J Heuristics (2019) 25:539–564
https://doi.org/10.1007/s10732-018-9373-1

Intensification, learning and diversification in a hybrid metaheuristic: an efficient unification

Vinícius R. Máximo[1] · **Mariá C. V. Nascimento**[1]

Received: 13 June 2017 / Revised: 23 March 2018 / Accepted: 7 April 2018 /
Published online: 24 April 2018

Abstract Hybrid heuristic methods have lately been pointed out as an efficient approach to combinatorial optimization problems. The main reason behind this is that, by combining components from different metaheuristics, it is possible to explore solutions (which would be unreachable without hybridization) in the search space. In particular, evolutionary algorithms may get trapped into local optimum solutions due to the insufficient diversity of the solutions influencing the search process. This paper presents a hybridization of the recently proposed metaheuristic—intelligent-guided adaptive search (IGAS)—with the well-known path-relinking algorithm to solve large scale instances of the maximum covering location problem. Moreover, it proposes a slight change in IGAS that was tested through computational experiments and has shown improvement in its computational cost. Computational experiments also attested that the hybridized IGAS outperforms the results found in the literature.

Keywords Intelligent-guided adaptive search · Path-relinking · Maximum covering location problem · Large scale

1 Introduction

The maximal covering location problem (MCLP) aims at identifying the best sites to locate p facilities to cover customers' demands as much as possible. Moreover,

✉ Mariá C. V. Nascimento
mcv.nascimento@unifesp.br

Vinícius R. Máximo
vinymax10@gmail.com

[1] 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

each facility has a coverage ratio r establishing whether a given facility can meet the customer demand. A set of facilities $M = \{1, \ldots, m\}$ and customers $N = \{1, \ldots, n\}$ compose each instance of the problem as points in a cartesian coordinate. A facility can address the demand k_i of a customer i within its coverage ratio as long as it is open (one of the p chosen sites). The distance between a facility j and a customer i is here defined as d_{ij}. Let $Y_s = \{Y_s^1, Y_s^2 \ldots, Y_s^p\}$ be a set of variables that indicates which facilities are chosen in the solution s. MCLP can be formulated as follows:

$$\max f(s) = \sum_{i \in N} k_i \min\{1, |\{j : j \in Y_s,\ d_{ij} \leq r\}|\} \tag{1}$$

subject to

$$|Y_s| = p \tag{2}$$

Church and ReVelle (1974) first introduced MCLP that Garey and Johnson (1979) have later proved to be $\mathscr{NP}$-hard.

Several studies in the literature rely on solving benchmark MCLP instances (Galvão and ReVelle 1996; Galvão et al. 2000; Karasakal and Karasakal 2004; ReVelle et al. 2008; Jia et al. 2007; Resende 1998). However, only few studies handle large-scale applications properly which are, nowadays, the primary focus of study in data analysis. For example, locating facilities (central points) with million of points in communication networks is highly relevant. In particular, recent versions of commercial packages, such as CPLEX (2014), can solve instances with up to 2000 nodes in reasonable time. However, large-scale applications remain a challenge in MCLP.

Máximo et al. (2017) have recently introduced a metaheuristic named intelligent-guided adaptive search (IGAS), which solved MCLP problem instances with up to 7730 nodes. The authors compared the results achieved by IGAS with those found in the literature and with CPLEX limited to 20,000 s. It was verified that the proposed solution method was very efficient in solving large-scale MCLP instances outperforming the other methods.

IGAS is a recent heuristic and thus has not been widely explored yet. There are indications that its performance may be improved by hybridizing it with an intensification heuristic. Similarly to the largely studied greedy randomized search procedures metaheuristic (Feo and Resende 1989), this type of hybridization may speed up the convergence to a good quality solution, as suggested in Máximo et al. (2017). Likewise, hybridizations of GRASP (Feo and Resende 1989) with path-relinking (Glover 1996) are common in the literature, to develop a memory mechanism to GRASP that is iteration-independent.

Different from GRASP, IGAS has a memory mechanism in its original form. This is due to the influence of a topological neural network known as growing neural gas (GNG) (Fritzke 1995). As a consequence, if hybridized with IGAS, path-relinking would play the roles of intensification and diversification by combining features of the elite set of solutions.

The primary contribution of this paper is the enhancement in quality of the best MCLP solutions for large-scale instances, achieved by hybridizing IGAS with path-relinking. This paper also presents a detailed performance analysis to attest the efficiency of IGAS with path-relinking. Computational experiments indicate that IGAS with path-relinking outperformed IGAS showing outstanding results. Another contribution of this study is the refinement of solutions constructed by IGAS.

The remainder of this paper is organized as follows. Section 2 sets forth IGAS with path-relinking, named IGASPR, to solve the MCLP. Section 3 details the computational experiments carried out. To sum up, Sect. 4 provides insights into the main contributions of this paper and suggests future work directions.

2 IGAS with path-relinking (IGASPR)

IGAS was first proposed as an attempt to address the gap between memory and iteration independency in the widely studied GRASP metaheuristic (Resende and Werneck 2006). GRASP is a two-phase and multi-iterative algorithm. The first phase, known as construction phase, is incremental and semi-greedy, producing a feasible solution at the end. The second phase is an improvement step, to which each solution found in the first phase is submitted.

Máximo et al. (2017) have shown a comprehensive comparison between IGAS and GRASP demonstrating that IGAS clearly excells GRASP in performance when it comes to solving MCLP large-scale instances. The main difference between GRASP and IGAS is that the latter has a learning step that enables the storage of information from previous iterations. This happens by means of an unsupervised learning algorithm called growing neural gas (Fritzke 1995). GNG influences the construction phase, informing the features of the best solutions up to that iteration.

This paper presents an IGAS-path-relinking hybrid, referred to here as IGASPR. Through this solution method, path-relinking works in every iteration of the IGAS search process. Consequently, each IGASPR iteration may be divided into four parts: construction phase, local search phase, learning process and path-relinking search. All parts are repeated until a stop criterion is reached. The flowchart in Fig. 1 illustrates the proposed method.

In the beginning of an iteration, the solution construction algorithm incrementally searches for a feasible solution using accumulated information from the neural network. This means that there is no or low effect from the GNG network (learning) in the first iterations. After this, the solution is refined in the local search phase (in case it is not a local optimum already). The average solution value $\bar{f}$ is then updated, since it consists in the mean solution values obtained in the local search phase. Value f^* is the best solution found so far. Parameter $\bar{b} = \bar{f} + \eta(f^* - \bar{f})$ aims at restricting the solutions to be presented to the GNG network. Value $\bar{b}$ varies in the interval $[\bar{f}, f^*]$, as parameter $\eta \in [0, 1]$. Set $\mathscr{E}$ stores the best solutions found in previous iterations, the elite solutions.

2.1 Updating structure

Before describing each phase of IGASPR, this section discusses the impact of adding and/or removing a facility j on the objective function of a given solution, i.e., the sum of the demands that facility j exclusively meets in the solution.

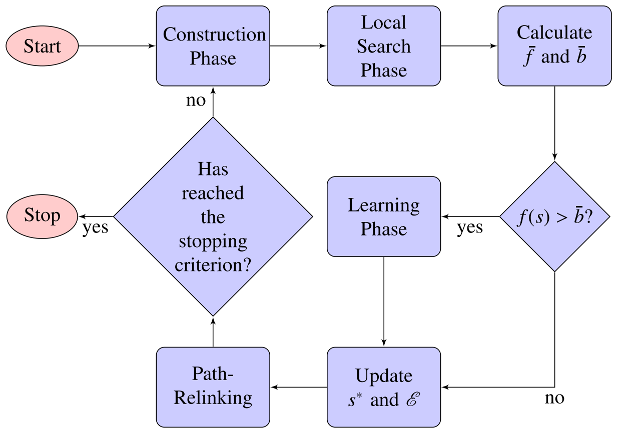

Fig. 1 IGAS with path-relinking (IGASPR)

Let set $C_j = \{i \in N \mid d_{ij} \leq r\}$ represent all clients within the coverage ratio of facility j. The impact of facility j on solution s can be calculated as presented in Eq. (3).

$$\Delta_j = \sum_{i \in C_j} k_i (1 - \min\{1, |\{l \in Y_s \backslash \{j\} \mid d_{il} \leq r\}|\}) \tag{3}$$

2.2 Construction phase

Algorithm 1 displays a pseudocode of the IGASPR construction phase. It can be roughly divided into two phases. The first phase consists in inserting $\lfloor \omega p \rfloor$ facilities following a semi-greedy criterion guided by information contained in the GNG network. The second phase refers to the insertion of $p - \lfloor \omega p \rfloor$ facilities at random in the partial solution.

