European Journal of Operational Research 266 (2018) 1100–1112

Contents lists available at ScienceDirect

European Journal of Operational Research

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

Innovative Applications of O.R.

Block layout for attraction-based enterprises

Jinhua Li[a], Alice E. Smith[b,*]

[a] School of Economics & Management, South China Normal University, Guangzhou, Guangdong, China
[b] Department of Industrial and Systems Engineering, Auburn University, Auburn, AL, USA

ARTICLE INFO

Article history:
Received 17 September 2016
Accepted 13 October 2017
Available online 23 October 2017

Keywords:
Facilities planning and design
Metaheuristics
OR in entertainment
Moran's *I*
Tabu search

ABSTRACT

This paper proposes an approach for block layout for attraction based enterprises such as theme parks, museums, casinos and exhibitions. The approach takes into account the attraction of individual entities within the enterprise and any adjacency requirements among entities while enforcing reasonable shapes. Using Moran's *I* of spatial statistics and a newly devised shape control method, an optimization mathematical model is established. To solve the model, a space filling curve mechanism works with a tabu search heuristic. The effectiveness and practicality of the approach are demonstrated using three layout cases including the SeaWorld San Diego Park. The approach is easy to use, effective and can readily accommodate varying degrees of preference regarding adjacencies and fixed entities.

1. Introduction

In many museums, exhibitions and theme parks, visitor crowding occurs when there are a large number of visitors or a highly uneven distribution of visitors among the enterprise's entities. Visitor crowding negatively affects visitor satisfaction (Rathnayake, 2015). Because individual entities of a site can differentially attract volumes of visitors, strategic layout of entities is critical. Researchers have studied how to guide visitors to reduce crowding and improve satisfaction (Jaén, Mocholí, Catalá, & Navarro, 2011; Kawamura, Kataoka, Kurumatani, & Ohuchi, 2004; Yu, Lin, & Chou, 2010), but no previous research has explicitly considered the actual layout of the attractors.

The design of such an enterprise can be classified as a block layout problem. Layout problems are generally NP-hard (Garey & Johnson, 1979) and have been extensively studied (e.g., Ahmadi & Jokar, 2016; Drira, Pierreval, & Hajri-Gabouj, 2007; Saraswat, Venkatadr, & Castillo, 2015). Problems can be formulated as discrete or continuous. The slicing tree (Komarudin & Wong, 2010; Ripon, Glette, Khan, Hovin, & Torresen, 2013; Shayan & Chittilappilly, 2004) and the flexible bay structure (Kulturel-Konak, 2012; Kulturel-Konak & Konak, 2011; Tate & Smith, 1995) are popularly used to model layouts in continuous space. For a discrete representation, the planar site is divided into square blocks of the same size and each block is assigned to a department or entity. Space filling curves (abbr. SFC) are generally used to model these layouts (Bozer, Meller, & Erlebacher, 1994; Islier, 1998; Wang, Hu, & Ku, 2005). SFC can ensure the blocks assigned to a department are contiguous, but cannot ensure reasonable shapes on their own. While a slicing tree or flexible bay construct can model attraction based enterprises, they are not as flexible as the SFC in the shapes allowed. The SFC allows rectangles but also L's and other complex shapes. This greater freedom in attraction shape may be an advantage for some applications.

Various methods and procedures have been proposed to solve layout problems. Because exact approaches are only applicable to small size problems, approximated approaches such as heuristics and meta-heuristics have been extensively used (Drira et al., 2007; Gonçalves & Resende, 2015). The meta-heuristics mainly includes genetic algorithms, ant colony algorithms, simulated annealing algorithms and tabu search algorithms (Bashiri & Karimi, 2012; Drira et al., 2007). These approaches can be applicable to both discrete and continuous layout problems.

Traditionally, the main objective of layout problems is to minimize a material flow cost function, perhaps while considering adjacencies (Aiello, Scalia, & Enea, 2012; Lee, Roh, & Jeong, 2005). In a few cases, maximizing an area utilization factor and minimizing shape irregularities have been considered (e.g., Wang et al., 2005). However, the layout of attraction based entities has largely been ignored in the literature. Although some studies have simulated crowd evacuation in emergency situations of large-scale events, these have not considered block layout design.

The contributions of this paper include: (1) the first analytic formulation of an attraction-based block layout design problem; (2) a new department shape control method using a discrete layout representation; (3) the adaptation of Moran's *I* to measure the

* Corresponding author.
E-mail address: smithae@auburn.edu (A.E. Smith).

https://doi.org/10.1016/j.ejor.2017.10.028
0377-2217/

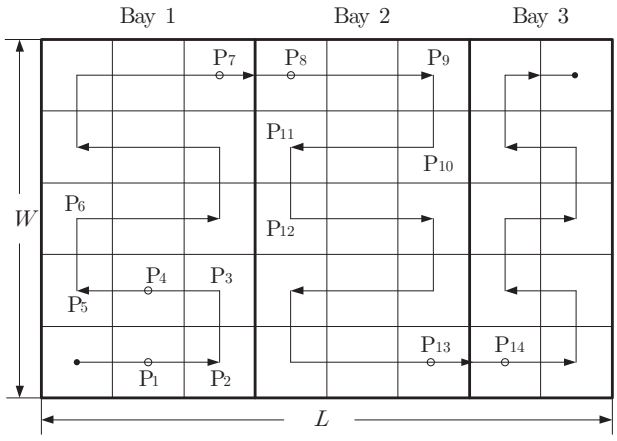


Fig. 1. Site size 8 by 5 with 3 bays.

attraction distribution of the layout to reduce visitor crowding; and (4) improvements on the classical SFC with a set of rules and procedures to ensure reasonable department shapes including rectangles and L's.

2. Problem formulation

This section provides the formulation of the model for block layout design of attraction-based enterprises. We use terminology in this section, without loss of generalization, appropriate to exhibits within an exhibition.

2.1. Problem description

N entities denote N exhibits in an exhibition. These entities are to be placed in a rectangular planar site which is divided into a grid of square blocks of L by W. Each block has the same unit size, such as one meter square. Every entity has a required area (e.g., A_k denotes area required by entity k), and needs to occupy contiguous blocks whose total area is equal to the required entity area.

Each entity has a certain attraction to visitors, which is determined by its popularity and the value of its exhibits. This is termed the attraction value and is normalized to [0, 1]. Attraction values of entities vary considerably. In addition, each entity has a predefined correlation or affinity with each other entity, termed adjacency.

In brief, this problem can be represented as an unequal-area facility block layout problem. The objective function takes into account an attraction distribution requirement and an adjacency requirement, i.e., maximize the uniformity of attraction and maximize specified adjacency. Other objectives might be included. The constraints include: (1) a block cannot be shared by more than one entity; (2) the area required by any entity must be satisfied; (3) the total area required by all entities must not exceed the available area; (4) the blocks that form the total area of a given entity are contiguous and of reasonable shape.

2.2. Space filling curve and shape control

Since the planar site is divided into a grid of square blocks, a discrete layout representation using a SFC is adopted. The SFC connects every block in the planar site, and ensures that the blocks allocated to an entity are contiguous. All entities are placed in the planar site along the SFC in a sequence.

The planar site is vertically divided into N_b bays as shown in Fig. 1. B_i is the width of the ith bay and $L = \sum_{i=1}^{N_b} B_i$. The SFC is generated based on certain rules. The detailed procedure of generating the SFC is described in Section 3.1.

When an entity becomes irregular in shape, it may be impractical. Bozer et al. (1994) proposed a method to control department shape as follows:

$$\varphi_i = \frac{P_i/A_i}{P_i^*/A_i} = \frac{1}{4} P_i A_i^{-0.5} \tag{1}$$

where P_i is the perimeter of department i, A_i is the area of department i, and P_i^* is the perimeter of department i when it is square. With this measure, φ increases as the shape becomes more irregular. Although this method is better than the other shape control methods when using a SFC, it still has limitations, e.g., it cannot enforce specific shapes. Furthermore, this method considers the square as the ideal shape when a rectangle or L shape may work perfectly well for attraction-based entities.

To address these shortcomings for our application, we devised a new shape control method termed the corner-control method. Generally, if an entity consists of equally sized blocks in a planar site, the entity shape is more regular when it has fewer corners. For example, a square or rectangular entity has four corners, and an L-shaped entity has six corners. There are four possible shapes when the number of corners of an entity is eight, as shown in Fig. 2. Shapes become more irregular when the number of corners increases. Therefore, entity shape can be controlled by constraining the number of corners. If set to four, the shape will be square or rectangular. If set to six, the shape will be square, rectangular or L-shaped. These three shapes are very reasonable and practical.

Calculating the number of corners of an entity in a layout is straightforward and computationally simple. It will be formulated and explained in Section 2.5.3. However, if this is a strict constraint, there will be fewer feasible layouts or possibly no feasible layouts when L and W are fixed to equal to the sum of the entity areas. Therefore, we can optionally relax the bounding facility area to allow for feasible layouts as explained later in the paper.

To further refine the shape of each entity, we consider the ratio of perimeter to area as defined in Eq. (1). Even if an entity is rectangular, it may be impractical by being too long and narrow. To achieve reasonable entity shapes, we add an objective function factor termed the shape ratio for each entity. This is defined later in Eq. (6).

