Annals of Operations Research (2023) 320:355–377
https://doi.org/10.1007/s10479-022-04907-w

ORIGINAL RESEARCH

A sequential solution heuristic for continuous facility layout problems

Mehmet Burak Şenol[1,2] · Ekrem Alper Murat[3]

Accepted: 2 August 2022 / Published online: 6 October 2022

Abstract
We propose a novel heuristic approach, sequential solution method (SSM), for the efficient solution of Continuous Facility Layout Problems (CFLPs). The proposed SSM approach is compared with exact solution methods as well as Genetic Algorithm (GA) and Simulated Annealing (SA) metaheuristic algorithms. We also improved the metaheuristic approaches based on approximating the facility coordinates with the coordinates of the Center of the Smallest Rectangle (CSR) that covers all facilities in the solution. The proposed SSM approach is a recursive heuristic based on the exact solutions of reduced layout problems. Instead of solving the original CFLP with many variables, SSM first generates subproblems (facility clusters) of smaller sizes using a clustering model and then sequentially solves layout subproblems where non-member facilities locations are constrained. Based on an experimental study, we report that the proposed SSM substantially outperforms exact approaches and meta-heuristic approaches and hence provide an alternative approach for efficiently solving large CFLP instances.

Keywords Heuristics · Genetic Algorithm · Simulated Annealing · Clustering model · Sequential solution method · Facility layout

1 Introduction

Optimal design of facility layouts is a classical problem in the operational research literature with applications in manufacturing and service systems. Facility layout planning (FLP) aims to determine the layout of facilities to optimize one or more operational objectives by minimizing cost objectives (e.g., travel time, material handling costs) and maximizing benefit objectives (e.g., area utilization, closeness rating, satisfaction of decision maker's

✉ Mehmet Burak Şenol
mburaksenol@gazi.edu.tr; mehmetburak.senol@tai.com.tr

Ekrem Alper Murat
amurat@wayne.edu

[1] Department of Industrial Engineering, Gazi University Maltepe, Ankara, Türkiye

[2] Turkish Aerospace Industries (TUSAŞ), Kahramankazan Ankara, Türkiye

[3] Department of Industrial and Systems Engineering, Wayne State University, Detroit, MI, USA

preferences) (Hosseini-Nasab et al., 2018; Pérez-Gosende et al., 2021). Its significance manifest through the impact on the efficiency of production systems and their productivity level. Facility layout plans determine the system's ability to meet the production schedules at an affordable cost and poor plans create process bottlenecks, material flow congestion, underutilization. These plans also determine the degree of flexibility for replanning in the future. There are variations of the classical facility layout planning problem such as multi-objective FLPs (Saraswat et al., 2015), single-row FLPs(Keller & Buscher, 2015), reconfigurable FLPs (Maganha et al., 2019) and multiperiod FLPs (Turanoğlu & Akkaya, 2018). FLPs have been studied in various problem domains such as finding layout of temporary buildings on a construction site (Kumar & Cheng, 2015), designing hospitals (Hahn & Krarup, 2001), and developing microprocessor layouts (Funke, Hougardy, Schneider, 2016). FLP is known to be NP-hard and exact solutions are limited to small sized problems.

Continuous Facility Layout Problem (CFLP) is concerned with determining an efficient layout design for a set of facilities within a given area to minimize the total material handling costs. Considering facility shapes and dimension restrictions, layout design problems can be categorized into equal area and unequal area layout problems (Chae & Regan, 2016; Hosseini-Nasab et al., 2018). We consider the unequal area CFLPs with orthogonal linear flow of material between facilities. The distances are commonly based on the rectilinear distance (Benjaafar, Heragu, Irani, 2002) and less often based on Euclidean distance (Ripon et al., 2013). Accordingly, given the coordinates of the center of the facility as x_i and y_i, then the orthogonal linear distance from a location (a, b) is calculated as $d = |x_i - a| + |y_i - b|$.

An optimal layout is then defined as a non-overlap packing of rectangles with the minimal total material handling cost given p_{ij}, unit distance handling cost between facilities i and j. Let indices i and j to denote one of the N rectangular facilities. The parameters w_i and h_i denote the width and height of the facility i. Let (x_i, y_i) denote the location of the center of facility i on the two-dimensional plane. The CFLP problem studied in this work is formulated as follows:

CFLP Model:

$$MinZ_{CFLP} = \sum_{\text{(i,j)}\in \text{N: i <j}} p_{ij}\left(\left|x_i - x_j\right| + \left|y_i - y_j\right|\right)$$

s.t.

$$\left|x_i - x_j\right| \geq \frac{1}{2}\left(w_i + w_j\right) \text{ or}$$

$$\left|y_i - y_j\right| \geq \frac{1}{2}\left(h_i + h_j\right)\forall i, j \in N \text{ and } i < j$$

$$\text{x}_\text{i},\ \text{y}_\text{i} \geq 0,\ \forall i, j$$

In this study, we propose a novel heuristic method called Sequential Solution Method (SSM). We compared the proposed SSM with popular heuristic methods, i.e., Simulated Annealing (SA), Genetic Algorithms (GA) and our improved versions of simulated annealing (SA-CSR) and genetic algorithms (GA-CSR). Improved versions are based on approximating the facility coordinates to the coordinates of the center of smallest rectangle (CSR) that covers all facilities in the solution. In addition, we report on the comparison of SSM with four exact solution approaches, i.e., MIP1 (Papageorgiou & Rotstein, 1998), MIP2 (Ozyurt & Realff, 1999), MIP3 (Yang & Peters, 1998) and BB (Xie & Sahinidis, 2008) in the literature. Based on an experimental study, we report that SSM approach is promising in terms of solution times and quality.

SSM is based on the iterative decomposition idea where, instead of solving the original problem with many facilities (variables), we sequentially solve smaller subproblems generated via a Clustering Model (CM). SSM approach alternates between clustering of facilities using the CM and selecting the best layout for each cluster based on the objective function value. Accordingly, SSM is a recursive heuristic based on the exact solutions of reduced CFLPs. In SSM, the optimum layout solutions for the reduced CFLP are obtained using exact solution methods. Efficiency of the SSM is tested on the various test problems and results are compared with other heuristics and exact methods. In addition to its computational performance, the proposed approach, due to its decomposition-based solution strategy, allows for dynamically balancing the solution quality and time. Further, the clustering-based solution improvement gives the ability to prioritize the layout planning facility sets based on preferences of the decision maker.

The rest of the paper is organized as follows: In Sect. 2, we provide an overview of the relevant literature and recent studies on layout planning and CFLP formulation. Section 3 describes the CM and introduces the SSM as well as the CSR enhancements to the SA and GA. In Sect. 4, we present the results of a computational study comparing the proposed SSM with exact and heuristic methods using test instances with varying complexity. Section 5 concludes with a summary of findings and suggestions for future research. Results indicate that our novel clustering and sequential solution strategy is more successful in solving larger CFLP instances.

2 Literature

CFLP extends the well-known Quadratic Assignment Problem (QAP), as a facility allocation problem, determines minimum cost assignment of facilities to locations (Koopmans & Beckman, 1957). QAP assumes that all potential locations for each facility are known in advance and is based on the selection of one or more locations from potential locations in order to minimize a certain cost function (Chae & Regan, 2016, 2020). In contrast to QAP, there is no restriction for the choice of locations in the CFLP.

The unequal area CFLPs are NP-hard and global optimal solutions are difficult or impossible to compute in reasonable time. Both exact algorithms and heuristics have been developed to solve CFLP. Montreuil (1991) modeled facility location problem as a Mixed Integer Linear Programming (MIP) model. Binary (1–0) variables were employed to prevent overlapping of rectangular facilities. However, studies with MIP model indicate that it can find the optimal solution up to five facilities and Montreuil's approach (1991) is only applicable for five facilities. Meller et al. (1998) proposed a linear approach to solve the CFLP and developed cutting methods to reduce the solution time. However, their approach could only solve the problem for up to seven facilities. Yang and Peters' (1998) Mixed Integer Linear programming model made it possible to align long edge of each facility with the horizontal and vertical axes of the facility area. Ozyurt and Realff (1999) improved the model of Jeroslow (1989). In their study, absolute objective function value was expressed in a piecewise linear function and they formulated mixed-integer facility programming model excluding Big-M variables (Ozyurt & Realff, 1999). Sherali et al. (2003) significantly improved the solution time by developing symmetry reduction techniques and improving the branch and bound rules. Xie and Sahinidis (2008) proposed a branch-and-bound based algorithm to improve the pruning effectiveness.

