Production, Manufacturing and Logistics

A biased random-key genetic algorithm for the unequal area facility layout problem

José Fernando Gonçalves and Mauricio G. C. Resende

1. Introduction

The facility layout design problem (FLP) is a challenging non-linear combinatorial optimization problem encountered in many service and manufacturing organizations. The problem has been extensively studied in the literature and good reviews can be found in Kusiak and Heragu (1987) and Meller and Gau (1996).

The variant of the FLP we focus on in this paper was originally formulated by Armour and Buffa (1963) and involves determining the most cost-effective arrangement of a given number of rectangular facilities with unequal area requirements within a given rectangular floor space. This problem is denoted as UA-FLP. The objective of the problem is to minimize the cost associated with the interactions between facilities (the cost of material-handling flows). This cost is commonly represented by the sum of the products (over all facility pairs) of the weighted rectilinear distance and the material-handling cost between the centroids of the facilities. The constraints of the problem include facility area requirements and shape restrictions, as well as assuring that the facilities do not overlap and are located within the boundary of the space floor.

The UA-FLP is NP-hard since it is a generalization of the quadratic assignment problem (QAP), where the optimization is carried over a finite set of possible facility locations. The QAP was shown to be NP-hard by Sahni and Gonzalez (1976). Various methods and procedures have been proposed to solve the FLP and can be classified into:

- **Exact procedures** - Montreuil (1991) proposed one of the first MIP formulations of the FLP on the continuous plane. The model includes disjunctive constraints to prevent facility overlaps and bounded perimeter constraints to enforce specified facility area and shape requirements. The largest problem instance solved optimally by Montreuil’s original formulation had six facilities. Meller, Narayanan, and Vance (1998) used valid inequalities to tighten Montreuil’s formulation and were able to optimally solve problems with up to eight facilities. Sherali, Fraticelli, and Meller (2003) used a polyhedral outer approximation to the facility area to improve the model of Meller et al. (1998) and solved a nine-facility problem optimally. Castillo and Westerlund (2005) developed an \( \epsilon \)-accurate approximation. However, the largest problem solved had only nine facilities. By reformulating the FLP with the sequence-pair representation of Murata, Fujiyoshi, Nakatake, and Kajitani (1996), Meller, Chen, and Sherali (2007) were able to solve problems with up to 11 facilities.

- **Heuristics and meta-heuristics** – Tam and Li (1991) proposed a hierarchical approach that employs a divide-and-conquer strategy consisting of three phases: (1) cluster analysis, (2) initial layout, and (3) layout refinement. The cluster analysis generates a hierarchical structure of the layout. The second phase produces an initial layout of each cluster which is subsequently refined in the third phase. Langevin, Montreuil, and Riopel (1994) developed a heuristic approach based on the MIP model of Montreuil (1991) to solve spine layout problems. Kado (1996) developed six types of genetic algorithms using a slicing-tree structure representation. Garces-Perez, Schoenefeld, and Wainwright (1996) proposed a multi-purpose genetic programming kernel to generate slicing trees that are converted into candidate solutions. Schnecke and Vornberger (1997) introduced a genetic algorithm with a tree-structured genotype representation and hybrid problem-specific operators. Dunker, Radons, and Westkämper (2003) developed a coevolutionary approach that improved mutation and crossover operators and clustered the facilities into groups. Scholz, Petrick, and Domshke (2009) proposed a tabu search algorithm with a slicing-tree representation and incorporated a bounding curve for solving fixed and flexible facilities in UA-FLPs. Their tabu search incorporated four types of neighborhood moves to find better solutions. Komarudin and Wong (2010) proposed an ant system to solve the UA-FLP using a slicing-tree representation and several types of local search to improve its search performance. Wong and Komarudin (2010) developed an ant system algorithm for solving UA-FLPs using an improved flexible bay structure representation called modified-FBS (mFBS). McKendall Jr and Hakobyan (2010) introduced an approach which uses a boundary search construction technique that places facilities along the boundaries of already placed facilities. The solution is improved using a tabu search. Kulturel-Konak and Konak (2011b) use an ant colony optimization approach which uses a flexible bay structure representation. Kulturel-Konak and Konak (2011a) produced a hybrid particle swarm optimization and local search approach using a relaxed flexible bay structure (RFBS). Kulturel-Konak and Konak (2014) proposed a hybrid simulated annealing and MIP algorithm for the cyclic facility layout problem which can be applied to various facility layout problems including the UA-FLP. A large-scale hybrid simulated annealing algorithm (LS-HSA) is proposed to solve the formulated problem and shown to be effective and versatile as it can be applied to various facility layout problems.

- **Matheuristics** – Gau and Meller (1999) proposed an algorithm that iterates between a genetic algorithm with a slicing-tree representation and a mixed-integer program with a subset of the binary variables set via the genetic algorithm. Montreuil, Brotherton, Ouazzani, and Norellfath (2004) put together an algorithm based on ant colony optimization (ACO) and the zone-based MIP (Montreuil, Brotherton, & Marcotte, 2002) where the ACO-based heuristic searches for assignments of facilities to zones, and then the zone-based MIP is used to determine the detailed layout, including the input/output points of the facilities. Liu and Meller (2007) proposed a GA combining the sequence-pair representation (Murata et al., 1996) with the MIP model of Sherali et al. (2003). For a given sequence-pair, the corresponding layout is determined in this hybrid approach using the linear programming (LP) relaxation of the MIP model. Bozer and Wang (2012) introduced a hybrid approach based on a new representation called the graph-pair representation. The graph-pair representation encodes the relative locations of the facilities and the shape and uses an LP to determine the exact location of each facility. A simulated annealing algorithm is used to search for new layouts based on the graph-pair representation. Recently, Kulturel-Konak and Konak (2013) proposed a hybrid genetic algorithm and linear programming approach to solve the UA-FLP which uses a new encoding scheme, called location/shape and which represents the relative facility positions based on the centroids and orientations of the facilities. After the relative facilities positions are set by the GA, the actual facility locations and shapes are determined by solving an LP problem.

Our contribution to solve the UA-FLP is a matheuristic which consists of a biased random-key genetic algorithm (BRKGA) with a decoder which combines a novel placement strategy and a linear programming model to fine-tune the solutions.

The remainder of the paper is organized as follows. Section 2 presents a formal model of the UA-FLP. Section 3 introduces the new approach, describing the BRKGA in detail, the novel placement strategy, and the fitness function. Finally, in Section 4, we report on computational experiments, and in Section 5 we make concluding remarks.

### 2. Problem formulation

Let $N$ be the number of rectangular facilities of unequal areas to be placed, without any overlap, on a rectangular floor space with dimensions $(W, H)$ along the $X$- and $Y$-axes, respectively. Each facility $i = 1, \ldots, N$ is defined by its dimensions along the horizontal and vertical axis $(w_i, h_i)$, its area $A_i = w_i \times h_i$, and the maximum aspect ratio, $R_{\text{max}}$, which due to practical reasons imposes the maximum permissible ratio between its longest and shortest dimensions, i.e., $R_{\text{max}} \geq \max \{w_i/h_i\}/\min \{w_i/h_i\} \geq 1$.