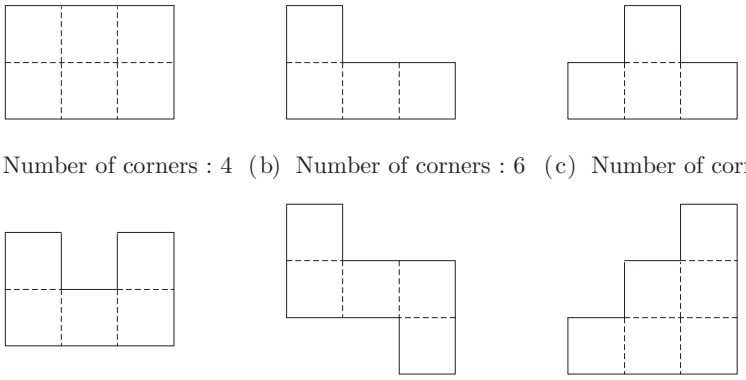
2.3. Attraction distribution consideration

A uniform distribution of attraction is an important objective factor while designing an enterprise's layout. This depends on the spatial distribution of the entities. We use Moran's I from spatial statistical analysis to characterize the attraction distribution. Moran's I is a measure of spatial autocorrelation that measures the correlation of an attribute among nearby locations in space (Cliff & Ord, 1981; Moran, 1950). Moran's I is defined as

$$I = \frac{N}{\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}} \cdot \frac{\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}(v_i - \bar{v})\left(v_j - \bar{v}\right)}{\sum_{i=1}^{N}(v_i - \bar{v})^2} \tag{2}$$

where N is the number of spatial objects indexed by i and j, v_i is the attribute (e.g., attraction value) for spatial object i, $\bar{v} = \frac{\sum_{i=1}^{N} v_i}{N}$ is the mean of v, and w_{ij} is an element of a matrix of spatial weights ($i \neq j$). Usually there are two methods to set the w_{ij} value. One is an adjacency criterion: when spatial object i and spatial object j are adjacent, w_{ij} is 1, otherwise 0. The other is a distance criterion: when spatial object i and spatial object j are within a given distance, w_{ij} is 1, otherwise 0. However, w_{ij} might also be a real number reflecting the distance between spatial objects i and j.

Values of Moran's I range from -1 to $+1$. A negative value denotes negative spatial autocorrelation and a positive value denotes positive spatial autocorrelation. A zero value denotes a random spatial pattern. The greater I is, the greater the correlation of spatial distribution is, that is, the spatial distribution is



(a) Number of corners : 4 (b) Number of corners : 6 (c) Number of corners : 8

(d) Number of corners : 8 (e) Number of corners : 8 (f) Number of corners : 8

Fig. 2. Possible entity shapes with different values of the number of corners.

Table 1
The rules of setting b_{ij} and c_{ij} (from Lee et al., 2005).

b_{ij}		c_{ij}	
1.0	if $0 < d_{ij} \le d_{max}/6$	0	It is undesirable for entities i and j to be located close together
0.8	if $d_{max}/6 < d_{ij} \le d_{max}/3$	1	It is unimportant for entities i and j to be located close together
0.6	if $d_{max}/3 < d_{ij} \le d_{max}/2$	2	It is ordinary for entities i and j to be located close together
0.4	if $d_{max}/2 < d_{ij} \le 2d_{max}/3$	3	It is important for entities i and j to be located close together
0.2	if $2d_{max}/3 < d_{ij} \le 5d_{max}/6$	4	It is especially important for entities i and j to be located close together
0.0	if $5d_{max}/6 < d_{ij} \le d_{max}$	5	It is absolutely necessary for entities i and j to be located close together

more clustered. The less I is, the less the correlation of spatial distribution is, that is, the spatial distribution is more dispersed. To reduce crowding, a low I is desirable. Note that this method also disperses low attraction entities as well as high attraction ones. If this aspect is not desirable for a certain exhibition, then adjacencies can be used to encourage certain designs such as grouping low attraction entities.

Once all entities are placed in a layout, the attraction distribution is formed. Then Moran's I is used to measure it. A minimum Moran's I results in the greatest dispersion of attractions. We use a distance criterion to set w_{ij}. d_{ij} is the rectilinear distance between the centroids of the two entities for a given layout as defined in Section 2.5.2. d_{max} is the largest d_{ij}. (Though we have used the rectilinear distance between centroids, any other distance metric could be used.) The distance criterion is set as follows: if $d_{ij} \le d_{max}/4$, $w_{ij} = 1$, otherwise $w_{ij} = 0$. We also tested $w_{ij} = 1/d_{ij}$ and $w_{ij} = 1/{d_{ij}}^2$, which are continuous scales, but these are more complicated and did not yield better results.

2.4. Adjacency requirement

How well a layout meets the desired adjacencies of entities is another objective factor. For example, in a museum it may be desirable to group entities from similar time frames or similar geographic origins. In a casino, it might be undesirable to put high stakes tables near the slot machines. We adopt two indicators defined by Lee et al. (2005), namely, the adjacency factor and the adjacency value. The adjacency factor b_{ij}, which denotes the adjacency ratio between entities i and j, is determined by distance d_{ij} between entities i and j as shown in Table 1. d_{ij} is the rectilinear distance between the centroids of entities i and j for a given layout as defined in Section 2.5.2. d_{max} is the largest d_{ij}. (As with the weights for Moran's I, other distance metric could be used if desired.) The adjacency value c_{ij} between entities i and j is a functional relationship as shown in Table 1, however any similar relationship could be used. The magnitude of adjacency between entities i and j is equal to $b_{ij} \cdot c_{ij}$, which is a "bigger is better" objective.

2.5. Mathematical model

2.5.1. Parameters and variables

The parameters and variables for the model are listed below:

Parameters:

N	number of entities
L	length of planar site
W	width of planar site
A_k	area required by entity k
v_i	attraction value of entity i, $v_i \in [0, 1]$
c_{ij}	adjacency value between entities i and j
N_k^c	upper limit number of corners for entity k

Decision variables:

a_{ijk}	decision variable which indicates whether entity k is placed in the location at the ith row and the jth column, 1: Yes; 0: No

Dependent variables:

w_{ij}	element of a matrix of spatial weights
b_{ij}	adjacency factor between entities i and j
$(x_k,\ y_k)$	coordinates of the centroid of entity k, namely (column, row)
d_{ij}	rectilinear distance between the centroids of entities i and j
n_k^c	number of corners of entity k

2.5.2. Distance equation

$(x_k,\ y_k)$ is formulated as Eq. (3) and d_{ij} is formulated as Eq. (4) as below:

$$x_k = \frac{\sum_{i=1}^{W}\sum_{j=1}^{L} a_{ijk} \cdot j}{\sum_{i=1}^{W}\sum_{j=1}^{L} a_{ijk}},\ y_k = \frac{\sum_{i=1}^{W}\sum_{j=1}^{L} a_{ijk} \cdot i}{\sum_{i=1}^{W}\sum_{j=1}^{L} a_{ijk}},\ k \in [1, N] \tag{3}$$

$$d_{ij} = \left|x_i - x_j\right| + \left|y_i - y_j\right|,\ i \text{ and } j \in [1, N] \tag{4}$$

2.5.3. Corner-control method

The corner-control method works as follows: Each block (i, j) has a value of a_{ijk} which indicates whether it is occupied by entity k or not. So it is straightforward to conclude from these

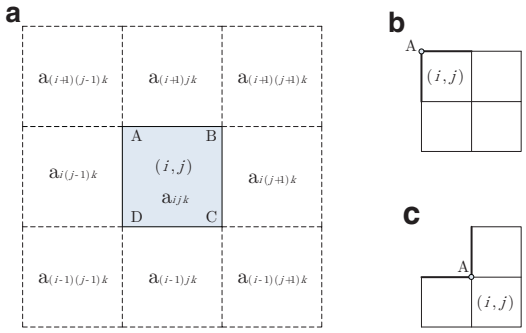


Fig. 3. Diagram of the corner-control method.

values whether a vertex of a block is a corner of an entity or not. In Fig. 3(a), vertex A of block (i, j) is a corner of entity k only in two cases: (1) if $a_{ijk} = 1$ and $a_{i(j-1)k} = 0$ and $a_{(i+1)jk} = 0$ and $a_{(i+1)(j-1)k} = 0$, then it is a corner as shown in Fig. 3(b); (2) if $a_{ijk} = 1$ and $a_{i(j-1)k} = 1$ and $a_{(i+1)jk} = 1$ and $a_{(i+1)(j-1)k} = 0$, then it is a corner as shown in Fig. 3(c). The other vertices B, C and D have similar cases. Checking all blocks assigned to entity k, the number of corners of entity k will be obtained. These rules equate to Eq. (5) to calculate the number of corners of an entity.

$$\begin{aligned} n_k^c = \sum_{i=1}^{W}\sum_{j=1}^{L}[&a_{ijk}(1-a_{i_2jk})(1-a_{ij_1k})(1-a_{i_2j_1k}) \\ &+ a_{ijk}a_{i_2jk}a_{ij_1k}(1-a_{i_2j_1k}) + a_{ijk}(1-a_{i_2jk})(1-a_{ij_2k})(1-a_{i_2j_2k}) \\ &+ a_{ijk}a_{i_2jk}a_{ij_2k}(1-a_{i_2j_2k}) + a_{ijk}(1-a_{i_1jk})(1-a_{ij_2k})(1-a_{i_1j_2k}) \\ &+ a_{ijk}a_{i_1jk}a_{ij_2k}(1-a_{i_1j_2k}) + a_{ijk}(1-a_{i_1jk})(1-a_{ij_1k})(1-a_{i_1j_1k}) \\ &+ a_{ijk}a_{i_1jk}a_{ij_1k}(1-a_{i_1j_1k})], \\ i \in [1,W], \quad & j \in [1,L], \quad k \in [1,N], \quad i_1 = i-1, \\ i_2 = i+1, \quad & j_1 = j-1, j_2 = j+1 \end{aligned} \tag{5}$$

where $a_{ijk} = 0$ if $i = 0$ or $i = W + 1$ or $j = 0$ or $j = L + 1$.