The GNG network is an undirected graph. Then, let $G = (V, E)$ be such network. A neuron $v \in V(G)$, picked at random, guides the semi-greedy process. More specifically, every neuron has an associated vector of atributtes $w^v \in [0, 1]^m$. The element at each position l of w^v, w_l^v, represents the weight of facility l in that neuron. On the one hand, if $w_l^v \approx 1$, then it was verified that neuron v was mostly influenced by solutions that contained facility l. On the other hand, if $w_l^v \approx 0$, l did not belong to the majority of the solutions that trained node v.

The construction phase is a semi-greedy strategy that builds a solution by preferably choosing the elements that provide the largest priorities to the partial solution according to node v. Obviously, in the beginning of the routine the partial solution is empty. In line with this, the procedure iteratively updates the list of possible facilities to enter the partial solution, referred to as CL. In this list, the facilities are sorted in decreasing order of w^v. A restricted list of candidates (RCL) storing the best candidates (facilities) from CL contains the first φ elements of CL.

```
   Data: $\alpha$, $\varphi$, $\omega$, $G$
   Result: Feasible solution $s$
1  $Y_s = \emptyset$
2  Randomly choose a $v \in V(G)$
3  $CL \leftarrow M$
4  Sort $CL$ in decreasing order of $w_l^v$
5  for $j \leftarrow 1$ to $\lfloor \omega p \rfloor$ do
6      $RCL = \{l \in CL :$ position of $l$ in $CL \le \varphi\}$
7      Randomly choose $l \in RCL$ and make $Y_s \leftarrow Y_s \cup \{l\}$
8      $CL \leftarrow M \backslash Y_s$
9  end
10 Randomly add $p - \lfloor \omega p \rfloor$ facilities to $s$.
```

Algorithm 1: IGASPR Construction Phase

The procedure then randomly selects some facilities, without any influence from the GNG network to ensure diversity in the process of finding solutions. Parameter $\omega \in [0, 1]$ controls this diversification, as it is the percentage of facilities chosen by the influence of a node from the GNG network.

This construction phase differs from that proposed in Máximo et al. (2017) by line 10. In that case, IGAS selected the $p - \lfloor \omega p \rfloor$ facilities using another semi-greedy strategy. Accordingly, the remaining facilities j (candidates to be in solution s) were sorted in a list *CL* according to their aggregated values, Δ_j. Then, after randomly choosing one of the best $\alpha|CL|$ facilities, the aggregated costs were updated (in case the solution was not complete) and the process was repeated until it provided a feasible solution. This slight change in IGAS substantially simplified the construction phase and diminished its CPU-time. It is enough to observe that, in line 10, Algorithm 1 does not employ any sorting algorithm, different from the version of IGAS found in the literature.

Algorithm 1 returns a feasible solution and its complexity is $O(m \log m)$. This phase has an asymptotic complexity of $O(pmn)$ in the version found in the literature, demonstrating an advantage of the strategy presented here.

2.3 Local search phase

Local search (LS) consists of an intensification heuristic to identify local optimal solutions. Considering a neighborhood of solution s, $\mathcal{N}(s)$, Definition 1 states a local optimum.

Definition 1 In a maximization problem, a local optimum solution is a solution s in which, for every $s' \in \mathcal{N}(s)$, $f(s') \le f(s)$.

The probing process of LS, therefore, depends on the investigated neighborhood. Moreover, the systematics behind LS involves replacing a given solution s' by the best solution within its neighborhood (best improvement strategy), or by a solution better evaluated than s' (first improvement strategy). Then, by iteratively searching for a better solution, a local optimum is returned at some point (in case it exists). Otherwise, the problem has no local optimum and, consequently, no global optimum.

IGAS employs a 1-neighborhood LS method that has the following definition.

Definition 2 The 1-neighborhood of a solution s, $\mathcal{N}^1$(s), includes all solutions s' that for a distance metric D, $D(s', s) \leq 1$.

The distance metric between a pair of solutions s and s' here employed is the cardinality of the symmetric difference between sets Y_s and $Y_{s'}$ divided by 2. Therefore $D(s, s') = |Y_s \triangle Y_{s'}|/2$.

In line with this, given a solution s, the used LS will identify better valued solutions in the $\mathcal{N}^1$ here defined by simply changing a facility $l \in Y_s$ for another facility $j \in M \backslash Y_s$, such as $\Delta_j > \Delta_l$. This process is repeated until it is no longer possible to find a j and l that satisfy this criterion. It is worth mentioning that this method is similar to that used in Resende (1998). Algorithm 2 displays the first improvement LS method employed in the introduced metaheuristic.

```
Data: $s$
Result: $s^*$
1 $s^* \leftarrow s$
2 repeat
3     Find the first pair $(j, l)$ such that $j \in M\backslash Y_{s^*}$ and $l \in Y_{s^*}$ such that $\Delta_j > \Delta_l$
4     $Y_{s^*} \leftarrow Y_{s^*} \backslash \{l\} \cup \{j\}$
5 until $(j, l)$ is $\emptyset$;
```

Algorithm 2: Local Search.

The asymptotic complexity of LS is $O(pmn)$.

2.4 Learning phase

Machine learning algorithms based on topological mappings, such as self-organizing maps (Kohonen 1982) and GNG (Fritzke 1995), have been widely employed in solving unsupervised problems. The primary goal of these techniques is to extract meaningful subsampling of high dimensional feature spaces. In particular, GNG is a self-organizing network that uses unsupervised competitive learning to perform such task.

GNG is the result of a combination between competitive Hebbian learning (CHL) and growing cell structures (GCS). On the one hand, to define CHL, Martinetz (1993) introduced the concept of creating an edge between the two winning neurons of a given input. On the other hand, to define GCS, Fritzke (1994) determined a dynamic mechanism for creating neurons. Then, Fritzke (1995) introduced the GNG network, based on the so-called neural gas (NG) network (Martinetz and Schulten 1991), which starts with all neurons pre-defined. At the beginning, the GNG network is empty and so the neurons pop up in an iterative manner in the learning process.

Therefore, GNG is a dynamic neural network that self-adapts to the set of inputs. This characteristic for IGAS is highly relevant because the solutions are presented to the network as they are found along during the search process. During this process and guided by the inputs, the GNG network topology is formed and updated, thus intensifying the identification of regions with best solutions. The neural network is able to disregard irrelevant regions and migrate to those where the solutions are more promising.

It is worth mentioning that the GNG learning has a low computational cost and consequently does not degrade the performance of the solution method. This is one of the reasons for choosing the GNG network in IGAS. Moreover, its topology restricts disseminating unnecessary information through the network.

A graph $G = (V, E)$ represents the GNG network, where $V(G)$ is its set of vertices and $E(G)$ its set of edges. $\delta(v)$ indicates the neighborhood of a vertex $v \in V(G)$. Each neuron $v \in V(G)$ has an m-dimensional vector of weights w^v that maps its set of features as well as its accumulated error, a scalar θ^v. An edge $\{u, v\} \in E(G)$ is aged $A_{\{u,v\}}$ and this attribute is used to update the network in the learning process.

G starts with a single neuron v, and the values for w_j^v, $\forall j \in M$, are chosen at random within the interval [0, 1] and its accumulated error initiates with a null value.