A layout is defined by the coordinates of the centroid $(x_i, y_i)$ and the horizontal $(w_i)$ and vertical $(h_i)$ dimensions of each facility $i$. The cost function to be minimized is

$$\text{Cost} = \sum_{i=1}^{N} \sum_{j=1}^{N} c_{ij} f_{ij} d_{ij}$$

where $f_{ij}$ represents the flow between facilities $i$ and $j$ (we assume that $f_{ij} = 0$), $c_{ij}$ is the cost per unit distance between $i$ and $j$, and $d_{ij}$ is the distance between the centroids of facilities $i$ and $j$.

The distance $d_{ij}$ can be defined according to one of the following distance norms:

- **Rectilinear distance (R):** $d_{ij} = |x_i - x_j| + |y_i - y_j|$;
- **Euclidean distance (E):** $d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$;
- **Squared Euclidean distance (SE):** $d_{ij} = (x_i - x_j)^2 + (y_i - y_j)^2$.

Since most real-world layout problems make use of the rectilinear distance norm, we will model the problem using this distance norm. In practice the dimensions of the floor space and the area of the facilities are usually not hard constraints and can accommodate small variations. To be able to include this flexibility when searching for a solution we will add to our model the variables $\Delta W$ and $\Delta H$ to represent, respectively, the amount that each facility $i$ exceeds the horizontal and vertical dimensions $(W, H)$ and include a term in the objective function to penalize this excess.

The problem can be cast in an intuitive way in the form of a non-linear mixed integer programming model, FLP-NMIP. In the model we will use the following additional notation:

$$P_{ij}, Q_{ij} = \text{Binary variables used to model the non-overlapping constraints:}$$

- If $(P_{ij}, Q_{ij})$ is equal to $(0, 0)$ then facility $i$ is forced to the right of $j$;
between all pairs of facilities \((P_i, Q_i)\) is equal to \((0, 1)\) then facility \(i\) is forced to the right of \(j\);

- if \((P_i, Q_i)\) is equal to \((1, 0)\) then facility \(i\) is forced to the bottom of \(j\);

\(M^x, M^y =\) Parameters defining upper bounds on the horizontal and vertical distance between any two facilities, respectively.

\(d_{ij}^x, d_{ij}^y =\) Variables representing the distances between the facilities \(i\) and \(j\) along the \(X\)- and \(Y\)-axes, respectively.

\(M =\) Constant used to penalize the solutions for exceeding the floor space. If no variations are allowed, then \(M = \infty\).

A nonlinear integer programming model of the facility layout problem is given in (2)–(15). The objective function (2) minimizes the total cost using the appropriate distance norm. Constraints (3)–(4) define the horizontal and vertical dimensions of the facility according to the area and maximum allowed ratio. Constraints (5)–(8) impose the non-overlapping constraints by forcing the facilities to be separated horizontally and/or vertically. Constraints (9)–(10) force each facility to be within the horizontal and vertical limits of the floor space, respectively. Constraints (11)–(14) define the distances between all pairs of facilities \((i, j)\) according to the rectilinear norm distance function. Finally, constraints (15) define the domains for the different variables.

In the model FLP-NMIP constraints (3) are non-linear resulting in a hard-to-solve model. However, Castillo and Westerlund (2005) developed a linear approximation based on a cutting plane representation of the actual area constraint that guarantees that, at optimality, the final area of each facility is within an \(\varepsilon\)-error of the required area regardless of the aspect ratio of the facilities. To linearize the model and make it easier to solve, we replace the non-convex and hyperbolic area constraints (3) and the aspect ratio constraints 4 with the \(\varepsilon\)-accurate representation:

\[
-h_i - \frac{A_i}{W_{i,k}} w_i \leq 2A_i \frac{1}{W_{i,k}}, \quad k = 0, \ldots, C_i, \quad \forall i
\]

where \(W_{i,k}\) corresponds to the tangent points of the cutting planes on the real curve and \(C_i\) is the total number of points being used in the approximation according to the chosen \(\varepsilon\)-error value. The resulting model with constraints (1) replacing constraints (3) will be denoted as FLP-MIP.

\[
\text{(FLP-NMIP). Minimize } \text{Cost} = \sum_{i=1}^{N} \sum_{j=1}^{N} c_{ij} f_{ij} \left(d_{ij}^x + d_{ij}^y\right)
\]

\[
+ M \sum_{i=1}^{N} \left(\Delta w_i + \Delta h_i\right)
\]

Subject to:

- Facility aspect constraints

\[
w_i \times h_i = A_i, \quad \forall i
\]

\[
\frac{1}{R_{\max}} \leq \frac{w_i}{h_i} \leq R_{\max}, \quad \forall i
\]

- Non-overlapping constraints

\[
x_i - x_j + M^x \left(P_{ij} + Q_{ij}\right) \geq \frac{w_i + w_j}{2}, \quad \forall i, j | j > i
\]

\[
x_j - x_i + M^x \left(1 - P_{ij} + Q_{ij}\right) \geq \frac{w_i + w_j}{2}, \quad \forall i, j | j > i
\]

\[
y_i - y_j + M^y \left(1 + P_{ij} - Q_{ij}\right) \geq \frac{h_i + h_j}{2}, \quad \forall i, j | j > i
\]

\[
y_j - y_i + M^y \left(2 - P_{ij} - Q_{ij}\right) \geq \frac{h_i + h_j}{2}, \quad \forall i, j | j > i
\]

\[
\frac{w_i}{2} \leq x_i \leq W - \frac{w_i}{2} + \Delta w_i, \quad \forall i
\]

\[
h_i \leq y_i \leq H - h_i + \Delta h_i, \quad \forall i
\]

\[
x_i - x_j \leq d_{ij}^x, \quad \forall i, j | j > i
\]

\[
x_j - x_i \leq d_{ij}^x, \quad \forall i, j | j > i
\]

\[
y_i - y_j \leq d_{ij}^y, \quad \forall i, j | j > i
\]

\[
y_j - y_i \leq d_{ij}^y, \quad \forall i, j | j > i
\]

- Domain constraints

\[
x_i, y_i, w_i, h_i, d_{ij}^x, d_{ij}^y \geq 0, \quad \forall i \quad \text{and} \quad P_{ij}, Q_{ij} \in \{0, 1\}, \quad \forall i, j | j > i
\]

The mixed integer model FLP-MIP developed above is still computationally difficult and fails to provide optimal or even near-optimal solutions on real-size problems due to the large number of binary variables. To overcome this problem we developed a new solution methodology, which combines a biased random-key genetic algorithm with a novel placement strategy. In the next section we describe the new methodology.

3. Biased random-key genetic algorithm

We begin this section with an overview of the proposed solution methodology. This is followed by a discussion of the biased random-key genetic algorithm, including detailed descriptions of the solution encoding and decoding, evolutionary process, and novel placement strategy.

We will first describe the algorithm for the case where the dimensions of the floor space are unconstrained (i.e., we do not take into account Eqs. (9) and (10)). Later in Section 4.2 we extend the approach to the constrained case.

3.1. Overview

The new approach is based on a constructive heuristic algorithm which places the facilities, one at a time, on the rectangular floor space. To avoid overlapping of the facilities, we propose a novel placement strategy which uses the concept of empty maximal-spaces, as described in Lai and Chan (1997), and a novel, very efficient, placement procedure to position a facility within a given empty maximal space. The new approach combines a biased random-key genetic algorithm and the novel placement strategy.