Due to the complex nature of the layout problem, even the state-of-the-art exact algorithm is only able to solve instances with nearly ten facilities, while practical instances are often concerned with more facilities. Hence, most research in literature is based on heuristics to generate good solutions (Garc´ıa-Herna´ndez et al., 2015; Garcia-Hernandez et al., 2019; Kang & Chae, 2017; Kim & Chae, 2019; Palomo-Romero et al., 2017; Scalia et al., 2019; Tavakkoli-Moghaddam & Panahi, 2007; Ulutas & Kulturel, 2012). Given the high complexity of the CFLP, numerous metaheuristics have been proposed for layout design (Banerjee et al., 1992; Montreuil, Venkatadri, Ratliff, 1993; Chwif, Barretto, Moscato, 1998; Langevin, Montreuil, Riopel, 1994; Lacksonen, 1997; Kim & Kim, 2000; Montreuil et al., 2004a, 2004b; Hu et al., 2007; Ou-yang and Utamina, 2013; Keller and Bucher, 2015; Palubeckis, 2017). Various meta-heuristic are developed for these problems GA (Tate & Smith, 1995), SA (Tam, 1992), Tabu Search (Scholz, Petrick, Domschke, 2009), Particle Swarm Optimization (Kulturel & Konak, 2011) Ant Colony Optimization (Liu, 2019) and Harmony Search (Kang, Kim, Chae, 2018) /

In addition to the solution methodology for the standard CFLP, the extant literature also presents methods for its extensions. Some studies assumed that there may be uncertainty in the cost matrix and dynamic facility layout design models were developed (Balakrishnan & Cheng, 1998; Benjaafar, Heragu, Irani, 2002; Yang & Peters, 1998). Other variants and approaches in the literature include SA for dynamic layout problem (Baykasoglu & Gindy, 2001), simulation based non-sorting GA (Chen, 2019), a slicing tree based parallel tempering heuristic (Friedrich, Klausnitzer, Lasch 2018), a cuckoo search based algorithm for the closed loop layout design problem (Kang et al., 2018), hybrid ant colony optimization and simulated annealing (Ebrahimi et al., 2020). Several studies addressed the special instance of placing facilities on a linear path (Cravo & Amaral, 2019; Guan & Lin, 2016). A concurrent block layout, path design and aisle design (Klausnitzer & Lasch, 2019), packing algorithms for CFLPs (Crainic et al., 2008; Lodi et al., 2002a, 2002b).Some studies proposed optimizing over the software generated layouts (Abdollahi et al., 2019).

Others studies include layout problems with variations in the facility dimensions and number of objectives (Mendes & Themido, 2004; Hahn et al., 2010; Covas, et al., 2013; Opasanon & Lertsanti, 2013; M'Hallah & Bouziri, 2016; Pa Pagès Bernaus et al., 2019; Li et al., 2019; Amorim-Lopes, 2020). Schneuwly and Widmer (2003) investigated the arrangement of the exhibition spaces in a fair. Braglia et al. (2005) discussed the layout problem in dynamic environments and proposed a measure of the layout robustness. De Giovanni et al. (2013), focused on the gate matrix layouts and optimization of electronic circuits.

3 Methodology

Solution times of CFLPs increase exponentially with the growing number of facilities. To address this complexity, we developed a clustering model (CM) that identifies sets of facilities based on an objective criterion. Given the clusters of facilities, respective layout optimization subproblems for each cluster (reduced CFLPs hereafter) can then be solved in a sequential manner (SSM). We note that this sequential approach is a heuristic method which can generate locally optimal solutions that are suboptimal for the original CFLP problem. In this section, we describe the CM and SSM methods as well as improved versions of SA and GA applied to CFLP, SA-CSR and GA-CSR, based on approximating the facility coordinates to the coordinates of the center of smallest rectangle (CSR).

3.1 Clustering model

CM generates reduced continuous facility layout subproblems with smaller number of facilities. Instead of solving the original problem with many variables, these reduced CFLPs are solved exactly by the subsequent SSM approach. It is known that exact algorithms can achieve the results quickly for problems with few facilities, e.g. up to 5 facilities. Thus, we limit the number of facilities in each cluster with a maximal threshold set size (K) in the CM. The choice of this threshold (K) as well as number of clusters (C) determines the performance of the overall approach. Note that, in the extreme case of $K = N$ and $C = 1$, the proposed approach is equivalent to the original CFLP where there is a single cluster containing all the facilities. As the cluster size decrease, e.g., tightening the K threshold, subsequent solution of reduced CFLPs becomes easier albeit at the expense of increased suboptimality risk. Hence there is a trade-off between the cluster size and optimality of the proposed approach. While the clustering approach reduces the computational complexity, it can result in myopic optimization of facility clusters during the subsequent SSM iterations. Hence, we also allow for commonality of facilities across multiple clusters.

In the CM model, the material handling costs and flow frequencies between facility i and j are employed to assign the most associated facilities to the same clusters. The clusters should include most associated facilities in terms of material handling costs to minimize CFLP in SSM, so the objective of CM model is maximization. In other words, we want to assign the most associated facilities (maximization) to the same clusters in CM and if the most associated facilities are in the same cluster the overall objective function of CFLP could be minimized in SSM. CM is employed only for finding most efficient clusters and SSM is for solving and minimizing the CFLP objective function value. The developed approach leverages the idea of divide-and-conquer where the set of facilities are partitioned into subsets based on their material flow associations (maximization) through CM. Accordingly, CM model maximizes the intra-cluster material flow and handling costs across all clusters in (1). Next, the SSM approach sequentially optimizes the layouts of facility clusters (minimization).The formulation of the clustering model is provided below.

Clustering Model (CM):

$$MaxZ_{CM} = \sum_{c \in C} \sum_{\substack{i, j \in N \\ \mathrm{i} \neq \mathrm{j}}} p_{ij} z_{ic} z_{jc} \tag{1}$$

s.t.

$$\sum_{\mathrm{c} \in \mathrm{C}} z_{ic} = v \quad i \in N \tag{2}$$

$$\sum_{\mathrm{i} \in \mathrm{N}} z_{ic} \geq 1 \quad c \in C \tag{3}$$

$$\sum_{\mathrm{i} \in \mathrm{N}} z_{ic} \leq K \quad c = 1 \ldots C \tag{4}$$

$$\forall z_{ic} \in 0 - 1 i \in N, c \in C \tag{5}$$

In CM, N denotes the set of facilities, where $i, j \in N$ and c and C denotes the index and set of clusters, respective, i.e., $c \in C$. Decisions are binary indicating the cluster assignment of facilities, i.e. $z_{ic} = 1$ *if* facility i is assigned to cluster c, 0 otherwise. Constraint in (2) ensures that a facility is assigned to exactly v clusters where v is the inter-cluster mutuality parameter. Note that $v = 1$ corresponds to the case where all clusters are mutually exclusive. For a non-trivial clustering solution, we have $v < N$. The constraints in (3) and (4) ensure that the number facilities in each cluster are greater than *1* and smaller than *K*.