The learning process consists in updating the network with the information contained in the input data Y_s. To perform this update, one must find the two closest neurons to the input data. This is a criterion from the competitive learning. Function ρ [Eq. (4)] calculates the proximity of data input Y_s and a neuron v.

$$\rho(s, v) = \sum_{j \in Y_s} \left(1 - w_j^v\right)^2 + \sum_{j \in M \setminus Y_s} \left(w_j^v\right)^2 \tag{4}$$

The closest neuron to input Y_s is here called v_1 and the second nearest neuron v_2. After identifying these neurons, if edge $\{v_1, v_2\}$ does not belong to $E(G)$ and v_2 exists, the algorithm adds an edge $\{v_1, v_2\}$ to the set $E(G)$ assigning null to its age, i.e., $A_{\{v_1,v_2\}} = 0$. In case $\{v_1, v_2\}$ already exists, its age restarts, i.e., it receives a null value. This edge indicates that there exists an input in this part of the graph. Then, the algorithm moves neuron v_1 and its neighborhood toward the input data. The variation on v_1 must be greater than on its neighborhood. For this reason, the update rates of the winner neuron and its neighborhood are represented by ε_b and ε_n, respectively. The update is indicated in Eqs. (5)–(8):

$$w_j^{v_1} = \varepsilon_b \left(1 - w_j^{v_1}\right) \qquad \forall j \in Y_s \tag{5}$$

$$w_j^{v_1} = \varepsilon_b - w_j^{v_1} \qquad \forall j \in M \setminus Y_s \tag{6}$$

$$w_j^{v'} = \varepsilon_n \left(1 - w_j^{v'}\right) \qquad \forall v' \in \delta(v_1), \forall j \in Y_s \tag{7}$$

$$w_j^{v'} = \varepsilon_n - w_j^{v'} \qquad \forall v' \in \delta(v_1), \forall j \in M \setminus Y_s \tag{8}$$

The primary feature of the GNG network is its growth over time. To implement this behavior, first it is necessary to find the appropriate place to insert a new neuron in the network. This place is where the algorithm introduced a significant amount of data and where the region is represented by a low number of neurons. The GNG network keeps track of such regions by accumulating, in each neuron, the sum of the distances between the input data for which the neuron was the winner—the accumulated error.

$$\theta^{v_1} = \theta^{v_1} + \rho(s, v_1) \tag{9}$$

According to the GNG learning, old errors must not stand out in relation to new ones. For this reason, in every iteration, the algorithm diminishes the errors of the nodes by a factor β. Equation (10) shows how this update works.

$$\theta^v = \beta\theta^v \quad \forall v \in V(G) \tag{10}$$

In order for the network to have a dynamic behavior, it is necessary to disregard the regions where no data have been presented for a number of iterations. The GNG network has an aging process on the edges. It works by incrementing the age of the edges incident to the winner neuron v_1 by one, as indicated in Eq. (11).

$$A_{\{v_1,v'\}} = A_{\{v_1,v'\}} + 1, \quad \forall\, v' \in \delta(v_1) \tag{11}$$

In this way, the learning algorithm removes all edges with ages exceeding a threshold, set to a_{max}. This step plays the role of the migration process. The process may result in disconnected neurons, if the neurons are exclusively incident to the removed edges. In this case, the algorithm also removes the corresponding neurons.

During the learning process, the insertion of new neurons and edges occurs periodically. However, the number of nodes in the network cannot exceed the imposed limit of max_G. At each λ iterations, as long as it is possible, the growing phase adds a new neuron $\hat{v}$ in the network. The ideal position for inserting $\hat{v}$ is between neurons $v_e = \arg\max_{v \in V(G)} \theta^v$ and $v'_e = \arg\max_{v' \in \delta(v_e)} \theta^{v'}$. Then, the procedure removes edge $\{v_e, v'_e\}$ and inserts edges $\{\hat{v}, v_e\}$ and $\{\hat{v}, v'_e\}$. Weight $w^{\hat{v}}$ receives the average value of w^{v_e} and $w^{v'_e}$. The learning proceeds by reducing the accumulated errors of v_e and v'_e by half to assign this reduction to $\theta^{\hat{v}}$. When the GNG network has only one neuron, v_e, the updates are slightly different. A new neuron $\hat{v}$ is added to the network as well as an edge $\{\hat{v}, v_e\}$. Both weights and accumulated error of the new neuron $\hat{v}$ receive the same values as those of v_e. Then, both $\theta^{\hat{v}}$ and θ^{v_e} are reduced by half.

Algorithm 3 shows the detailed process of inserting new elements in the network.

The detailed process of inserting new elements in the network occurs as shown in Algorithm 4.

The asymptotic complexity of the learning strategy is $O(m)$.

2.5 Path-relinking

The metaheuristic path-relinking (PR) is an intensification strategy proposed in Glover (1996) to further investigate good quality solutions hybridized with a tabu search (Glover 1998, 1999).

PR consists in defining paths between a pair of solutions (initial solution and guiding solution) with the purpose of finding intermediate high-quality solutions. This metaheuristic works by iteratively adding attributes from the guiding solution to the path solution, which starts as a copy of the initial solution, until they coincide.

```
Data: $\beta, \varepsilon_b, \varepsilon_n, s, count, \lambda, a_{max}, max_G, G$
Result: Updated $G$
1  if $|V(G)| > 1$ then
2      Find the two closest neurons to $s$, $v_1$ and $v_2$
3      Add edge $\{v_1, v_2\}$ with age zero
4      Update the weights of the neighborhood of $v_1$: Eqs. (7) and (8)
5      Increase the age of the incident edges of $v_1$ by one
6  end
7  else
8      Assign the single vertex from $V(G)$ to $v_1$
9  end
10 Update the data input error of $v_1$: Eq. (9)
11 Update the weight of $v_1$: Eqs. (5) and (6)
12 Increase count by one
13 if $|V(G)| > 2$ then
14     Remove the disconnected neurons
15 end
16 if $count \bmod \lambda = 0$ and $|V(G)| < max_G$ then
17     $G \leftarrow$Neuron insertion $(G)$
18 end
19 Update the error of all neurons: Eq. (10)
```

Algorithm 3: GNG presentation

```
Data: $G$
Result: Updated $G$
1  if $|V(G)| > 1$ then
2      Find the neuron with the largest error, $v_e$
3      Find the neighbor of $v_e$ with the largest error, $v'_e$
4      Add a new neuron $\hat{v}$ between $v_e$ and $v'_e$
5      Add edges $\{\hat{v}, v_e\}$ and $\{\hat{v}, v'_e\}$, both with null age, and remove edge $\{v_e, v'_e\}$
6      Initiate the error of $\hat{v}$ with the average error of $v_e$ and $v'_e$ and diminish the errors of $v_e$ and $v'_e$ by
       half
7  end
8  else
9      Let $v_e \in V(G)$
10     Add a new neuron $\hat{v}$ to $G$ with $\theta^{\hat{v}} = \theta^{v_e}$ and $w^{v_e} = w^{\hat{v}}$
11     Add the edge $\{\hat{v}, v_e\}$ to $G$
12     Diminish the errors of $v_e$ and $\hat{v}$ by half
13 end
```

Algorithm 4: Neuron insertion

In PR, to better address the steps of the path, it is necessary to define a distance metric between solutions. Then, the distance between the path solution and the guiding solution iteratively decreases until it is null. The distance metric enables one to estimate the number of iterations needed to accomplish the path.

The literature regarding solution methods based on the PR is vast (Resende and Ribeiro 2005).

2.5.1 Proposed PR

Let $\mathscr{E}$ be the set of elite solutions, which stores the best overall solutions. Let D be the previously defined distance metric, i.e., half the cardinality of the symmetric difference between a pair of sets.

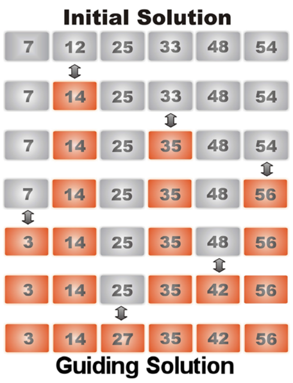

Fig. 2 A PR example

The proposed PR takes a 1-neighborhood at each step to define the path solutions. The neighborhood is restricted to the set of facilities that belong to the symmetric difference and that decreases the distance between the path and the guiding solutions. Figure 2 illustrates a PR between two solutions with 6 facilities.

Considering $j \in \{1, \ldots, 6\}$, the it-th vector in Fig. 2 indicates the path solution at the it-th step of the path. The positions colored in orange are the facilities that do not belong to the initial solution added to the corresponding path solution. The most recently swapped facilities in the path are indicated by a two-point arrow.

