European Journal of Operational Research 246 (2015) 86–107

Contents lists available at ScienceDirect

European Journal of Operational Research

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

Production, Manufacturing and Logistics

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

José Fernando Gonçalves[a,*], Mauricio G. C. Resende[b]

[a] LIAAD, INESC TEC, Faculdade de Economia, Universidade do Porto Rua Dr. Roberto Frias s/n, 4200-464, Porto, Portugal
[b] Mathematical Optimization and Planning, Amazon.com, 333 Boren Avenue North, Seattle, WA 98109, USA

ARTICLE INFO

*Article history:*
Received 5 September 2014
Accepted 19 April 2015
Available online 24 April 2015

*Keywords:*
Facilities planning and design
Facility layout
Biased random-key genetic algorithms
Random-keys

ABSTRACT

This paper presents a biased random-key genetic algorithm (*BRKGA*) for the unequal area facility layout problem (*UA-FLP*) where a set of rectangular facilities with given area requirements has to be placed, without overlapping, on a rectangular floor space. The objective is to find the location and the dimensions of the facilities such that the sum of the weighted distances between the centroids of the facilities is minimized. A hybrid approach combining 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, is developed. The proposed approach is tested on 100 random datasets and 28 of benchmark datasets taken from the literature and compared with 21 other benchmark approaches. The quality of the approach was validated by the improvement of the best known solutions for 19 of the 28 extensively studied benchmark datasets.

## 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 flow between the centroids. 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 $\varepsilon$-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. Konak, Kulturel-Konak, Norman,

☆ This work has been partially supported by projects PTDC/EGE-GES/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).

* Corresponding author. Tel.: +351937061379.
*E-mail addresses:* jfgoncal@fep.up.pt, jfgoncal@gmail.com (J.F. Gonçalves), resendem@amazon.com (M.G.C. Resende).

http://dx.doi.org/10.1016/j.ejor.2015.04.029
0377-2217/© 2015 Elsevier B.V. and Association of European Operational Research Societies (EURO) within the International Federation of Operational Research Societies (IFORS). All rights reserved.

and Smith (2006) modeled the facility area constraints exactly using a set of linear constraints derived from the structure of the flexible bay structure representation. They report solving problems having up to 14 facilities. Banerjee, Montreuil, Moodie, and Kashyap (1992), Montreuil, Venkatadri, and Ratliff (1993), and Banerjee, Zhou, and Montreuil (1997) used design skeletons to reduce the complexity of MIP formulations for the FLP. Lacksonen (1997) fixed the orientations of obvious facility pairs using a preprocessing heuristic.

- *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 Domschke (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 Nourelfath (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 a 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, $R_{\max}$, which due to practical reasons imposes the maximum permissible ratio between its longest and shortest dimensions, i.e., $R_{\max} \geq \max\{w_i, h_i\} / \min\{w_i, h_i\} \geq 1$.

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

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

where $f_{i,j}$ represents the flow between facilities $i$ and $j$ (we assume that $f_{i,i} = 0$), $c_{i,j}$ is the cost per unit distance between $i$ and $j$, and $d_{i,j}$ is the distance between the centroids of facilities $i$ and $j$. The distance $d_{i,j}$ can be defined according to one of the following distance norms:

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

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 $\triangle w_i$ and $\triangle h_i$ 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 nonlinear mixed integer programming model, FLP-NMIP. In the model we will use the following additional notation:

$P_{i,j}, Q_{i,j}$ = Binary variables used to model the non-overlapping constraints:

- if $(P_{i,j}, Q_{i,j})$ is equal to $(0, 0)$ then facility $i$ is forced to the right of $j$;

- if $(P_{i,j}, Q_{i,j})$ is equal to $(1, 0)$ then facility $i$ is forced to the left of $j$;
- if $(P_{i,j}, Q_{i,j})$ is equal to $(0, 1)$ then facility $i$ is forced to the above of $j$;
- if $(P_{i,j}, Q_{i,j})$ is equal to $(1, 0)$ then facility $i$ is forced to the below of $j$.

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

$d^x_{ij}, d^x_{ij}$, = 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 dimensions. The value of this constant should be chosen according to the available flexibility to accommodate small variations in the dimensions of 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:

$$-h_i - \frac{A_i}{\overline{w}^2_{i,k}} w_i \leq 2\frac{A_i}{\overline{w}_{i,k}}, \qquad k = 0, \dots, C_i, \qquad \forall i \tag{1}$$

where $\overline{w}_{i,k}$ 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.

$$\text{(FLP-NMIP)}, \quad \text{Minimize} \quad \text{Cost} = \sum_{i=1}^{i=N}\sum_{j=1}^{j=N} c_{i,j} f_{i,j}\left(d^x_{i,j} + d^x_{i,j}\right) + M\sum_{i=1}^{i=N}(\triangle w_i + \triangle h_i) \tag{2}$$

**Subject to:**
**Facility aspect constraints**

$$w_i \times h_i = A_i \quad \forall i \tag{3}$$

$$\frac{1}{R_{\max}} \leq \frac{w_i}{h_i} \leq R_{\max} \quad \forall i \tag{4}$$

**Non-overlapping constraints**

$$x_i - x_j + M^x\left(P_{i,j} + Q_{i,j}\right) \geq \frac{w_i + w_j}{2} \quad \forall i, j \mid j > i \tag{5}$$

$$x_j - x_i + M^x\left(1 - P_{i,j} + Q_{i,j}\right) \geq \frac{w_i + w_j}{2} \quad \forall i, j \mid j > i \tag{6}$$

$$y_i - y_j + M^y\left(1 + P_{i,j} - Q_{i,j}\right) \geq \frac{h_i + h_j}{2} \quad \forall i, j \mid j > i \tag{7}$$

$$y_j - y_i + M^y\left(2 - P_{i,j} - Q_{i,j}\right) \geq \frac{h_i + h_j}{2} \quad \forall i, j \mid j > i \tag{8}$$

**Floor space constraints**

$$\frac{w_i}{2} \leq x_i \leq W - \frac{w_i}{2} + \triangle w_i \quad \forall i \tag{9}$$

$$\frac{h_i}{2} \leq y_i \leq H - \frac{h_i}{2} + \triangle h_i \quad \forall i \tag{10}$$

**Distance constraints**

$$x_i - x_j \leq d^x_{i,j} \quad \forall i, j \mid j > i \tag{11}$$

$$x_j - x_i \leq d^x_{i,j} \quad \forall i, j \mid j > i \tag{12}$$

$$y_i - y_j \leq d^y_{i,j} \quad \forall i, j \mid j > i \tag{13}$$

$$y_j - y_i \leq d^y_{i,j} \quad \forall i, j \mid j > i \tag{14}$$

**Domain constraints**

$$x_i, y_i, w_i, h_i, d^x_{i,j}, d^x_{i,j} \geq 0 \quad \forall i \quad \text{and} \quad P_{i,j}, Q_{i,j} \in \{0, 1\} \quad \forall i, j \mid j > i \tag{15}$$

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.

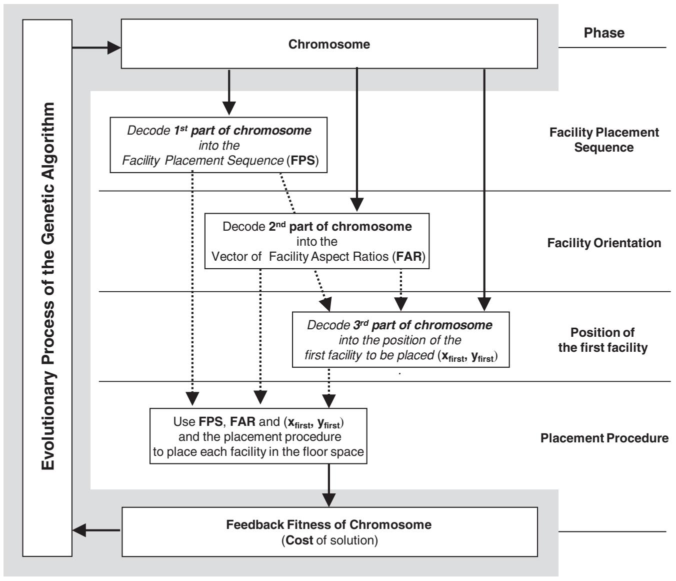

Phase
Chromosome
Evolutionary Process of the Genetic Algorithm
Decode 1st part of chromosome into the Facility Placement Sequence (FPS)
Facility Placement Sequence
Decode 2nd part of chromosome into the Vector of Facility Aspect Ratios (FAR)
Facility Orientation
Decode 3rd part of chromosome into the position of the first facility to be placed (x_first, y_first)
Position of the first facility
Use FPS, FAR and (x_first, y_first) and the placement procedure to place each facility in the floor space
Placement Procedure
Feedback Fitness of Chromosome (Cost of solution)

**Fig. 1.** Architecture of the algorithm.

3. *Position of the first facility decoder*: The third phase decodes part of the chromosome into the coordinates $(x_{\text{first}}, y_{\text{first}})$ of the first facility to be placed on the floor space.
4. *Placement strategy*: The fourth phase makes use of FPS, FAR, and$(x_{\text{first}}, y_{\text{first}})$ defined in Phases 1, 2, and 3, respectively, and places all the facilities on the floor space using the novel placement strategy.
5. *Fitness evaluation*: The final phase computes the fitness of the solution obtained in Phase 4 (a measure of quality of the facility placement) using Eq. (2).

Fig. 1 illustrates the sequence of steps applied to each chromosome generated by the BRKGA.

The remainder of this section describes the genetic algorithm, the decoding procedure, and the placement strategy in detail.

### 3.2. Biased random-key genetic algorithms

Genetic algorithms with random keys, or *random-key genetic algorithms* (RKGA), for solving sequencing problems were introduced in Bean (1994). In a RKGA, chromosomes are represented as vectors of randomly-generated real numbers in the interval [0, 1]. A *decoder* is a deterministic algorithm that takes as input a chromosome and associates with it a solution of the combinatorial optimization problem for which an objective value or *fitness* can be computed.

In a RKGA evolves a *population* of random-key vectors over a number of *generations* (iterations). The initial population is made up of $p$ vectors of $r$ random keys. Each component of the solution vector, or random key, is generated independently at random in the real interval [0, 1]. After the fitness of each individual is computed by the decoder in generation $g$, the population is partitioned into two groups of individuals: a small group of $p_e$ *elite* individuals, i.e. those with the best fitness values, and the remaining set of $p - p_e$ *non-elite* individuals. To evolve the population of generation $g$, a new generation $(g + 1)$ of individuals is produced. All elite individuals of the population of generation $g$ are copied without modification to the population of generation $g + 1$. RKGAs implement mutation by introducing *mutants* into the population. A mutant is a vector of random keys generated in the same way that an element of the initial population is generated. Its role is similar to that of mutation in other genetic algorithms (Goldberg, 1989), i.e. to introduce noise into the population and avoid convergence of the entire population to a local

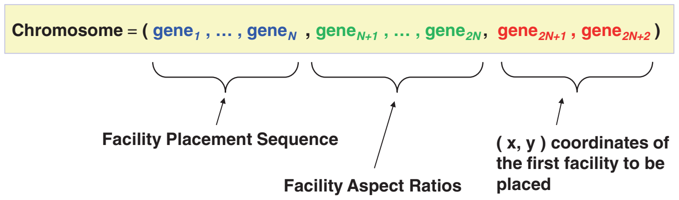

Chromosome = ( gene_1 , ... , gene_N , gene_N+1 , ... , gene_2N , gene_2N+1 , gene_2N+2 )
Facility Placement Sequence
Facility Aspect Ratios
( x, y ) coordinates of the first facility to be placed

**Fig. 2.** Solution encoding.

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 $\rho_e$ 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 $c(i)$ of the offspring vector $c$ takes on the value of the $i$-th component $e(i)$ of the elite parent $e$ with probability $\rho_e$ and the value of the $i$-th component $\bar{e}(i)$ of the non-elite parent $\bar{e}$ with probability $1 - \rho_e$. While in a BRKGA $\rho_e > \frac{1}{2}$, 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,

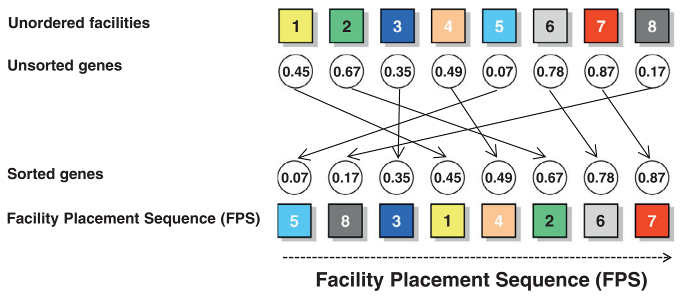


**Fig. 3.** Decoding of the facility placement sequence.

described in Section 3.3.3, makes use of FPS, FAR, and $(x_{\text{first}}, y_{\text{first}})$ to construct a solution from the chromosome.