The role of the genetic algorithm is to evolve the encoded parameters, or chromosomes, that represent the facility placement sequence (FPS), the vector of facility aspect ratios (FAR), and the position of the first facility \((x_{\text{first}}, y_{\text{first}})\). The vectors of encoded parameters are decoded using a novel placement strategy which results in the placement of each facility. For each chromosome, the following phases are applied to decode the chromosome:

1. **Facility placement sequence decoder:** This first phase decodes part of the chromosome into the FPS, i.e., the sequence in which the facilities are placed on the floor space.

2. **Facility aspect ratio decoder:** The second phase decodes part of the chromosome into the FAR, i.e., the vector of facility aspect ratios.
3.2. Biased random-key genetic algorithms

In biased random-key genetic algorithms (Goldberg, 1989), i.e. to introduce noise into the population and avoid convergence of the entire population to a local optimum. At each generation, a small number \( p_m \) of mutants is introduced into the population. With \( p_e + p_m \) individuals accounted for in population \( g + 1 \), \( p - p_e - p_m \) additional individuals need to be generated to complete the \( p \) individuals that make up population \( g + 1 \). This is done by producing \( p - p_e - p_m \) offspring solutions through the process of mating or crossover.

A biased random-key genetic algorithm (Gonçalves & Resende, 2011), or simply BRKGA, differs from a RKGA in the way parents are selected for mating and how mating is carried out. While in the RKGA of Bean (1994) both parents are selected at random from the entire current population, in a BRKGA each element is generated combining a parent selected at random from the elite partition in the current population and one from the rest of the population. Repetition in the selection of a mate is allowed and therefore an individual can produce more than one offspring in the same generation. As in a RKGA, parametrized uniform crossover (Spears & Dejong, 1991) is used to implement mating in a BRKGA. Let \( p_e \) be the probability that an offspring inherits the vector component of its elite parent. Recall that \( r \) denotes the number of components in the solution vector of an individual. For \( i = 1, \ldots, r \), the \( i \)-th component \( e(i) \) of the offspring vector \( e \) takes on the value of the \( i \)-th component \( e(i) \) of the elite parent \( e \) with probability \( p_e \) and the value of the \( i \)-th component \( e(i) \) of the non-elite parent \( e \) with probability \( 1 - p_e \). While in a BRKGA \( p_e > \frac{1}{2} \), in a RKGA this is not necessarily the case.

When the next population is complete, i.e. when it has \( p \) individuals, fitness values are computed for all of the newly created random-key vectors and the population is partitioned into elite and non-elite individuals to start a new generation.

A BRKGA searches the solution space of the combinatorial optimization problem indirectly by searching the continuous \( r \)-dimensional hypercube, using the decoder to map solutions in the hypercube to solutions in the solution space of the combinatorial optimization problem where the fitness is evaluated.

To specify a biased random-key genetic algorithm, we simply need to specify its parameters, how solutions are encoded and decoded, and how their corresponding fitness values are computed. We specify our algorithm next by first showing how the facility placement problem is encoded and then decoded into a solution and how their fitness evaluation is computed.

3.2.1. Chromosome representation and decoding

A chromosome encodes a solution to the problem as a vector of random keys. In a direct representation, a chromosome represents a solution of the original problem and is called genotype, while in an indirect representation it does not. In an indirect representation, special procedures are needed to extract a solution, or phenotype, from it. In the present context, solutions will be represented indirectly by parameters that are later used by a decoding procedure to obtain a solution. To obtain the phenotype we use the decoding procedures described in Section 3.3.3.

Each solution chromosome is made of \( 2N + 2 \) genes as depicted in Fig. 2, where \( N \) is the number of facilities to be laid out. The first \( N \) genes are used to obtain the facility placing sequence (FPS), genes \( N + 1 \) to \( 2N \) are used to obtain the vector of facility aspect ratios (FAR), and genes \( 2N + 1 \) and \( 2N + 2 \) are used to obtain the coordinates \((x_{\text{first}}, y_{\text{first}})\) of the first facility to be placed. The placement procedure,
3.2.2. Fitness function

The decoding (mapping) of the first N genes of each chromosome into a FPS is accomplished by sorting the key values of the genes in increasing order. The indices of the sorted keys correspond to the sequence in which facilities will be laid out. Fig. 3 shows an example of the decoding process for the FPS. There are eight facilities in this example. Sorting the eight random keys in increasing order produces the following FPS = (5, 8, 3, 1, 4, 2, 6, 7).

The decoding of the vector of facility aspect ratios (FAR) is accomplished for i = 1, . . . , N, using the following expression:

\[ \text{FAR}_i = \frac{1}{R_{\text{max}}} + \text{gene}_i \times \left( R_{\text{max}} - \frac{1}{R_{\text{max}}} \right). \]  

(16)

Since FAR = \( \frac{w_i}{h_i} \), the dimensions of each facility i can now be computed as

\[ w_i = \sqrt{A_i \times \text{FAR}_{R_{\text{max}}}} , \]

(17)

and

\[ h_i = \frac{A_i}{w_i} . \]

(18)

The first facility to be laid out is defined by the FPS and is denoted as FPS. Its coordinates, i.e., (x(first), y(first)) are decoded as:

\[ x_{\text{first}} = \frac{w_1}{2} + \text{gene}_{2N+1} \times (W - w_1) . \]

(19)

\[ y_{\text{first}} = \frac{h_1}{2} + \text{gene}_{2N+2} \times (H - h_1) . \]

(20)

FPS, FAR, and (x(first), y(first)) are used later by the placement procedure to construct a layout with all the facilities placed on the floor space.

3.2.2. Fitness function

To evolve the solutions, the evolutionary process needs a measure of the solution fitness, or quality. A natural fitness function for this type of problem is the layout cost defined as

\[ \text{Cost} = \sum_{i=1}^{N} \sum_{j=1}^{N} c_{ij} f_{ij} d_{ij} + M \sum_{i=1}^{N} (\Delta w_i + \Delta h_i) , \]

where, as defined above, \( c_{ij} \) and \( f_{ij} \) are respectively, the cost per distance unit and the flow between facilities \( i \) and \( j \), and \( d_{ij} \) is the distance according to the appropriate norm. The second term in the fitness corresponds to the penalty for exceeding the dimensions of the floor space. Note that when the dimensions of the floor space are unconstrained the value of the penalty term will be equal to zero, i.e., \( M = 0 \).

3.3. Placement strategy

In the next sections we describe the main components of the placement strategy.

3.3.1. Maximal-spaces and the difference process

While trying to place a facility on the floor space we use a list \( S \) of empty maximal-spaces (EMSs), i.e., largest empty rectangular spaces available on the floor space. An empty maximal-space \( s \) is represented by its vertices with minimum and maximum coordinates \((\min_x, \min_y, \max_x, \max_y)\). When searching for a place to position a facility, we need to consider only the available EMSs where the facility being placed fits. This way, we guarantee that there will be no overlap between facilities. To generate and keep track of the EMSs, we make use of the difference process (DP), developed by Lai and Chan (1997). Fig. 4 depicts an example of the application of the DP process. In the example, we assume that we have two facilities to be placed on the floor space. Since the floor space is initially empty, we only have the empty maximal-space \( \text{EMS}_0 \) which is the entire floor space (see Fig. 4a). After placing facility 1 in \( \text{EMS}_0 \) we update the list \( S \) of available empty maximal-spaces so we can try to place facility 2. Fig. 4b shows the four newly generated EMSs. Facility 2 is placed in \( \text{EMS}_2 \) and the DP process results in the six EMSs shown in Fig. 4c. Every time a facility is placed on the floor space, we reapply the DP process to update list \( S \) before we place the next facility.