Algorithm 5 presents the PR procedure. It is possible to note that the elite solution, set $\mathscr{E}$, and a solution s are relevant input data for constructing the paths.

```
Data: $\mathscr{E}, s^*, s, \beta, \varepsilon_b, \varepsilon_n, count, \lambda, a_{max}, max_G, G$
Result: The best solution $s^*$
1  Select a solution $s_{\mathscr{E}} \in \mathscr{E}$ according to the probability distribution indicated in Eq. (12)
2  Randomly select between $s$ and $s_{\mathscr{E}}$ and assign it to $s_{guide}$
3  Assign the solution not chosen to be $s_{guide}$ in line 2 to $s_{path}$
4  while $Y_{s_{guide}} \neq Y_{s_{path}}$ do
5      $(j, l) \leftarrow \arg\max_{j \in Y_{path} \setminus Y_{s_{guide}},\ l \in Y_{s_{guide}} \setminus Y_{s_{path}}} (\Delta_l - \Delta_j)$
6      $Y_{s_{path}} \leftarrow Y_{s_{path}} \setminus \{j\} \cup \{l\}$
7      if $f(s^*) < f(s_{path})$ then
8          $s^* \leftarrow s_{path}$
9          $s^* \leftarrow$ Local Search($s^*$)
10         Update $\mathscr{E}$ with $s^*$
11         $G \leftarrow$ GNG presentation($\beta, \varepsilon_b, \varepsilon_n, s^*, count, \lambda, a_{max}, max_G, G$)
12     end
13 end
```

Algorithm 5: PR algorithm.

Algorithm 5 aims at reconnecting paths between the current solution s and a solution $s_{\mathscr{E}} \in \mathscr{E}$. First, $s_{\mathscr{E}}$ is chosen considering a probability distribution that favours the best quality solutions:

$$P\left(s_{\mathscr{E}}\right)=\frac{e-p_{s_{\mathscr{E}}}}{\left(e^2-e\right)/2} \tag{12}$$

where $e=|\mathscr{E}|$ and $p_{s_{\mathscr{E}}}$ is the position of $s_{\mathscr{E}}$ in $\mathscr{E}$, and where $\mathscr{E}$ is sorted in decreasing order of objective function values. This formula prioritizes the best solutions with the highest objective functions.

After choosing $s_{\mathscr{E}}$, the algorithm randomly selects between s and $s_{\mathscr{E}}$ which will be the initial and the guiding solutions, as presented in Pessoa et al. (2013). Then, as part of PR, it chooses a facility that belongs to the symmetric difference between the path solution and the guiding solution and that pertains to the initial solution. The algorithm picks the facility that incurs in the best objective function value to the path solution, even though it may worsen the quality of the current solution. If the path solution is better than the best overall solution, it applies a local search to the path solution and updates it as the best overall solution. If the best overall solution has just been updated, it is presented to the GNG network and the algorithm replaces the worst solution in the elite set $\mathscr{E}$ with the new best overall solution. The complexity of this algorithm is $O(pmn)$.

A PR closely related to the one proposed here can be found in Resende and Werneck (2006). In that study, the authors address the uncapacitated facility location problem and the criterion of the facility swaps is the same as the PR here proposed; the differences lie on the elite solutions set and the post-optimization process.

The elite set $\mathscr{E}$ comprises the best solutions found in the search process and cannot exceed the size limit $max_{\mathscr{E}}$.

Algorithm 6 shows a detailed procedure of IGASPR, that has as input the instance data. The instance data refer to all the parameters of the MCLP.

3 Computational experiments

The experiments were carried out in a cluster with 104 nodes, each with 2 Intel Xeon E5-2680v2 processors of 2.8 GHz, 10 cores and 128 GB DDR3 1866 MHz of RAM. All the metaheuristics were implemented in Java.

The computational experiments used the same set of instances presented in Máximo et al. (2017), which are based on four data sets representing the demographic density of the following counties: Bronx, Manhattan, San Francisco and Kings. The authors compiled the data sets from the 2010 United States census. Their primary features are shown in Table 1. Additionally, the set of instances contained a total of 24 instances, as $p \in \{50, 60, 70, 80, 90, 100\}$.

The metric to measure the distance between two points i and j, d_{ij}, is the 2-norm (Euclidean). Then, it is enough to consider their latitude-longitude coordinates, (lat_i, lon_i) and (lat_j, lon_j), to calculate d_{ij} in meters, using Eq. (13). This equation follows the spherical law of cosines, where R the radius of the earth ($R = 6{,}378{,}100$).

```
   Data: Instance data
   Result: The best solution $s^*$
1  Set parameter values for: $\eta, \beta, \alpha, \varphi, \omega, \lambda, a_{max}, max_G$
2  Initiate $G$ with a single vertex $\hat{v}$, $\theta^{\hat{v}} \leftarrow 0$ and the vector $w^{\hat{v}}$ with random values [0, 1]
3  $\bar{f}, f^*, \bar{b}, iter, count \leftarrow 0$
4  while elapsed time < time limit do
5      $s \leftarrow$ IGASPR Construction Phase($\alpha, \varphi, \omega, G$)
6      $s \leftarrow$ Local Search($s$)
7      if $f^* < f(s)$ then
8          $s^* \leftarrow s$
9          $f^* \leftarrow f(s)$
10     end
11     $\bar{f} \leftarrow (\bar{f} \times iter + f(s))/(iter + 1)$
12     $\bar{b} \leftarrow \bar{f} + \eta(f^* - \bar{f})$
13     if $f(s) > \bar{b}$ then
14         $G \leftarrow$ GNG presentation( $\beta, \varepsilon_b, \varepsilon_n, s, count, \lambda, a_{max}, max_G, G$)
15     end
16     Update $\mathscr{E}$ with $s$ if $D(s', s) > 0 \; \forall \; s' \in \mathscr{E}$
17     $s^* \leftarrow$ PR algorithm($\mathscr{E}, s^*, s, \beta, \varepsilon_b, \varepsilon_n, count, \lambda, a_{max}, max_G, G$)
18     $iter \leftarrow iter + 1$
19 end
```

Algorithm 6: IGASPR.

Table 1 Summary of the data sets used in the experiments

County–state	m	r (m)
Manhattan–NY	2713	400
Bronx–NY	3839	600
San Francisco–CA	5137	600
Kings–NY	7730	800

$$d_{ij} = \arccos(\cos(lat_i)\cos(lat_j)\cos(lon_j - lon_i) + \sin(lat_i)\sin(lat_j))R \qquad (13)$$

Let UB be the linear relaxation solution value of a problem. The quality of a solution s of this problem is assessed by the gap between its objective function value and UB. Equation (14) presents how this gap can be calculated.

$$gap = 100\frac{(UB - f(s))}{UB} \qquad (14)$$

The performances of IGASPR, GRASP (Resende 1998) and IGAS were assessed and contrasted. The parameter configuration for both GRASP and IGAS followed what is suggested in the literature. In addition, GRASP was hybridized with the introduced PR (GRASPPR).

This section presents the experiments reported in this paper and is subdivided as follows: (1) an experiment that shows how the parameters of IGASPR were defined; (2) a performance analysis presenting the improvement of the quality of the solutions when the learning step in the search process is considered; (3) a performance analysis to demonstrate that the modified IGAS proposed in this paper outperforms IGAS in its original setting; (4) an experiment with time-to-target plots (TTT-plots) (Aiex et al. 2007) to prove that IGASPR outperforms IGAS and GRASPPR; (5) an experiment that reports the results obtained using IGASPR, GRASP (Resende 1998) and IGAS, considering the entire set of instances of Máximo et al. (2017); (6) and a comparison between IGASPR and CPLEX v.12.6 time limited in 20,000 s on the model proposed by Church and ReVelle (1974).

3.1 Parameters for the IGASPR

This subsection describes the tuning of the IGASPR parameters. To achieve this, first the performance of IGASPR was analyzed by varying the values of ω, η and φ. The GNG parameters were fixed with the same values suggested in the literature after their fine tuning with ParamILS (Hutter et al. 2009): $\varepsilon_b = 0.6$, $\varepsilon_n = 0.003$, $a_{max} = 110$, $\lambda = 60$, $\beta = 0.980$ and $max_G = 10$. Heinke and Hamker (1998) pointed out that all GNG parameters are not sensitive to changes, then a set of default values are usually adopted. The size of the elite set of the PR phase, $max_{\mathscr{E}}$, was empirically defined as 6.

The investigated ranges for parameters ω, η and φ were:

- $\omega \in \{0.5, 0.55, \dots, 0.9\}$;
- $\eta \in \{0, 0.1, \dots, 0.9\}$;
- $\varphi \in \{20, 30, 40, 50\}$.

The analysis consisted in running 30 independent executions for each combination of all possible values for the parameters, in the defined ranges, and using the 24 instances. The time limit for each execution was 300 s. A summary of the achieved results is displayed in Fig. 3. It is possible to notice that the combination of parameters $\omega = 0.7$, $\eta = 0.4$ and $\varphi = 40$ resulted in the best mean gaps.