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 $i = 1, \ldots, N$, using the following expression:

$$\text{FAR}_i = \frac{1}{R_{\max}} + gene_i \times \left(R_{\max} - \frac{1}{R_{\max}}\right). \tag{16}$$

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

$$w_i = \sqrt{A_i \times \text{FAR}_i}\,, \tag{17}$$

and

$$h_i = A_i / w_i. \tag{18}$$

The *first* facility to be laid out is defined by the FPS and is denoted asFPS$_1$. Its coordinates, i.e. $(x_{\text{first}}, y_{\text{first}})$ are decoded as:

$$x_{\text{first}} = \frac{w_i}{2} + gene_{2N+1} \times (W - w_i), \tag{19}$$

$$y_{\text{first}} = \frac{h_i}{2} + gene_{2N+2} \times (H - h_i). \tag{20}$$

FPS, FAR, and $(x_{\text{first}}, y_{\text{first}})$ 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

$$\text{Cost} = \sum_{i=1}^{i=N} \sum_{j=1}^{j=N} c_{i,j} f_{i,j}\, d_{i,j} + M \sum_{i=1}^{i=N} (\triangle w_i + \triangle h_i),$$

where, as defined above, $c_{i,j}$ and $f_{i,j}$ are respectively, the cost per distance unit and the flow between facilities $i$ and $j$, and $d_{i,j}$ 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 $(\text{min\_}x_s,\ \text{min\_}y_s$ and $\text{max\_}x_s,\ \text{max\_}y_s$, respectively). 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 $EMS_0$ which is the entire floor space (see Fig. 4a). After placing facility 1 in $EMS_0$ 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 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.,

$$\text{EMS}_0 = \left(0\,,\ 0,\ \sum_i \sqrt{A_i \times R_{\max}}\,,\ \sum_i \sqrt{A_i \times R_{max}}\right).$$

### 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:

$$(\text{FLP\_EMS}) \quad \textbf{Minimize} \quad \text{Cost}(i) = \sum_{k \in K} c_{i,k} f_{i,k}\, d_{i,k} \tag{21}$$

**Subject to**:

$$\text{min\_}x_s + \frac{w_i}{2} \le x_i \le \text{max\_}x_s - \frac{w_i}{2}, \tag{22}$$

$$\text{min\_}y_s + \frac{h_i}{2} \le y_i \le \text{min\_}y_s - \frac{h_i}{2}. \tag{23}$$

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, Martínez, and Raydan (2000) (FORTRAN source code in 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

$$\left(\frac{\text{min\_}x_s + \text{max\_}x_s}{2}\,,\ \frac{\text{min\_}y_s + \text{max\_}y_s}{2}\right).$$

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.,

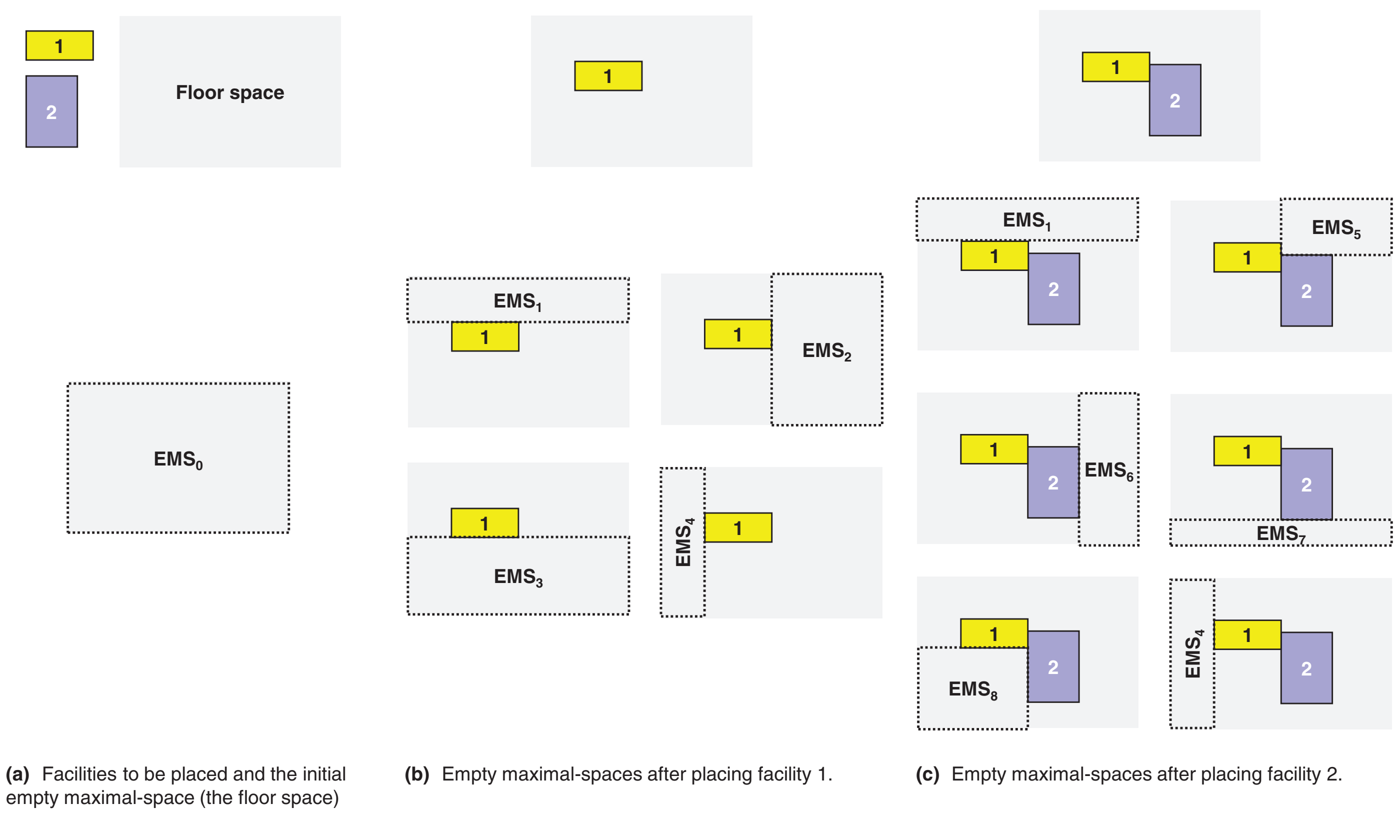

1
2
Floor space
EMS$_0$
EMS$_1$
EMS$_2$
EMS$_3$
EMS$_4$
EMS$_5$
EMS$_6$
EMS$_7$
EMS$_8$
(a) Facilities to be placed and the initial empty maximal-space (the floor space)
(b) Empty maximal-spaces after placing facility 1.
(c) Empty maximal-spaces after placing facility 2.

**Fig. 4.** Difference process (DP) example.

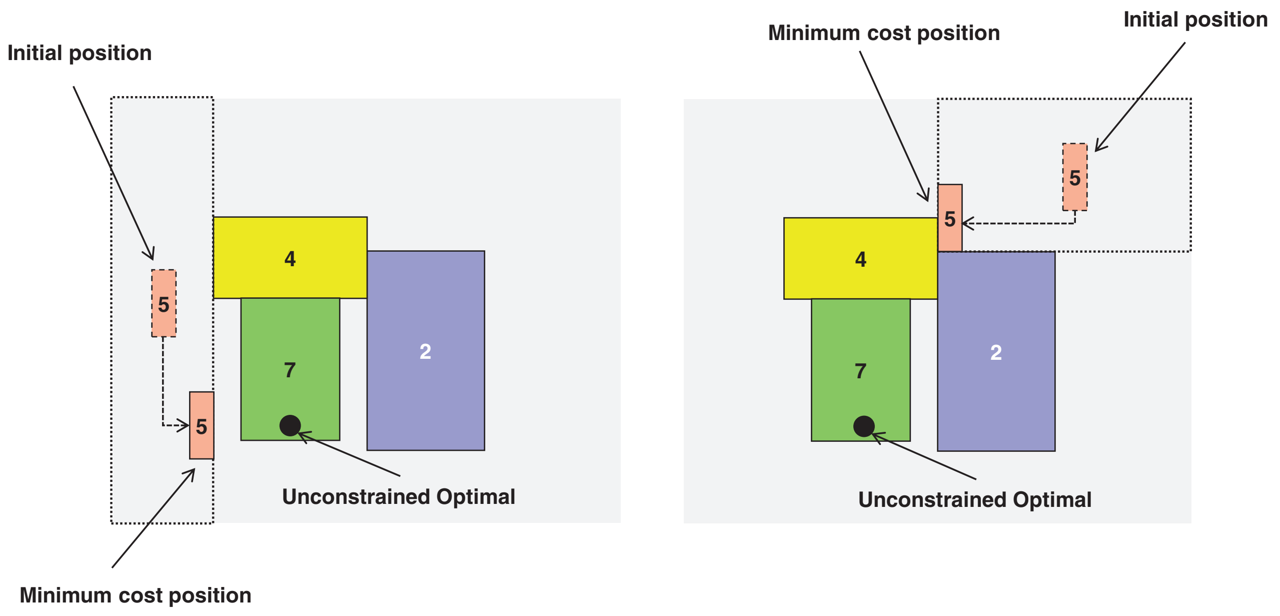


**Fig. 5.** Example of the novel approach used to solve FLP_EMS.

$f_{i,k} = 0 \ \forall \, k \in K$. 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.

```
procedure PLACEMENT ($FPS$, $FAR$, $x_{first}$, $y_{first}$)
1  Let $S$ be the set of $EMSs$ available;

   // ** Initialization
2  $S \leftarrow \{FloorSpace\}$;

3  for $i = 1, \ldots, N$ do

·      // ** Facility selection
4      $FacToPlace \leftarrow FPS_i$ // select the facility in the $i^{th}$ position of $FPS$;

·      // ** Computation of facility aspect ratio and dimensions
5      Compute $FAR_i$, $w_i$ and $h_i$, using equations 16, 17 , and 18, respectively;

·      // ** Facility placement
6      if $i > 1$ then // not first facility
·          // Place facility $FacToPlace$ in the position having the minimum cost
7          $BestCost = \infty$;
8          for all $s$ in $EMSs$ do
9              $Cost = FLP\_EMS(FacToPlace, s, x, y)$;
10             if $Cost \le BestCost$ then
11                 $BestCost = Cost$;
12                 $x^* = x$, $y^* = y$;
13             endif
14         end for
15     else
16         $x^* = x_{first}$, $y^* = y_{first}$
17     endif

18     Place the centroid of $FacToPlace$ at position ($x^*$, $y^*$);

·      // ** State information update
19     Update the list of $S$ of using
·              the DP procedure of Lai and Chan (1997);

20 end for
end PLACEMENT;
```

**Fig. 6.** Pseudo-code for the PLACEMENT procedure.

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, usingthe 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.,

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

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 floor space 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 where 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 $S^{\mathrm{BRKGA}}$ 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 $S^{\mathrm{BRKGA}}$ to be equal to $\infty$ and skip Steps 2 and 3.
2. Based on the solution $S^{\mathrm{BRKGA}}$ obtained in the previous step, formulate the corresponding linear model FLP-LP($S^{\mathrm{BRKGA}}$), where the non-overlapping constraints (5), (6), (7), and (8) in the FLP-MIP model are replaced by a single constraint defined according to $S^{\mathrm{BRKGA}}$. The single separating constraint that will replace the original separating constraints in the FLP-MIP model can

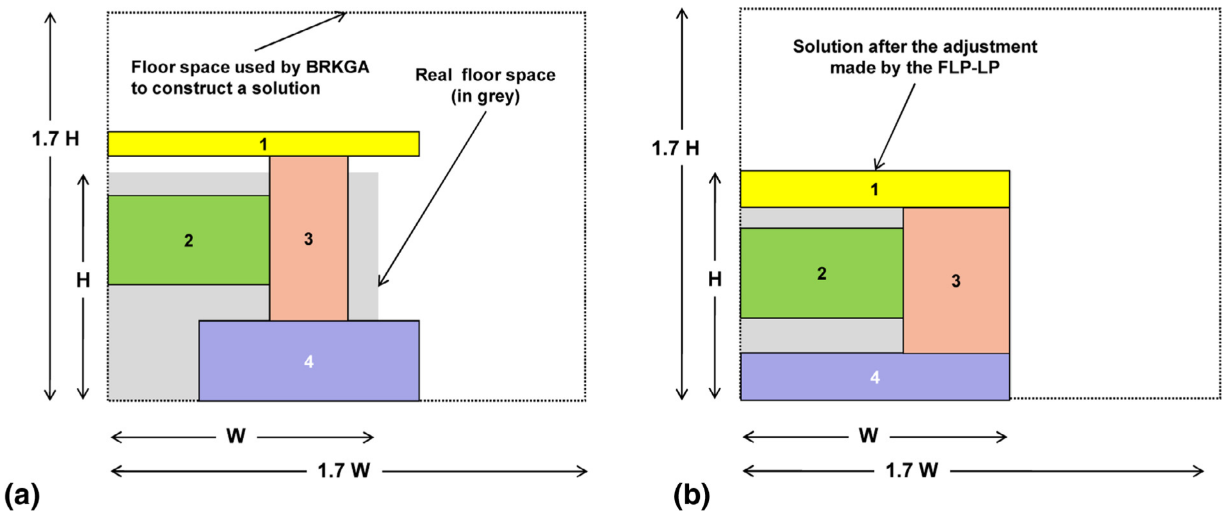
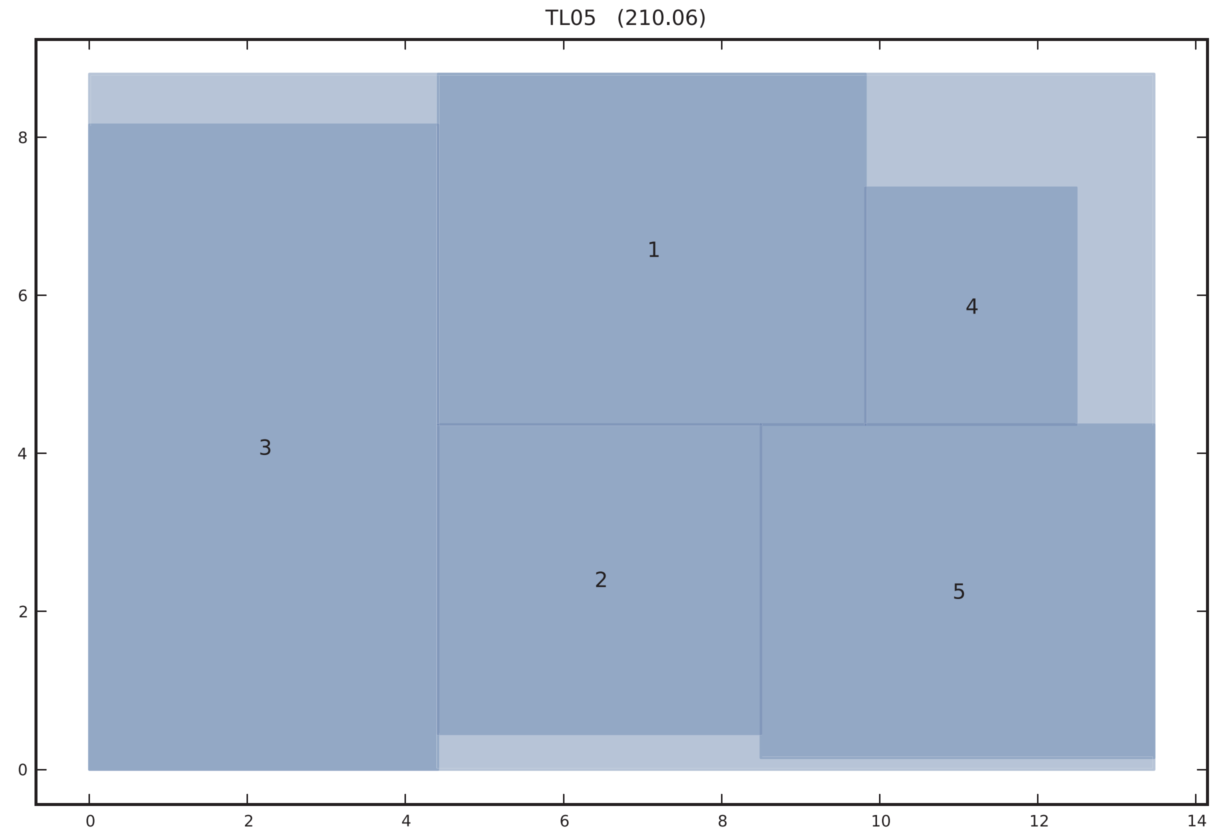