3.2 Sequential solution method (SSM)

Subsequent to CM, the SSM procedure, a recursive heuristic based on the exact solutions of reduced CFLPs is executed. SSM approach is an iterative procedure where we optimize the reduced CFLP for each cluster at a time while fixing the locations of all other facilities not a member of the cluster. Note that in solving the reduced CFLP of each cluster we consider the interaction between not only the facilities within the cluster but also with those external to the cluster. The salient aspect is that external facilities' locations are kept fixed. In each iteration r of the SSM, given the clusters of facilities C, we solve a reduced version of the CFLP model shown below. We denote the objective value of the reduced CFLP subproblem solved in iteration r for cluster c with $Z_{CFLP}^{c,r}$. Given a set of clusters (C), the reduced CFLP model optimizing the layout of facilities in cluster c is identical to the original formulation with the exception of the last constraint which fixes the layout decisions of all the facilities other than those in the current cluster, i.e.,$\forall i \in C \backslash c$.

Reduced CFLP Model (C, c, r): Cluster c at iteration r with set of clusters C

$$MinZ_{CFLP}^{c,r} = \sum_{\text{(i,j)} \in \text{C: i <j}} p_{ij}\left(\left|x_i - x_j\right| + \left|y_i - y_j\right|\right) \tag{6}$$

s.t.

$$\left|x_i - x_j\right| \geq \frac{1}{2}\left(w_i + w_j\right) \; \forall i, j \in N, \text{ and } i < j \text{ or} \tag{7}$$

$$\left|y_i - y_j\right| \geq \frac{1}{2}\left(h_i + h_j\right) \forall i, j \in N, \text{and } i < j \tag{8}$$

$$(x_i, y_i) = (x_i, y_i)^r \forall i \in C \backslash c, \; \text{x}_\text{i}, \; \text{y}_\text{i} \geq 0, \; \forall i, j \tag{9}$$

We now present the SSM algorithm which is executed at least $C + 1$ and at most R iterations, i.e., $r = 1...R$ in Table 1. SSM is initialized with a clustering solution C^* (set of clusters) obtained by solving the CM model and then sequentially constructed optimal layouts of the clusters. This sequential optimization of the layout gradually appends the clusters to the incumbent set of clusters (C') which is initialized with the first cluster. Next, the SSM solves the reduced CFLPs using the reduced CFLP model with all $c \in C^*$. SSM algorithm terminates if the iteration limit is reached, there is no change in the layout solution or the CPU time limit is reached. Upon termination, SSM algorithm outputs the layout solutions for all facilities and the total objective function across all clusters.

3.3 Heuristics

We improved SA and GA algorithms by approximating the facility coordinates to the coordinates of the center of the smallest rectangle (CSR) that covers all facilities in the current solution. In the next two sections we present the standard SA and GA as well as their improved versions with pseudo codes.

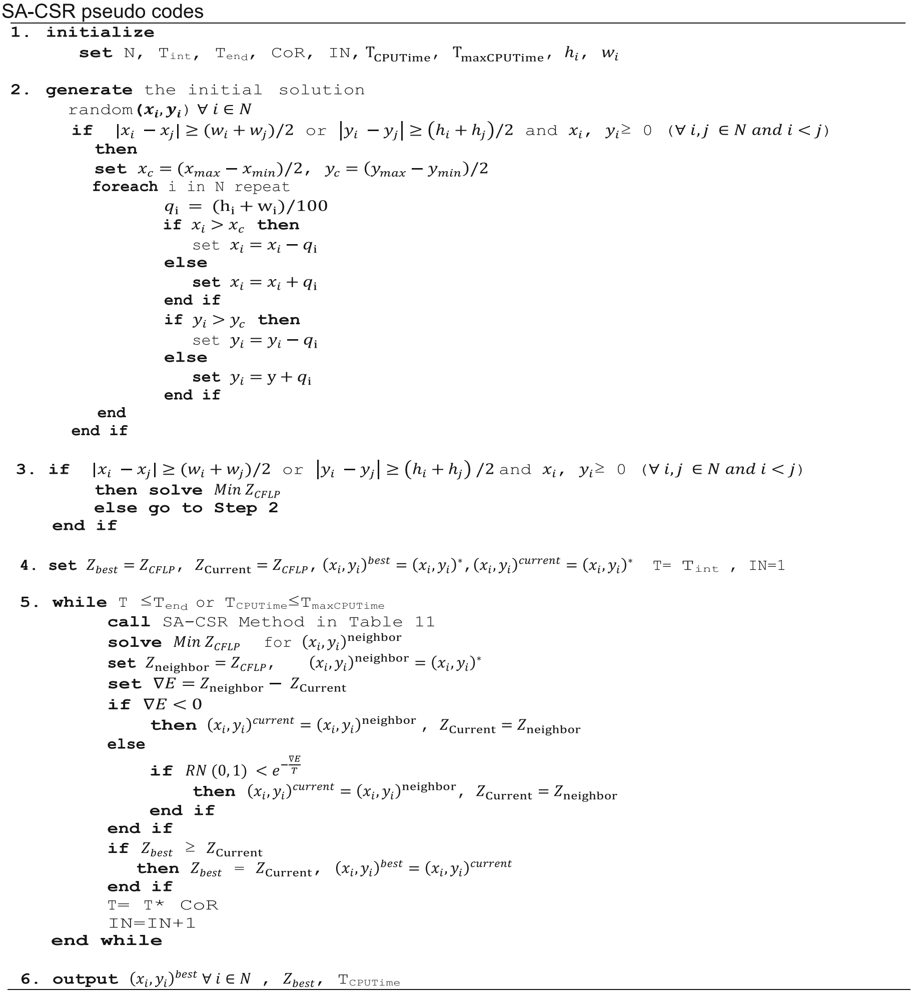
Table 1 SSM algorithm

```
SSM algorithm
initialize
        Set |C|, K, $v$, $T_{CPUTime}$, $T_{maxCPUTime}$
                Solve Cluster Model (CM) and obtain optimal clustering solution $C^*$
        Set $r := 1$ and $C' = \emptyset$
        foreach $c$ in $C^*$ repeat
                        $C' = C' \cup c$
                        Solve Reduced CFLP Model($C'$,$c$,$r$)
                        Set $(x_i,y_i)^r := (x_i,y_i)^* \ \forall i \in c$
                        $r := r + 1$
        end|
while $r \leq R$ or $T_{CPUTime} \leq T_{maxCPUTime}$
        Solve reduced CFLPs in sequence
                foreach $c$ in $C^*$ repeat
                        Solve Reduced CLFP Model($C^*$,$c$,$r$)
                        Set $(x_i,y_i)^r := (x_i,y_i)^* \ \forall i \in c$
                end|
        If $(x_i,y_i)^r \neq (x_i,y_i)^{r-1} \forall i \in c$, then exit with convergence.
        $r = r + 1$
end
Output $(x_i,y_i)^r \ \forall i \in N$ and $Z_{CFLP}^{SSM} = \sum_{c \in C^*} Z_{CFLP}^{c,r}$, $T_{CPUTime}$
```