In the unconstrained case we assume that the floor space can include all the facilities laid out horizontally or vertically at their maximum horizontal or vertical dimensions, i.e.,

\[ \text{EMS}_0 = (0, 0, \sqrt{A_i \times R_{\text{max}}} \times \sqrt{A_j \times R_{\text{max}}} ) . \]

3.3.2. Placement of a facility in an empty maximal-space

When attempting to place a facility \( i \) on the floor space we want the facility to have the least cost with respect to the facilities already laid out. To achieve this we solve the following problem for every available empty maximal-space \( s \) and all facilities \( k \in K \) already placed on the floor space:

\[(\text{FLP}_{\text{EMS}}) \text{ Minimize } \text{Cost}(i) = \sum_{k \in K} c_{ik} f_{ik} d_{ik} \]

(21)

Subject to:

\[ \min_x x_i + \frac{w_i}{2} \leq x_i \leq \max_x x_i - \frac{w_i}{2} , \]

(22)

\[ \min_y y_i + \frac{h_i}{2} \leq y_i \leq \max_y y_i - \frac{h_i}{2} . \]

(23)

Note that we solve FLP_{EMS} only for empty maximal-spaces in which facility \( i \) fits.

FLP_{EMS} has a non-linear objective function (21) and the variables \( x_i, y_i \) are subject only to box constraints (22) and (23). This problem can be solved with the non-monotone spectral projected gradient method proposed in Birgin, Martinez, and Raydan (2000) (FORTRAN source code at \http://www.ime.usp.br/~egbirgin/tango/codes.php#spg\). However, we developed a new more efficient surrogate approach. The new approach starts by computing the unconstrained optimal (i.e., without the box constraints), denoted by UO (details on how to compute UO for the three distance norms can be found in Heragu (1997)). Next, facility \( i \) is tentatively positioned at the geometric center of the EMS \( s \), i.e., at the point

\[ \left( \frac{\min_x x_i + \max_x x_i}{2}, \frac{\min_y y_i + \max_y y_i}{2} \right) . \]

The final position of facility \( i \) is obtained by moving its centroid as close as possible to UO without leaving the boundaries of the EMSs. We first try to get as close as possible to UO by moving vertically and then by moving horizontally (or vice-versa). Fig. 5 illustrates this approach for two empty maximal spaces.

In some cases there is no flow between the facility \( i \) being placed and all the other facilities \( k \in K \) already placed on the floor space, i.e.,

---

**Fig. 3.** Decoding of the facility placement sequence.
Facilities to be placed and the initial empty maximal-space (the floor space)

Empty maximal-spaces after placing facility 1.

Empty maximal-spaces after placing facility 2.

Fig. 4. Difference process (DP) example.

Unconstrained Optimal

Minimum cost position

Initial position

Minimum cost position

Initial position

Fig. 5. Example of the novel approach used to solve FLP EMS.

\[ f_i, k = 0 \quad \forall k \in K. \]

When this occurs we consider UO to be equal to the geometric center of all laid-out facilities, i.e.,

\[
(UO_x, UO_y) = \left( \frac{1}{|K|} \sum_{k \in K} x_k, \frac{1}{|K|} \sum_{k \in K} y_k \right).
\]

From this point on we will denote by FLP EMS\((i, s, x, y)\) the procedure that determines the minimum-cost position \((x, y)\) of facility \(i\) in EMS.

3.3.3. Placement procedure

The placement procedure follows a sequential process which places a single facility at each stage. The procedure combines FPS, FAR, and \((x_{\text{first}}, y_{\text{first}})\) evolved by the BRKGA. Each stage is comprised of four main steps:

1. Facility selection;
2. Computation of the facility aspect ratio and its dimensions;
3. Facility placement;
4. State information update.
Pseudo-code of the placement procedure is given in Fig. 6. The facility selection at stage $i$ chooses for placement the facility in the $i$th position of the FPS (line 4 of the pseudo-code). The facility dimensions are defined by the $i$th position of the FAR (line 5). The facility placement is carried out in lines 6–18. The coordinates of the first facility placed are defined in line 16 while the coordinates of the other facilities are defined in lines 7–14 of the pseudo-code. The facility placement is carried out in line 18. The final step, state information update, consists in updating the list of empty maximal spaces, according to the facility being placed and the corresponding coordinates, using the DP procedure (line 19 of the pseudo-code).

3.4. Constrained case

Our first approach to extend the algorithm for the case where the dimensions of the floor space are constrained was to use BRKGA with a large value assigned to the penalty constant $M$, i.e.,

$$M = (W + H) \sum_{i=1}^{N} \sum_{j=1}^{N} f_{ij}.$$  

This approach works quite well, for all the three distance norms presented above when the area of the floor space exceeds the sum of the areas of the facilities by more than about 25%. However, when the area of the floor space is close to the sum of the areas of the facilities, the quality of the solution worsens significantly and most of the time no feasible solution can be found. To overcome this problem, we developed a new hybrid matheuristic approach, denoted by BRKGA-LP, to solve problems with floor space constraints. In the hybrid BRKGA-LP approach, the BRKGA produces unconstrained solutions defining the relative locations of facilities in the floor space. The relative locations are then used to define the separation constraints of the FLP-MIP model transforming it into a linear program, denoted by FLP-LP. The solution of the FLP-LP determines the new locations and dimensions of all the facilities that may improve the BRKGA solution. The constrained approach, BRKGA-LP, can be described for each chromosome by the following steps:

1. Solve the problem with BRKGA, the unconstrained approach. In this case and after some experimentation we came to the conclusion that limiting the unconstrained floorspace to 1.7 times of the real horizontal and vertical dimensions of the floor space produces better results, when solving in the next step the FLP-LP, than if the two dimensions were both unconstrained, i.e., were very large. Therefore, when solving the unconstrained problem we use a floor space with dimensions $(1.7 \times W, 1.7 \times H)$, as shown in Fig. 7a. Let $S_{BRKGA}$ be the solution produced by the BRKGA in this step. Note that sometimes, due to the limitations of the floor dimensions, it is not possible to position all the facilities in the floor space. In that case we consider the fitness of the solution $S_{BRKGA}$ to be equal to $\infty$ and skip Steps 2 and 3.

2. Based on the solution $S_{BRKGA}$ obtained in the previous step, formulate the corresponding linear model FLP-LP($S_{BRKGA}$), where the non-overlapping constraints (5), (6), (7), and (8) in the FLP-MIP model are replaced by a single constraint defined according to $S_{BRKGA}$. The single separating constraint that will replace the original separating constraints in the FLP-MIP model can
Fig. 8. BRKGA solutions for datasets TL05 and TL06.
be determined as follows. Let $\text{Dis}_W_{i,j} = |x_i - x_j| - \frac{w_i}{2} - \frac{w_j}{2}$ and $\text{Dis}_H_{i,j} = |y_i - y_j| - \frac{h_i}{2} - \frac{h_j}{2}$ be, respectively, the horizontal and vertical distances separating facilities $i$ and $j$. If $\text{Dis}_W_{i,j} \geq \text{Dis}_H_{i,j}$, then separate facilities $i$ and $j$ horizontally. If $x_i \leq x_j$, use constraint $x_i + \frac{w_i}{2} + \frac{w_j}{2} \leq x_j$. Otherwise, use $x_i + \frac{w_i}{2} + \frac{w_j}{2} \leq x_j$. If $\text{Dis}_W_{i,j} < \text{Dis}_H_{i,j}$, then separate facilities $i$ and $j$ vertically. If $y_i \leq y_j$ use constraint $y_i + \frac{h_i}{2} + \frac{h_j}{2} \leq y_j$. Otherwise, use $y_i + \frac{h_i}{2} + \frac{h_j}{2} \leq y_j$. Note that this way of defining the constraint guarantees that we will always obtain a feasible solution in terms of the relative position and dimensions of the facilities.