Furthermore, after establishing the best values for the three parameters, the size of the elite set considering $max_{\mathscr{E}} \in \{2, 4, \dots, 20\}$ was varied. The mean results showed that $max_{\mathscr{E}}$ was not sensitive to such variation since it did not have a significant impact on the performance of IGASPR. Therefore, $max_{\mathscr{E}}$ was kept with the value 6.

After analyzing these four parameters, we then studied the behavior of the six GNG parameters considering the following ranges:

- $\varepsilon_b \in \{0.1, 0.2, \dots, 1\}$;
- $\varepsilon_n \in \{0.001, 0.002, \dots, 0.01\}$;
- $\lambda \in \{10, 20, \dots, 100\}$;
- $max_G \in \{2, 4, \dots, 20\}$;
- $a_{max} \in \{50, 60, \dots, 150\}$;
- $\beta \in \{0.970, 0.975, \dots, 0.995\}$.

Like parameter $max_{\mathscr{E}}$, no relevant impact on the performance of IGASPR was noticed by varying these parameters, and therefore, the values suggested in Máximo et al. (2017) were used for the GNG parameters.

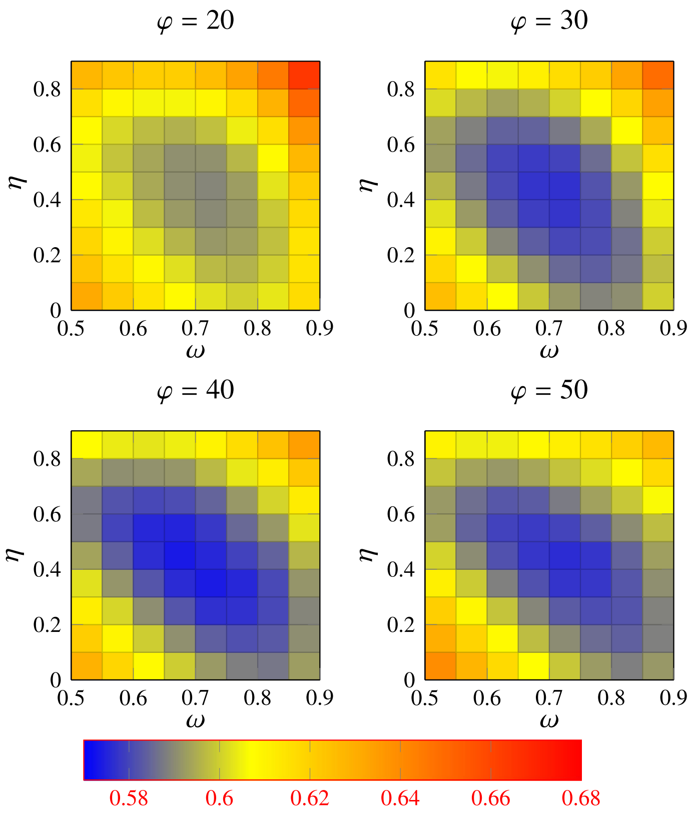

Fig. 3 Mean gaps of 24 instances varying the parameters φ, ω and η

3.2 Improvement with learning

This section presents an experiment that shows the quality of the solutions found in the construction phase and in the local search phase with different values for ω. Note that ω is the percentage of elements added to the solution in the construction phase using knowledge learnt at previous iterations. Therefore, this experiment aims at demonstrating the impact of the learning step on the search process.

Thus, by varying this parameter, it is possible to notice the interference of the learning step during the iterations of the algorithm. In this experiment, 1000 iterations were equally distributed in the interval [0, 0.7] in order to define ω to solve the Bronx instance with $p = 70$ and $r = 600$. Therefore, considering a step of 0.0007 in the interval [0, 0.7] to define ω, Fig. 4 illustrates the solutions found by the construction and local search phases of IGASPR.

The best solutions curve up to a given iteration demonstrates the improvement in quality of the LS solutions after the construction phase and after PR.

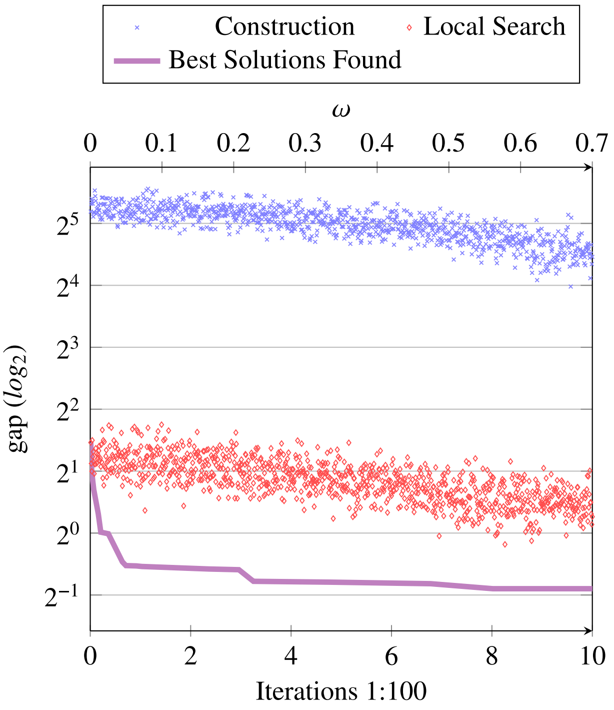

Fig. 4 Figure that highlights the best gaps found after the construction and local search phases, and the best gaps found after path-relinking varying ω along the iterations

3.3 Proving the better performance of the introduced IGAS

In this experiment, we present the performance profiles introduced by Dolan and Moré (2002) to attest that the modified IGAS outperforms IGAS (Máximo et al. 2017). The graphic displays the relation between the percentage of the np problems solved (y-axis) by a given algorithm s with a performance better than or equal to a factor $\tau \in \mathbb{R}$ (x-axis) multiplied by the value of the best performance metric found among all ns algorithms. Let us consider that the metric must be minimized. Therefore, for a given τ, such percentage is denoted by $\phi_s(\tau)$:

$$\phi_s(\tau) = \frac{\left|\left\{\frac{t_{ps}}{\min_{s' \in \{1,\ldots,ns\}} t_{ps'}} \leq \tau : \forall p\right\}\right|}{np}$$

Figure 5 compares the performance profiles of IGAS and the modified IGAS. The metric considered is the time to achieve the best solution, since both versions obtained the same final solutions. The results indicate that the modified IGAS outperforms IGAS. One may observe that when $\tau = 1$, the modified IGAS presented 90% for $\phi_s(\tau)$, which indicates that the modified IGAS algorithm showed a better time than IGAS in 90% of the solutions. As τ grows, it is noteworthy that the curves only cross when $\phi_s(\tau) = 1$, showing that the modified IGAS is better than IGAS. Moreover,

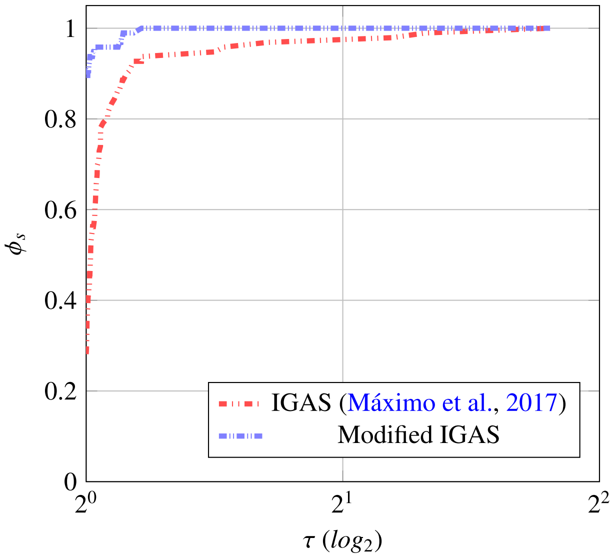

Fig. 5 Performance profiles (Dolan and Moré 2002) comparing IGAS (Máximo et al. 2017) with the modified IGAS here proposed using a new strategy to construct solutions

$\phi_s(\tau) = 1$ for the modified IGAS when $\tau = 1.1$, whereas for IGAS, this happens when $\tau = 3.5$.

As this version of IGAS is better than the previous one, it was employed in the hybridization with PR, IGASPR.

3.4 TTT-plots analysis

In addition to the quality expressed by the objective function, this paper shows the TTT-plots (Aiex et al. 2007, commonly used to compare metaheuristics hybridized with PR. This graphic illustrates the cumulative probability of an algorithm to achieve a target solution within a considered time for a given instance. In this experiment, we considered 100 runs for each of the following algorithms: GRASPPR, IGAS and IGASPR.