Floor space used by BRKGA to construct a solution
Real floor space (in grey)
1.7 H
H
1
2
3
4
W
1.7 W
(a)
Solution after the adjustment made by the FLP-LP
1.7 H
H
1
2
3
4
W
1.7 W
(b)

**Fig. 7.** BRKGA solution in a) and the corresponding *FLP-LP* improved solution in b).



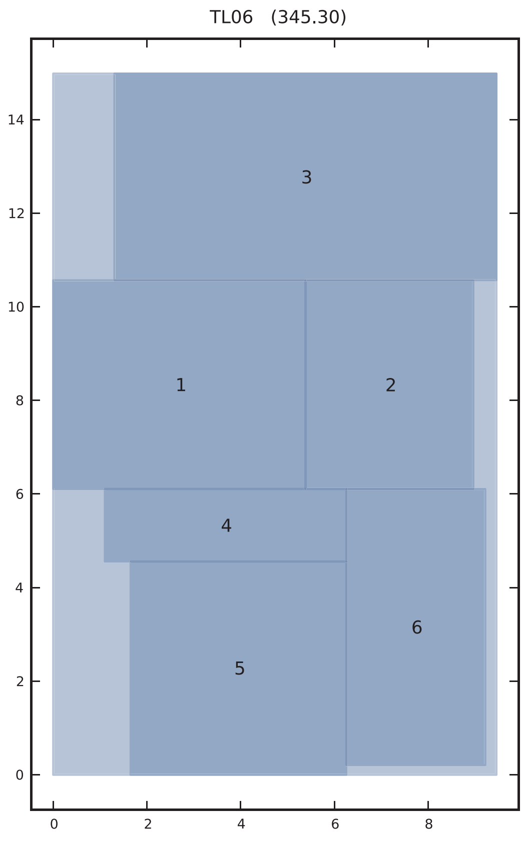


**Fig. 8.** BRKGA solutions for datasets TL05 and TL06.

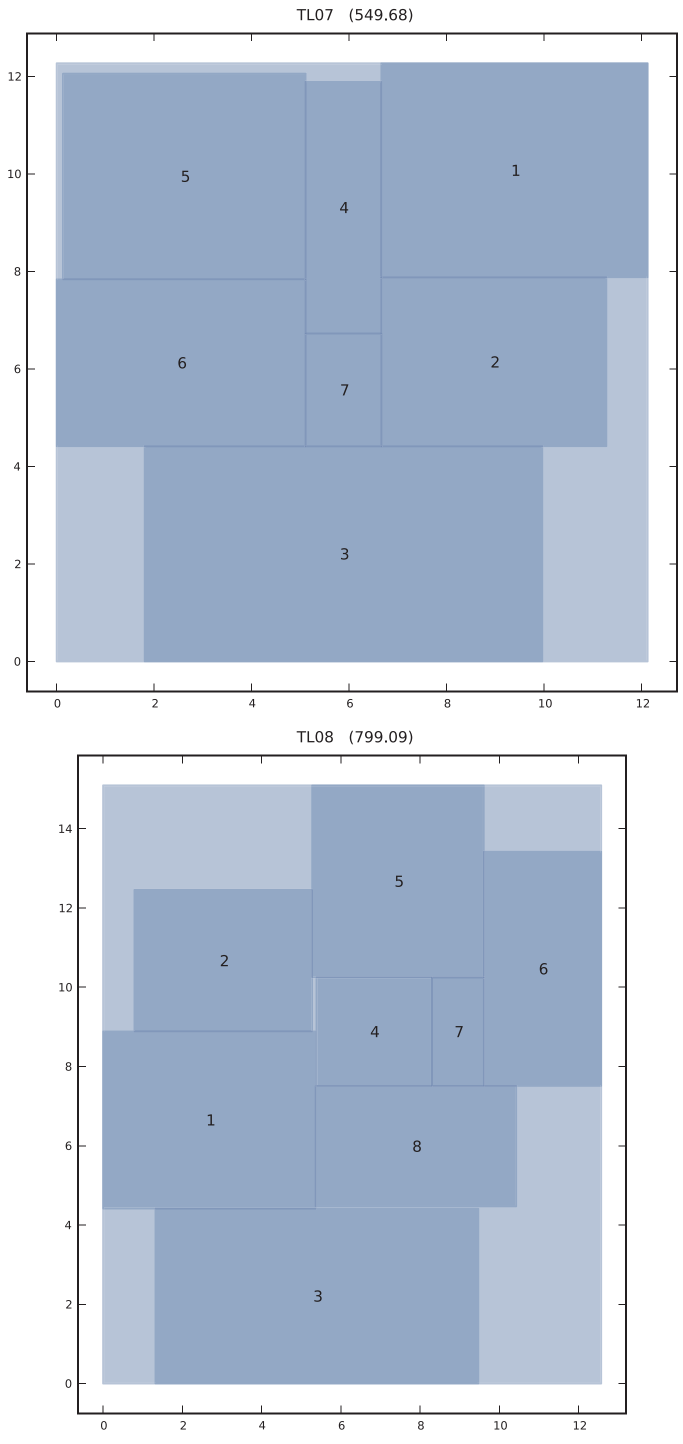


**Fig. 9.** BRKGA solutions for datasets TL07 and TL08.

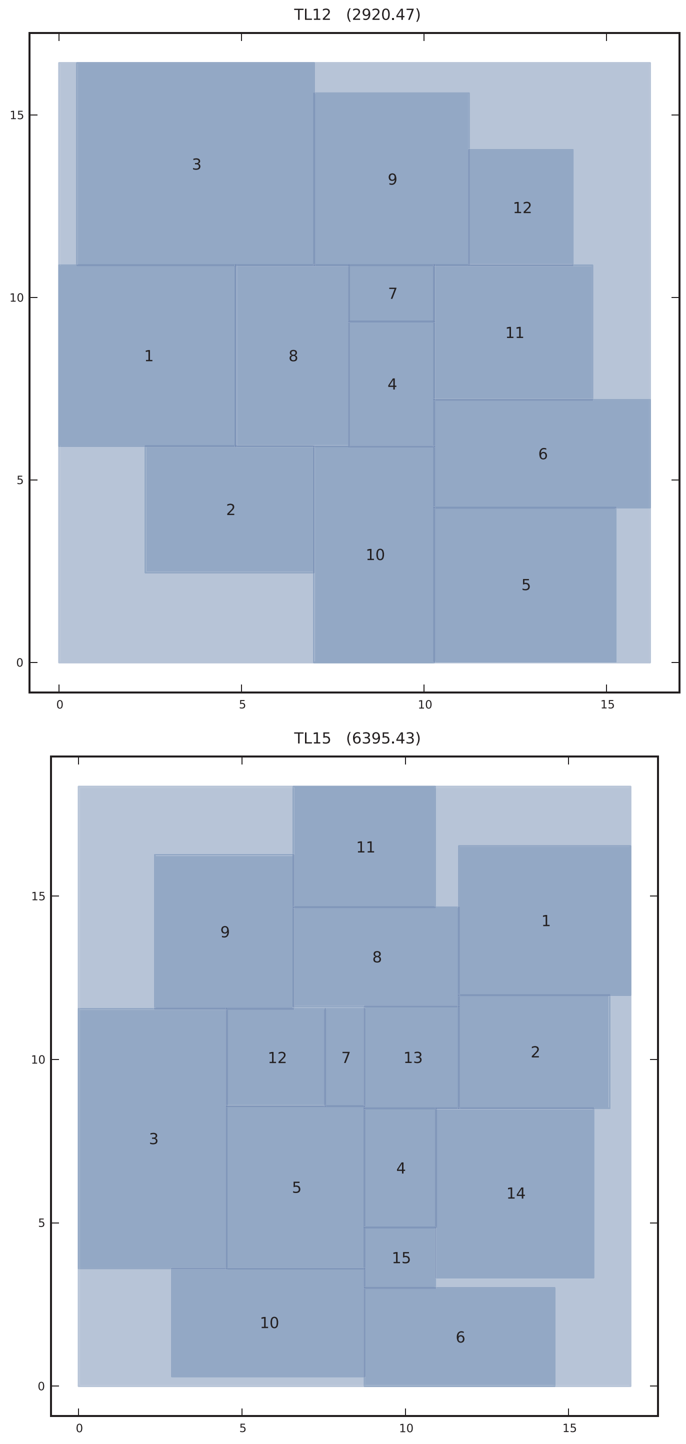


**Fig. 10.** BRKGA solutions for datasets TL12 and TL15.

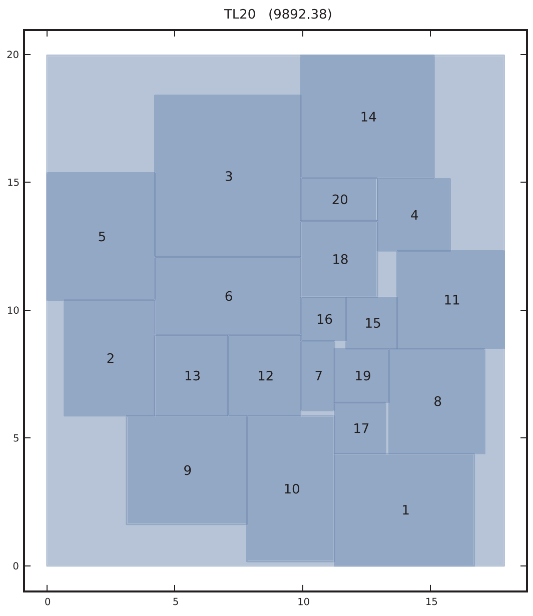
be determined as follows. Let $DisW_{i,j} = |x_i - x_j| - \frac{w_i}{2} - \frac{w_j}{2}$ and $DistH_{i,j} = |y_i - y_j| - \frac{h_i}{2} - \frac{h_j}{2}$ be, respectively, the horizontal and vertical distances separating facilities *i* and *j*. If $DistW_{i,j} \geq DistH_{i,j}$, then separate facilities *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 $x_j + \frac{w_i}{2} + \frac{w_j}{2} \leq x_i$. If $DistW_{i,j} < DistH_{i,j}$, 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 $y_j + \frac{h_i}{2} + \frac{h_j}{2} \leq y_i$. 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.,

Area of the facilities outside of the real floor space $\leq 0.45 \times W \times H$.