2.5.4. Objective function and constraints

The problem described above is mathematically formulated as follows:

$$max\, z = \left(1 - \text{Moran}'\text{s } I\right) \cdot \left(\prod_{i=1}^{N}\frac{1}{\varphi_i}\right)^{1/N} \cdot \sum_{i=1}^{N-1}\sum_{j=i+1}^{N} b_{ij}c_{ij} \tag{6}$$

s.t.

$$\sum_{k=1}^{N} a_{ijk} \le 1, \quad i \in [1,W], \quad j \in [1,L] \tag{7}$$

$$\sum_{i=1}^{W}\sum_{j=1}^{L} a_{ijk} = A_k, \quad k \in [1,N] \tag{8}$$

$$\begin{aligned} &\sum_{i' \in I} a_{i'jk} + \sum_{j' \in J} a_{ij'k} \ge a_{ijk}, \quad i \in [1,W], j \in [1,L], k \in [1,N], \\ &I = \left\{i' \middle| |i-i'| = 1\right\}, J = \left\{j' \middle| |j-j'| = 1\right\}, A_k > 1 \end{aligned} \tag{9}$$

$$n_k^c \le N_k^c \text{ or } n_k^c = N_k^c, \quad k \in [1,N] \tag{10}$$

$$a_{ijk} \in \{0,1\}, \quad i \in [1,W], \quad j \in [1,L], \quad k \in [1,N] \tag{11}$$

Eq. (6) is the objective function, which considers three factors: the attraction distribution, the shape ratio of all entities and the adjacency requirement, where φ_i is defined as in Eq. (1). Eq. (7) prevents more than one entity being placed in the same location. Eq. (8) ensures that each entity gets its required area. Eq. (9) ensures each block is adjacent to at least one block that has been assigned to the same entity whose required area is greater than 1. Eq. (10) controls the number of corners of each entity, and is flexible to adopt the former constraint or the latter constraint. If the former constraint is chosen, the shape of an entity has greater freedom like classical SFC; or else, the shape is more specific. n_k^c is defined in Eq. (5) in the previous section.

If a solution violates constraint Eq. (10), a penalty will be applied. The penalty function is:

$$p\left(n_{inf}\right) = k n_{inf} z / N \tag{12}$$

where n_{inf} is the number of entities which violate Eq. (10). Experimentally we find that using $k = 2$ is appropriate though the method is not sensitive to this particular parameter. The resulting fitness function for evaluating the quality of a solution is:

$$f = z - p\left(n_{inf}\right). \tag{13}$$

3. Methodology

The model above cannot be solved readily to optimality. We chose to use a tabu search (TS) meta-heuristic because of the combinatorial nature of the model and the strong neighborhood structure of the block layout designs. This section details an enhanced SFC and the TS. The SFC is used to generate feasible candidate layouts with regular entity shapes and the TS is used to search over among candidate layouts.

3.1. The improved SFC

3.1.1. Procedure of generating the SFC

We use an *xy*-oscillatory model (Islier, 1998) to generate the SFC as shown in Fig. 1 which is applicable to cases where W is odd. But when W is even, an entity's curve on the top or bottom of the site may need to be adjusted to be contiguous. We do this as described below.

A key step for generating the SFC is how to move to the next step. A method is proposed to establish the rules governing this movement. Let (i, j, D_h, D_v) denote the current position and the current moving directions, where i denotes the ith row, j denotes the jth column, D_h denotes the horizontal moving direction and D_v denotes the vertical moving direction. $D_h = 1$ indicates horizontal moving from left to right; $D_h = -1$ indicates horizontal moving from right to left. $D_v = 1$ indicates vertical moving from down to up; $D_v = -1$ denotes vertical moving from up to down. The start position is at the bottom-left, where $(i, j, D_h, D_v) = (1, 1, 1, 1)$. It can be deduced that the current position is in the j_bth column of the bth bay from j. B_b denotes the width of the bth bay. The rules of seeking the next position are shown in Table 2.

In Table 2, when the SFC moves from a row to its adjacent row such as P2→P3 in Fig. 1, the horizontal moving direction changes, i.e., $D_h = -D_h$; when the SFC moves from a bay to its adjacent bay such as P7→P8 in Fig. 1, the vertical moving direction changes, i.e., $D_v = -D_v$.

The pseudo code of the procedure of generating the SFC is given below:

```
procedure GenerateSFC (Path[], Sfc[][])
    //Path[]: array of entities to be placed in sequence
    //Sfc[][]: array of entities placed in position (i, j)s
    for each block (i, j),  Sfc[i][j]←0;  // initialization: set all positions to be unassigned
    //---generate SFC---
```

Table 2
The rules of seeking the next position of the SFC.

No.	If	Then	Notes (see Fig. 1)
(1)	($B_b = 1$ and $D_v = 1$ and $i \neq W$) or ($B_b = 1$ and $D_v = -1$ and $i \neq 1$)	$D_h = D_h$, $D_v = D_v$, $i = i + D_v$, $j = j$	
(2)	($j_b \neq B_b$ and $D_h = 1$) or ($j_b \neq 1$ and $D_h = -1$ and $B_b \neq 1$)	$D_h = D_h$, $D_v = D_v$, $i = i$, $j = j + D_h$	$P_1 \rightarrow P_2$ $P_3 \rightarrow P_4$
(3)	($j_b = B_b$ and $D_h = 1$ and $D_v = 1$ and $i \neq W$ and $B_b \neq 1$) or ($j_b = B_b$ and $D_h = 1$ and $D_v = -1$ and $i \neq 1$ and $B_b \neq 1$) or ($j_b = 1$ and $D_h = -1$ and $D_v = 1$ and $i \neq W$) or ($j_b = 1$ and $D_h = -1$ and $D_v = -1$ and $i \neq 1$)	$D_h = -D_h$, $D_v = D_v$, $i = i + D_v$, $j = j$	$P_2 \rightarrow P_3$ $P_9 \rightarrow P_{10}$ $P_5 \rightarrow P_6$ $P_{11} \rightarrow P_{12}$
(4)	($j_b = B_b$ and $D_h = 1$ and $D_v = 1$ and $i = W$) or ($j_b = B_b$ and $D_h = 1$ and $D_v = -1$ and $i = 1$)	$D_h = D_h$, $D_v = -D_v$, $i = i$, $j = j + 1$	$P_7 \rightarrow P_8$ $P_{13} \rightarrow P_{14}$
(5)	($j_b = 1$ and $D_h = -1$ and $D_v = 1$ and $i = W$) or ($j_b = 1$ and $D_h = -1$ and $D_v = -1$ and $i = 1$)	$D_h = -D_h$, $D_v = -D_v$, $i = i$, $j = j + B_b$	When W is even

```
    i ← 0;
    j ← 0;
    m←1;   //the mth entity to be placed
    do
        k←0;   //count the blocks assigned to the mth entity
        if i = 0 and j = 0 then
            (i, j, D_h, D_v)←(1, 1, 1, 1);   // set the start position
            Sfc[i][j]←Path[m];
            k←k + 1;
        end if
        do
            SeekNextPosition(i, j, D_h, D_v); //according to the rules of Table 2;
            if W is even and k = 0, then AdjustSFC (i, j, D_h, D_v, Path[m]);
            Sfc[i][j]←Path[m];   //assign block (i, j) to the mth entity
            k←k + 1;
        while (k < A[Path[m]]);   //A[]: the area required by a entity
        m←m+1;
    while (m < Deptnum+1);   // Deptnum: the number of entities
end GenerateSFC
```

When W is even, in order to enforce an entity's curve to be contiguous, the following procedure is performed:

```
procedure AdjustSFC (i, j, D_h, D_v, n)
    // Current position (i, j, D_h, D_v) is the first position of entity n because k = 0
    if (i = 1 or i = W) and D_h = −1, then
        if j_b ≠ B_b and A[n] > j_b, then
            if moving Sfc[i][j_1], j_1 = j + 1, ..., j + B_b − j_b to Sfc[i][j_1 − j_b] is permitted,
then   // check whether the blocks are contiguous if moved
                Sfc[i][j_1 − j_b] ← Sfc[i][j_1], Sfc[i][j_1] ← 0, j_1 = j + 1, ..., j + B_b − j_b;
                (i, j, D_h, D_v)←(i, j + B_b − 2j_b + 1, −D_h, D_v);
            else
                (i, j, D_h, D_v)←(i, j + B_b − j_b + 1, −D_h, −D_v);   //move to next bay
            end if
        end if
    end if
end AdjustSFC
```

If there are any fixed entities in the planar site, the algorithm of generating the SFC can be slightly modified as follows:

(1) Place the fixed entities and label these blocks assigned to the fixed entities at the initial state.
(2) When assigning a new block to an entity, check whether this block is occupied or not. If yes, seek the next block; if no, go to (3).
(3) If it is not the first block in the entity to be placed, check whether this block has adjacent blocks or not according to Eq. (9). If no, reset this entity's placement (that is, it is ready to place again). If yes, assign this block to the entity.

3.1.2. Modifying the generation of the SFC

Given that L and W are fixed to agree with the sum of the entity areas, there may be few or no feasible solutions when the upper limit on the number of corners becomes small (e.g., 4 or 6). Because most layouts consist of entities with regular shapes, it is necessary to optionally relax the procedure of generating the SFC to adhere to the shape constraints.

To generate enough feasible layouts, we use a flexible size of the planar site and a flexible area of each entity. An entity can obtain more area than its minimum requirement to achieve a feasible shape. If the initial size of the planar site is not large enough to accommodate the increase in entity area, the site extends in length. Specifically, if the current position is at the last block of the site and there are still one or more entities to be placed, a new bay will be generated. Note that the last bay is as small as possible per the algorithm below.

Set A_r to the sum of the remaining area of the entities not yet placed.

(1) Set $A_B \leftarrow B_b \times W$;
(2) Check whether (($D_v = 1$ and $D_h = 1$ and $i = 1$) or ($D_v = -1$ and $D_h = 1$ and $i = W$)) and $j_b = 1$ and $A_r < A_B$. If yes, set $B_b \leftarrow \text{int}(B_b \times \frac{A_r}{A_B}) + 1$.

The modified rules of generating the SFC are listed as follows. We use these when the upper limit on the number of corners is 4 or 6.