Figure 6 shows the results when the value of the target solution is defined as 1,573,000 for the Manhattan County instance with $p = 100$ and $r = 400$. This means the time every algorithm took to obtain the first solution (whose objective function was at least the value of the target solution) was considered. The results for GRASP are not reported in this experiment because it did not achieve the target solution in any of the one hundred executions carried out in the time limit of 10^6 s.

IGASPR was more effective than IGAS, as can be seen in Fig. 6. IGASPR took on average 46.9 s to obtain the target solution, whereas IGAS took on average 487.9 s. Therefore, IGASPR proved to be 10 times faster than IGAS. GRASPPR presented the worst computational performance, taking 25,507.4 s, on average, to achieve the target solution.

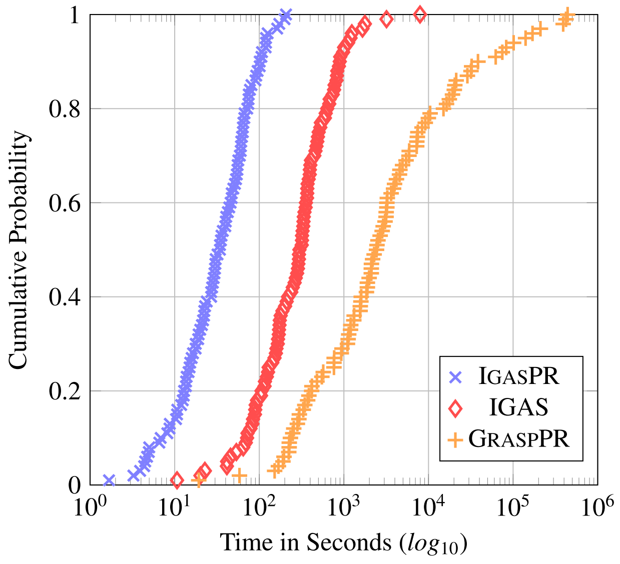

Fig. 6 TTT-plots comparing IGASPR with IGAS and GRASPPR using the Manhattan instance with $p = 100$ and $r = 400$

In this same experiment, 100 independent executions of the Kings instance were carried out, with $p = 90$ and $r = 800$, and considering a time limit of 1000 s. The gap between the best solution up to that moment and its UB was registered for each run, and the average gap among the 100 executions for that period in time was then calculated. Figure 7 displays the relation between this average gap (y-axis) and the time to achieve the gap (x-axis).

It can be observed in Fig. 7 that IGAS and IGASPR presented the highest convergence rates when compared to the other algorithms. In particular, in just 8 s, IGASPR achieved an average gap lower than the one achieved by GRASPPR in 1000 s. Both IGAS and IGASPR obtained gaps that were better than the best gap achieved by GRASP in 1000 s.

Finally, in order to assess the robustness of the metaheuristics, a box plot diagram was designed. It shows the 100 executions of the four analyzed algorithms under a time limit of 300 s, as can be seen in Fig. 8. The San Francisco instance with $p = 100$ and $r = 600$ was employed in this analysis. This graphic shows the distribution of the final gaps obtained by the metaheuristics.

The results show that IGAS and IGASPR had a tight data distribution in comparison to the other algorithms. Moreover, IGASPR obtained the best results among all algorithms, as previously mentioned.

3.5 Comparative performance analysis

In this experiment, the performance of IGASPR was compared to the performance of the other three algorithms using the entire set of 24 instances. Each algorithm was

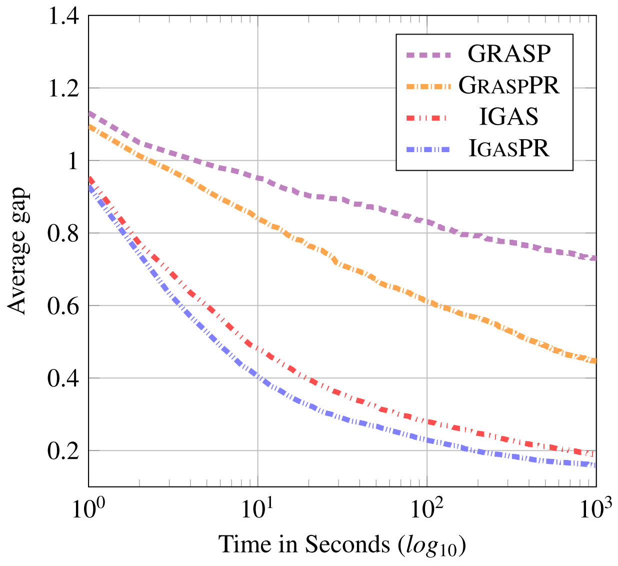

Fig. 7 Relation between the average gap in 100 executions within the corresponding time using the Kings instance with $p = 90$ and $r = 800$

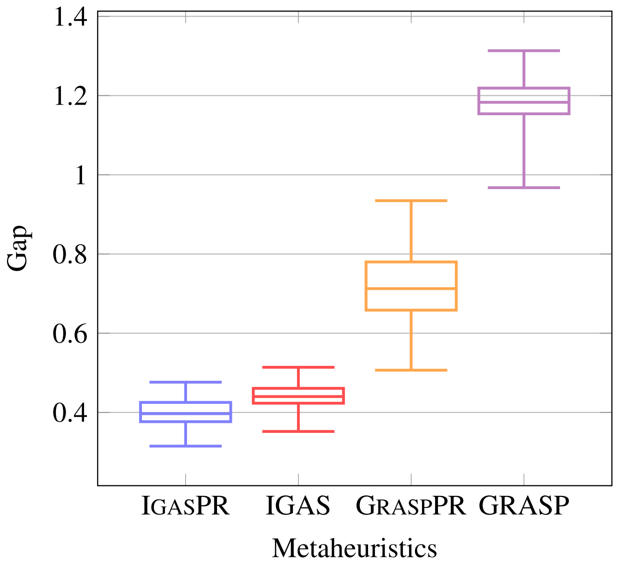

Fig. 8 Box plot of the final gaps achieved by the four metaheuristics considering 100 independent runs and $p = 100$ and $r = 600$ for the San Francisco instance

carried out 30 times with a time limit of 300 s. Tables 2 and 3 present the results of this experiment:

- **Best** Best gap found in 30 executions.
- **Worst** Worst gap found in 30 executions.

Table 2 Results of Experiment II

County	p	GRASP			GRASPPR			IGAS			IGASPR		
		Best	Worst	Median	Best	Worst	Median	Best	Worst	Median	Best	Worst	Median
Manhattan	50	**0.175**	0.479	0.329	**0.175**	0.207	**0.175**	**0.175**	**0.175**	**0.175**	**0.175**	**0.175**	**0.175**
	60	0.472	0.911	0.745	**0.262**	0.300	0.271	**0.262**	0.279	**0.262**	**0.262**	0.279	**0.262**
	70	0.901	1.226	1.083	**0.331**	0.538	0.411	**0.331**	0.360	**0.331**	**0.331**	**0.331**	**0.331**
	80	0.985	1.520	1.345	**0.419**	0.814	0.463	**0.419**	0.462	0.426	**0.419**	0.430	**0.419**
	90	1.306	1.656	1.562	0.582	0.850	0.653	**0.573**	0.660	0.589	**0.573**	0.617	0.574
	100	1.056	1.364	1.249	0.419	0.545	0.473	0.369	0.479	0.433	**0.366**	0.439	0.405
Bronx	50	1.080	1.395	1.291	0.660	0.972	0.782	**0.614**	0.788	0.660	**0.614**	0.692	**0.614**
	60	1.180	1.580	1.431	0.761	1.062	0.874	**0.691**	0.921	0.755	**0.691**	0.736	0.712
	70	1.105	1.450	1.275	0.508	0.878	0.615	0.496	0.608	0.507	**0.476**	0.562	0.478
	80	0.894	1.098	1.004	0.301	0.646	0.432	**0.262**	0.367	0.292	**0.262**	0.311	0.275
	90	0.434	0.504	0.467	0.097	0.276	0.163	0.059	0.112	0.075	**0.049**	0.075	0.060
	100	0.064	0.127	0.098	0.003	0.013	0.006	**0.001**	0.004	0.002	**0.001**	0.004	0.002
San Francisco	50	0.811	1.322	1.119	**0.600**	0.927	0.676	**0.600**	0.624	**0.600**	**0.600**	**0.600**	**0.600**
	60	1.305	1.684	1.507	0.915	1.195	1.055	**0.844**	1.052	0.933	**0.844**	0.990	**0.844**
	70	1.512	1.962	1.843	1.200	1.530	1.360	0.988	1.202	1.051	**0.974**	1.039	1.010
	80	1.931	2.176	2.055	1.267	1.795	1.525	1.003	1.270	1.084	**0.984**	1.170	1.022
	90	1.877	2.074	1.963	1.238	1.615	1.399	0.879	1.102	0.965	**0.872**	1.053	0.968
	100	1.033	1.249	1.159	0.542	0.866	0.713	0.359	0.506	0.434	**0.340**	0.463	0.397

