on 15 instances. On all the 6 BKS proven to be optimal solutions, the average solution found by AILS-PR was the same as the BKS. The average solution obtained by IGAS-PR was the same as the BKS on three out of these six instances. Finally, AILS-PR achieved a better average time to reach the final solution than IGAS-PR and presented a better mean average gap.

For a better analysis of the results, we also present in Figure 2 the performance profiles of Dolan and Moré (2002), considering the average gaps achieved by IGAS-PR and AILS-PR. Roughly, these performance profiles give the relation between the proportion of instances for which a given algorithm outperformed (y-axis) considering the average gap multiplied by the exponent of the value on x-axis.

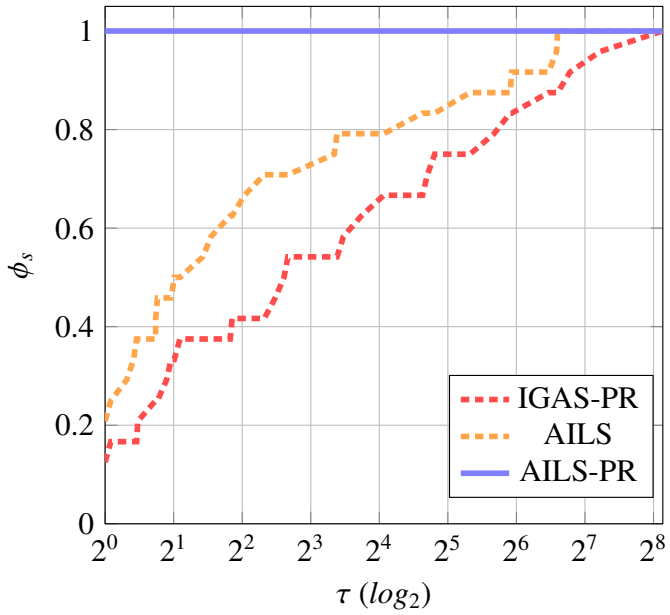

Fig. 2. Performance profiles (Dolan and Moré, 2002) comparing the average gaps obtained by the IGAS-PR and AILS-PR.

According to Figure 2, in addition to the fact that AILS-PR has achieved the best solutions in 100% of the instances, for over a half of the instances its gaps were twice better than IGAS-PR's.

4.3. Experiment II

This section presents the performance of CPLEX on the tested instances. The instances for which the reported gap is higher than 0 refer to those where CPLEX ran out of memory during execution. The table shows the *gap* to the *BKS*, the *gap* to the best lower bound provided by CPLEX (gap_o), the time to achieve the best upper bound (column "Time (s)") and the time required for the solver to prove its optimality (column "Total Time (s)").

Table 5
Performance of CPLEX on the tested instances.

			CPLEX				
Instance	p	*BKS*	Solution	*gap*	gap_o	Time (s)	Total Time (s)
Manhattan	50	**1153640***	**1153640**	0.0000	0.00	6.32	6.52
	60	**1288174***	**1288174**	0.0000	0.0	7.68	7.79
	70	**1396333***	**1396333**	0.0000	0.0	19.26	37.69
	80	**1481434***	**1481434**	0.0000	0.0	27.30	384.08
	90	**1539992***	**1539992**	0.0000	0.00	63889.95	257051.22
	100	1574256	1574230	0.0017	0.11	79149.39	711798.41
Bronx	50	**1205051***	**1205051**	0.0000	0.00	1568.85	6009.70
	60	**1290635***	**1290635**	0.0000	0.00	120202.08	678993.29
	70	1348232	**1348232**	0.0000	0.10	739239.35	1202077.97
	80	1376061	**1376061**	0.0000	0.07	345501.67	1207623.39
	90	1384376	1384364	0.0009	0.03	362846.58	990398.93
	100	1385099	1385095	0.0003	0.00	2434.02	1206201.02
San Francisco	50	**617592***	**617592**	0.0000	0.00	6700.64	342336.56
	60	683139	682766	0.0546	0.46	208810.46	1000000.70
	70	733772	731978	0.2445	0.90	190427.66	1207301.72
	80	769781	768918	0.1121	0.83	931266.53	1202010.90
	90	791562	791243	0.0403	0.72	24175.29	1203371.75
	100	802307	801334	0.1213	0.40	292615.72	1208496.56
Kings	50	2022077	**2022077**	0.0000	0.22	168534.68	1165544.78
	60	2227025	2222485	0.2039	1.31	965197.78	1198873.90
	70	2380660	2375431	0.2196	1.34	617943.58	1207128.38
	80	2467701	2456592	0.4502	1.31	500311.72	1202041.53
	90	2502669	2498931	0.1494	0.23	362584.87	1190689.55
	100	**2504700***	**2504700**	0.0000	0.00	70923.39	70935.31
Average				0.0666			

Table 5 demonstrates that CPLEX could prove the optimality of 12 out of the 24 tested instances, 9 from Manhattan and Bronx instances, which are the instances with the fewest points. To achieve the best upper bound on San Francisco and Kings instances, CPLEX required more than two days on most instances.