Rule 1: If $N_k^c = 4$, the first position (1 subscript) and the last position (2 subscript) assigned to an entity should meet one of the conditions as follows:

(1) $i_1 = i_2$ and ($b_1 = b_2$ or $i_1 = W$ or $i_1 = 1$);
(2) $|i_1 - i_2| = 1$ *and* $j_1 = j_2$;
(3) ($D_{h1} = 1$ and $j_{b1} = 1$) and (($D_{h2} = 1$ and $j_{b2} = B_{b2}$) or ($D_{h2} = -1$ and $j_{b2} = 1$)) and ($b_1 = b_2$ or $i_1 = i_2$);
(4) ($D_{h1} = -1$ and $j_{b1} = B_{b1}$) and (($D_{h2} = -1$ and $j_{b2} = 1$) or ($D_{h2} = 1$ and $j_{b2} = B_{b2}$)) and ($b_1 = b_2$ or $i_1 = i_2$) .

Rule 2: If $N_k^c = 6$, the first position (1 subscript) and the last position (2 subscript) assigned to an entity should meet one of the conditions as follows:

(1) $|i_1 - i_2| \leq 1$ and ($b_1 = b_2$ or $i_1 = W$ or $i_1 = 1$);
(2) ($b_1 = b_2$) and (($D_{h2} = -1$ and $j_{b2} = 1$) or ($D_{h2} = 1$ and $j_{b2} = B_{b2}$));
(3) ($D_{h1} = 1$ and $j_{b1} = 1$) and ($b_1 = b_2$ or ($i_2 = W$ or $i_2 = 1$) or ($D_{h2} = 1$ and $j_{b2} = B_{b2}$) or ($D_{h2} = -1$ and $j_{b2} = 1$));
(4) ($D_{h1} = -1$ and j_{b1} $m = B_{b1}$) and ($b_1 = b_2$ or ($i_2 = W$ or $i_2 = 1$) or ($D_{h2} = 1$ and $j_{b2} = B_{b2}$) or ($D_{h2} = -1$ and $j_{b2} = 1$)).

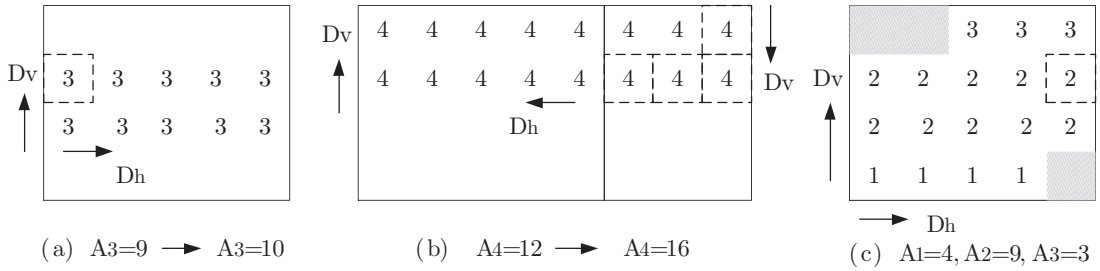


Fig. 4. Diagram of the algorithm for Rule 1 (with dashed boxes are extended to form the required shape, and the shaded blocks are unassigned).

Below are the corresponding algorithms to ensure that each entity observes Rule 1 and Rule 2.

The algorithm for Rule 1:

(1) At the completion of the placement of an entity, check whether it adheres to Rule 1. If not, go to step (2); if yes, exit.
(2) Check whether the entity meets ($D_{h1} = 1$ and $j_{b1} = 1$) or ($D_{h1} = -1$ and $j_{b1} = B_{b1}$) and meets:
   (I) $B_{b1}(W - i_1 + 1) \geq A_k$ or $i_1 = W + 1 - k - int(A_k / (B_{b1} + B_{b1+1}))$ or $i_1 = 1$ when $D_{v1} = 1$, or
   (II) $B_{b1} i_1 \geq A_k$ or $i_1 = k + int(A_k / (B_{b1} + B_{b1+1}))$ or $i_1 = W$ when $D_{v1} = -1$, where $k = 0$ *if* A_k *mode* $(B_{b1} + B_{b1+1}) = 0$, or else $k = 1$.
   If not, move the first position of the entity forward until it meets these conditions. The skipped space which is not used is classified as unassigned.
(3) Place the unit blocks of the entity until it obtains an area not less than the required area and meets Rule 1.

The examples are given in Fig. 4.

The algorithm for Rule 2:

(1) At the completion of the placement of an entity, check whether it adheres to Rule 2. If not, go to step (2); if yes, exit.
(2) Check whether the entity meets ($D_{h1} = 1$ and $j_{b1} = 1$) or ($D_{h1} = -1$ and $j_{b1} = B_{b1}$). If not, move the first position of the entity forward until it meets these conditions. The skipped space which is not used is classified as unassigned.
(3) Place the unit blocks of the entity until it obtains an area not less than the required area and meets Rule 2.

Although the above rules and algorithms may not make the shape of an entity in full compliance with the shape requirement (that is, the shape ratio might still be violated), they greatly increase the probability of obtaining a design where the entities all have the desired shapes.

3.2. Tabu search

TS is a neighborhood search algorithm with a mechanism which prohibits certain moves within the neighborhood to avoid cyclical behavior. TS has been widely applied into many combinatorial, NP-hard problems including unequal area facility layout (Yapicioglu and Smith, 2012a,b). Our implementation of TS is discussed in the following sections.

3.2.1. Solution representation

The bay widths and the placement sequence of the entities are critical to generate a layout. The solution representation in TS includes two segments as shown in Fig. 5. The first segment represents a placement sequence of entities to be located. The second segment represents the bay widths. $B_{min} \leq B_i \leq B_{max}$, where $B_{min} = \sqrt{A_s}$, $B_{max} = \sqrt{A_l}$, A_s is the area of the smallest entity and A_l is the area of the largest entity, B_i, B_{min}, B_{max} are integer numbers. These settings make entities squarer. The initial solution is generated randomly.

If L is fixed, $\sum_{i=1}^{N_b} B_i = L$. Adjust the last bay width to meet this constraint. The pseudo code of the procedure of generating the initial bay widths is given below:

```
procedure GenerateBaywidths (B[])
    //B[]: array of bay widths, B[i] is B_i
    sum←0;
    i←1;
    while(sum<L)
    {
        B[i]← rand() mode (B_max − B_min + 1)+ B_min;
        sum←sum +B[i];
        i←i +1;
    }
    i←i - 1;
    sum←sum - B[i];
    B[i]←L - sum;
end GenerateBaywidths
```

Once the sequence of all entities is determined, the placement procedure in Section 3.1 is used to allocate the blocks in the planar site to all ordered entities from the start position bay by bay.

3.2.2. Move operator and neighborhood

A different move operator is used in each of the two segments of the solution and these produce two distinct neighborhoods. The first move operator is a 2-opt swap operator, which exchanges the entities located in the ith position and the jth position of the first segment (entity permutation) of the current solution while keeping the second segment unchanged, where $i = 1, \ldots, N-1$ and $j = i+1, \ldots, N$. If N is large (for example $N \geq 30$), we only select a portion of all possible swaps randomly, so the speed of the TS can be increased.

The second move operator is a single-point move operator, which increases 1 or decreases 1 to the ith bay width of the second segment of current solution while keeping the first segment of current solution unchanged, where $i = 1, \ldots, N_b$. If a bay width is greater than B_{max}, set it to B_{min}; if a bay width is less than B_{min}, set it to B_{max}. Then, adjust the final bay width to keep $\sum_{i=1}^{N_b} B_i = L$.

These two neighborhoods are alternatively used in TS, that is, swap the sequence of the first segment for $Iteration_{nbh1}$ iterations and then perturb the values of the second segment for $Iteration_{nbh2}$ iterations. This simple but effective move structure works well with the encoding of the sequence and bay widths.

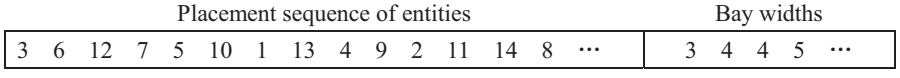


Fig. 5. An example of the solution representation for the TS.

3.2.3. Candidate list

The solutions of each neighborhood are sorted in descending order. The top $Size_{cand1}$ elements of the first neighborhood are selected to form candidate list 1. All elements in the second neighborhood form candidate list 2 (because the neighborhood is much smaller). The best solution of a candidate list which is not tabu is selected as the solution for the next iteration. If all solutions are tabu, the best solution of a candidate list will be chosen as the solution for next iteration.

3.2.4. Tabu list

There are two tabu lists corresponding to the two candidate lists. The most recent pairs which are swapped are kept on tabu list 1 for a certain number of iterations. The most recent fitness values are kept on tabu list 2. The size of tabu list 1 is $Size_{tabu1}$. The size of tabu list 2 is $Size_{tabu2}$.

3.2.5. Aspiration criterion

For both candidate lists, if a solution on the candidate list has a better fitness value than the best found solution, a move to that solution is allowed even if the move is tabu.

3.2.6. Diversification strategy

If the best found solution has no improvement for $Iteration_{dive}$ consecutive iterations, regenerate a totally new random solution and reset the tabu lists. Then restart the search with the empty tabu lists. Note that only $Iteration_{nbh1}$ iterations account for the non-improving consecutive iterations.

3.2.7. Termination criterion

If the best found solution has no improvement for $Iteration_{term}$ consecutive iterations, the search terminates. Note that only $Iteration_{nbh1}$ iterations (that is, permutation swap iterations) account for the non-improving consecutive iterations.

4. Case studies

A hypothetical layout case and two practical layout cases are presented in this section. The above algorithms are implemented using Visual C++ 6.0. Experiments are done on an HP EliteBook 820 G3 with a 2.5 gigahertz Intel CPU and 8 gigabytes RAM. (All case study data will be provided in an on line supplement to the published paper.)

4.1. A hypothetical exhibition case

An exhibition case study is constructed, where 40 attractions (entities) (No. 1–40) are to be placed. The area required by each entity ranges from 4 to 25, $B \in [2, 5]$. 20% of the entities have high attraction values ranging [0.7, 1] and the others have lower attraction values ranging [0.1, 0.5]. The adjacency value of any pair of entities ranges from 0 to 5. All case data needed is generated randomly according to the above rules. The areas required by the 40 entities are listed in Table 3. For simplicity, we set all the maximum numbers of corners for all entities to be 8, i.e., $N_c = 8$. $L = 25$ and $W = 21$ and $L \times W \geq \sum_{k=1}^{N} A_k$. Entities 8, 12, 27 and 29 have high attraction values.