Table 2 continued

County	p	GRASP			GRASPPR			IGAS			IGASPR		
		Best	Worst	Median	Best	Worst	Median	Best	Worst	Median	Best	Worst	Median
Kings	50	0.840	1.304	1.101	0.584	1.016	0.705	**0.570**	0.642	0.583	**0.570**	0.637	0.583
	60	1.842	2.098	1.971	1.440	1.903	1.630	**1.272**	1.413	1.329	**1.272**	1.390	1.308
	70	2.018	2.439	2.271	1.387	2.270	1.877	1.259	1.518	1.399	**1.252**	1.490	1.309
	80	1.689	2.095	1.892	1.221	1.771	1.589	0.942	1.240	1.055	**0.932**	1.120	1.006
	90	0.678	0.902	0.768	0.341	0.677	0.503	0.130	0.230	0.172	**0.120**	0.212	0.166
	100	0.112	0.195	0.149	0.002	0.040	0.019	**0.000**	0.002	0.001	**0.000**	0.001	**0.000**
Mean		1.054	1.367	1.237	0.636	0.946	0.765	0.546	0.667	0.588	**0.541**	0.617	0.563

This table shows the best, the worst and the median gaps of the 30 executions for every tested metaheuristic
Bold values indicate the best results considering all algorithms in the comparison

Table 3 Results of Experiment II

County	*p*	GRASP		GRASPPR		IGAS		IGASPR	
		Average	Time	Average	Time	Average	Time	Average	Time
Manhattan	50	0.334	171.5	0.178	54.9	**0.175**	8.4	**0.175**	4.3
	60	0.728	145.5	0.273	129.2	0.269	107.6	**0.267**	85.8
	70	1.091	166.2	0.405	136.5	0.336	66.7	**0.331**	27.2
	80	1.324	162.9	0.495	162.0	0.427	185.8	**0.422**	120.3
	90	1.532	150.2	0.686	148.9	0.598	161.1	**0.579**	108.5
	100	1.226	147.6	0.474	159.7	0.436	209.0	**0.404**	156.8
Bronx	50	1.262	158.6	0.792	110.0	0.667	120.6	**0.616**	57.5
	60	1.414	138.2	0.900	135.6	0.768	210.8	**0.709**	134.4
	70	1.295	123.3	0.653	176.4	0.519	202.7	**0.491**	134.4
	80	1.000	160.2	0.452	213.1	0.299	214.1	**0.278**	167.5
	90	0.470	129.6	0.164	239.6	0.078	214.5	**0.059**	184.0
	100	0.097	178.8	0.007	143.1	**0.002**	212.1	**0.002**	106.4
San Francisco	50	1.106	169.2	0.704	142.4	0.604	51.0	**0.600**	40.6
	60	1.484	145.0	1.068	164.5	0.934	167.5	**0.872**	137.1
	70	1.828	148.0	1.357	198.9	1.077	210.3	**1.003**	188.6
	80	2.051	141.7	1.530	200.5	1.097	208.6	**1.034**	161.9
	90	1.965	127.8	1.393	217.2	0.984	217.4	**0.966**	171.3
	100	1.153	155.2	0.707	232.3	0.436	230.7	**0.405**	200.0

Table 3 continued

County	p	GRASP		GRASPPR		IGAS		IGASPR	
		Average	Time	Average	Time	Average	Time	Average	Time
Kings	50	1.095	135.8	0.713	159.9	0.596	185.3	**0.581**	127.7
	60	1.962	142.1	1.638	171.2	1.340	217.7	**1.319**	153.2
	70	2.271	143.0	1.879	201.2	1.391	216.6	**1.325**	192.3
	80	1.907	204.3	1.578	244.4	1.063	252.1	**1.011**	202.8
	90	0.774	159.0	0.500	235.7	0.180	220.6	**0.169**	199.1
	100	0.150	114.1	0.020	244.1	0.001	228.8	**0.000**	163.9
Mean		1.230	150.7	0.774	175.9	0.595	180.0	**0.567**	134.4

The table shows the average gaps and the mean times to obtain the best solution at each of the 24 instances and for each metaheuristic
Bold values indicate the best results considering all algorithms in the comparison

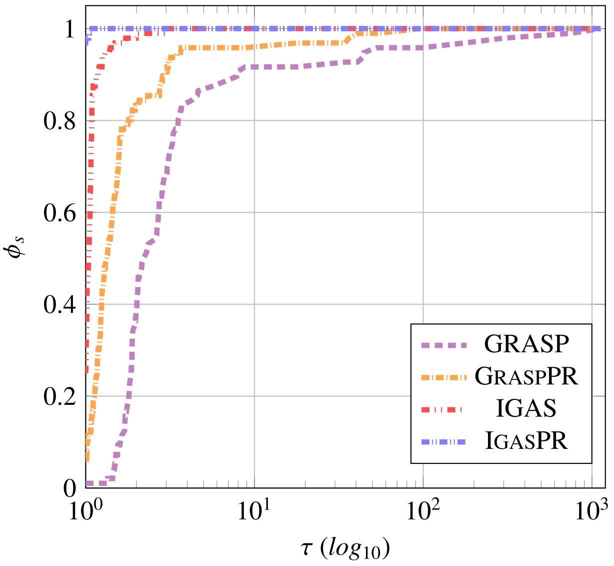

Fig. 9 Performance profiles (Dolan and Moré 2002) comparing GRASP, GRASPPR, IGAS and IGASPR

- **Median** Median gap found in 30 executions.
- **Average** Average gap found in 30 executions.
- **Time** Average time in seconds that an algorithm took to find the best solution in each of the 30 executions.

The performance profiles from Dolan and Moré (2002), explained in Sect. 3.3, was used in this experiment. Figure 9 compares the best, worst, median and average gaps of the four algorithms. One may observe through the IGASPR performance profile curve that it dominates the curves of the other algorithms. This means that IGASPR had the largest percentage of problems solved within any τ value. Moreover, we point out that IGASPR presented the best results in 97% of the problems, whereas IGAS, GRASPPR and GRASP presented the best results in 25, 6 and 1% of the problems, respectively. These results are observed when $\tau = 1$.

3.6 IGASPR versus CPLEX

In this experiment, the results obtained by IGASPR were compared with results on the model introduced by Church and ReVelle (1974) using CPLEX v.12.6 (CPLEX 2014) and run on the same machine. To that purpose, IGASPR was run 100 times, for each of the 24 instances, with a time limit of 1000 s. Table 4 shows the best solutions found by IGASPR and the results of CPLEX limited in 20,000 s. Column 'Best' shows the gap of the best lower bound found by CPLEX to the corresponding linear relaxation and the elapsed time it took to find it. Moreover, column 'UB' reports the linear relaxation found by CPLEX. The gaps highlighted with asterisks indicate that the best lower bound found by either CPLEX or IGASPR corresponded to the optimal solution. We