3. Solve FLP-LP($S_{\text{BRKGA}}$) to try to improve the solution $S_{\text{BRKGA}}$ by using new locations and dimensions for all facilities. Fig. 7b depicts a possible improved solution obtained by FLP-LP($S_{\text{BRKGA}}$) for the BRKGA solution presented in Fig. 7a.

Since solving FLP-LP($S_{\text{BRKGA}}$) is computationally expensive we only carry on to Steps 2 and 3 if the solution $S_{\text{BRKGA}}$ seems promising. A solution is considered promising if the following two conditions are satisfied:

1. The area of the facilities outside the real floor space dimensions is smaller than 45% of the real floor space, i.e.,
   \[
   \text{Area of the facilities outside of the real floor space} \leq 0.45 \times W \times H.
   \]
2. The cost of the solution $S^{BRKGA}$ is at most 40% above the cost of the best solution found so far in the solution process. At the beginning of the solution process the cost of the best solution is equal to $\frac{221}{e}$.

4. Numerical experiments

To evaluate the performance and the capabilities of our BRKGA and BRKGA-LP algorithms, we performed a series of computational experiments. The numerical experiments were conducted on a computer with an Intel Xeon E5-2630 @ 2.30GHz CPU and 16 GB of physical memory running the Linux operating system with Fedora release 18. BRKGA and BRKGA-LP were coded in C++ and the linear programs were solved with GUROBI OPTIMIZER version 5.5.

Two different types of FLPs were investigated. In the unconstrained case, we consider problems in which the dimensions of the floor space are allowed to be determined by the optimization algorithm. In the constrained case, we consider problems with given dimensions for the final layout.

4.1. BRKGA configuration

The parameters of the BRKGA were configured based on our past experience with genetic algorithms using the same evolutionary strategy (see, Gonçalves and Almeida (2002), Ericsson, Resende, and Pardalos (2002), Gonçalves and Resende (2004), Gonçalves, Mendes, and Resende (2005), Mendes, Gonçalves, and Resende (2009), Gonçalves, Mendes, and Resende (2009), Gonçalves, Mendes, and Resende (2009), Gonçalves and Resende (2011), Gonçalves and Sousa (2011), Gonçalves and Resende (2012), Fontes and Gonçalves (2013), Gonçalves and Resende (2013), Gonçalves and Resende (2014), Gonçalves, Resende, and Costa (2014a) and Gonçalves, Resende, and F. (2014b)). Experience has shown that good results can be obtained with the values of $p_e$, $p_m$, and crossover probability ($\rho_e$) listed in Table 1.

The next subsections report the details of the experiments and the results obtained by the approaches proposed in this paper.
For population size we obtained good results by indexing population size to the size of the problem, i.e., use small populations for small problems and larger populations for larger problems. The configuration shown in Table 2 was held constant for all problem instances in the experiments. The experimental results demonstrate that this configuration not only provides high-quality solutions, but it is also very robust.

4.2. Unconstrained case

As mentioned above for the unconstrained case, the dimensions of the floor space are free and are determined by the optimization algorithm. Additionally, note that the unconstrained approach can be used with any of the distance measures (R-Rectilinear, E-Euclidean and SE-Squared Euclidean).

4.2.1. Datasets and benchmark approaches

To compare the performance of the BRKGA with other approaches for the unconstrained case, we used 16 datasets from the literature and 100 randomly generated datasets which were constructed in such a way that the optimal solution is known. We do that in order to measure, in absolute terms, the deviation from the optimal values. A summary of the datasets used is presented in Table 3.

We compare the unconstrained version of the BRKGA with the approaches listed in Table 4.

4.2.2. Experimental results

In Tables 5, 6, and 7 we report the best cost and average times obtained over ten runs of BRKGA for all datasets. For the other approaches we also report the best cost. Even though the computational times reported by the other approaches might not be comparable we report them if they are available.
Table 2: Configuration parameters for the BRKGA algorithm.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p )</td>
<td>( 100 \times N )</td>
</tr>
<tr>
<td>( p_e )</td>
<td>( \min(0.25 \times p, \ 50) )</td>
</tr>
<tr>
<td>( p_m )</td>
<td>( 0.25 \times p )</td>
</tr>
<tr>
<td>( \rho_e )</td>
<td>0.70</td>
</tr>
<tr>
<td>Fitness</td>
<td>Cost of layout given by Eq. (2) (to be minimized)</td>
</tr>
<tr>
<td>Stopping criterion</td>
<td>50 generations for the unconstrained case and 100 generations for the constrained case</td>
</tr>
</tbody>
</table>

In Table 5 we evaluate the performance of BRKGA on the datasets TL05 to TL30. As can be seen in column %Imp, BRKGA improved the best solution for six out of the eight datasets. The improvements vary from 1.4% for TL08 to 25% for TL30. For dataset TL06 GA-TSG ranks first and BRKGA ranks second while for dataset TL15 HA-C ranks first and BRKGA second. It is clear that overall the BRKGA has the best performance in terms of solution quality.

In Table 6 we evaluate the performance of BRKGA on the datasets L020 to L125B and Dunker62. The best cost values of VIP-PLANOPT on datasets L020 to L125B are obtained from the 2010 demo version while the times are taken from the user manual of the 2006 version since they are not reported elsewhere. The values for the cost and CPU times for dataset Dunker62 are taken from McKendall Jr and Hakobyan (2010). As can be seen in column %Imp., BRKGA improved the best solution for all eight datasets. The improvements vary from 1.86% for L020 to 11.05% for L125A. It is clear that, overall, the BRKGA-based approach has the best performance in terms of solution quality. In terms of computational times, the BRKGA solves all the datasets in less than 2 min.

To study the absolute error of layouts produced by BRKGA we generated 100 datasets with known optimal solution (ten datasets with 10, 20, 30, 40, 50, 60, 70, 70, 80, 90, and 100 facilities each, respectively). In Table 7 we report on the average and maximum percentage deviation from optimal, \( \% \) DO_{avg} and \( \% \) DO_{max}, (over all the datasets having the same number of facilities) of the best cost of BRKGA over 10 runs. Additionally, we also include the average deviation from optimal solution value obtained by GUROBI when solving the FLP-MIP model for a maximum of 3600 CPU seconds. The times for the BRKGA correspond to the total time for ten BRKGA runs.

The results in Table 7 show that BRKGA performs quite well in terms of absolute deviation from optimal. From 10 to 40 facilities the \( \% \) DO_{avg} and \( \% \) DO_{max} equal zero. For datasets having between 50 and 100 facilities, the value of \( \% \) DO_{avg} increases from 0.11% to 7.36% while the value of \( \% \) DO_{max} increases from 1.12% to 10.97%. The relative performance of BRKGA when compared to GUROBI with the FLP-MIP model is also good both in terms of CPU time and solution quality. Note that as the number of facilities increases the quality of the solutions found by GUROBI with FLP-MIP decreases, e.g., for datasets with 100 facilities GUROBI has a \( \% \) DO_{avg} value equal to 101.78% while for BRKGA this value is only 7.36%.