The parameters of the TS are set as follows: 50% of all possible swaps are selected to be the permutation neighborhood, $Size_{cand1} = 200$, $Size_{tabu1} = 50$, $Size_{tabu2} = 20$, $Iteration_{nbh1} = 200$, $Iteration_{nbh2} = 100$, $Iteration_{dive} = 200$, $Iteration_{term} = 2000$. These parameters were set rather arbitrarily and the method is not sensitive to exact values of them.

We performed 10 runs of the TS under this combination of parameters, each with a different starting point. The test results are shown in Table 4. The fitness value evolving with iterations in the best run is shown in Fig. 6. It can be seen that the standard variation is very small, and entities 8, 12, 27 and 29 with high attraction values are dispersed to some extent. The three objective factors (i.e., the attraction distribution, the shape ratio of the entities and the adjacency requirement) are 1.3408, 0.9154 and 1321.2000, respectively. The run time is 231.86 seconds. Also, we found the three individual best objective function values by performing the same 10 runs but with one objective function only (attraction or shape ratio or adjacency requirement). The comparison between these single three objective factors with the ones from the layout which considers all of them simultaneously is found in Table 5. As can be seen, our design achieves more than 90% of the best possible values while balancing all three aspects of the design – attraction, shape ratio and adjacency.

Table 3
The areas and attractions of the 40 entities.

No.	Area	Attraction	No.	Area	Attraction	No.	Area	Attraction
1	12	0.25	15	12	0.43	29	12	0.96
2	8	0.31	16	15	0.32	30	16	0.20
3	20	0.13	17	12	0.42	31	10	0.44
4	25	0.21	18	12	0.41	32	6	0.72
5	15	0.73	19	8	0.48	33	8	0.84
6	10	0.44	20	6	0.42	34	8	0.21
7	6	0.16	21	8	0.18	35	10	0.39
8	20	0.97	22	15	0.12	36	10	0.15
9	6	0.17	23	10	0.42	37	12	0.19
10	16	0.19	24	20	0.33	38	6	0.19
11	16	0.33	25	25	0.15	39	16	0.31
12	15	0.94	26	8	0.13	40	16	0.27
13	20	0.35	27	20	0.96			
14	6	0.45	28	10	0.21			

Table 4
The results of 10 runs.

Average fitness value	Best fitness value	Standard variation	Average run time (seconds)
1609.82	1621.60	5.46	215.07

Table 5
Comparing the individual best objective function values with those of our design, $N_c = 8$.

	Attraction distribution	Shape ratio	Adjacency
Values for our best design	1.3408	0.9154	1321.2000
Values for single objective function	1.4759	0.9579	1434.2000
Ratio of the two	90.84%	95.56%	92.12%

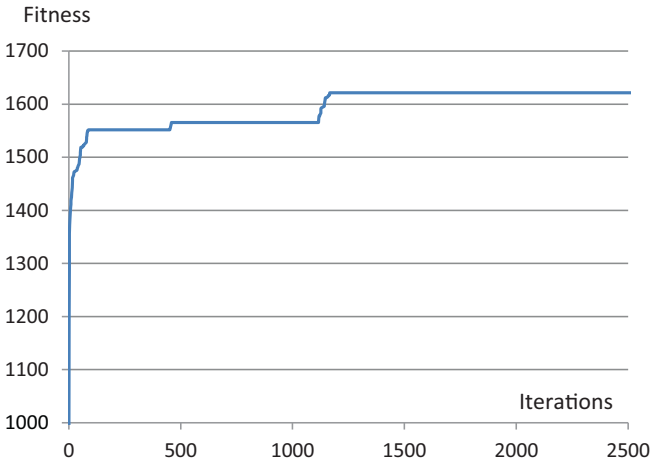


Fig. 6. Fitness vs iterations of the best run, site size 25 by 21, $N_c = 8$.

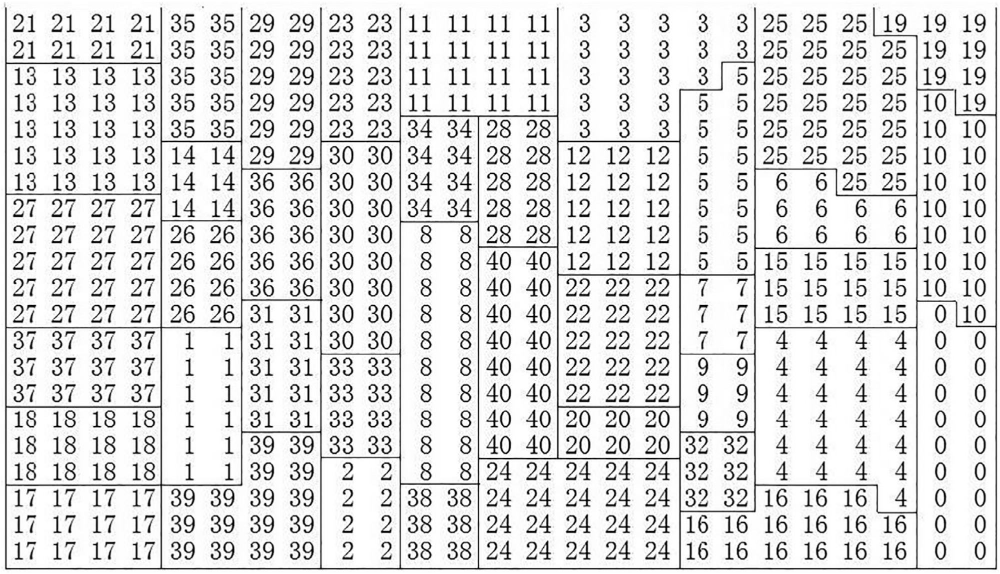


Fig. 7. The best layout, 40 entities, site size 25 by 21, $N_c = 8$.

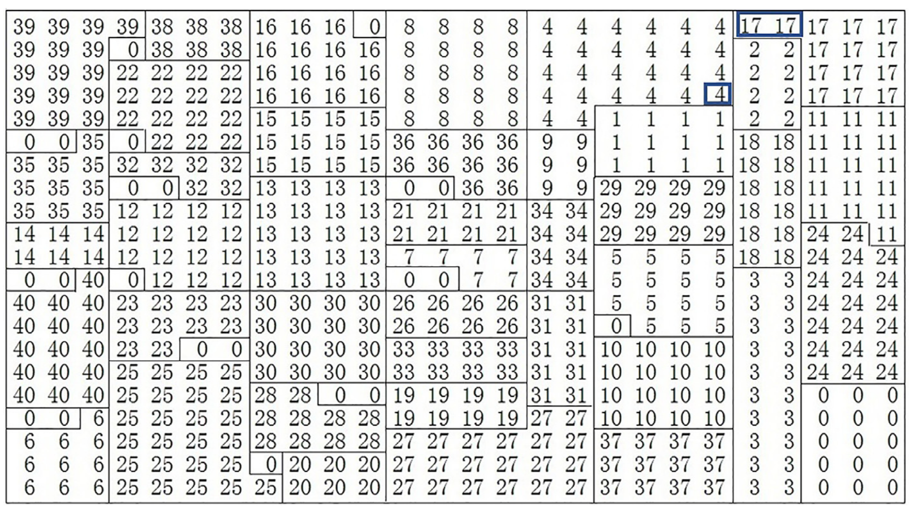


Fig. 8. The best layout, 40 entities, site size 26 by 21, $N_c = 6$.

The best layout is shown in Fig. 7, where number 0 denotes blocks which are unassigned, and numbers 1–40 are the attraction numbers. This layout complies to the procedure of generating SFC, which shows that the procedure is correct. All entities have eight or fewer corners, which indicates the corner-control method is effective.

4.2. Considering more shape constrained layouts

We now consider the situation above but impose a stronger constraint on the shape by reducing the maximum number of corners from 8 to 6. To accomplish this, L might need to be extended. We performed 10 runs using the combination of parameters of Section 4.1 and $N_c = 6$. The best fitness value is 1720.15. The three objective factors (i.e., the attraction distribution, the shape ratio of the entities and the adjacency requirement) are 1.2963, 0.9255 and 1433.8000 respectively. The run time is 405.21 seconds. The best layout is shown in Fig. 8. Entities 4 and 17 are enlarged as marked by thick border. There also exists a little unused space labeled by 0. Accordingly, the site extends in length from 25 to 26, a minor increase which allows the identification of a design which adheres to the corner constraint value imposed on all entities. Entities 8, 12, 27 and 29 with high attraction values are scattered to some extent.

Table 6
Comparing the individual best objective function values with those of our design, $N_c = 6$.

	Attraction distribution	Shape ratio	Adjacency
Values for our best design	1.2963	0.9255	1433.8000
Values for single objective function	1.4839	0.9633	1600.2000
Ratio of the two	87.36%	96.08%	89.60%

For this more constrained problem, we again compare with the best achieved individual objective functions and find our design performs very well on balancing the goals. See Table 6.