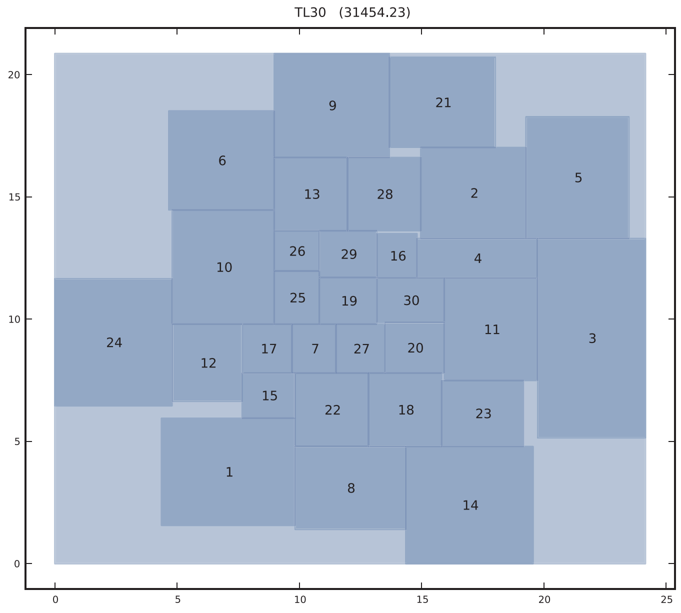
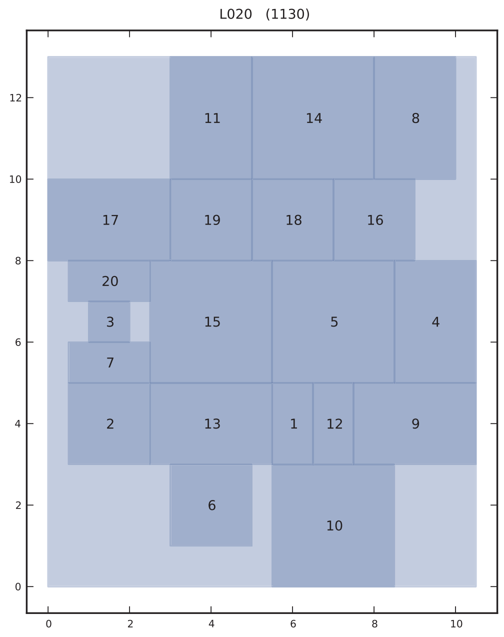


**Fig. 11.** BRKGA solutions for datasets TL20 and TL30.



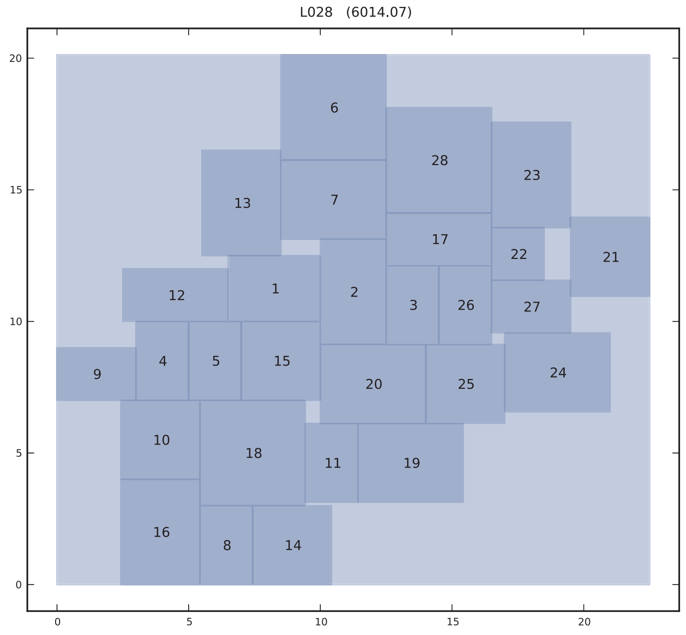


**Fig. 12.** BRKGA solutions for datasets L020 and L028.

2. The cost of the solution $S^{\text{BRKGA}}$ 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 $\infty$.

The next subsections report the details of the experiments and the results obtained by the approaches proposed in this paper.

## 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, Resende, and Mendes (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.

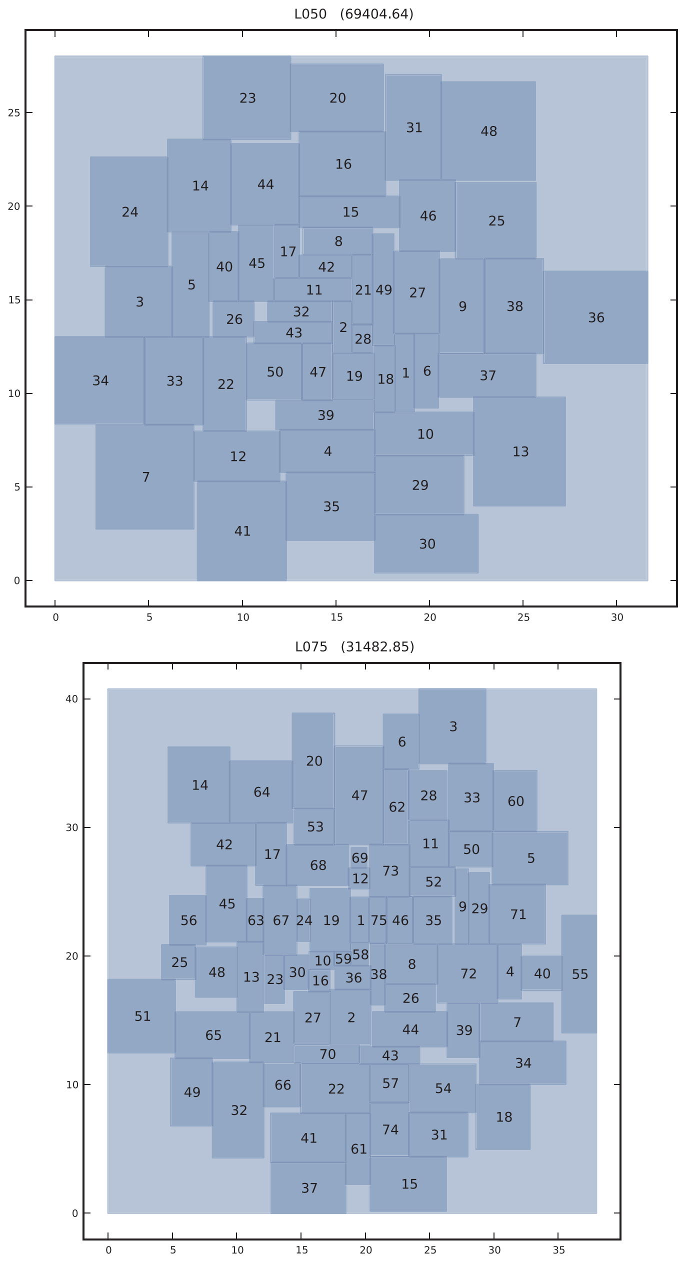


**Fig. 13.** BRKGA solutions for datasets L050 and L075.

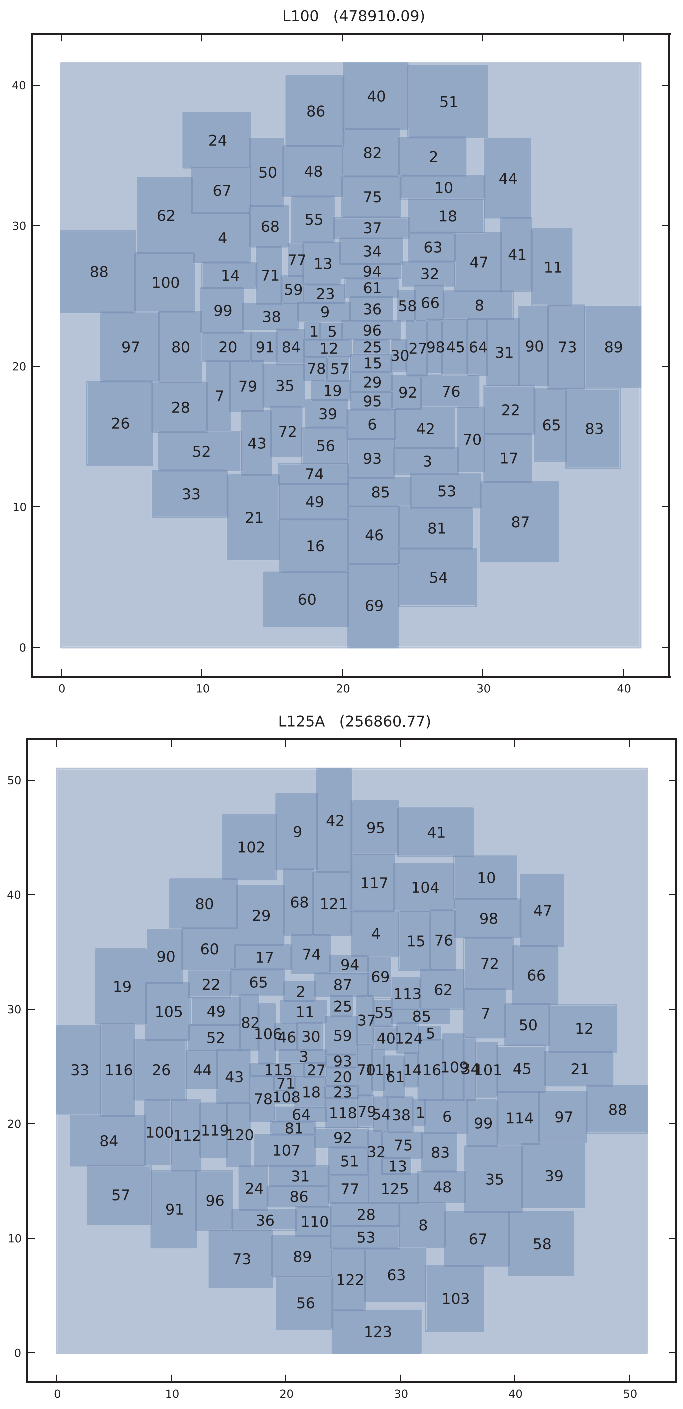


**Fig. 14.** BRKGA solutions for datasets L100 and L125A.

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.

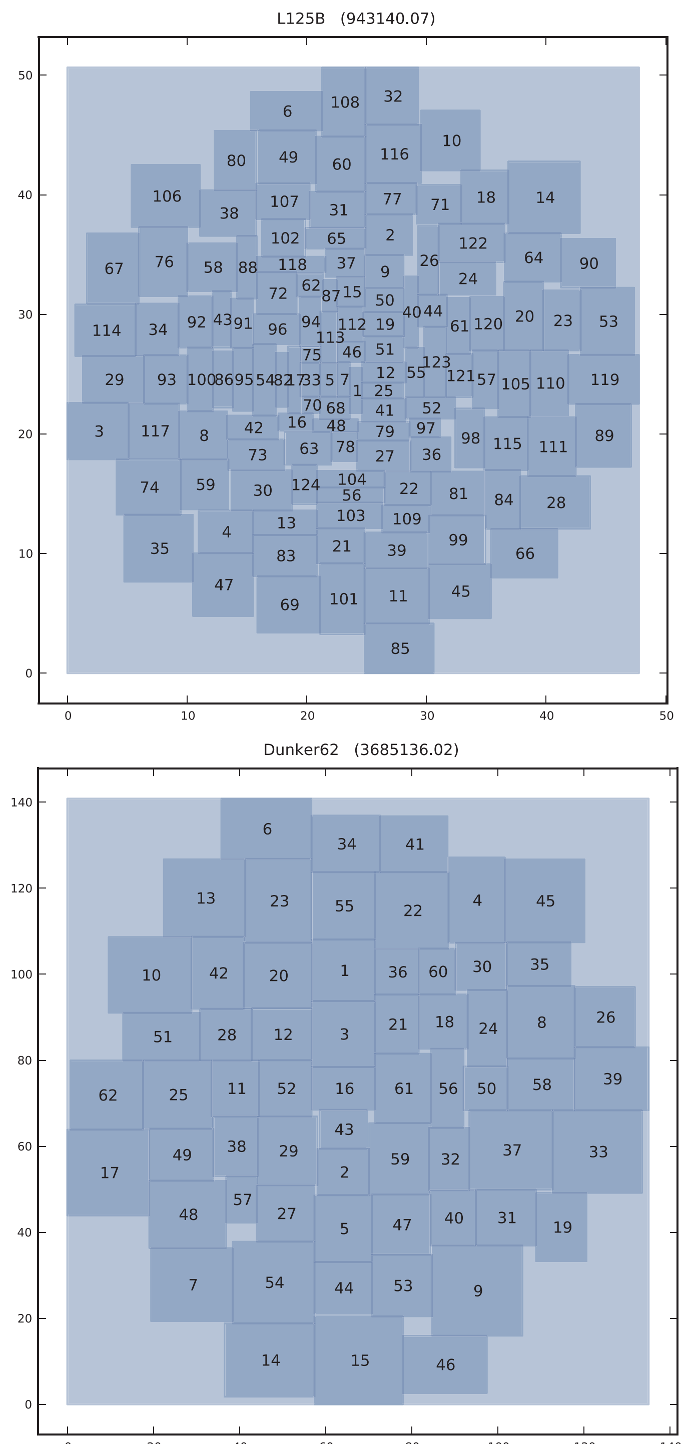


**Fig. 15.** BRKGA solutions for datasets L125B and Dunker62.

**Table 1**
Range of parameters for the evolutionary strategy.

| Parameter | Interval |
|---|---|
| $p_e$ | 0.10–0.25 |
| $p_m$ | 0.15–0.30 |
| Crossover probability ($\rho_e$) | 0.70–0.85 |

**Table 2**
Configuration parameters for the *BRKGA* algorithm.

| Parameter | Value |
|---|---|
| $p =$ | $100 \times N$ |
| $p_e =$ | $\min(0.25 \times p, 50)$ |
| $p_m =$ | $0.25 \times p$ |
| $\rho_e =$ | 0.70 |
| Fitness = | Cost of layout given by Eq. (2) (to be minimized) |
| Stopping criterion = | 50 generations for the unconstrained case and 100 generations for the constrained case |

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, TL 08, 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.

| Dataset | Distance measure | Description | Source |
|---|---|---|---|
| L020, L028, L050, L075, L100, L125A, L125B | **R** L020, L075, L100, L125A and L125B **SE** L028 **E** L050 | Seven datasets with sizes equal to 20, 28, 50, 75, 100, 125, and 125 respectively. There are no constraints in the floor space dimensions. | Imam and Mir (1993), Mir and Imam (1996), Imam and Mir (1998), Mir and Imam (2001) and VIP-PLANOPT (2010) available from the commercial software Engineering Optimization Software, VIP-PLANOPT 2006 and 2010 at http://www.planopt.com. |
| Dunker62 | **R** | A dataset with 63 facilities. There are no constraints in the floor space dimensions. | Dunker et al. (2003) |
| TL05-30 | **SE** | 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. | Tam and Li (1991) |
| RND10-100 | **R** | 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. | Available from the authors upon request. |