3.3.1 Simulated annealing algorithms (SA, SA-CSR)

In our experimental study, we employ two different Simulated Annealing algorithms, e.g. the classical SA from the literature and an improved version as SA-CSR. Here we will explain the details and present pseudo codes of our improved version. Chwif et al. (1998), proposes a solution to the facility layout problem using simulated annealing, which includes two procedures working interchangeably. The first procedure makes a pairwise exchange between facilities and the second procedure makes random moves on the planar site at four main directions upwards, downwards, leftwards and rightwards or makes random changes of aspect ratios. We improved solution approach of the Chwif et al. (1998), and present our new approach in Table 2, referred to as Simulated Annealing based on the Center of Smallest Rectangle (SA-CSR). Here, x and y coordinates are generated such that they are within total of the width of facilities for x and height of facilities for y so that an initial solution could be efficiently generated. The main steps of the SA algorithm are described as follows: Step 1 initializes the parameters, i.e., facility number N, initial temperature T_{int} final temperature T_{end}, cooling rate CoR, iteration number IN, CPU running time $T_{CPUTime}$ and maximum CPU running time $T_{maxCPUTime}$. Step 2 checks the feasibility of a random initial solution; if the feasibility condition holds, we find center of facilities (x_c, y_c) in the random solution and move all facility coordinates q step to the center of facilities. The step size $\mathrm{q_i}$ is defined as a percentage of the facility height plus facility width. Instead of percentage we can use a bigger step size etc. *$(h_i + w_i)$/50* or *$(h_i + w_i)$/10*, that may decrease the iteration number but on the other hand it also may lead the infeasibility (e.g. overlapping of the facilities). Furthermore, if we use a smaller step size etc. *$(h_i + w_i)$/1000*, that may increase the iteration number and solution times. The search for the best value of the step size is a separate issue and can be investigated as a future extension of this paper. Step 3 re-checks the feasibility; and if the feasibility condition holds, we solve Z_{CFLP}, otherwise we repeat Step 2. Z_{CFLP} objective function value and facility coordinates are assigned in Step 4 as $\mathrm{Z_{best}} = \mathrm{Z_{CFLP}}$, $\mathrm{Z_{Current}} = \mathrm{Z_{CFLP}}$,$\left(\mathrm{x_i}, \mathrm{y_i}\right)^{\mathrm{best}} = \left(\mathrm{x_i}, \mathrm{y_i}\right)^*$,$\left(\mathrm{x_i}, \mathrm{y_i}\right)^{\mathrm{current}} = \left(\mathrm{x_i}, \mathrm{y_i}\right)^*$. Step 5 executes the main iterations SA. Method in Table 3 is used to create neighbors $(x_i, y_i)^{\mathrm{neighbor}}$ and solved Z_{CFLP} for $(x_i, y_i)^{\mathrm{neighbor}}$. Differences (∇E) between neighbor $\mathrm{Z_{neighbor}}$ and

Table 2 SA-CSR pseudo codes



SA-CSR pseudo codes

```
1. initialize
      set N, T_int, T_end, CoR, IN, T_CPUTime, T_maxCPUTime, $h_i$, $w_i$

2. generate the initial solution
     random$(x_i, y_i)$ $\forall i \in N$
     if $|x_i - x_j| \geq (w_i + w_j)/2$ or $|y_i - y_j| \geq (h_i + h_j)/2$ and $x_i$, $y_i \geq 0$ $(\forall i,j \in N\ and\ i < j)$
        then
        set $x_c = (x_{max} - x_{min})/2$, $y_c = (y_{max} - y_{min})/2$
        foreach i in N repeat
              $q_i = (h_i + w_i)/100$
              if $x_i > x_c$ then
                 set $x_i = x_i - q_i$
              else
                 set $x_i = x_i + q_i$
              end if
              if $y_i > y_c$ then
                 set $y_i = y_i - q_i$
              else
                 set $y_i = y + q_i$
              end if
        end
     end if

3. if $|x_i - x_j| \geq (w_i + w_j)/2$ or $|y_i - y_j| \geq (h_i + h_j)/2$ and $x_i$, $y_i \geq 0$ $(\forall i,j \in N\ and\ i < j)$
        then solve $Min\, Z_{CFLP}$
        else go to Step 2
   end if

4. set $Z_{best} = Z_{CFLP}$, $Z_{Current} = Z_{CFLP}$, $(x_i, y_i)^{best} = (x_i, y_i)^*$, $(x_i, y_i)^{current} = (x_i, y_i)^*$ T= T_int , IN=1

5. while T $\leq$T_end or T_CPUTime$\leq$T_maxCPUTime
        call SA-CSR Method in Table 11
        solve $Min\, Z_{CFLP}$ for $(x_i, y_i)^{neighbor}$
        set $Z_{neighbor} = Z_{CFLP}$, $(x_i, y_i)^{neighbor} = (x_i, y_i)^*$
        set $\nabla E = Z_{neighbor} - Z_{Current}$
        if $\nabla E < 0$
           then $(x_i, y_i)^{current} = (x_i, y_i)^{neighbor}$ , $Z_{Current} = Z_{neighbor}$
        else
           if $RN\,(0,1) < e^{-\frac{\nabla E}{T}}$
              then $(x_i, y_i)^{current} = (x_i, y_i)^{neighbor}$, $Z_{Current} = Z_{neighbor}$
           end if
        end if
        if $Z_{best} \geq Z_{Current}$
          then $Z_{best} = Z_{Current}$, $(x_i, y_i)^{best} = (x_i, y_i)^{current}$
        end if
        T= T* CoR
        IN=IN+1
   end while

6. output $(x_i, y_i)^{best}$ $\forall i \in N$ , $Z_{best}$, T_CPUTime
```

current solution Z$_{\text{Current}}$ objective function values are calculated to compare both solutions. If (∇E) $<$ 0, neighbor solution is accepted as the current solution. If $(\nabla E) \geq 0$, a random number (RN) is produced. If $(\nabla E) \geq 0$ and $RN < e^{-\frac{\nabla E}{T}}$, neighbor solution $(x_i, y_i)^{\text{neighbor}}$ is accepted as the current solution $(x_i, y_i)^{current}$. Otherwise current solution is not changed. If current objective function Z_{Current} is better than local optimum objective function Z_{best}, current solution is accepted as local optimum. Temperature (T) is decreased by multiplying Cooling Rate (CoR). If $T = T_{end}$ or maximum CPU running time is reached, the algorithm terminates, and otherwise we repeat Step 5.

SA-CSR improves the SA implementation in Chwif et al. (1998), by improving p step the initial solution, proposing a new neighboring strategy and improving the random moves on the planar site. Instead of random moves, it employs o step moves to the coordinates of the Center of the Smallest Rectangle (CSR) that covers all facilities layout in the current solution to create a neighbor solution. Neighbors are generated from solutions by approximating the

Table 3 SA-CSR neighborhood generation pseudo codes

SA-CSR neighborhood generation pseudo codes

```
1.foreach $i$ in $N$ repeat
    generate random $x_i$ in $( 0 - \sum_{i=1}^{n} w_i)$
    generate random $y_i$ in $(0 - \sum_{i=1}^{n} h_i)$
  end

2. if $|x_i - x_j| \geq (w_i + w_j)/2$ or $|y_i - y_j| \geq (h_i + h_j)/2$ and $x_i$, $y_i \geq 0$ $(\forall i,j \in N$ and $i < j)$
    then Goto Step 3
    else
     repeat Step 1
   end if

3. set $o = (1 - (\mathrm{IN} - 1)/\mathrm{IN}))$, $mid_x = \frac{\sum_{i=1}^{n} w_i}{2}$, $mid_y = \frac{\sum_{i=1}^{n} h_i}{2}$

4. foreach i in N repeat
    rand (Direction, (V/H)
    case H
          if $y_i > mid_y$ then
          set $y_i = y_i - o$
          else
          set $y_i = y_i + o$
          end if
    case W
          if $x_i > mid_x$ then
          set $x_i = x_i - o$
          else
          set $x_i = x_i + o$
          end if
   end|

5. if $|x_i - x_j| \geq (w_i + w_j)/2$ or $|y_i - y_j| \geq (h_i + h_j)/2$ and $x_i$, $y_i \geq 0$ $(\forall i,j \in N$ and $i < j)$
    then output $(x_i, y_i)^{\text{neighbor}}$
    else
        go to step 1
   end if
```

facility coordinates o step to the coordinates of the center of the smallest rectangle that covers all facilities layout. Improved SA-CSR's neighborhood generation pseudo code is presented in Table 3. Here, we take the central coordinate of factory layout as the mid-point of the layout (mid_x, mid_y) as shown in Fig. 1 given the set of facilities.

The steps of the neighborhood generation used in SA-CSR are provided in Table 3. In Step 1, we randomly generate coordinates of neighbor solution as $\mathrm{x_i}$ and $\mathrm{y_i}$ within 0 and sum of wides $\sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{w_i}$ and heights $\sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{h_i}$ of all facilities (i = 1 to N). Next in Step 2, we check feasibility and proceed to Step 3 where step size o is calculated with respect to iteration number IN. Furthermore, we calculate the mid-point of the layout $(\mathrm{mid_x}, \mathrm{mid_y})$ as shown in Fig. 1 given the set of facilities. Step 4 randomly establishes neighbor solutions' coordinates and Step 5 checks for feasibility.