Table 4 Best solutions for every tested instance

County	p	UB	CPLEX		IGASPR	
			Best	Time	$f(s)$	Gap
Manhattan	50	1,155,665.95	**0.175***	8.6	1,153,640	**0.175***
	60	1,291,555.70	**0.262***	16.6	1,288,174	**0.262***
	70	1,400,966.42	**0.331***	36.6	1,396,333	**0.331***
	80	1,487,666.36	**0.419***	54.1	1,481,434	**0.419***
	90	1,548,872.76	0.577	8669.6	1,539,992	**0.573**
	100	1,579,918.14	0.360	12,325.8	1,574,256	**0.358**
Bronx	50	1,212,490.99	**0.614***	4053.0	1,205,051	**0.614***
	60	1,299,609.96	0.736	3210.3	1,290,635	**0.691**
	70	1,354,680.04	0.534	19,268.5	1,348,232	**0.476**
	80	1,379,346.03	0.350	19,547.6	1,376,061	**0.238**
	90	1,384,989.60	0.073	18,790.8	1,384,372	**0.045**
	100	1,385,108.00	**0.001**	6776.4	1,385,095	**0.001**
San Francisco	50	621,321.75	**0.600**	18,575.1	617,592	**0.600**
	60	688,955.63	0.939	19,997.9	683,139	**0.844**
	70	740,942.12	1.168	19,922.8	733,772	**0.968**
	80	777,434.54	1.096	19,165.9	769,781	**0.984**
	90	798,237.81	1.010	19,958.3	791,562	**0.836**
	100	804,711.15	0.612	13,440.4	802,211	**0.311**
Kings	50	2,033,678.47	0.660	10,696.8	2,022,077	**0.570**
	60	2,255,707.14	1.477	19,724.5	2,227,025	**1.272**
	70	2,410,847.41	1.556	7048.3	2,380,660	**1.252**
	80	2,490,711.39	1.518	7231.1	2,467,701	**0.924**
	90	2,504,700.00	18.923	2782.6	2,502,169	**0.101**
	100	2,504,700.00	0.138	18,581.2	2,504,700	**0.000***

Bold values indicate the best results considering all algorithms in the comparison

also reported the value of the objective function of the best solutions found by IGASPR, as well as the gap to their linear relaxation.

CPLEX confirmed the optimality of the lower bound found by IGASPR to the Manhattan instances when $p \in \{50, 60, 70, 80\}$, and Bronx when $p = 50$. In particular, for Kings with $p = 100$, the solution obtained by IGASPR was proved to be optimum because the value of its objective function was equal to the UB.

Experiments were also carried out with the instances proposed in Lorena and Pereira (2002), but are not reported in this paper due to the low dimensionality of the instances. These results can be found at https://sites.google.com/site/nascimentomcv/downloads/sjcinstances. As a result, it is possible to observe that IGASPR outperformed all the other heuristics in the comparative analysis.

4 Final remarks

This paper reports a study performed on intelligent-guided adaptive search (IGAS), a recently proposed metaheuristic that considers the Growing Neural Gas (GNG) network as a memory mechanism in the search process.

IGAS efficiently solved large instances of the maximum covering location problem (MCLP) as proposed in the literature. In spite of the good performance of IGAS, we provided new insights into this metaheuristic by slightly modifying one of its steps and hybridizing it with path-relinking.

In a first experiment, we proved that the new version of IGAS is consistently better than the original IGAS in solving large-scale MCLP instances. In a second experiment, we evaluated the performance of the hybrid, named IGASPR, comparing it with IGAS, GRASP and GRASPPR. IGASPR further enhanced the quality of the solutions achieved by the modified IGAS for the MCLP, significantly outperforming every tested algorithm.

In future works, the authors intend to apply IGAS and its hybridization with path-relinking with other combinatorial optimization problems, such as the capacitated facility location problem.

Acknowledgements The authors are grateful to Fundação de Amparo á Pesquisa do Estado de São Paulo (FAPESP) (Grant No. 15/21660-4) and Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) (Grant No. 308708/2015-6, 448614/2014-6) for their financial support. Research carried out using the computational resources of the Center for Mathematical Sciences Applied to Industry (CeMEAI) funded by FAPESP (Grant No. 2013/07375-0).

References

Aiex, R.M., Resende, M.G.C., Ribeiro, C.C.: TTT plots: a perl program to create time-to-target plots. Optim. Lett. **1**, 355–366 (2007)

Church, R., ReVelle, C.: The maximal covering location problem. Pap. Reg. Sci. Assoc. **32**, 101–118 (1974)

CPLEX: IBM ILOG CPLEX Optimization Studio CPLEX User's Manual Version 12 Release 6, IBM (2014)

Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. Math. Program. **91**, 201–213 (2002)

Feo, T.A., Resende, M.G.C.: A probabilistic heuristic for a computationally difficult set covering problem. Oper. Res. Lett. **8**, 67–71 (1989)

Fritzke, B.: Growing cell structures—a self-organizing network for unsupervised and supervised learning. Neural Netw. **7**, 1441–1460 (1994)

Fritzke, B.: A growing neural gas network learns topologies. In: Tesauro, G., Touretzky, D.S., Leen, T.K. (eds.): Advances in Neural Information Processing Systems 7, pp. 625–632. MIT Press, Cambridge (1995)

Galvão, R.D., ReVelle, C.: A Lagrangean heuristic for the maximal covering location problem. Eur. J. Oper. Res. **88**, 114–123 (1996)

Galvão, R.D., Espejo, L.G.A., Boffey, B.: A comparison of Lagrangean and surrogate relaxations for the maximal covering location problem. Eur. J. Oper. Res. **124**, 377–389 (2000)

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

Glover, F.: Tabu search and adaptive memory programing—advances, applications and challenges. In: Barr, R.S., Helgason, R.V., Kennington, J.L. (eds.) Interfaces in Computer Science and Operations Research, pp. 1–75. Kluwer, Dordrecht (1996)

Glover, F.: A template for scatter search and path relinking. In: Hao, J.-K., Lutton, E., Ronald, E., Schoenauer, M., Snyers, D. (eds.) Artificial Evolution, Volume 1363 of Lecture Notes in Computer Science, pp. 1–51. Springer, Berlin (1998)

Glover, F.: Scatter search and path relinking. In: Corne, D., Dorigo, M., Glover, F. (eds.) New Ideas in Optimization, pp. 297–316. McGraw Hill (1999)

Heinke, D., Hamker, F.H.: Comparing neural networks: a benchmark on growing neural gas, growing cell structures, and fuzzy artmap. IEEE Trans. Neural Netw. **9**, 1279–1291 (1998)

Hutter, F., Hoos, H.H., Leyton-Brown, K., Stützle, T.: ParamILS: an automatic algorithm configuration framework. J. Artif. Int. Res. **36**, 267–306 (2009)

Jia, H., Ordóñez, F., Dessouky, M.M.: Solution approaches for facility location of medical supplies for large-scale emergencies. Comput. Ind. Eng. **52**, 257–276 (2007)

Karasakal, O., Karasakal, E.K.: A maximal covering location model in the presence of partial coverage. Comput. Oper. Res. **31**, 1515–1526 (2004)

Kohonen, T.: Self-organized formation of topologically correct feature maps. Biol. Cybern. **43**, 59–69 (1982)

Lorena, L.A.N., Pereira, M.A.: A Lagrangean/surrogate heuristic for the maximal covering location problem using Hillman's edition. Int. J. Ind. Eng. **9**, 57–67 (2002)

Martinetz, T.: Competitive Hebbian learning rule forms perfectly topology preserving maps. In: Gielen, S., Kappen, B. (eds.) Proceedings of the International Conference on Artificial Neural Networks (ICANN-93), pp. 427–434. Springer, Amsterdam (1993)

Martinetz, T., Schulten, K.: A "neural-gas" network learns topologies. In: Kohonen, T., Makisara, K., Simula, O., Kangas, J. (eds.) Artificial Neural Networks, pp. 397–402. Elsevier Science Publishers B. V., North-Holland (1991)

Máximo, V.R., Nascimento, M.C.V., Carvalho, A.C.P.L.F.: Intelligent-guided adaptive search for the maximum covering location problem. Comput. Oper. Res. **78**, 129–137 (2017)

Pessoa, L.S., Resende, M.G.C., Ribeiro, C.C.: A hybrid Lagrangean heuristic with GRASP and path-relinking for set k-covering. Comput. Oper. Res. **40**, 3132–3146 (2013)

Resende, M.G.C.: Computing approximate solutions of the maximum covering problem with GRASP. J. Heurist. **4**, 161–177 (1998)

Resende, M.G.C., Ribeiro, C.C.: GRASP with path-relinking: recent advances and applications. In: Ibaraki, T., Nonobe, K., Yagiura, M. (eds.) Metaheuristics: Progress as Real Problem Solvers, Volume 32 of Operations Research/Computer Science Interfaces Series, pp. 29–63. Springer, New York (2005)

Resende, M.G.C., Werneck, R.F.: A hybrid multistart heuristic for the uncapacitated facility location problem. Eur. J. Oper. Res. **174**, 54–68 (2006)

ReVelle, C., Scholssberg, M., Williams, J.: Solving the maximal covering location problem with heuristic concentration. Comput. Oper. Res. **35**, 427–435 (2008). Part Special Issue: Location Modeling Dedicated to the memory of Charles S. ReVelle