Finally, we performed 10 runs using the combination of parameters of Section 4.1 and $N_c = 4$ so all entities must be rectangular, the most shape constrained design. The best fitness value is 1839.90. The three objective factors (i.e. the attraction distribution, the shape ratio of the entities and the adjacency requirement) are 1.2364, 0.9490 and 1568.0000, respectively. The run time is

27	27	27	27	5	5	5	5	5	25	25	25	25	25	25	21	21	21	21	14	14	14	14	14	0	0	0
27	27	27	27	5	5	5	5	5	25	25	25	25	25	25	21	21	21	21	14	14	14	14	14	17	17	17
27	27	27	27	5	5	5	5	5	25	25	25	25	25	25	22	22	22	22	16	16	16	16	16	17	17	17
27	27	27	27	31	31	31	23	23	25	25	25	25	25	25	22	22	22	22	16	16	16	16	16	17	17	17
27	27	27	27	31	31	31	23	23	25	25	25	25	25	25	22	22	22	22	16	16	16	16	16	17	17	17
0	0	36	36	31	31	31	23	23	0	0	13	13	13	13	22	22	22	22	24	24	24	24	24	18	18	18
0	0	36	36	31	31	31	23	23	0	0	13	13	13	13	19	19	19	19	24	24	24	24	24	18	18	18
0	0	36	36	37	37	37	23	23	0	0	13	13	13	13	19	19	19	19	24	24	24	24	24	18	18	18
28	28	36	36	37	37	37	26	26	0	0	13	13	13	13	29	29	29	29	24	24	24	24	24	18	18	18
28	28	36	36	37	37	37	26	26	0	0	13	13	13	13	29	29	29	29	6	6	6	6	6	9	9	9
28	28	38	38	37	37	37	26	26	20	20	12	12	12	12	29	29	29	29	6	6	6	6	6	9	9	9
28	28	38	38	30	30	30	26	26	20	20	12	12	12	12	33	33	33	33	10	10	10	10	10	1	1	1
28	28	38	38	30	30	30	7	7	20	20	12	12	12	12	33	33	33	33	10	10	10	10	10	1	1	1
35	35	0	0	30	30	30	7	7	2	2	12	12	12	12	15	15	15	15	10	10	10	10	10	1	1	1
35	35	0	0	30	30	30	7	7	2	2	40	40	40	40	15	15	15	15	10	10	10	10	10	1	1	1
35	35	0	0	30	30	30	0	0	2	2	40	40	40	40	15	15	15	15	4	4	4	4	4	39	39	39
35	35	0	0	30	30	30	0	0	2	2	40	40	40	40	3	3	3	3	4	4	4	4	4	39	39	39
35	35	8	8	8	8	8	11	11	11	11	40	40	40	40	3	3	3	3	4	4	4	4	4	39	39	39
32	32	8	8	8	8	8	11	11	11	11	34	34	34	34	3	3	3	3	4	4	4	4	4	39	39	39
32	32	8	8	8	8	8	11	11	11	11	34	34	34	34	3	3	3	3	4	4	4	4	4	39	39	39
32	32	8	8	8	8	8	11	11	11	11	0	0	0	0	3	3	3	3	0	0	0	0	0	39	39	39

Fig. 9. The best layout, 40 entities, site size 27 by 21, $N_c = 4$.

Table 7
Comparing the individual best objective function values with those of our design, $N_c = 4$.

	Attraction distribution	Shape ratio	Adjacency
Values for our best design	1.2364	0.9490	1568.0000
Values for single objective function	1.5588	0.9656	1683.6000
Ratio of the two	79.32%	98.28%	93.13%

297.75 seconds. The best layout is shown in Fig. 9. Some entities are enlarged to form a rectangular shape, as marked by thick border. There also exists some unused space labeled by 0. So the site expands in length from 25 to 27, which is slightly larger than the size needed for $N_c = 6$, a remarkable efficiency. Entities 8, 12, 27 and 29 with high attraction values are still properly scattered. Comparing with the individual best objective function values in Table 7 we see our design does a good job of balancing them even under the severe shape constraints of $N_c = 4$.

These results on this hypothetical case show that our approach can effectively control entity shape and still enable the entities with high attraction values to be located evenly.

4.3. An actual exhibition case

Fig. 10 is the layout of the exhibition "Lost Formats" held in 2011 (https://rebeccacivil.wordpress.com/2011/03/). This exhibition took place in the United Kingdom and showed gaming consoles from the 1980s including a wide range from the Sega Mega Drive

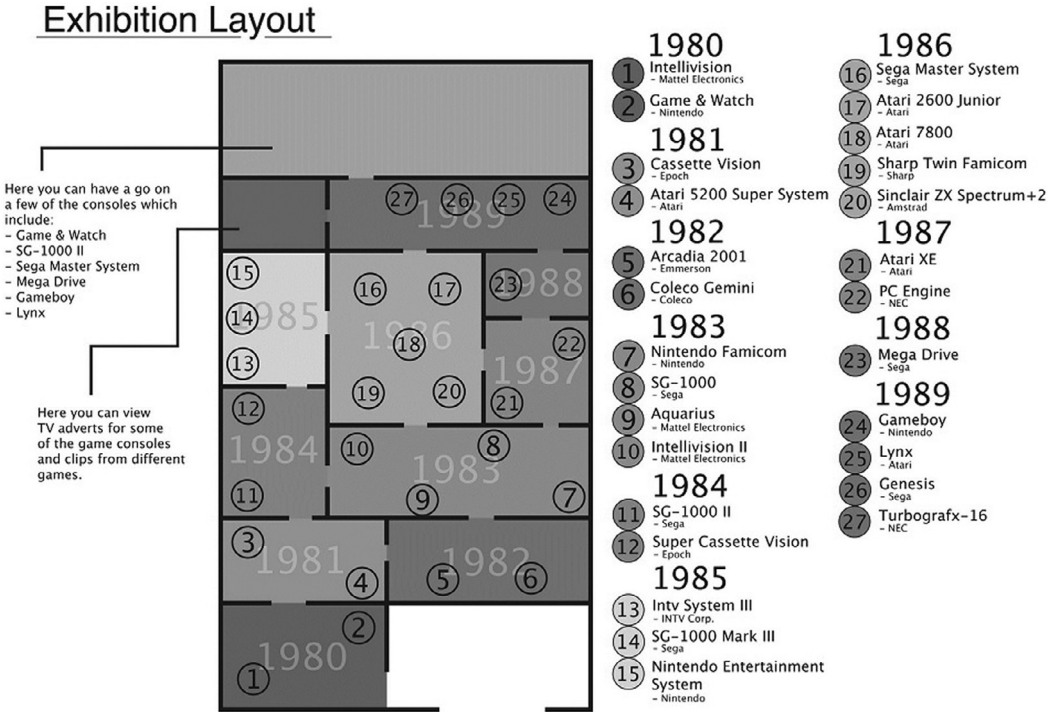


Fig. 10. The layout of the exhibition "Lost Formats" held in 2011 (data source: https://rebeccacivil.wordpress.com/2011/03/).

Table 8
The parameters of the entities in Fig. 11.

No.	Area	Attraction	No.	Area	Attraction	No.	Area	Attraction
1(1980)	40	0.48	6(1985)	35	0.28	11(TVs)	20	0.75
2(1981)	32	0.26	7(1986)	72	0.45	12(Demos)	95	0.90
3(1982)	44	0.35	8(1987)	30	0.38	13(Lobby)	55	0.5
4(1983)	56	0.42	9(1988)	24	0.32			
5(1984)	30	0.40	10(1989)	56	0.85			

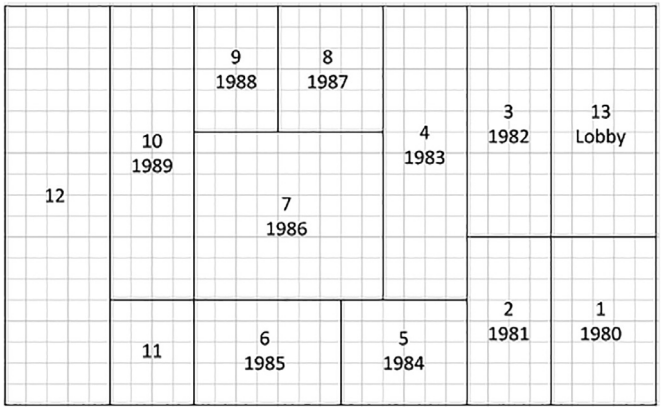


Fig. 11. The layout converted from Fig. 10, site size 31 by 19.

to the Nintendo Gameboy. Fig. 10 can be approximately converted to Fig. 11. Note that we kept the exhibits within a single year together so merged the individual numbers of Fig. 10 to the years of Fig. 11. Most exhibits are by year but there is also a small area with historic television advertisements (entity 11 in Fig. 11), an interactive demo area (entity 12 in Fig. 11), and a lobby (entity 13 in Fig. 11).

The areas and attraction values of all entities are listed in Table 8. Entities 10, 11 and 12 have high attraction while the other entities have lower attraction to visitors. Because each of entities 1 to 10 correspond to a specific year, we first set the adjacency values of these according to the following rules: for $i, j \in [1, 10]$, if $|year_i - year_j| = 1$, $c_{ij} = 5$; if $|year_i - year_j| = 2$, $c_{ij} = 4$; if $|year_i - year_j| = 3$, $c_{ij} = 3$; if $|year_i - year_j| = 4$, $c_{ij} = 2$; if $|year_i - year_j| = 5$, $c_{ij} = 1$; if $|year_i - year_j| > 5$, $c_{ij} = 0$. Let $c_{1(13)} = 5$ and $c_{2(13)} = 4$ to enforce entities 1 and 2 to be near the lobby (a natural starting point for a visitor). Because of the specific location of entity 13 (the lobby), we place it last, so it can be at the right-upper corner of the site. As in the other cases, $c_{ij} = 1$. $L = 31$ and $W = 19$, but L can be extended slightly to accommodate the corner constraint if needed. The parameters of TS are set as follows: $Size_{cand1} = 200$, $Size_{tabu1} = 30$, $Size_{tabu2} = 20$, $Iteration_{nbh1} = 200$, $Iteration_{nbh2} = 100$, $Iteration_{dive} = 200$, $Iteration_{term} = 2000$.

We performed 10 runs and the best layout is shown in Fig. 12. The best fitness value is 182.69 which improves considerably compared with the fitness of the actual layout (Fig. 11) of 67.04. The run time is about 31 seconds. Some entities were enlarged to form a rectangular shape, as marked by thick border. There also exists some unused space labeled by 0 and the site expands in length from 31 to 34. If this expansion would be unallowable, our method would still generate a viable layout but there would be at least one entity violating the corner constraint of 4. Entities 10, 11 and 12 with high attraction values are separated by entities 2, 6, 9 with lower attraction values. This layout would have less crowding. However, this layout does not always maintain the years in the proper sequence (for example, 1985 is not close to 1986). Therefore, we performed two additional cases where the weighting for consecutive years is increased.