3.3.2 Genetic algorithms (GA, GA-CSR)

In our experimental study, we employ two different GA, e.g. the classical GA from the literature and an improved version as GA-CSR. Crossover operator in Mak et al. (1998), is based on randomly chosen cutting sections. We propose a new GA strategy in which we combine grids $(n\mathrm{x}n)$ layout in the study of Mak et al. (1998), with cross-over operator in the study of El Baz (2004). We combined and improved both Mak et al. (1998), and El Baz (2004) studies. All parents in population were improved by approximating the facility coordinates o step to the coordinates of Center of Smallest Rectangle (CSR) that covers all facilities in the current population. Here, we take the central coordinate of facility layout as the mid-point of the layout (mid_x, mid_y) as shown in Fig. 1 given the set of facilities. We refer

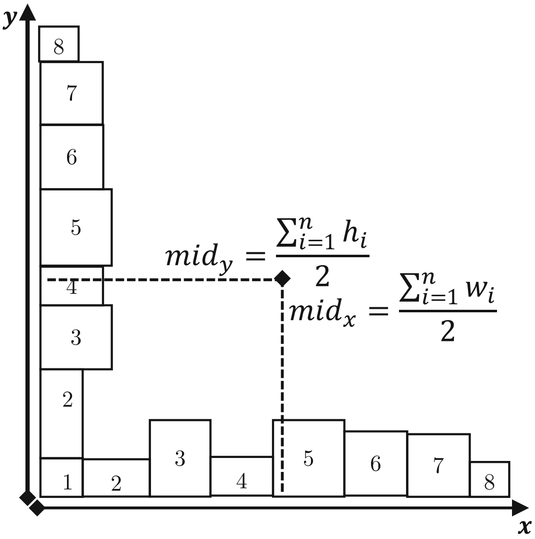

y
x
8
7
6
5
4
3
2
1
2
3
4
5
6
7
8
$mid_y = \frac{\sum_{i=1}^{n} h_i}{2}$
$mid_x = \frac{\sum_{i=1}^{n} w_i}{2}$

Fig. 1 Illustration of the central coordinate (mid-point) of factory layout given eight facilities

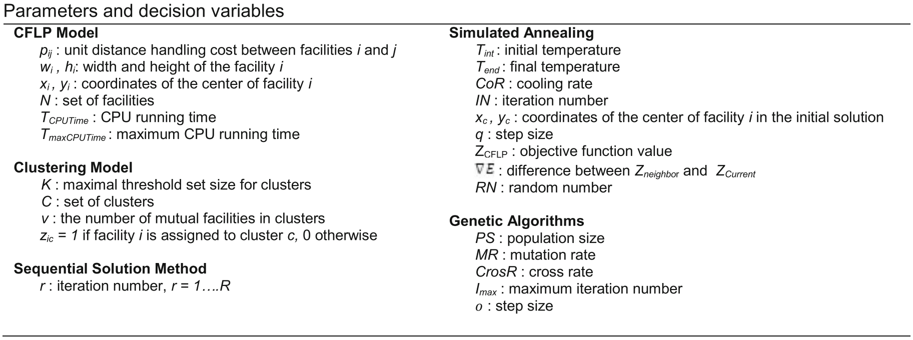
to this approach as Genetic Algorithm based on Center of Smallest Rectangle (GA-CSR) and present its pseudocode in Table 4. Natural selection is performed as explained in Lipowski and Lipowska's (2012) study and probabilities of individuals are calculated accordingly. After generating the offspring, the population is updated by "One Point Crossover Operator individuals" as explained in Kellegöz et al. (2008), research. In conclusion, we present a new GA implementation approach by improving the initial solution and improving the random moves on the planar site.

The GA-CSR algorithm executes in five main steps. Step 1 initializes the algorithm parameters, i.e., facility number N, population size PS, mutation rate MR, cross rate $CrosR$, iteration number IN, maximum iteration number I_{max}, CPU running time $T_{CPUTime}$ and maximum CPU running time $T_{maxCPUTime}$. Random initial solutions (PS) are generated and checked for feasibility in Step 2. Provided feasible, center of facilities (x_c, y_c) coordinates are calculated and all facility coordinates in PS are moved q step towards the center of facilities. The step size $\mathrm{q_i}$ is defined as a percentage of the facility height plus facility width. Instead of percentage we can use a bigger step size etc. $(h_i + w_i)/50$ or $(h_i + w_i)/10$, that may decrease the iteration number but on the other hand it also may lead the infeasibility (e.g. overlapping of the facilities). Furthermore if we use a smaller step size etc. $(h_i + w_i)/1000$, that may increase the iteration number and solution times. The search for the best value of the step size is a separate issue and can be investigated as a future extension of this paper. After checking for the feasibility of these moves, we solve Z_{CFLP} for all individuals in PS and set current solution and factory coordinates $(x_i, y_i)^*$ to following variables $\mathrm{Z_{Current,g}} = \mathrm{Z_{CFLP}}\left(\mathrm{x_i}, \mathrm{y_i}\right)^{\mathrm{current,g}} = \left(\mathrm{x_i}, \mathrm{y_i}\right)^*$, otherwise we repeat Step 2. Minimum objective function value and facility coordinates in PS are assigned in Step 3 as $\mathrm{Z_{dummy}} = \text{minimum of } \mathrm{Z_{Current,g}}$, $\left(\mathrm{x_i}, \mathrm{y_i}\right)^{\mathrm{dummy}} = \text{minimum of } \left(\mathrm{x_i}, \mathrm{y_i}\right)^{\mathrm{current,g}}$, $\mathrm{Z_{best}} = \mathrm{Z_{dummy}}$, $\left(\mathrm{x_i}, \mathrm{y_i}\right)^{\mathrm{best}} = \left(\mathrm{x_i}, \mathrm{y_i}\right)^{\mathrm{dummy}}$. Step 4 executes the main GA iterations. Parent with the best objective function value in the population is overwritten to a randomly selected parent (Natural Selection). Two parents are randomly selected, and then a random number $Rand(0, 1)$ is produced in between 0–1. If $Rand(0, 1) <$ Cross Rate ($CrosR$), two parents' chromosomes are crossed (Crossing). A parent is randomly selected in PS, and then a random