**Table 4**
Benchmark approaches used in the unconstrained case.

| Name | Type of method | Source |
|---|---|---|
| HA-C | Hierarchical approach with clusters. | Tam and Li (1991) |
| GA-STS | Genetic algorithm with a Slicing Tree Structure method. | Kado (1996) |
| GP-STS | Genetic programming algorithm that generates Slicing Tree Structures later converted to FLP solutions. | Garces-Perez et al. (1996) |
| GA-TSG | Genetic algorithms with tree-structured genotype representation and hybrid problem-specific operators. | Schnecke and Vornberger (1997) |
| TSaST | Tabu search with slicing-tree. | Scholz et al. (2009) |
| VIP-PLANOPT | 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. | http://www.planopt.com |
| TS-BST | Tabu search with boundary search technique. | McKendall Jr and Hakobyan (2010) |
| GUROBI | Version 5.5 of commercial software solver from Gurobi Optimization. | http://www.gurobi.com |

**Table 5**
Experimental results: Datasets TL05 to TL30.

| | HA-C | GA-STS | GP-STS | GA-TSG | TSaST | | BRKGA | | |
|---|---|---|---|---|---|---|---|---|---|
| Dataset | Cost | Cost | Cost | Cost | Cost | $T$ (s) | Cost | $T$ (s) | %Imp. |
| TL05 | 247 | 228 | 226 | 214 | 213.48 | 2.3 | **210.06** | 0.035 | 1.60 |
| TL06 | 514 | 361 | 384 | **327** | 348.76 | 3.0 | 345.03 | 0.049 | −5.51 |
| TL07 | 559 | 596 | 568 | 629 | 562.91 | 2.5 | **549.68** | 0.060 | 1.67 |
| TL08 | 839 | 878 | 878 | 833 | 810.43 | 4.7 | **799.09** | 0.080 | 1.40 |
| TL12 | 3162 | 3283 | 3220 | 3164 | 3054.23 | 12.5 | **2920.47** | 0.162 | 4.38 |
| TL15 | **5862** | 7384 | 7510 | 6813 | 6615.81 | 17.0 | 6395.43 | 0.251 | −9.10 |
| TL20 | – | 16393 | 14033 | 13190 | 13198.40 | 50.0 | **9892.38** | 0.443 | 25.00 |
| TL30 | – | 41095 | 39018 | 35358 | 33721.20 | 95.4 | **31454.23** | 1.132 | 6.72 |

Best values are in bold.

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

| | VIP-PLANOPT | | TSaST | | TS-BST | | BRKGA | | |
|---|---|---|---|---|---|---|---|---|---|
| Dataset | Cost | $T$ (s) | Cost | $T$ (s) | Cost | $T$ (s) | Cost | $T$ (s) | %Imp. |
| L20 | 1157 | 0.3 | – | – | 1151.4 | 10351.86 | **1130.00** | 0.48 | 1.86 |
| L28 | 6447.25 | 1.5 | – | – | – | – | **6014.07** | 0.95 | 6.72 |
| L50 | 78224.68 | 7 | – | – | 71291.4 | 7626.52 | **69404.64** | 6.26 | 2.65 |
| L75 | 34396.38 | 13 | – | – | | | **31482.84** | 11.60 | 8.47 |
| L100 | 538193.1 | 14 | – | – | 496820.4 | 11397.19 | **478910.09** | 57.03 | 3.60 |
| L125A | 288774.6 | 110 | – | – | | | **256860.78** | 83.58 | 11.05 |
| L125B | 1084451 | 70 | – | – | 1008839 | 9250.28 | **943140.06** | 118.65 | 6.51 |
| Dunker62 | 3939362 | 4996 | 3871510 | 252.0 | 3812825 | 7304.05 | **3685136.02** | 9.07 | 3.35 |

Best values are in bold.

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

| Number of | GUROBI | | | BRKGA | | |
|---|---|---|---|---|---|---|
| facilities | $T$ (s) | $\%DO_{\text{avg}}$ | $\%DO_{\text{max}}$ | $T$ (s) | $\%DO_{\text{avg}}$ | $\%DO_{\text{max}}$ |
| 10 | 3600 | 0.21 | 1.66 | 1.76 | 0.00 | 0.00 |
| 20 | 3600 | 0.01 | 0.12 | 6.13 | 0.00 | 0.00 |
| 30 | 3600 | 0.32 | 2.14 | 15.00 | 0.00 | 0.00 |
| 40 | 3600 | 2.37 | 7.10 | 28.67 | 0.00 | 0.00 |
| 50 | 3600 | 3.99 | 9.30 | 48.30 | 0.11 | 1.12 |
| 60 | 3600 | 16.65 | 29.73 | 72.86 | 0.02 | 0.15 |
| 70 | 3600 | 12.21 | 22.70 | 102.90 | 1.44 | 5.29 |
| 80 | 3600 | 22.31 | 50.97 | 143.37 | 3.31 | 7.10 |
| 90 | 3600 | 36.11 | 52.99 | 186.87 | 6.00 | 9.09 |
| 100 | 3600 | 101.78 | 235.31 | 235.84 | 7.36 | 10.97 |

the literature. Table 8 summarizes the parameters of the datasets. These datasets were chosen because of their variety in size (from seven up to 35 facilities) and because they are frequently used in the literature to benchmark alternative approaches for solving the FLP.

Some authors have relaxed the dimensions of the facilities, or of the floor space, or both, when conducting the computational experiments. Although, in practice, adjusting dataset input parameters to find practical solutions is acceptable, modifying the datasets parameters makes it difficult to benchmark alternative approaches. BRKGA-LP finds feasible solutions for all datasets without modifying the original dataset parameters. Therefore, the BRKGA-LP will only be compared against previous approaches which use the original dataset parameters. Table 9 lists the approaches used for comparison with BRKGA-LP. We included approaches which do not impose any additional constraints in the solutions sought (Castillo and Westerlund (2005), Castillo, Westerlund, Emet, and Westerlund (2005), Liu and Meller (2007) and Kulturel-Konak and Konak (2013)) and the other approaches based on zone-based layouts, slicing-tree representation, and flexible bay structure representation which impose additional limitations in the search domain. The first group of approaches require longer computational times since they have a larger search domain but usually find better solutions. The second group of approaches takes less computing times due to the reduced search domain but usually generate worse solutions (with higher cost).

### 4.3.2. Experimental results

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 (△) 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:

$$\%E_{\text{avg}} = \frac{100\%}{N} \sum_{i=1}^{N} \frac{|A_i - w_i h_i|}{A_i},$$

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

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%

**Table 8**
Datasets used in the constrained case.

| Dataset | $N$ | Floor dimensions | | Facility requirements | Source |
|---|---|---|---|---|---|
| | | $W$ | $H$ | | |
| O7 | 7 | 8.54 | 13 | $R_{\text{max}} = 4$ | Meller et al. (1998) |
| O8 | 8 | 11.31 | 13 | $R_{\text{max}} = 4$ | Meller et al. (1998) |
| O9 | 9 | 12 | 13 | $R_{\text{max}} = 4$ | Meller et al. (1998) |
| F10 | 10 | 90 | 95 | $R_{\text{max}} = 3$ | Montreuil et al. (2004) |
| VC10 | 10 | 25 | 51 | $w_{\text{min}} = h_{\text{min}} = 5$ | Van Camp, Carter, and Vannelli (1992) |
| BA12 | 12 | 10 | 6 | $w_{\text{min}} = h_{\text{min}} = 1$ | Bazaraa (1975) |
| BA14 | 14 | 9 | 7 | $w_{\text{min}} = h_{\text{min}} = 1$ | Bazaraa (1975) |
| AB20 | 20 | 30 | 20 | $R_{\text{max}} = 4$ | Armour and Buffa (1963) |
| TAM20 | 20 | 40 | 35 | $R_{\text{max}} = 5$ | Tam (1992) and Gau and Meller (1999) |
| TAM30 | 30 | 45 | 40 | $R_{\text{max}} = 5$ | Tam (1992) and Gau and Meller (1999) |
| SC30 | 30 | 15 | 12 | $R_{\text{max}} = 5$ | Liu and Meller (2007) |
| SC35 | 35 | 16 | 15 | $R_{\text{max}} = 4$ | Liu and Meller (2007) |

**Table 9**
Benchmark approaches used in the unconstrained case.

| Name | Type of method | Source |
|---|---|---|
| GA-ST | GA with slicing-tree | Gau and Meller (1999) |
| ACO-ZB | Ant Colony Optimization with zone-based layout | 2004 Montreuil et al. (2004) |
| I-MIP | Improved mixed-integer programming model | Sherali et al. (2003) |
| MIP-$\varepsilon$ | MIP $\varepsilon$-accurate | Castillo and Westerlund (2005) |
| MIP-MINLP | Mixed-integer linear and mixed-integer nonlinear optimization methods | Castillo et al. (2005) |
| GA-SP-MIP | GA with sequence pair and MIP | Liu and Meller (2007) |
| STaTS | Tabu search with slicing-tree | Scholz et al. (2009) |
| ACO-ST | Ant colony optimization with slicing tree | Komarudin and Wong (2010) |
| AS-FBS | Ant system with flexible bay structure | Wong and Komarudin (2010) |
| ACO-LS-FBS | Ant colony optimization with local search and flexible bay structure | Kulturel-Konak and Konak (2011b) |
| PSO-RFBS | Particle swarm optimization with relaxed flexible bay structure | Kulturel-Konak and Konak (2011a) |
| GA-LP | Linear programming based genetic algorithm | Kulturel-Konak and Konak (2013) |
| SA-MIP | Hybrid simulated annealing and MIP algorithm | Kulturel-Konak and Konak (2014) |

**Table 10**
Experimental results: layout costs for datasets 07-09.

| Dataset | I-MIP | MIP-$\varepsilon$ | MIP-MINLP | GA-SP-MIP | STaTS | AS-FBS | *BRKGA-LP* | *BRKGA-LP* | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | $\%E_{\text{avg}}$ | $\%E_{\max}$ |
| 07 | 131.64 | 131.57 | 131.64 | 131.63 | 132 | 131.68 | 131.56 | 0.0019 | 0.0030 |
| 08 | 242.89 | 242.77 | 242.73 | 242.88 | 243.16 | 243.12 | 242.92 | 0.0004 | 0.0012 |
| 09 | 235.95 | 235.87 | 236.14 | 235.95 | 239.07 | 236.14 | 236.57 | 0.0013 | 0.0040 |

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

| Dataset | I-MIP | MIP-$\varepsilon$ | MIP-MINLP | GA-SP-MIP | STaTS | AS-FBS | *BRKGA-LP* |
|---|---|---|---|---|---|---|---|
| 07 | 7700 | 2301 | 1195 | 790 | 4.3 | 4032 | 6.85 |
| 08 | 43000 | 54443 | 18392 | 3860 | 6.2 | 4248 | 10.18 |
| 09 | 60000 | 85255 | 83211 | 5384 | 8.9 | 5184 | 12.77 |

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

| | Datasets | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Approaches | F10 | VC10 | BA12 | BA14 | AB20 | TAM20 | TAM30 | SC30 | SC35 |
| GA-ST | | | 8485.4 | 4804.1 | | 9513.5 | 20658 | | |
| ACO-ZB | 8567 | | | | | | | | |
| MIP-MINLP | | 21297.98 | 8180 | | | | | | |
| GA-SP-MIP | | 19997 | 8702 | 5004 | 8180.00 | | | 3707 | 3604 |
| STaTS | | 19994.1 | 8264 | 4712.33 | 5225.96 | | | | |
| ACO-ST | | 19967 | 8252.67 | 4724.68 | 5073.82 | | | 3868.55 | 4132.36 |
| AS-FBS | | | 8299.5 | 4913.22 | | | | 3679.85 | 3962.72 |
| ACO-LS-FSB | 9020.75 | | 8801.33 | 4904.67 | 5360.8 | 9003.82 | 19667.45 | 3794.82 | 4011.31 |
| PSO-RFBS | 9020.75 | 22899.65 | 8129 | 4780.91 | 5336.36 | 8753.57 | 19462.41 | 3443.34 | 3700.75 |
| GA-LP | 7651.28 | | **8021** | 4686.81 | 5196.38 | 8058.06 | 19009.9 | 3370.98 | 3385.48 |
| SA-MIP | | | | | 5016.93 | | | 3318.76 | 3469.40 |
| *BRKGA-LP* | **7650.95** | 19951.17 | **8020.97** | **4628.79** | 5021.17 | **7986.48** | **18740.3** | 3367.87 | **3316.77** |
| %Imp | 0.00 | 0.21 | 0.00 | 1.24 | – 0.08 | 0.89 | 1.42 | – 1.48 | 2.03 |
| $\%E_{\text{avg}}$ | 0.0016 | 0.0054 | 0 | 0.0016 | 0.0019 | 0.0011 | 0.0016 | 0.0012 | 0.0010 |
| $\%E_{\max}$ | 0.0054 | 0.0016 | 0 | 0.0095 | 0.0046 | 0.0065 | 0.0061 | 0.0052 | 0.0048 |

Best values are in bold.