5. Final Remarks

This paper presented a study of a recently proposed metaheuristic called AILS that so far has been applied to routing problems with excellent results. The target problem of this paper is a classical location problem, known as the Maximal Covering Location Problem (MCLP). The state-of-the-art solution method for large-scale instances of this problem is a hybrid metaheuristic that integrates learning, intensification and exploration. As AILS also considers this triad, it has a clear potential of achieving good results for this problem.

To further explore the intensification aspect, an AILS-PR hybrid is proposed. Computational experiments confirmed the better performance of AILS-PR in comparison to the existing metaheuristic. Moreover, this paper shows an experiment evaluating the performance of CPLEX on the tested large-scale instances, showing that it runs out of memory in most of them.

A future work direction is the investigation of AILS for integrated location-routing problems. As it presented an outstanding performance for variants of the individual problems, we expect that it achieves good results in their integration mainly on large-scale instances.

Acknowledgments

The authors are grateful for the financial support provided by CNPq (403735/2021-1) and FAPESP (2016/01860-1, 2019/22067-6); the Laboratory for Simulation and High Performance Computing (LaSCADo), funded by (FAPESP) through project 2010/50646-6; and the Center for Mathematical Sciences Applied to Industry (CeMEAI), funded by FAPESP by through the 2013/07375-0 project, for the availability of computer resources.

References

Adenso-Díaz, B., Rodríguez, F., 1997. A simple search heuristic for the MCLP: Application to the location of ambulance bases in a rural region. *Omega* 25, 2, 181–187.

Atta, S., Sinha Mahapatra, P.R., Mukhopadhyay, A., 2018. Solving maximal covering location problem using genetic algorithm with local refinement. *Soft Computing* 22, 12, 3891–3906.

Church, R., ReVelle, C., 1974. The maximal covering location problem. *Papers of the Regional Science Association* 32, 1, 101–118.

Cordeau, J.-F., Furini, F., Ljubić, I., 2019. Benders decomposition for very large scale partial set covering and maximal covering location problems. *European Journal of Operational Research* 275, 3, 882–896.

Dolan, E.D., Moré, J.J., 2002. Benchmarking optimization software with performance profiles. *Mathematical Programming* 91, 2, 201–213.

Downs, B.T., Camm, J.D., 1996. An exact algorithm for the maximal covering problem. *Naval Research Logistics (NRL)* 43, 3, 435–461.

Galvão, R.D., Espejo, L.G.A., Boffey, B., 2000. A comparison of Lagrangean and surrogate relaxations for the maximal covering location problem. *European Journal of Operational Research* 124, 2, 377 – 389.

Galvão, R.D., ReVelle, C., 1996. A Lagrangean heuristic for the maximal covering location problem. *European Journal of Operational Research* 88, 1, 114–123.

Garey, M.R., Johnson, D.S., 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA.

Glover, F., 1997. Tabu search and adaptive memory programming—advances, applications and challenges. In *Interfaces in Computer Science and Operations Research*. Springer, pp. 1–75.

Lorena, L.A., Pereira, M.A., 2002. A Lagrangean/surrogate heuristic for the maximal covering location problem using hillman's edition. *International Journal of Industrial Engineering* 9, 57–67.

Lourenço, H.R., Martin, O.C., Stützle, T., 2003. Iterated local search. In *Handbook of Metaheuristics*, *International Series in Operations Research and Management Science*. Vol. 57. Springer US, Boston, MA, pp. 320–353.

Máximo, V.R., Cordeau, J.-F., Nascimento, M.C.V., 2022. An adaptive iterated local search heuristic for the heterogeneous fleet vehicle routing problem. *Computers & Operations Research* 148, 105954.

Máximo, V.R., Nascimento, M.C.V., 2019. Intensification, learning and diversification in a hybrid metaheuristic: an efficient unification. *Journal of Heuristics* 25, 4, 539–564.

Moore, G.C., ReVelle, C., 1982. The hierarchical service location problem. *Management Science* 28, 7, 775–780.

Máximo, V.R., Nascimento, M.C.V., 2021. A hybrid adaptive iterated local search with diversification control to the capacitated vehicle routing problem. *European Journal of Operational Research* 294, 3, 1108–1119.

Máximo, V.R., Nascimento, M.C.V., Carvalho, A.C., 2017. Intelligent-guided adaptive search for the maximum covering location problem. *Computers & Operations Research* 78, 129 – 137.

Penna, P.H.V., Subramanian, A., Ochi, L.S., 2013. An iterated local search heuristic for the heterogeneous fleet vehicle routing problem. *Journal of Heuristics* 19, 2, 201–232.

Resende, M.G., 1998. Computing approximate solutions of the maximum covering problem with GRASP. *Journal of Heuristics* 4, 161–171.

ReVelle, C., Scholssberg, M., Williams, J., 2008. Solving the maximal covering location problem with heuristic concentration. *Computers & Operations Research* 35, 2, 427 – 435.

Subramanian, A., Battarra, M., Potts, C.N., 2014. An iterated local search heuristic for the single machine total weighted tardiness scheduling problem with sequence-dependent setup times. *International Journal of Production Research* 52, 9, 2729–2742.

Zarandi, M.F., Davari, S., Sisakht, S.H., 2011. The large scale maximal covering location problem. *Scientia Iranica* 18, 6, 1564 – 1570.