We strengthened the adjacency values of these entities according to the following rules: for $i, j \in [1, 10]$, if $|year_i - year_j| = 1$, $c_{ij} = 5$; if $|year_i - year_j| = 2$, $c_{ij} = 4$; if $|year_i - year_j| > 2$, $c_{ij} = 0$. Let $c_{1(13)} = 5$ and $c_{2(13)} = 4$. In other cases, $c_{ij} = 1$. The best result of 10 runs is given in Fig. 13.

The fitness value of this layout is 142.32 compared to a fitness value of the existing layout in Fig. 11 of 52.00. However, there is still not total organization by year. For example, 1982 and 1983 are not close to each other.

In the final version of this case study, we set the adjacency values of these entities according to the following rules: for $i, j \in [1, 10]$, if $|year_i - year_j| = 1$, $c_{ij} = 5$; $|year_i - year_j| = 2$, $c_{ij} = 1$; if $|year_i - year_j| > 2$, $c_{ij} = 0$. Let $c_{1(13)} = 5$ and $c_{2(13)} = 4$. In other cases, $c_{ij} = 1$. The best result of 10 runs is given in Fig. 14.

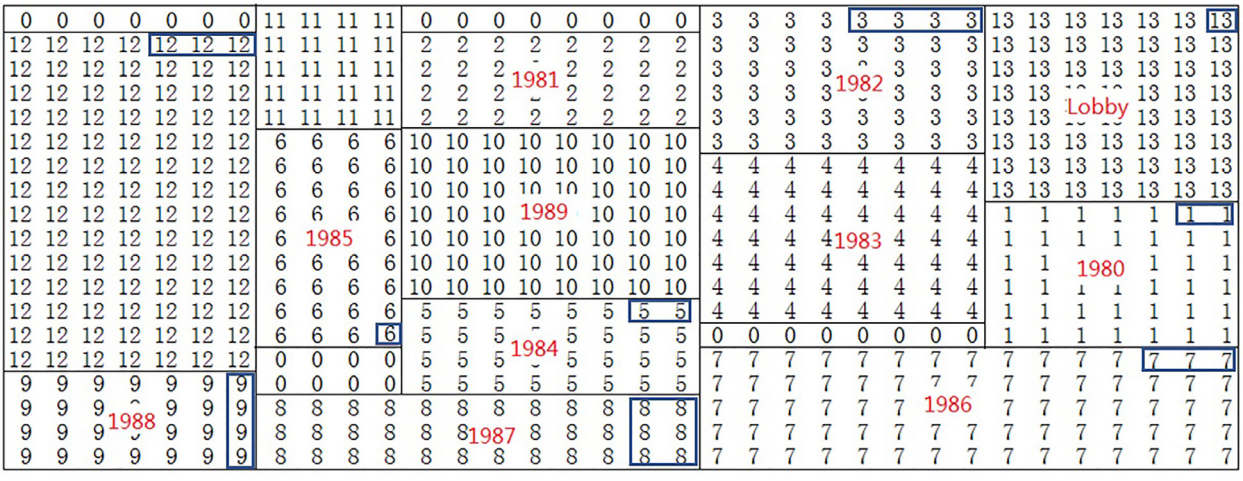


Fig. 12. The new layout of the exhibition “Lost Formats”, site size 34 by 19 with moderate adjacency by year.

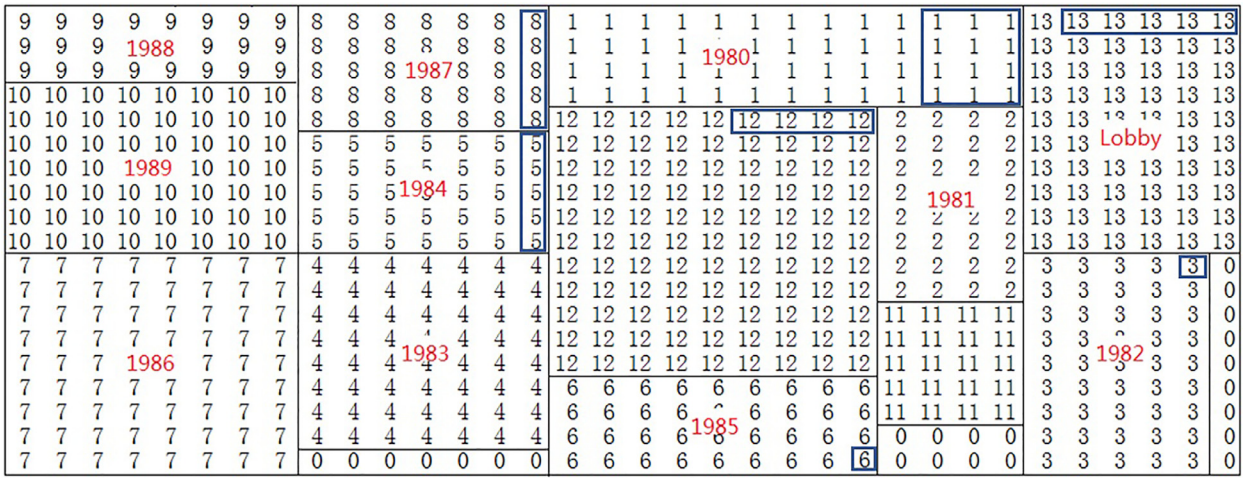


Fig. 13. The new layout of the exhibition "Lost Formats", site size 34 by 19 with adjacency by year strengthened.

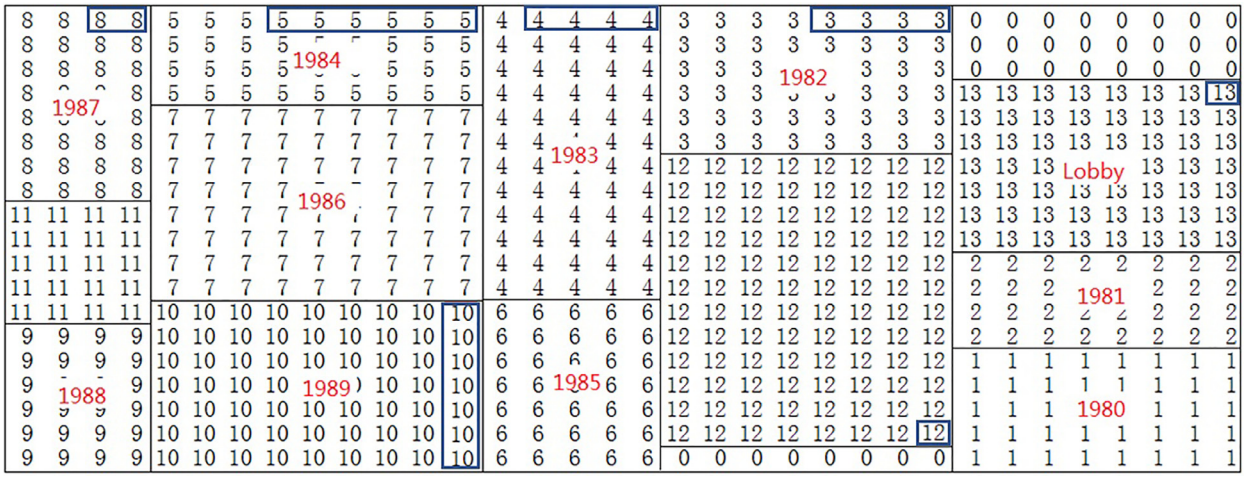


Fig. 14. The new layout of the exhibition "Lost Formats", site size 34 by 19 with strong adjacency by year.

The fitness value of this layout is 117.17 whereas the fitness value of the existing layout in Fig. 11 is 42.48. This final design significantly decreases visitor crowding and keeps a reasonable flow from year to year in the exhibition.

4.4. SeaWorld San Diego Park layout case

The SeaWorld San Diego Park is a famous theme park in the USA and attracts a large number of visitors from all over the world every day. According to online reviews (https://en.wikipedia.org/wiki/SeaWorld; http://articles.latimes.com/2011/may/29/news/la-trb-seaworld-shamu-one-ocean-05201128; etc.), One Ocean (also known as the killer whale show) is the SeaWorld's most popular attraction. Visitor crowding often occurs in the park. The attractions for the park are listed in Table 9.

The park map from the official website (https://seaworldparks.com/en/seaworld-sandiego/park-info/interactive-map?from=Top_Nav) was divided into a grid of square blocks. This translates to the layout shown in Fig. 15. Attractions 1, 5 and 6 are fixed because they have special geographic requirements. Label X represents unusable space because these areas are water. Label 0 represents unassigned but usable space. Attraction 3 has 8 corners and the other attractions are rectangular or *L*-shaped.

Because the park has the live queue times and popularity data can be obtained through surveys, the appeal of each attraction can be evaluated according to its average queue time and popularity.

Table 9
The list of attractions for SeaWorld San Diego.