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

| | Datasets | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Approaches | F10 | VC10 | BA12 | BA14 | AB20 | TAM20 | TAM30 | SC30 | SC35 |
| GA-ST | | | 412 | 396 | | 499 | 1397 | | |
| ACO-ZB | 26300 | | | | | | | | |
| STaTS | | 8.9 | 13.6 | 16.1 | 13.6 | | | | |
| ACO-ST | | 1008 | 4104 | 8568 | 17820 | | | 23544 | 83988 |
| AS-FBS | | | 164 | 292 | 919 | | | 19135 | 28671 |
| ACO-LS-FSB | 51 | | 146 | 131 | 106 | 226 | 623 | 902 | 1185 |
| PSO-RFBS | 2 | 3 | 10 | 19 | 85 | 104 | 924 | 873 | 1842 |
| MIP-MINLP | | 43200 | 43200 | | | | | | |
| GA-LP | 3000 | | 6000 | 7500 | 22500 | 22500 | 73500 | 22038.3 | 29538.9 |
| GA-SP-MIP | | 6660 | 2988 | 2880 | 4919.47 | | | 14652 | 57744 |
| SA-MIP | | | | | 4001 | | | 7799 | 9524 |
| *BRKGA-LP* | 49.15 | 46.40 | 113.81 | 160.46 | 426.09 | 686.00 | 3004.55 | 998.74 | 1556.03 |

or very close to 0%. Tables 10 and 12 present the experimental results, respectively, for the datasets O7 to O9 and F10 to SC35. For all the best found solutions we report $\%E_{\text{avg}}$ and $\%E_{\max}$. As can be observed, the largest value of $\%E_{\max}$ was 0.0095 %, which for a facility with an area of 100 units corresponds to a negligible reduction of only 0.0095 units.

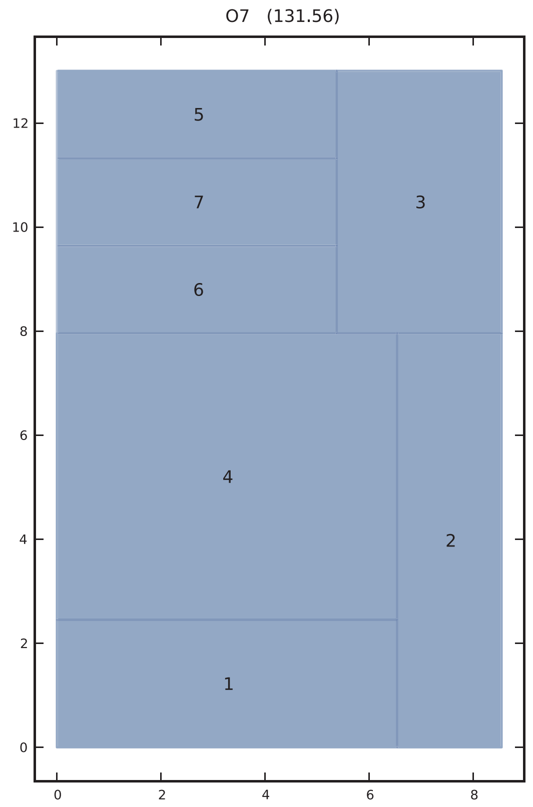
Table 10 reports the best cost solutions found by BRKGA-LP, over 10 runs, and six other approaches from the literature for datasets O7 to O9. Even though the computational times reported by the other approaches might not be comparable, we report them, if available, in Table 11.

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 $\%E_{\max}$ equal to 0.05, 0.15 and 0.30, respectively, for O7, O8 and O9, while BRKGA-LP has $\%E_{\max}$ values equal to 0.0030, 0.0012, and 0040 which are at least one order of magnitude smaller).

As can be observed in Table 11 the computational times of BRKGA-LP and *STaST* 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 O7-O9 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



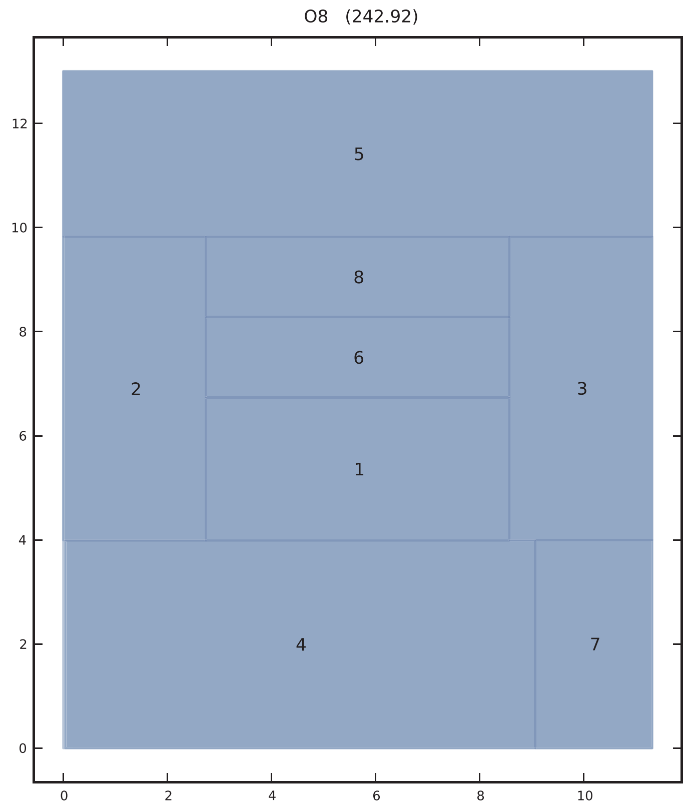
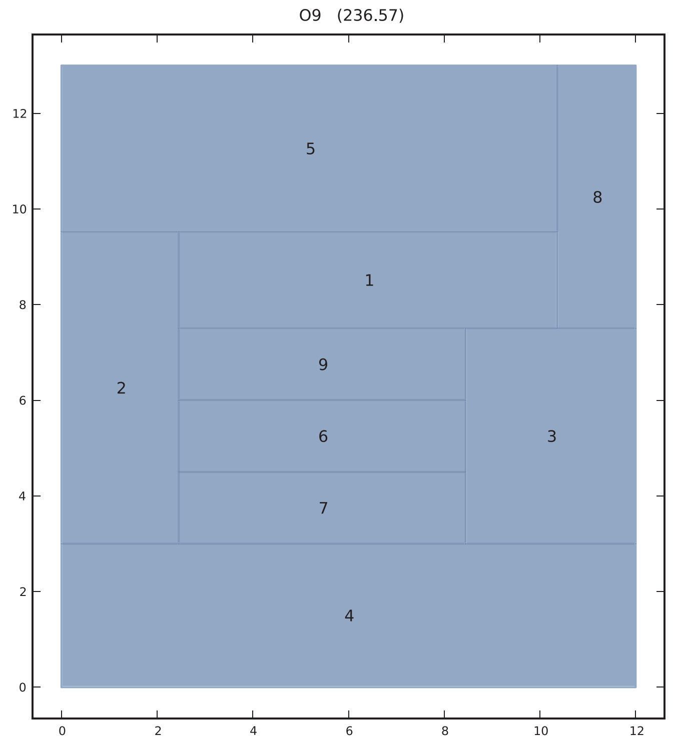


Fig. 16. BRKGA-LP solutions for datasets O7 and O8.



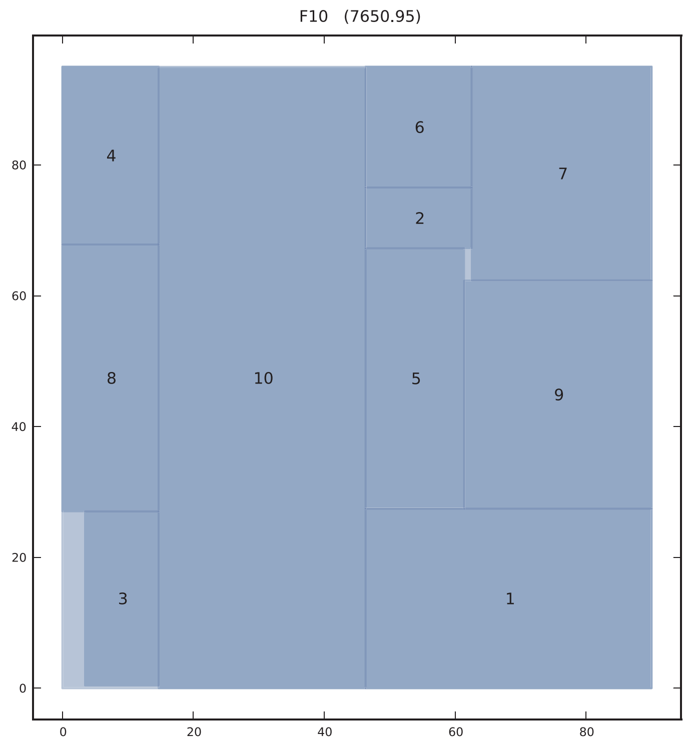
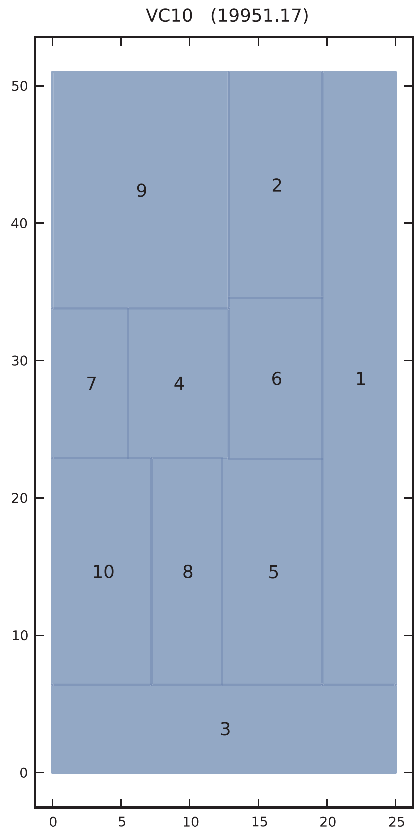


Fig. 17. BRKGA-LP solutions for datasets O9 and F10.

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.



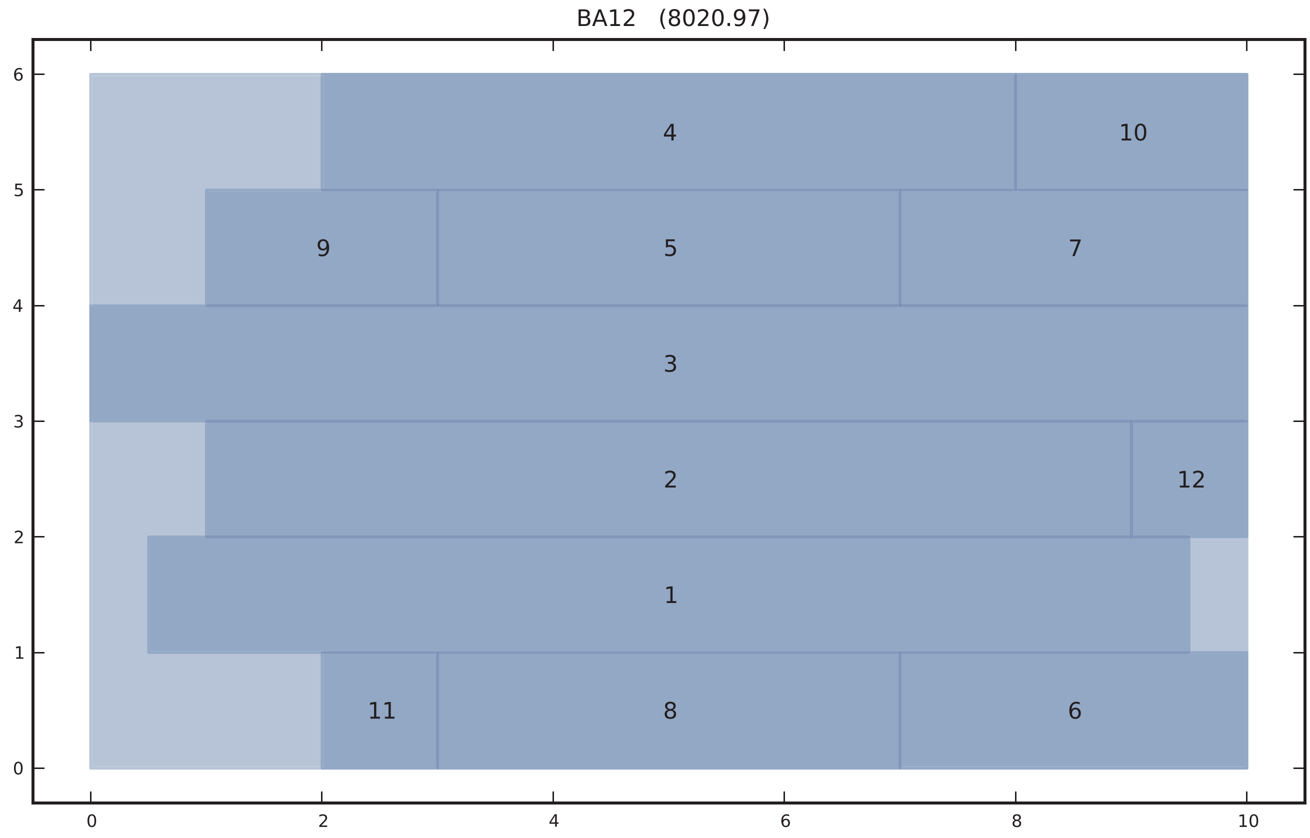
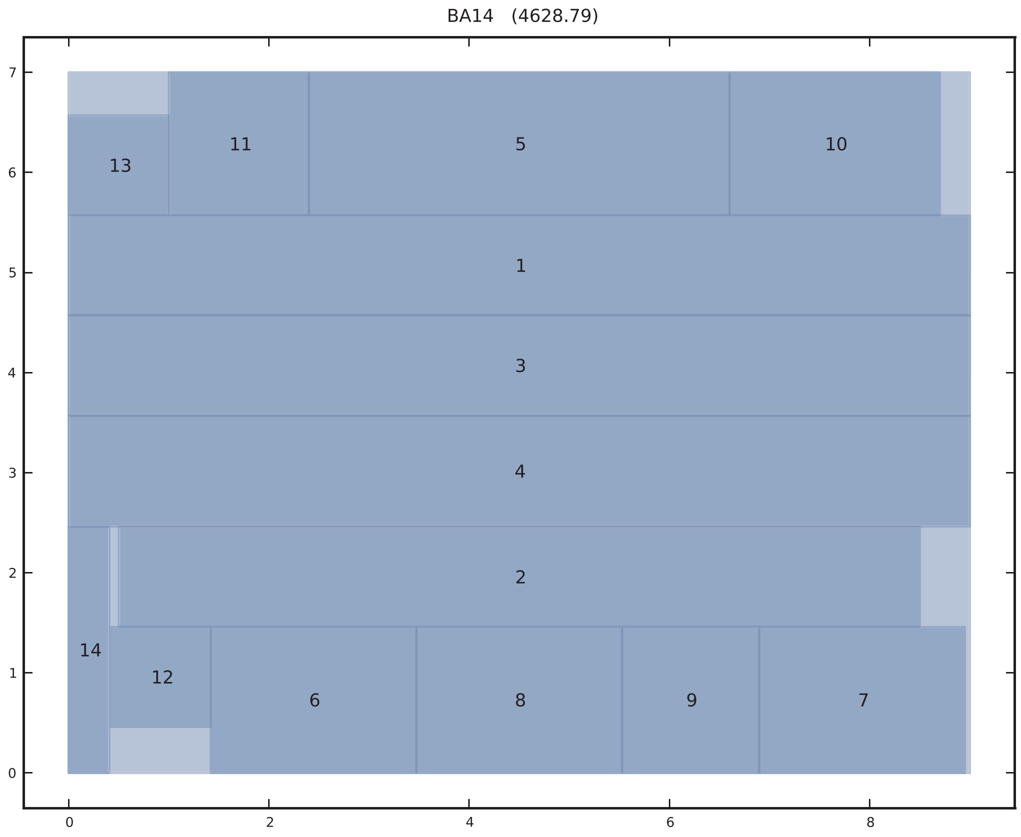