The final solutions generated by BRKGA for datasets TL05, TL06, TL07, TL08, TL12, TL15, TL20, TL30, L020, L028, L050, L075, L100, L125A, L125B and Dunker62 are shown in Figs. 8–15.

4.3. Constrained case

As mentioned above, the dimensions of the floor space in the constrained case are fixed and are given as input. Additionally, note that the constrained approach can be only be used for the rectilinear distance measure (\( R \)).

4.3.1. Datasets and benchmark approaches

The performance of the BRKGA-LP approach for the constrained case is investigated using a comprehensive set of test problems from
Table 3
Benchmark datasets used in the unconstrained case.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Distance measure</th>
<th>Description</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dunker62</td>
<td>R</td>
<td>A dataset with 63 facilities. There are no constraints in the floor space dimensions.</td>
<td>Dunker et al. (2003)</td>
</tr>
<tr>
<td>TL05–30</td>
<td>SE</td>
<td>Eight datasets with sizes equal to 5, 6, 7, 8, 12, 15, 20, and 30 facilities. There are no constraints in the floor space dimensions.</td>
<td>Tam and Li (1991)</td>
</tr>
<tr>
<td>RND10–100</td>
<td>R</td>
<td>100 random generated datasets. Ten datasets for facility sizes equal to 10, 20, 30, 40, 50, 60, 70, 70, 80, 90, and 100. There are no constraints in the floor space dimensions. The dimensions of each of the facilities are individually given in the datasets. These datasets were generated in such a way that the optimal solution is known.</td>
<td>Available from the authors upon request.</td>
</tr>
</tbody>
</table>

Table 4
Benchmark approaches used in the unconstrained case.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type of method</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>HA-C</td>
<td>Hierarchical approach with clusters.</td>
<td>Tam and Li (1991)</td>
</tr>
<tr>
<td>GP-STS</td>
<td>Genetic programming algorithm that generates Slicing Tree Structures later converted to FLP solutions.</td>
<td>Garces-Perez et al. (1996)</td>
</tr>
<tr>
<td>TSaST</td>
<td>Tabu search with slicing-tree.</td>
<td>Scholz et al. (2009)</td>
</tr>
<tr>
<td>VIP-PLANOPT</td>
<td>VIP-PLANOPT 2010 is a commercial software from Engineering Optimization Software which is based on the algorithms presented in Mir and Imam (1996), Imam and Mir (1998), and Mir and Imam (2001). Since VIP-PLANOPT presents results that are better or equal to the ones reported in the papers we use it for comparisons purposes instead of the above three approaches.</td>
<td><a href="http://www.planopt.com">http://www.planopt.com</a></td>
</tr>
<tr>
<td>TS-BST</td>
<td>Tabu search with boundary search technique.</td>
<td>McKendall Jr and Hakobyan (2010)</td>
</tr>
<tr>
<td>GUROBI</td>
<td>Version 5.5 of commercial software solver from Gurobi Optimization.</td>
<td><a href="http://www.gurobi.com">http://www.gurobi.com</a></td>
</tr>
</tbody>
</table>

Table 5
Experimental results: Datasets TL05 to TL30.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>HA-C</th>
<th>GA-STS</th>
<th>GP-STS</th>
<th>GA-TSG</th>
<th>TSaST</th>
<th>BRKGA</th>
</tr>
</thead>
<tbody>
<tr>
<td>TL05</td>
<td>247</td>
<td>228</td>
<td>226</td>
<td>214</td>
<td>213.48</td>
<td>210.06</td>
</tr>
<tr>
<td>TL06</td>
<td>514</td>
<td>361</td>
<td>384</td>
<td>327</td>
<td>348.76</td>
<td>345.03</td>
</tr>
<tr>
<td>TL07</td>
<td>559</td>
<td>596</td>
<td>568</td>
<td>629</td>
<td>562.91</td>
<td>549.68</td>
</tr>
<tr>
<td>TL08</td>
<td>839</td>
<td>878</td>
<td>878</td>
<td>833</td>
<td>810.43</td>
<td>799.09</td>
</tr>
<tr>
<td>TL12</td>
<td>3162</td>
<td>3283</td>
<td>3220</td>
<td>3164</td>
<td>3054.23</td>
<td>345.03</td>
</tr>
<tr>
<td>TL15</td>
<td>5862</td>
<td>7384</td>
<td>7510</td>
<td>6813</td>
<td>6615.81</td>
<td>6395.43</td>
</tr>
<tr>
<td>TL20</td>
<td>-</td>
<td>16393</td>
<td>14033</td>
<td>13190</td>
<td>13198.40</td>
<td>9892.38</td>
</tr>
<tr>
<td>TL30</td>
<td>-</td>
<td>41095</td>
<td>39018</td>
<td>35358</td>
<td>33721.20</td>
<td>31454.23</td>
</tr>
</tbody>
</table>

Best values are in bold.

Table 6
Experimental results: Datasets L020 to L125B and Dunker62.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>VIP-PLANOPT</th>
<th>TSaST</th>
<th>TS-BST</th>
<th>BRKGA</th>
</tr>
</thead>
<tbody>
<tr>
<td>L20</td>
<td>1157</td>
<td>0.3</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>L28</td>
<td>6447.25</td>
<td>1.5</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>L50</td>
<td>78224.68</td>
<td>7</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>L75</td>
<td>34396.38</td>
<td>13</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>L100</td>
<td>538193.1</td>
<td>14</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>L125A</td>
<td>288774.6</td>
<td>110</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>L125B</td>
<td>1084451</td>
<td>70</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Dunker62</td>
<td>3939362</td>
<td>4996</td>
<td>3871510</td>
<td>252.0</td>
</tr>
</tbody>
</table>

Best values are in bold.
The BRKGA-LP approach approximates the facility areas by tangential supports (see Eq. (1)) which tends to produce solutions with smaller facility areas than the actual area requirements. However, the area approximation quality can be increased by increasing the number of tangential supports ($\Delta$) at the expense of an increase in computational effort. To evaluate the area approximation error incurred for each facility, we use the average percent area approximation error ($\%E_{\text{avg}}$) and the maximum percent area approximation error ($\%E_{\text{max}}$) metrics (as proposed in Castillo and Westerlund (2005) and Kulturel-Konak and Konak (2013)). These are defined, respectively, as:

$$\%E_{\text{avg}} = \frac{1}{N} \sum_{i=1}^{N} \frac{|A_i - w_i h_i|}{A_i}.$$  

$$\%E_{\text{max}} = 100\% \max_{i=1,...,N} \left\{ \frac{|A_i - w_i h_i|}{A_i} \right\}.$$  

The solution process of the BRKGA-LP uses a strategy similar to the one implemented by Kulturel-Konak and Konak (2013) in their GA-LP approach, i.e., we used $\Delta = 25$ to obtain a solution and after the last generation is complete we solve the best solution found again using $\Delta = 100$ to ensure that area approximation error is negligible. Consequently, the best solutions reported in this paper for the BRKGA-LP approach have a facility area approximation error either equal to 0% or very close to it.

Table 7
Experimental results: Random generated datasets with known optimal solution.