No.	Name	No.	Name
1	Entrance and Exit	15	Mission Bay Theater
2	Shamu Stadium and Shamu's Underwater Viewing	16	Freshwater Aquarium
3	Shipwreck Rapids	17	Shark Encounter
4	World of Sea Aquarium	18	Turtle Reef
5	Cirque Stadium	19	Nautilus Pavilion
6	Bayside Skyride	20	Penguin Encounter
7	Manta	21	Pets Stadium
8	Dolphin Point	22	Wild Arctic
9	Explorer's Reef	23	Animal Connections
10	Skytower Ride	24	Journey to Atlantis
11	Sesame Street Bay of Play	25	Tide Pool
12	Dolphin Stadium	26	Otter Outlook
13	Pacific Point	27	Riptide Rescue
14	Sea Lion and Otter Stadium		

Here we refer to the online queue times (https://queue-times.com/parks/20/queue_times) and some online posted visitor comments about the park. Thus we have estimated the attraction values of attractions 1–27 as listed in Table 10. We do not consider adjacency here because we have no credible data to set those values. Therefore, the adjacency value of any pair of attractions is set to 3. $L = 16$ and $W = 11$, and $N_c = 8$. The parameters of the TS are as fol-

6	6	6	6	15	15	15	16	16	27	X	X	X	X	X	X
5	5	0	7	7	7	14	14	16	17	18	X	X	X	X	X
5	5	5	7	7	7	14	14	0	17	17	0	0	X	X	X
4	25	3	3	7	7	13	13	12	12	12	0	19	19	0	0
3	3	3	3	7	7	13	13	12	12	12	0	19	19	20	20
3	3	3	3	7	7	10	0	12	12	12	0	21	21	20	20
3	3	0	0	8	26	10	11	11	11	11	11	21	21	22	22
2	2	2	2	8	26	9	11	11	11	11	11	21	21	22	22
2	2	2	2	2	0	9	11	11	11	11	0	0	0	22	22
2	2	2	2	2	1	1	1	1	0	0	23	24	24	0	0
2	2	2	2	2	1	1	1	1	0	0	23	24	24	24	24

Fig. 15. The layout converted from the SeaWorld San Diego Park map, site size 16 by 11.

Table 10
The parameters of all entities in Fig. 15.

No.	Area	Attraction	No.	Area	Attraction	No.	Area	Attraction
1	8	0.10	10	2	0.65	19	3	0.20
2	19	0.98	11	14	0.55	20	4	0.40
3	12	0.95	12	9	0.90	21	6	0.72
4	1	0.45	13	4	0.42	22	6	0.60
5	5	0.48	14	4	0.85	23	2	0.32
6	4	0.80	15	3	0.35	24	6	0.76
7	12	0.87	16	3	0.26	25	1	0.38
8	2	0.37	17	3	0.50	26	2	0.30
9	2	0.24	18	1	0.28	27	1	0.65

6	6	6	6	3	7	7	7	7	7	x	x	x	x
5	5	0	3	3	7	7	7	7	20	20	x	x	x
5	5	5	3	3	23	7	7	7	20	20	11	11	x
26	26	2	3	3	23	16	16	16	14	14	11	11	8
2	2	2	3	3	27	19	19	19	14	14	11	11	8
2	2	2	3	3	12	12	12	12	10	10	11	11	0
2	2	2	3	9	12	12	12	12	15	15	11	11	0
2	2	2	18	9	12	24	24	24	25	15	11	11	0
2	2	2	21	21	4	24	24	24	22	22	11	11	0
2	2	2	21	21	1	1	1	1	22	22	13	13	0
17	17	17	21	21	1	1	1	1	22	22	13	13	0

Fig. 16. The new layout of the SeaWorld San Diego Park, site size 14 by 11.

lows: $Size_{cand1} = 200$, $Size_{tabu1} = 30$, $Size_{tabu2} = 20$, $Iteration_{nbh1} = 200$, $Iteration_{nbh2} = 100$, $Iteration_{dive} = 200$, $Iteration_{term} = 2000$.

We performed 10 runs and the best layout is shown in Fig. 16. The best fitness value is 985.48. The run time is about 52 seconds. Label 0 indicates unused space and X represents unusable space. Attractions 2, 3, 7, 12 and 14 with high attraction values are separated by several entities with lower attraction values. By contrast, attractions 2, 3, 7 and 12 are located together in Fig. 15. Because the layout in Fig. 15 has more unused space than the layout in Fig. 16, the new site size becomes 14 by 11, which is smaller than the original size 16 by 11. This shows a further value of our approach which is that the site may become more compact, allowing a smaller first cost or the potential for additional attractions to fit into the site. The fitness value of the layout in Fig. 15 is 706.50, and is considerably less than the best fitness value of 985.48 that we obtained. Again, customer satisfaction may be improved significantly using an approach as outlined in this paper.

5. Conclusions

There are many attractions based layout problems arising in enterprises such as casinos, theme parks, museums, technical exhibitions and science centers. While this block layout problem arises frequently in practice, there is virtually no previous analytical approach described in the literature. We describe a typical unequal area exhibition layout problem, and establish a mathematical model considering three objective factors, the attraction distribution of the layout, entity shape and the adjacency requirements of the entities. To control entity shape, a corner-control method based on a discrete layout representation is developed, and the classical SFC is improved. To measure the attraction distribution requirement of the layout, Moran's I of spatial statistical analysis is introduced. A tabu search is designed to find the best block design of the enterprise. This approach is very flexible regarding constraints and fixed areas and finds implementable layouts in a very reasonable computational time.

We considered three case studies. Each shows the value and practicality of the approach. Computational effort is low while the block layouts produced are reasonable and significantly reduce visitor crowding. Moreover, the needed adjacencies can be achieved and the user can set how regular the shape of each entity needs to be. Space unavailable for assignment can be readily blocked out and entities with fixed locations can also be enforced. Varying importance of adjacencies is straightforward to handle as illustrated in Case 2 where three potential layouts were identified depending on the designer's emphasis on year proximity of exhibits.

Our approach can control each entity shape to be rectangular, L-shaped or others by the corner-control method. No any other approach in the current literature uses this general approach. Possible extensions to this work are to consider aisles and walkways, entrance and exit points to each entity and length of typical visitor paths through the enterprise. Also, a bi-objective optimization model could be developed quite readily which would allow the objectives of equalizing visitor attraction and maximizing adjacencies to be handled separately to establish a Pareto optimal set of layout designs.

Acknowledgments

This study is supported by the project (# 14YJA630021) funded by the Humanities and Social Science Research Foundation of the Ministry of Education of China. We are grateful for the valuable comments and guidance of the two anonymous reviewers of this paper.

References

Ahmadi, A., & Jokar, M. R. A. (2016). An efficient multiple-stage mathematical programming method for advanced single and multi-floor facility layout problems. *Applied Mathematical Modelling, 40*(9–10), 5605–5620.

Aiello, G., Scalia, G. L., & Enea, M. (2012). A multi objective genetic algorithm for the facility layout problem based upon slicing structure encoding. *Expert Systems with Applications, 39*(12), 10352–10358.

Bashiri, M., & Karimi, H. (2012). Effective heuristics and meta-heuristics for the quadratic assignment problem with tuned parameters and analytical comparisons. *Journal of Industrial Engineering International, 8*(1), 1–9.

Bozer, Y. A., Meller, R. D., & Erlebacher, S. J. (1994). An improvement-type layout algorithm for single and multiple floor facilities. *Management Science, 40*(7), 918–932.

Cliff, A. D., & Ord, J. K. (1981). *Spatial processes, models and applications*. London: Pion.

Drira, A., Pierreval, H., & Hajri-Gabouj, S. (2007). Facility layout problems: A survey. *Annual Reviews in Control, 31*(2), 255–267.

Garey, M. R., & Johnson, D. S. (1979). *Computers and intractability: A guide to the theory of NP-completeness*. New York: WH Freeman.

Gonçalves, J. F., & Resende, M. G. C. (2015). A biased random-key genetic algorithm for the unequal area facility layout problem. *European Journal of Operational Research, 246*(1), 86–107.

Islier, A. A. (1998). A genetic algorithm approach for multiple criteria facility layout design. *International Journal of Production Research, 36*(6), 1549–1569.

Jaén, J., Mocholí, J. A., Catalá, A., & Navarro, E. (2011). Digital ants as the best cicerones for museum visitors. *Applied Soft Computing, 11*(1), 111–119.

Kawamura, H., Kataoka, T., Kurumatani, K., & Ohuchi, A. (2004). Investigation of global performance affected by congestion avoiding behavior in theme park problem. *IEEJ Transactions on Electronics Information and Systems, 124*(10), 1922–1929.

Komarudin, & Wong, K. Y. (2010). Applying ant system for solving unequal area facility layout problems. *European Journal of Operational Research, 202*(3), 730–746.

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

Kulturel-Konak, S. (2012). A linear programming embedded probabilistic tabu search for the unequal-area facility layout problem with flexible bays. *European Journal of Operational Research, 223*(3), 614–625.

Lee, K. Y., Roh, M. I., & Jeong, H. U. (2005). An improved genetic algorithm for multi-floor facility layout problems having inner structure walls and passages. *Computers & Operations Research, 32*(4), 879–899.

Moran, P. A. P. (1950). Notes on continuous stochastic phenomena. *Biometrika, 37*(1), 17–23.

Shayan, E., & Chittilappilly, A. (2004). Genetic algorithm for facilities layout problems based on slicing tree structure. *International Journal of Production Research, 42*(19), 4055–4067.

Wang, M-J., Hu, M. H., & Ku, M-Y. (2005). A solution to the unequal area facilities layout problem by genetic algorithm. *Computers in Industry, 56*(2), 207–220.

Rathnayake, R. M. W. (2015). How does 'crowding' affect visitor satisfaction at the Horton Plains National Park in Sri Lanka. *Tourism Management Perspectives, 16*, 129–138.

Ripon, K. S. N., Glette, K., Khan, K. N., Hovin, M., & Torresen, J. (2013). Adaptive variable neighborhood search for solving multi-objective facility layout problems with unequal area facilities. *Swarm and Evolutionary Computation, 8*(1), 1–12.

Saraswat, A., Venkatadr, U., & Castillo, I. (2015). A framework for multi-objective facility layout design. *Computers & Industrial Engineering, 90*(C), 167–176.

Tate, D. M., & Smith, A. E. (1995). Unequal-area facility layout by genetic search. *IIE Transactions, 27*(4), 465–472.

Yapicioglu, H., & Smith, A. E. (2012a). Retail space design considering revenue and adjacencies using a racetrack aisle network. *IIE Transactions, 44*(6), 446–458.

Yapicioglu, H., & Smith, A. E. (2012b). A bi-objective model for the retail spatial design problem. *Engineering Optimization, 44*(3), 243–266.

Yu, V. F., Lin, S.-W., & Chou, S-Y. (2010). The museum visitor routing problem. *Applied Mathematics and Computation, 216*(3), 719–729.