Table 4 GA-CSR pseudo codes

```
GA-CSR pseudo codes
1.initialize
    set $N$, $PS$, $MR$, $CrosR$, $IN$, $I_{max}$, $T_{CPUTime}$ $T_{maxCPUTime}$

2.generate the initial population
    foreach $g$ in $PS$ repeat
       generate Random$(x_i, y_i)$ $\forall i \in N$
       if $|x_i - x_j| \geq (w_i + w_j)/2$ or $|y_i - y_j| \geq (h_i + h_j)/2$ and $x_i$, $y_i \geq 0$ $(\forall i, j \in N \text{ and } i < j)$
       then set $x_c = (x_{max} - x_{min})/2$, $y_c = (y_{max} - y_{min})/2$
          foreach i in N repeat
              $q_i = (h_i + w_i)/100$
          end
          foreach i in N repeat
               if $x_i > x_c$ then
                  set $x_i = x_i - q_i$
               else
                  set $x_i = x_i + q_i$
               end if
               if $y_i > y_c$ then
                  set $y_i = y_i - q_i$
               else
                  set $y_i = y_i + q_i$
               end if
          end
       end if
       if $|x_i - x_j| \geq \frac{1}{2}(w_i + w_j)$ or $|y_i - y_j| \geq \frac{1}{2}(h_i + h_j)$ and $x_i$, $y_i \geq 0$ $(\forall i, j \in N \text{ and } i < j)$
       then solve Min $Z_{CFLP}$
       set $Z_{Current,g} = Z_{CFLP}$ $(x_i, y_i)^{current,g} = (x_i, y_i)^*$
       else go to repeat step 2
       end if
   end

3.set $Z_{dummy}$ = minimum of $Z_{Current,g}$, $(x_i, y_i)^{dummy}$ = minimum of $(x_i, y_i)^{current,g}$, $Z_{best}$=$Z_{dummy}$, $(x_i, y_i)^{best} = (x_i, y_i)^{dummy}$

4.while $IN \leq I_{max}$ or $T_{CPUTime} \leq T_{maxCPUTime}$
      Natural Selection
      Crossover(CrosR)
      Mutation (MR)
      set $o = (1 - (IN - 1)/IN))$
      foreach g in PS
         $(x_i, y_i)^{current} = (x_i, y_i)^{current,g}$
         set $mid_x = \frac{\sum_{i=1}^{n} w_i}{2}$, $mid_y = \frac{\sum_{i=1}^{n} h_i}{2}$
            foreach $i$ in $N$ repeat
                 rand (Direction, (V/H)
                 case H
                          if $y_i > mid_y$ then
                          set $y_i = y_i - o$
                          else
                          set $y_i = y_i + o$
                          end if
                 case W
                          if $x_i > mid_x$ then
                          set $x_i = x_i - o$
                          else
                          set $x_i = x_i + o$
                          end if
            end
            if $|x_i - x_j| \geq \frac{1}{2}(w_i + w_j)$ or $|y_i - y_j| \geq \frac{1}{2}(h_i + h_j)$ and $x_i$, $y_i \geq 0$ $(\forall i, j \in N \text{ and } i < j)$
                 then set $(x_i, y_i)^{current,g} = (x_i, y_i)^*$
            else
               $(x_i, y_i)^{current,g} = (x_i, y_i)^{current}$
            End if
      end

      foreach g in PS
         solve Min $Z_{CFLP}$
         set $Z_{Current,g} = Z_{CFLP}$ $(x_i, y_i)^{current,g} = (x_i, y_i)^*$
      end
      Set $Z_{dummy}$ = minimum of $Z_{Current,g}$        $(x_i, y_i)^{dummy}$ = minimum of $(x_i, y_i)^{current,g}$
      if $Z_{best} \geq Z_{dummy}$
            then
             set $Z_{best} = Z_{dummy}$, $(x_i, y_i)^{best} = (x_i, y_i)^{dummy}$
      end If
   $IN = IN + 1$
   end while

 5.output $(x_i, y_i)^{best}$ $\forall i \in N$ , $Z_{best}$, $T_{CPUTime}$
```

Table 5 Parameters and decision variables



Parameters and decision variables

CFLP Model

p_{ij} : unit distance handling cost between facilities i and j
w_i , h_i: width and height of the facility i
x_i , y_i : coordinates of the center of facility i
N : set of facilities
$T_{CPUTime}$: CPU running time
$T_{maxCPUTime}$: maximum CPU running time

Clustering Model

K : maximal threshold set size for clusters
C : set of clusters
v : the number of mutual facilities in clusters
$z_{ic} = 1$ if facility i is assigned to cluster c, 0 otherwise

Sequential Solution Method

r : iteration number, $r = 1....R$

Simulated Annealing

T_{int} : initial temperature
T_{end} : final temperature
CoR : cooling rate
IN : iteration number
x_c , y_c : coordinates of the center of facility i in the initial solution
q : step size
Z_{CFLP} : objective function value
∇E : difference between $Z_{neighbor}$ and $Z_{Current}$
RN : random number

Genetic Algorithms

PS : population size
MR : mutation rate
$CrosR$: cross rate
I_{max} : maximum iteration number
o : step size

number is produced $Rand(0, 1)$. If $Rand(0, 1) < MR$, parents' chromosomes are mutated. Here, step size o is calculated with respect to iteration number IN. Furthermore, we calculate the mid-point of the layout facilities (mid_x, mid_y) in PS as shown in Fig. 1. Facility coordinates are approached on randomly selected direction (V/H) o step to the coordinates of the center of the smallest rectangle (mid_x, mid_y) that covers all facilities layout. We again make a feasibility control of CFLP constraints. If the feasibility condition holds we replace current facility coordinates with approached coordinates $(x_i, y_i)^{\text{current,g}} = (x_i, y_i)^*$ otherwise we keep the current facility coordinates $(x_i, y_i)^{\text{current,g}} = (x_i, y_i)^{\text{current}}$. We solve Z_{CFLP} for all individuals in PS. We assign minimum objective function value and coordinates in PS to following parameters $\mathrm{Z}_{\text{dummy}} = \text{minimum of}\,\mathrm{Z}_{\text{Current,g}}\ \left(\mathrm{x}_{\mathrm{i}}, \mathrm{y}_{\mathrm{i}}\right)^{\text{dummy}} = \text{minimum of}\,\left(\mathrm{x}_{\mathrm{i}}, \mathrm{y}_{\mathrm{i}}\right)^{\text{current,g}}$. If the best solution is smaller than dummy solution, we set dummy solution as the best solution. $Z_{best} = \mathrm{Z}_{\text{dummy}}$, $(x_i, y_i)^{best} = (x_i, y_i)^{\text{dummy}}$. If $IN = I_{max}$ or $T_{CPUTime} = T_{maxCPUTime}$, iteration stop and repeat Step 4 otherwise.

We summarized the parameters and decision variables in Table 5 for all the models and algorithms.

In order to avoid local optimums in heuristics following issues should be considered. First population-based algorithms find neighbor solution depending on the current solution. In SA differences (∇E) between neighbor $\mathrm{Z}_{\text{neighbor}}$ and current solution $\mathrm{Z}_{\text{Current}}$ objective function values are calculated to compare both solutions. If $(\nabla \mathrm{E}) < 0$, neighbor solution is accepted as the current solution. If $(\nabla E) \geq 0$, a random number (RN) is produced. If $(\nabla E) \geq 0$ and $RN < e^{-\frac{\nabla E}{T}}$, neighbor solution $(x_i, y_i)^{\text{neighbor}}$ is accepted as the current solution $(x_i, y_i)^{current}$. Thus SA algorithm cannot get stuck at local optimums. However, GA might get stuck at a local optimum and we can tune the GA parameters e.g. population size, crossing rate and mutation rate to address this problem. In this study, in order to guarantee fair comparisons of all algorithms (SSM, GA, SA), the parameters of SA and GA were determined such that algorithms would run at least SSM CPU time. In particular, for the SA algorithms, the "cooling rate", "initial temperature" and final temperature" parameters were determined specific to each problem instance so that SA algorithm would run at least SSM CPU time. Similarly, the GA parameters (i.e., population size, crossing rate and mutation rate) were determined specific to each problem instance so that GA algorithm would run at least SSM CPU time. Finding the best values of these parameters to avoid local optimums is a separate issue and can be investigated as a future extension of this paper.

4 Results and discussion

The efficiency of algorithms is tested on test problems with varying sizes (e.g. *PA1, OZ3, T5*, *T6*, *T7*, *T8*, *T9* and *T10* and *YA2)* and compared in terms of solution times and objective function values. We used the same development and execution environment to code and run all the algorithms, Intel Core(TM) i5-3210 2.4 GHz CPU, 8 GB RAM. We used IBM ILOG CPLEX Optimization Studio to solve the CM problems and iterations of SSMs. Genetic and simulated-annealing based methods are programmed in Matlab. The facility sizes of these problem instances are given in Tables 6, 7 and 8, while the material handling cost matrices are presented in Tables 9, 10 and 11, respectively.

Problem instances denoted *T5*, *T6*, *T7*, *T8*, *T9* and *T10* are instances generated from *YA2* by taking the first *m* (5 to 10) facilities and the associated data from Table 8 and Table 11.