**Fig. 18.** BRKGA-LP solutions for datasets VC10 and BA12.



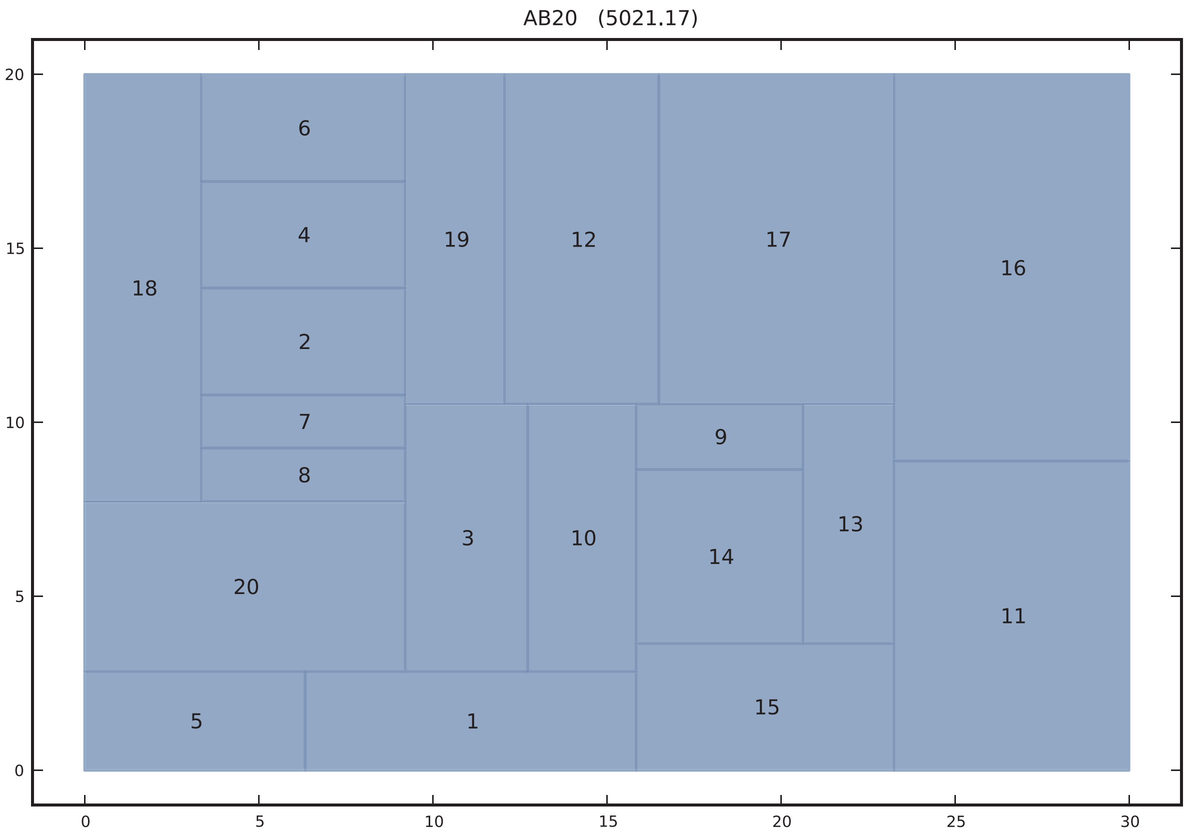
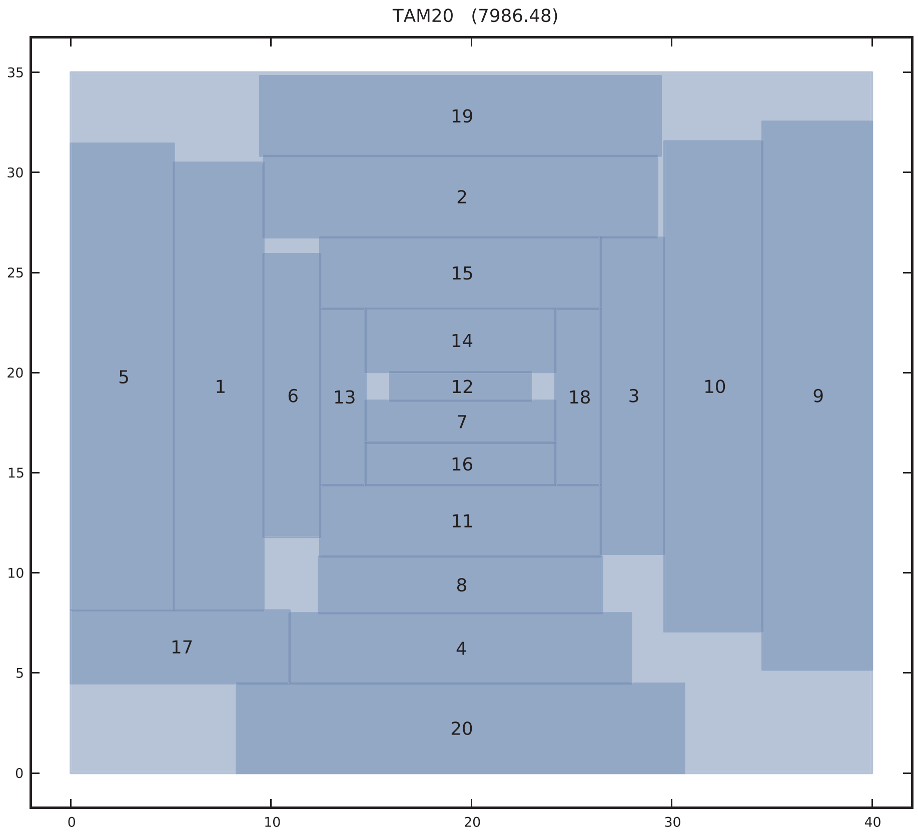


Fig. 19. BRKGA-LP solutions for datasets BA14 and AB20.



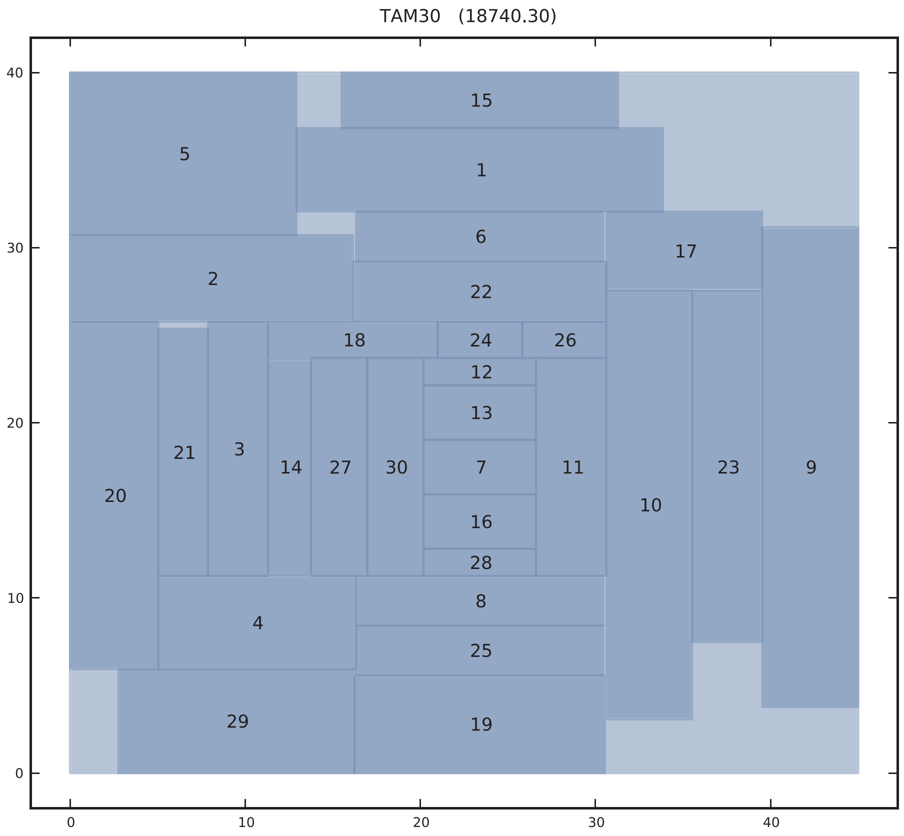
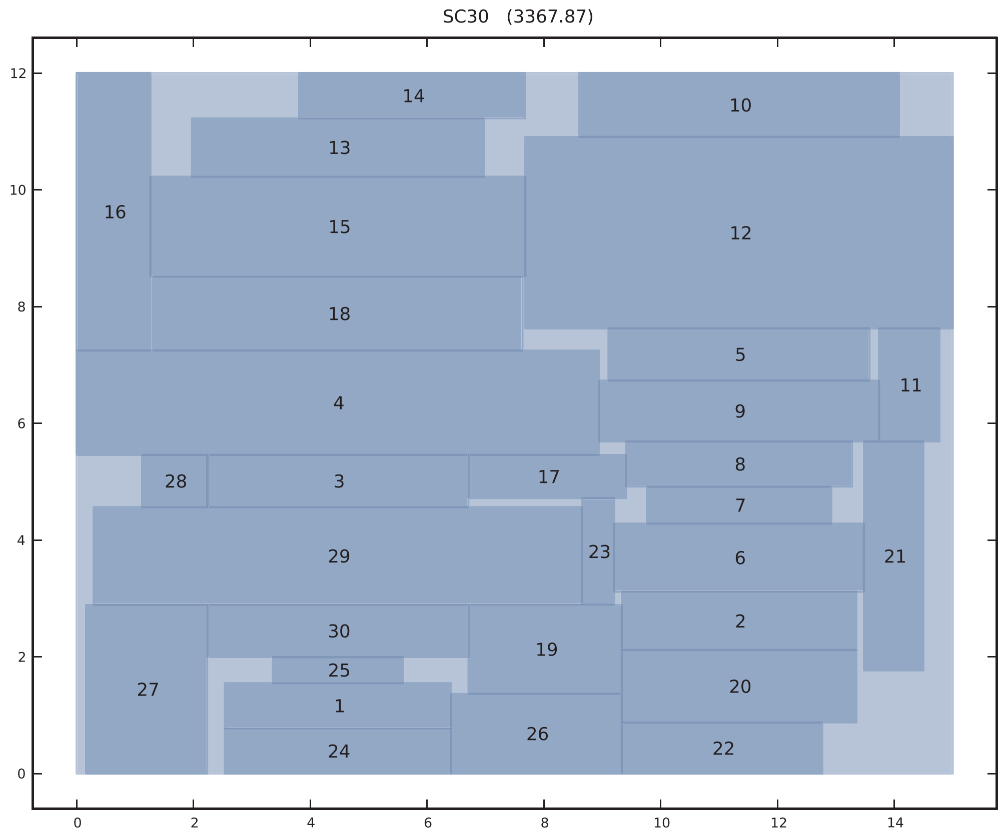