<table>
<thead>
<tr>
<th>Number of facilities</th>
<th>Gurobi</th>
<th>BRKGA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$T_s$</td>
<td>$%\Delta_{\text{avg}}$</td>
</tr>
<tr>
<td>10</td>
<td>3600</td>
<td>0.21</td>
</tr>
<tr>
<td>20</td>
<td>3600</td>
<td>0.01</td>
</tr>
<tr>
<td>30</td>
<td>3600</td>
<td>0.32</td>
</tr>
<tr>
<td>40</td>
<td>3600</td>
<td>2.37</td>
</tr>
<tr>
<td>50</td>
<td>3600</td>
<td>3.99</td>
</tr>
<tr>
<td>60</td>
<td>3600</td>
<td>16.65</td>
</tr>
<tr>
<td>70</td>
<td>3600</td>
<td>12.21</td>
</tr>
<tr>
<td>80</td>
<td>3600</td>
<td>12.21</td>
</tr>
<tr>
<td>90</td>
<td>3600</td>
<td>12.21</td>
</tr>
<tr>
<td>100</td>
<td>3600</td>
<td>101.78</td>
</tr>
</tbody>
</table>

Table 8
Datasets used in the constrained case.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>$T$</th>
<th>$H$</th>
<th>$W$</th>
<th>$\delta$</th>
<th>$\delta_{\text{max}}$</th>
<th>$\delta_{\text{min}}$</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAM20</td>
<td>20</td>
<td>40</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>0</td>
<td>Meller et al. (1998)</td>
</tr>
<tr>
<td>BA14</td>
<td>14</td>
<td>9</td>
<td>7</td>
<td>6</td>
<td>5</td>
<td>3</td>
<td>Bazaar (1975)</td>
</tr>
<tr>
<td>TAM30</td>
<td>30</td>
<td>40</td>
<td>35</td>
<td>35</td>
<td>30</td>
<td>0</td>
<td>Liu and Meller (2007)</td>
</tr>
<tr>
<td>SC35</td>
<td>35</td>
<td>16</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>0</td>
<td>Liu and Meller (2007)</td>
</tr>
</tbody>
</table>

Table 9
Benchmark approaches used in the unconstrained case.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type of method</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>GA-ST</td>
<td>GA with slicing-tree</td>
<td>Gau and Meller (1999)</td>
</tr>
<tr>
<td>I-MIP</td>
<td>Improved mixed-integer programming model</td>
<td>Sherali et al. (2003)</td>
</tr>
<tr>
<td>MIP-$\epsilon$</td>
<td>MIP $\epsilon$-accurate</td>
<td>Castillo and Westerlund (2005)</td>
</tr>
<tr>
<td>MIP-MINLP</td>
<td>Mixed-integer linear and mixed-integer nonlinear optimization methods</td>
<td>Castillo et al. (2005)</td>
</tr>
<tr>
<td>GA-SP-MIP</td>
<td>GA with sequence pair and MIP</td>
<td>Liu and Meller (2007)</td>
</tr>
<tr>
<td>STaTS</td>
<td>Tabu search with slicing-tree</td>
<td>Scholz et al. (2009)</td>
</tr>
<tr>
<td>ACO-ST</td>
<td>Ant colony optimization with slicing tree</td>
<td>Komarudin and Wong (2010)</td>
</tr>
<tr>
<td>AS-FBS</td>
<td>Ant system with flexible bay structure</td>
<td>Wong and Komarudin (2010)</td>
</tr>
<tr>
<td>ACO-ESFBS</td>
<td>Ant colony optimization with local search and flexible bay structure</td>
<td>Kulturel-Konak and Konak (2011b)</td>
</tr>
<tr>
<td>PSO-FS</td>
<td>Particle swarm optimization with relaxed flexible bay structure</td>
<td>Kulturel-Konak and Konak (2013a)</td>
</tr>
<tr>
<td>GA-LP</td>
<td>Linear programming based genetic algorithm</td>
<td>Kulturel-Konak and Konak (2013)</td>
</tr>
<tr>
<td>SA-MIP</td>
<td>Hybrid simulated annealing and MIP algorithm</td>
<td>Kulturel-Konak and Konak (2014)</td>
</tr>
</tbody>
</table>
Table 10
Experimental results: layout costs for datasets 07-09.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>I-MIP</th>
<th>MIP-ε</th>
<th>MIP-MINLP</th>
<th>GA-SP-MIP</th>
<th>STaTS</th>
<th>AS-FBS</th>
<th>BRKGA-LP</th>
</tr>
</thead>
<tbody>
<tr>
<td>07</td>
<td>131.64</td>
<td>131.57</td>
<td>131.64</td>
<td>131.63</td>
<td>132</td>
<td>131.68</td>
<td>131.56</td>
</tr>
<tr>
<td>08</td>
<td>242.89</td>
<td>242.77</td>
<td>242.73</td>
<td>242.88</td>
<td>243.16</td>
<td>243.12</td>
<td>242.92</td>
</tr>
<tr>
<td>09</td>
<td>235.95</td>
<td>235.87</td>
<td>236.14</td>
<td>235.95</td>
<td>239.07</td>
<td>236.14</td>
<td>236.57</td>
</tr>
</tbody>
</table>

Table 11
Experimental results: computing times for datasets 07-09.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>I-MIP</th>
<th>MIP-ε</th>
<th>MIP-MINLP</th>
<th>GA-SP-MIP</th>
<th>STaTS</th>
<th>AS-FBS</th>
<th>BRKGA-LP</th>
</tr>
</thead>
<tbody>
<tr>
<td>07</td>
<td>7700</td>
<td>2301</td>
<td>8221</td>
<td>13155</td>
<td>3055</td>
<td>1195</td>
<td>4132</td>
</tr>
<tr>
<td>08</td>
<td>43000</td>
<td>54443</td>
<td>18392</td>
<td>23600</td>
<td>6760</td>
<td>4248</td>
<td>10.18</td>
</tr>
<tr>
<td>09</td>
<td>60000</td>
<td>85255</td>
<td>82111</td>
<td>5384</td>
<td>5184</td>
<td>12.77</td>
<td></td>
</tr>
</tbody>
</table>

Table 12
Experimental results: layout costs for datasets F10-SC35.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Approaches</th>
<th>F10</th>
<th>VC10</th>
<th>BA12</th>
<th>BA14</th>
<th>AB20</th>
<th>TAM20</th>
<th>TAM30</th>
<th>SC30</th>
<th>SC35</th>
</tr>
</thead>
<tbody>
<tr>
<td>GA-ST</td>
<td>8485.4</td>
<td>4804.1</td>
<td>9513.5</td>
<td>20658</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACO-ZB</td>
<td>8607</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MIP-MINLP</td>
<td>21297.98</td>
<td>8180</td>
<td>5004</td>
<td>8180.00</td>
<td>3707</td>
<td>3604</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GA-SP-MIP</td>
<td>19997</td>
<td>8702</td>
<td>4712.33</td>
<td>5225.96</td>
<td>3866</td>
<td>6.2</td>
<td>4248</td>
<td>10.18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STaTS</td>
<td>19996.1</td>
<td>8264</td>
<td>4724.68</td>
<td>5073.82</td>
<td>3478</td>
<td>7.8</td>
<td>160</td>
<td>160.46</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AS-FBS</td>
<td>19996.1</td>
<td>8264</td>
<td>4724.68</td>
<td>5073.82</td>
<td>3478</td>
<td>7.8</td>
<td>160</td>
<td>160.46</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACO-LS-FSB</td>
<td>9020.75</td>
<td>8801.33</td>
<td>4904.67</td>
<td>5360.8</td>
<td>9003.82</td>
<td>19667.45</td>
<td>3794.82</td>
<td>4011.31</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSO-RFBS</td>
<td>9020.75</td>
<td>22899.65</td>
<td>4780.91</td>
<td>5336.36</td>
<td>8753.57</td>
<td>19462.41</td>
<td>3443.34</td>
<td>3700.75</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GA-LP</td>
<td>7651.28</td>
<td>8021</td>
<td>4686.81</td>
<td>5196.38</td>
<td>8058.06</td>
<td>19009.9</td>
<td>3370.98</td>
<td>3385.48</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SA-MIP</td>
<td>7650.95</td>
<td>8020.97</td>
<td>4628.79</td>
<td>5021.17</td>
<td>7986.48</td>
<td>18740.3</td>
<td>3367.87</td>
<td>3316.77</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BRKGA-LP</td>
<td>7650.95</td>
<td>8020.97</td>
<td>4628.79</td>
<td>5021.17</td>
<td>7986.48</td>
<td>18740.3</td>
<td>3367.87</td>
<td>3316.77</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Best values are in bold.