Table 6 Facility dimensions for *PA1* problem

	1	2	3	4	5	6	7
w_i	5.22	11.42	7.68	8.48	7.68	2.60	2.40
h_i	5.22	11.42	7.68	8.48	7.68	2.60	2.40

Table 7 Facility dimensions for *OZ3* problem

	1	2	3	4	5	6	7	8	9	10	11
w_i	5	6	6	5	6	5	5	2	3	6	4.5
h_i	3	6	6	3	6	5	5	1	2	6	4.5

Table 8 Facility dimensions for *YA2* problem

	1	2	3	4	5	6	7	8	9	10	11	12
w_i	5	7	6	4	6	5	10	7	6	5	5	6
h_i	4	5	5	4	6	4	7	5	5	5	5	4

Table 9 Material handling costs for *PA1* problem

c_{ij}	1	2	3	4	5	6	7
1	–	346	0	0	416.3	0	0
2	–	–	118	0	0	0	0
3	–	–	–	111	0	0	0
4	–	–	–	–	85.3	0	0
5	–	–	–	–	–	86.3	82.8
6	–	–	–	–	–	–	6.5

Table 10 Material handling costs for *OZ3* problem

c_{ij}	1	2	3	4	5	6	7	8	9	10	11
1	–	1	1	0	0	0	0	0	0	0	0
2	–	–	10	0	0	0	1	0	0	0	0
3	–	–	–	10	10	20	0	0	0	0	5
4	–	–	–	–	0	0	0	0	0	0	0
5	–	–	–	–	–	0	0	0	0	0	0
6	–	–	–	–	–	–	0	0	0	20	0
7	–	–	–	–	–	–	–	1	1	0	5
8	–	–	–	–	–	–	–	–	0	0	0
9	–	–	–	–	–	–	–	–	–	0	0
10	–	–	–	–	–	–	–	–	–	–	5

Table 11 Material handling costs for *YA2* problem

c_{ij}	1	2	3	4	5	6	7	8	9	10	11	12
1	–	18	6	12	2	20	18	10	38	20	26	26
2	–	–	0	0	0	0	0	0	0	0	18	0
3	–	–	–	0	4	4	14	30	16	36	32	38
4	–	–	–	–	8	0	0	0	0	0	0	0
5	–	–	–	–	–	10	2	34	30	6	14	24
6	–	–	–	–	–	–	0	0	0	14	0	0
7	–	–	–	–	–	–	–	36	12	20	4	28
8	–	–	–	–	–	–	–	–	0	0	6	0
9	–	–	–	–	–	–	–	–	–	8	22	12
10	–	–	–	–	–	–	–	–	–	–	0	0
11	–	–	–	–	–	–	–	–	–	–	–	6

For all problem instances, we varied the number of clusters $|C| \in \{2, 3, 4\}$ the maximum number of facilities $3 \leq K \leq 8$, and the number of mutual facilities $1 \leq v \leq 3$. Selected sets of clusters with the highest objective function values are presented in Table 12.

Next, we solved the SSM for each clustering solution in Table 12. The SSM is a recursive heuristic based on the exact solutions of reduced CFLPs, thus R must be chosen at least as the number of clusters (C) for a feasible layout solution. On the other hand, SSM should be executed at least "C + 1" iterations to check whether this feasible solution is the best solution (if there is no change in the layout solution) or not. Maximum number of cluster for a problem set is C = 4 in Table 12, so R should be taken at least 5 which is C + 1 = 4 + 1. We set the maximum number of iterations at 10, i.e., R = 10, which is 2 multiplied by 5 as an upper bound to make sure that the SSM algorithm runs smoothly and sufficiently enough. Note that in solving the reduced CFLP of each cluster we consider the interaction between not only the facilities within the cluster but also with those external to the cluster. SSM algorithm terminates if the iteration limit R is reached or there is no change in the layout

Table 12 Cluster sets identified through CM for each problem instance

Problem	Subproblems (clusters)	Mutual facilities
$T5_1$	{1,2,4} and {3,5}	None
$T5_2$	{1,2,4,5} and {3}	None
$T5_3$	{1,2,4,5} and {1,2,3,4}	{1,2,4}
$T6_1$	{1,2,6} and {3,4,5}	None
$T6_2$	{1,2,4,5,6} and {3}	None
$T6_3$	{1,2,3,6} and {1,4,5,6}	{1,6}
$T7_1$	{1,2,3,7} and {4,5,6}	None
$T7_2$	{1,2,3,6,7} and {4,5}	None
$T7_3$	{1,3,4,5,6,7} and {2}	None
$T7_4$	{1,2,3,7} and {1,4,5,6}	{1}
$T8_1$	{3,5,7,8} and {1,2,4,6}	None
$T8_2$	{1,3,5,6,7,8} and {2,4}	None
$T9_1$	{3,5,7,8,9} and {2,4} and {1,6}	None
$T10_1$	{1,3,5,6,7,8,9,10} and {2} and {4}	None
YA2	{3,7,10,12} and {4,5,6,8} and {1,2,9,11}	None
$OZ3_1$	{3,4,5,6,10} and {7,9,11} and {1,2,8}	None
$OZ3_2$	{6,7,10,11} and {3,6,10,11} and {2,3,4,5} and {1,7,8,9}	{6,10,11}, {3}, {7}
$OZ3_3$	{3,6,10,11} and {7,8,9,11} and {1,2,3,4} and {3,4,5,6}	{11}, {3}, {3,6}, {3,4}
$PA1_1$	{1,2,5,6} and {3,4,7}	None
$PA1_2$	{1,5,6,7} and {1,2,3,4}	{1}

solution. Results and objective function values at SSM iterations are presented in Table 13. It is observed that at most in 5 steps, the SSM algorithm is terminated (see Table 13).

Order of the clusters in the initialization step of SSM was chosen according to the CM objective function value of clusters from highest to smallest. The clusters in Table 12 are ranked with respect to CM objective function values from highest to smallest. The cluster with the highest CM objective function value is the first cluster executed in SSM, because it includes the most associated facilities in terms of material handling costs and the rest of clusters are appended the SSM algorithm accordingly. This is because we assign the most associated facilities to the same clusters in CM (maximization) and if the most associated facilities are in the same cluster, they should be deployed first to minimize the overall objective function of CFLP by SSM. We illustrate the SSM iterations and solutions obtained for $T9_1$ and $OZ3_1$ in Figs. 2 and 3.

4.1 Comparison of SSM with exact solution methods (MIP 1, MIP 2, MIP 3, and BB)

The comparison of SSM results with the existing approaches MIP1 (Papageorgiou & Rotstein, 1998), MIP2 (Ozyurt & Realff, 1999), MIP3 (Yang & Peters, 1998) and BB (Xie & Sahinidis, 2008) are presented in Table 14. Reported solution times are the total time required for generating clusters using CM and solving sub-problem sets using SSM. In all experiments, the SSM method's convergence is established if the number of iterations reaches the maximum

Table 13 SSM iteration objective function values (Z)

Problem	Step 1	Step 2	Step 3	Step 4	Step 5
$T5_{1}$	129	266	266 **Stop**		
$T5_{2}$	187	279	256	256 **Stop**	
$T5_{3}$	187	256	256 **Stop**		
$T6_{1}$	339	448	461	461 **Stop**	
$T6_{2}$	161	466	466 **Stop**		
$T6_{3}$	230	466	466 **Stop**		
$T7_{1}$	688	864	833	833 **Stop**	
$T7_{2}$	499	846	810	794	794 **Stop**
$T7_{3}$	333	846	846 **Stop**		
$T8_{1}$	1439	1793	1788	1710	1710 **Stop**
$T8_{2}$	828	1742	1742 **Stop**		
$T9_{1}$	520	1375	2594	2594 **Stop**	
$T10_{1}$	3149	3744	3756	3617	3617 **Stop**
YA2	945	2561	6730	6319	6319 **Stop**
$OZ3_{1}$	325	454.8	520.3	520.3 **Stop**	
$OZ3_{2}$	160	311.3	526	524.8	524.8 **Stop**
$OZ3_{3}$	275	324.3	514.8	627.8	627.8 **Stop**
$PA1_{1}$	6007.4	9978.5	9978.5	9978.5 **Stop**	
$PA1_{2}$	3612.2	9948.0	9948 **Stop**		