**Fig. 20.** BRKGA-LP solutions for datasets TAM20 and TAM30.



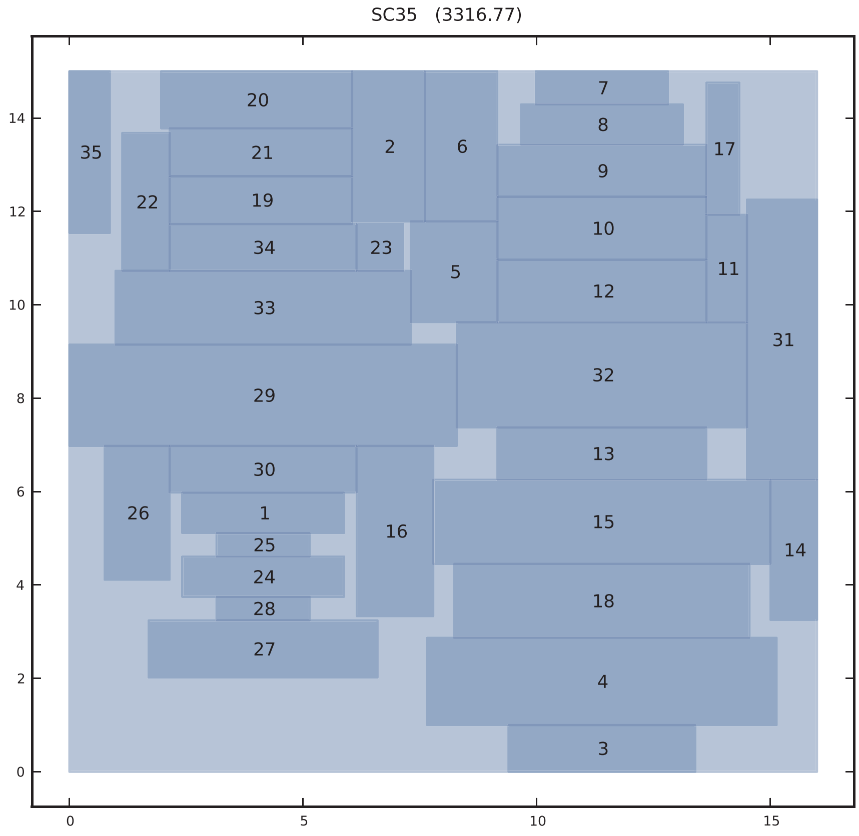


**Fig. 21.** BRKGA-LP solutions for datasets SC30 and SC35.

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-RFBS(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 requirements 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 improves 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 PTDC/EGE-GES/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 Mauricio G.C. Resende was done while he was employ ed by AT&T Labs Research in Middletown, New Jersey, U.S.A.

## References

Armour, G. C., & Buffa, E. S. (1963). A heuristic algorithm and simulation approach to relative location of facilities. *Management Science, 9*(2), 294–309.

Banerjee, P., Montreuil, B., Moodie, C., & Kashyap, R. (1992). A modelling of interactive facilities layout designer reasoning using qualitative patterns. *International Journal of Production Research, 30*(3), 433–453.

Banerjee, P., Zhou, Y., & Montreuil, B. (1997). Genetically assisted optimization of cell layout and material flow path skeleton. *IIE Transactions, 29*(4), 277–291.

Bazaraa, M. S. (1975). Computerized layout design: A branch and bound approach. *AIIE Transactions, 7*(4), 432–438.

Bean, J. C. (1994). Genetic algorithms and random keys for sequencing and optimization. *ORSA Journal on Computing, 6*, 154–160.

Birgin, E. G., Martínez, J. M., & Raydan, M. (2000). Nonmonotone spectral projected gradient methods on convex sets. *SIAM Journal on Optimization, 10*(4), 1196–1211.

Bozer, Y. A., & Wang, C.-T. (2012). A graph-pair representation and MIP-model-based heuristic for the unequal-area facility layout problem. *European Journal of Operational Research, 218*(2), 382–391.

Castillo, I., Westerlund, J., Emet, S., & Westerlund, T. (2005). Optimization of block layout design problems with unequal areas: A comparison of MILP and MINLP optimization methods. *Computers and Chemical Engineering, 30*(1), 54–69.

Castillo, I., & Westerlund, T. (2005). An $\varepsilon$-accurate model for optimal unequal-area block layout design. *Computers & Operations Research, 32*(3), 429–447.

Dunker, T., Radons, G., & Westkämper, E. (2003). A coevolutionary algorithm for a facility layout problem. *International Journal of Production Research, 41*(15), 3479–3500.

Ericsson, M., Resende, M., & Pardalos, P. (2002). A genetic algorithm for the weight setting problem in OSPF routing. *Journal of Combinatorial Optimization, 6*(3), 299–333. doi:10.1023/A:1014852026591.

Fontes, D. B. M. M., & Gonçalves, J. F. (2013). A multi-population hybrid biased random key genetic algorithm for hop-constrained trees in nonlinear cost flow networks. *Optimization Letters, 7*(6), 1303–1324. doi:10.1007/s11590-012-0505-5.

Garces-Perez, J., Schoenefeld, D. A., & Wainwright, R. L. (1996). Solving facility layout problems using genetic programming. In *Proceedings of the first annual conference on genetic programming* (pp. 182–190). MIT Press.

Gau, K.-Y., & Meller, R. D. (1999). An iterative facility layout algorithm. *International Journal of Production Research, 37*(16), 3739–3758.

Goldberg, D. (1989). *Genetic algorithms in search, optimization, and machine learning*. Addison-Wesley.

Gonçalves, J. F., & Almeida, J. (2002). A hybrid genetic algorithm for assembly line balancing. *Journal of Heuristics, 8*, 629–642.

Gonçalves, J. F., Mendes, J., & Resende, M. G. C. (2005). A hybrid genetic algorithm for the job shop scheduling problem. *European Journal of Operational Research, 167*, 77–95.

Gonçalves, J. F., Mendes, J., & Resende, M. G. C. (2009). A genetic algorithm for the resource constrained multi-project scheduling problem. *European Journal of Operational Research, 189*, 1171–1190.

Gonçalves, J. F., & Resende, M. G. C. (2004). An evolutionary algorithm for manufacturing cell formation. *Computers and Industrial Engineering, 47*, 247–273.

Gonçalves, J. F., & Resende, M. G. C. (2011). Biased random-key genetic algorithms for combinatorial optimization. *Journal of Heuristics, 17*, 487–525.

Gonçalves, J. F., & Resende, M. G. C. (2012). A parallel multi-population biased random-key genetic algorithm for a container loading problem. *Computers and Operations Research, 39*, 179–190.

Gonçalves, J. F., & Resende, M. G. C. (2013). A biased random key genetic algorithm for 2D and 3D bin packing problems. *International Journal of Production Economics, 145*, 500–510.

Gonçalves, J. F., & Resende, M. G. C. (2014). An extended Akers graphical with a biased random-key genetic algorithm for job-shop scheduling. *International Transactions in Operational Research, 21*, 215–246.

Gonçalves, J. F., Resende, M. G. C., & Costa, M. D. (2014a). A biased random-key genetic algorithm for the minimization of open stacks problem. *International Transactions in Operational Research*. doi:10.1111/itor.12109. (to appear).

Gonçalves, J. F., Resende, M. G. C., & F. , T. R. (2014b). An experimental comparison of biased and unbiased random-key genetic algorithms. *Pesquisa Operacional, 34*, 143–164.

Gonçalves, J. F., Resende, M. G. C., & Mendes, J. J. M. (2011). A biased random-key genetic algorithm with forward-backward improvement for the resource constrained project scheduling problem. *Journal of Heuristics, 17*, 467–486.

Gonçalves, J. F., & Sousa, P. S. A. (2011). A genetic algorithm for lot sizing and scheduling under capacity constraints and allowing backorders. *International Journal of Production Research, 49*, 2683–2703.

Heragu, S. (1997). *Facilities design*. PWS Publishing, Boston, MA.

Imam, M. H., & Mir, M. (1993). Automated layout of facilities of unequal areas. *Computers and Industrial Engineering, 24*(3), 355–366.

Imam, M. H., & Mir, M. (1998). Cluster boundary search algorithm for building-block layout optimization. *Advances in Engineering Software, 29*(2), 165–173.

Kado, K. (1996). University of Edinburgh (Ph.D. thesis).

Komarudin, & Wong, K. Y. (2010). Applying ant system for solving unequal area facility layout problems. *European Journal of Operational Research, 202*(3), 730–746. http://dx.doi.org/10.1016/j.ejor.2009.06.016.

Konak, A., Kulturel-Konak, S., Norman, B. A., & Smith, A. E. (2006). A new mixed integer programming formulation for facility layout design using flexible bays. *Operations Research Letters, 34*(6), 660–672.

Kulturel-Konak, S., & Konak, A. (2011a). A new relaxed flexible bay structure representation and particle swarm optimization for the unequal area facility layout problem. *Engineering Optimization, 43*(12), 1263–1287.

Kulturel-Konak, S., & Konak, A. (2011b). Unequal area flexible bay facility layout using ant colony optimisation. *International Journal of Production Research, 49*(7), 1877–1902.

Kulturel-Konak, S., & Konak, A. (2013). Linear programming based genetic algorithm for the unequal area facility layout problem. *International Journal of Production Research, 51*(14), 4302–4324.

Kulturel-Konak, S., & Konak, A. (2014). A large-scale hybrid simulated annealing algorithm for cyclic facility layout problems. *Engineering Optimization*, 1–16. in press.

Kusiak, A., & Heragu, S. S. (1987). The facility layout problem. *European Journal of Operational Research, 29*(3), 229–251.

Lacksonen, T. A. (1997). Preprocessing for static and dynamic facility layout problems. *International Journal of Production Research, 35*(4), 1095–1106.

Lai, K. K., & Chan, J. W. M. (1997). Developing a simulated annealing algorithm for the cutting stock problem. *Computers and Industrial Engineering, 32*, 115–127.

Langevin, A., Montreuil, B., & Riopel, D. (1994). Spine layout design. *International Journal of Production Research, 32*(2), 429–442.

Liu, Q., & Meller, R. D. (2007). A sequence-pair representation and mip-model-based heuristic for the facility layout problem with rectangular departments. *IIE Transactions, 39*(4), 377–394.

McKendall Jr, A. R., & Hakobyan, A. (2010). Heuristics for the dynamic facility layout problem with unequal-area departments. *European Journal of Operational Research, 201*(1), 171–182.

Meller, R. D., Chen, W., & Sherali, H. D. (2007). Applying the sequence-pair representation to optimal facility layout designs. *Operations Research Letters, 35*(5), 651–659.

Meller, R. D., & Gau, K.-Y. (1996). The facility layout problem: Recent and emerging trends and perspectives. *Journal of Manufacturing Systems, 15*(5), 351–366.

Meller, R. D., Narayanan, V., & Vance, P. H. (1998). Optimal facility layout design. *Operations Research Letters, 23*(3), 117–127.

Mendes, J. J. M., Gonçalves, J. F., & Resende, M. G. C. (2009). A random key based genetic algorithm for the resource constrained project scheduling problem. *Computers and Operations Research, 36*, 92–109.

Mir, M., & Imam, M. H. (1996). Analytic annealing for macrocell placement optimization. *Computers & Electrical Engineering, 22*(2), 169–177.

Mir, M., & Imam, M. H. (2001). A hybrid optimization approach for layout design of unequal-area facilities. *Computers and Industrial Engineering, 39*(1-2), 49–63.

Montreuil, B. (1991). A modeling framework for integrating layout design and flow network design. In J. White, & I. Pence (Eds.), *Progress in material handling and logistics: 2* (pp. 95–116). Springer-Verlag.

Montreuil, B., Brotherton, E., & Marcotte, S. (2002). Zone-based facilities layout optimization. In *Proceedings of the industrial engineering research conference, IIE annual conference* (pp. 1–6). Citeseer.

Montreuil, B., Brotherton, E., Ouazzani, N., & Nourelfath, M. (2004). Antzone layout metaheuristic: coupling zone-based layout optimization, ant colony system and domain knowledge. In *Progress in material handling research* (pp. 301–331). Charlotte: Material Handling Industry of America (MHIA).

Montreuil, B., Venkatadri, U., & Ratliff, D. H. (1993). Generating a layout from a design skeleton. *IIE Transactions, 25*(1), 3–15.

Murata, H., Fujiyoshi, K., Nakatake, S., & Kajitani, Y. (1996). VLSI module placement based on rectangle-packing by the sequence-pair. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 15*(12), 1518–1524.

Sahni, S., & Gonzalez, T. (1976). P-complete approximation problems. *Journal of the ACM (JACM), 23*(3), 555–565.

Schnecke, V., & Vornberger, O. (1997). Hybrid genetic algorithms for constrained placement problems. *IEEE Transactions on Evolutionary Computation, 1*(4), 266–277.

Scholz, D., Petrick, A., & Domschke, W. (2009). STaTS: A slicing tree and tabu search based heuristic for the unequal area facility layout problem. *European Journal of Operational Research, 197*(1), 166–178.

Sherali, H. D., Fraticelli, B. M., & Meller, R. D. (2003). Enhanced model formulations for optimal facility layout. *Operations Research, 51*(4), 629–644.

Spears, W. M., & Dejong, K. A. (1991). On the virtues of parameterized uniform crossover. In *Proceedings of the fourth international conference on genetic algorithms* (pp. 230–236).

Tam, K. Y. (1992). A simulated annealing algorithm for allocating space to manufacturing cells. *The International Journal of Production Research, 30*(1), 63–87.

Tam, K. Y., & Li, S. G. (1991). A hierarchical approach to the facility layout problem. *International Journal of Production Research, 29*(1), 165–184.

Van Camp, D. J., Carter, M. W., & Vannelli, A. (1992). A nonlinear optimization approach for solving facility layout problems. *European Journal of Operational Research, 57*(2), 174–189.

VIP-PLANOPT (2010). *Engineering optimization software,* <www.planopt.com> Accessed 08.13.

Wong, K. Y., & Komarudin (2010). Solving facility layout problems using flexible bay structure representation and ant system algorithm. *Expert Systems with Applications, 37*(7), 5523–5527.