Table 13
Experimental results: computing times for datasets F10-SC35.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Approaches</th>
<th>F10</th>
<th>VC10</th>
<th>BA12</th>
<th>BA14</th>
<th>AB20</th>
<th>TAM20</th>
<th>TAM30</th>
<th>SC30</th>
<th>SC35</th>
</tr>
</thead>
<tbody>
<tr>
<td>GA-ST</td>
<td>412</td>
<td>396</td>
<td>499</td>
<td>1397</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACO-ZB</td>
<td>26300</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STaTS</td>
<td>8.9</td>
<td>13.6</td>
<td>16.1</td>
<td>13.6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACO-LS-FSB</td>
<td>1008</td>
<td>4104</td>
<td>8568</td>
<td>17820</td>
<td>23544</td>
<td>83988</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSO-RFBS</td>
<td>51</td>
<td>146</td>
<td>131</td>
<td>106</td>
<td>226</td>
<td>623</td>
<td>902</td>
<td>1185</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SA-MIP</td>
<td>2</td>
<td>3</td>
<td>10</td>
<td>19</td>
<td>85</td>
<td>104</td>
<td>924</td>
<td>873</td>
<td>1842</td>
<td></td>
</tr>
<tr>
<td>BRKGA-LP</td>
<td>49.15</td>
<td>46.40</td>
<td>113.81</td>
<td>160.46</td>
<td>426.09</td>
<td>686.00</td>
<td>3004.55</td>
<td>998.74</td>
<td>1556.03</td>
<td></td>
</tr>
</tbody>
</table>

or very close to 0%. Tables 10 and 12 present the experimental results, respectively, for the datasets 07 to 09 and F10 to SC35. For all the best found solutions we report $\%E_{\text{avg}}$ and $\%E_{\text{max}}$. As can be observed, the largest value of $\%E_{\text{max}}$ was 0.0095%, which for a facility with an area of 100 units corresponds to a negligible reduction of only 0.0095 units.

Table 10 reports the best cost solutions found by BRKGA-LP, over 10 runs, and six other approaches from the literature for datasets 07 to 09. Even though the computational times reported by the other approaches might not be comparable, we report them, if available, in Table 11.

The best costs obtained by BRKGA-LP are very similar to those of the other approaches. The best solutions reported in Table 10 vary only slightly; this may be due to different approximations of the area of facilities (e.g., Castillo and Westerlund (2005) reports the best values but has a $\%E_{\text{max}}$ equal to 0.05, 0.15 and 0.30, respectively, for 07, 08 and 09, while BRKGA-LP has $\%E_{\text{max}}$ values equal to 0.0030, 0.0012, and 0.0040 which are at least one order of magnitude smaller).

As can be observed in Table 11 the computational times of BRKGA-LP and STaTS are small and similar and vary from 4.3 s to 12.77 s. However, the other approaches run in computational times that are between 115 and 6676 times greater than those of BRKGA-LP.

The final solutions generated by BRKGA-LP for datasets 07-09 are shown in the appendix in Figs. 16 and 17.

Table 12 reports the cost for the best solutions found by BRKGA-LP, over 10 runs, and ten other approaches from the literature for datasets F10 to SC35. Even though the computational times reported
by the other approaches might not be comparable we report them, if available, in Table 13.

As can be observed in Table 12 the layout costs obtained by BRKGA-LP, with the exception of those for dataset SC30, are better than or equal to (less than 0.1% deviation) the best found costs reported by any other approach in this study. BRKGA-LP improved the best known solution costs for datasets VC10, BA14, TAM20, TAM30, and SC35 by, 0.21%, 1.24%, 0.89%, 1.42%, and 2.03%, respectively. Given that these datasets have been extensively studied in the literature, the improvements over the previously best-known solutions are significant. Also, given the small area approximation errors for these datasets, we believe that the achieved improvements cannot be attributed to the area approximation.
Fig. 18. BRKGA-LP solutions for datasets VC10 and BA12.
Fig. 19. BRKGA-LP solutions for datasets BA14 and AB20.
Fig. 20. BRKGA-LP solutions for datasets TAM20 and TAM30.
Fig. 21. BRKGA-LP solutions for datasets SC30 and SC35.
The computational times reported in Table 13 show that BRKGA-LP takes significantly less computing time when compared with the similar approaches MIP-MINLP, GA-SP-MIP, GA-LP, and SA-MIP. However, the approaches using a smaller search domain with slicing trees or flexible bays, like PSO-RBFs (of Kulturel-Konak & Konak (2011a)) and ACO-LS-FSB (of Kulturel-Konak & Konak (2011b)) use significantly less computing time but generate solutions with higher layout costs. The final solutions generated by BRKGA-LP for datasets F10 to SC35 are shown in Figures 17, 18, 19, 20, and 21.

5. Concluding remarks

In this paper we present a biased random-key genetic algorithm (BRKGA) for the unequal area facility layout problem where a set of rectangular facilities with a given area requirement have to be placed, without overlapping, on a rectangular floor space, so as to minimize the quadratic cost of products of inter-facility flows and inter-facility distances. The hybrid approach combines a BRKGA, to determine the order of placement and the dimensions of each facility, a novel placement strategy, to position each facility, and a linear programming model, to fine-tune the solutions. The unconstrained version of the approach generates high-quality solutions in relatively small computing times. The constrained version of the algorithm, BRKGA-LP, uses the solutions generated by BRKGA and tries to improve them in terms of cost and feasibility using a linear programming model. The approach is tested on 100 random datasets and 28 benchmark datasets taken from the literature and compared against 21 other solution approaches proposed in the literature. The unconstrained version BRKGA improved the best known solutions for 14 of the 16 benchmark datasets while the constrained version BRKGA-LP improved the best known solutions for 5 of the 12 extensively studied benchmark datasets.

Acknowledgments

This work has been partially supported by projects PTDCE/EG/GE/117692/2010 and NORTE-07-0124-FEDER-000057 funded by the North Portugal Regional Operational Programme (ON.2 – O Novo Norte), under the National Strategic Reference Framework (NSRF), through the European Regional Development Fund (ERDF) and the Programme COMPETE, and by national funds, through the Portuguese funding agency, Fundação para a Ciência e a Tecnologia (FCT). The work of Maurício C.G. Resende was done while he was employed by AT&T Labs Research in Middletown, New Jersey, U.S.A.

References