Bold indicates the best objective function values for the relevant problems

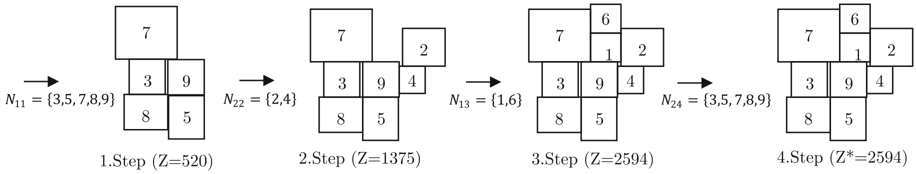


Fig. 2 Sequential solution of $T9_{1}$ problem, variables added to problem in each iteration and objective function values (Z)

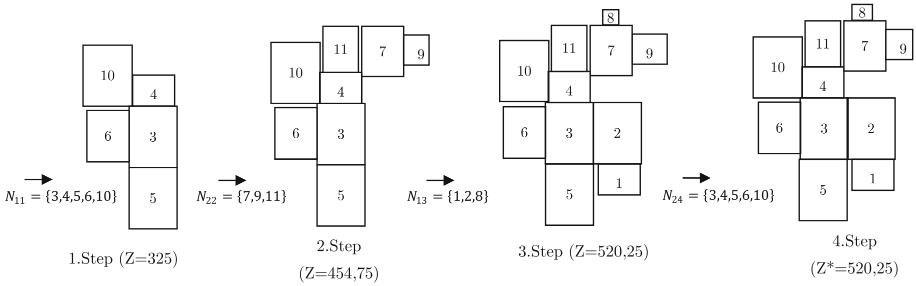


Fig. 3 Sequential solution of $OZ3_{1}$ problem, variables added to problem in each iteration and objective function values (Z)

Table 14 Comparison of the results of SSM with MIP 1, MIP 2, MIP 3, and BB

Problem	MIP 1		MIP 2		MIP 3		BB		SSM	
	Obj	**CPU(s)**	**Obj**	**CPU(s)**	**Obj**	**CPU(s)**	**Obj**	**CPU(s)**	**Obj**	**CPU(s)**
T5	**256**	0.3	**256**	1.2	**256**	0.6	**256**	**0.1**	**256**	1.3
T6	**461**	7.39	**461**	52.9	**461**	56.2	**461**	**0.7**	**461**	4.7
T7	**794**	551.5	**794**	> 6000	**794**	5829	**794**	14.6	**794**	**14.2**
T8	**1710**	3558	**1710**	> 6000	**1710**	> 6000	**1710**	56.8	**1710**	**44.3**
T9	2596	> 6000	2596	> 6000	2596	> 6000	**2594**	637.22	**2594**	**45.6**
T10	**3617**	> 6000	3707	> 6000	3634	> 6000	**3617**	> 6000	**3617**	**174**
PA1	**9948**	7.4	**9948**	> 6000	**9948**	16.4	**9948**	3.8	**9948**	**2.1**
OZ3	**520.3**	5179	528	> 6000	523	> 6000	**520.3**	565	**520.3**	**169.9**
YA2	6492	> 6000	6499	> 6000	6627	> 6000	**6319**	> 6000	**6319**	**100.3**

Bold indicates the best objective function values for the relevant problems number of iterations (R = 10), there is no change in the layout solution objective function value between consecutive steps or the CPU time limit (6000 s) is reached. The proposed approach performs better, in terms of both objective value and time, than MIP1, MIP2, and MIP3 in all problem instances but T5 and the difference is negligible given the problem complexity. In comparison, the BB method performs better compared to other methods from the existing literature but the proposed approach outperforms BB in medium and large problem instances. As expected for the solution of CFLP with many variables, SSM solution times are much better than all of the other methods (MIP1, MIP2, MIP3 and BB). In particular, for T10 and YA2 problems, while the proposed approach can find the optimal solution under three minutes, the BB method takes in excess of 100 min.

4.2 Comparison of SSM with metaheuristic algorithms (SA, SA-CSR, GA, and GA-CSR)

Initial solutions are generated in a three-step iteration for metaheuristics. In Step 1, new facility coordinates (x_{new}, y_{new}) are generated such that they are within total of the width of facilities for x and height of facilities for y. We check the feasibility condition in Step 2, whether the boundaries of the new facility overlap with all the facility boundaries in the solution set. If the new facility does not overlap with the other facilities in the solution set, it was included to solution set otherwise we repeat step 1. In Step 3, If iteration number equal to number of facilities, iteration stops, otherwise we repeat Step 1. We compared the initial solution times with total solution times of metaheuristic. Considering simulated annealing, the initial solution time is minimum %3,5 (T6), maximum %11,3 (T10) and average %6,4 of SA total solution time. Considering genetic algorithms, the initial solution time is minimum % 7,7 (T10), maximum %17,9 (YA2) and average %13,5 of GA total solution time. If the number of facilities is high it might be more difficult to generate a feasible initial solution, however this is not the case for our problem sets.

The test problems were solved with the four heuristics SA, SA-CSR, GA, and GA-CSR and compared with the SSM results in Table 15. In order to guarantee fair comparisons of all algorithms, the parameters of SA and GA were determined such that algorithms would run at least SSM CPU time. In particular, for the SA algorithms, the "cooling rate", "initial temperature" and final temperature" parameters were determined specific to each problem instance so that SA algorithm would run at least SSM CPU time. Similarly, the GA parameters (i.e., population size, crossing rate and mutation rate) were determined specific to each problem instance so that GA algorithm would run at least SSM CPU time.

Comparing the results of CSR based improvements; SA-CSR and GA-CSR perform better than SA and GA methods in terms of objective function values and solution times. However, SSM approach is the best among these five heuristics in terms of solution times and objective function values. The objective function values of SA and GA worsen as the number of facilities increase from 5 to 12 (e.g. *T5, T6, T7, T8, T9, T10, PA1, OZ3* and *YA2*). Although CSR based enhancement improved the performance of SA and GA, SSM yields overall better results than any of the metaheuristic algorithms.

5 Conclusions

In this study, we proposed a sequential heuristic approach (SSM) for continuous facility layout design problems and compared its performance with exact methods as well as several metaheuristic approaches. The developed approach leverages the idea of divide-and-conquer

Table 15 Comparison of the results of SSM with SA, SA-CSR, GA, and GA-CSR

Problem	SA		GA		SA-CSR		GA-CSR		SSM	
	Obj	**CPU(s)**	**Obj**	**CPU(s)**	**Obj**	**CPU(s)**	**Obj**.	**CPU(s)**	**Obj**.	**CPU(s)**
T5	982	365.5	628	218.7	328	181.6	399	130.2	**256**	**1.3**
T6	1226	826.8	1912	240.3	878	130.2	810	145.7	**461**	**4.7**
T7	3334	694.9	3942	267	1648	139.4	1427	135.7	**794**	**14.2**
T8	6466	693.4	8270	260	3218	108.5	3062	151.6	**1710**	**44.3**
T9	12,064	680.6	12,006	287.9	4358	93.4	5055	165.9	**2594**	**45.6**
T10	16,156	473.4	16,546	700.4	6506	200.3	6215	217.9	**3617**	**174**
PA1	45,476	677.5	44,153	262.9	14,938	232.2	17,059	1646	**9948**	**2.1**
OZ3	3265	1217	2908	319.2	1348	175.8	971.8	1082	**520.3**	**169.9**
YA2	34,690	560.5	32,142	307.2	13,616	101.1	10,847	189.8	**6319**	**100.3**

Bold indicates the best objective function values for the relevant problems where the set of facilities are partitioned into subsets based on their material flow associations through a clustering model. Next, the SSM approach sequentially optimizes the layouts of facility clusters (one cluster at a time) while fixing the locations of other facilities not a member of the cluster. The SSM approach trades off the optimality with the computational complexity through the clustering model parameters of number of clusters, maximum number of facilities in any cluster and commonality of facilities across clusters. The SSM approach converges to a solution whenever a threshold iteration limit is reached, or the layout solutions do not change. While not utilized in the experimental study, the method's clustering step can be flexibly incorporated into the SSM's iterations by allowing the procedure to update the cluster assignments between iterations. Further, the clustering step could be utilized to capture the decision maker's stated or unstated preferences regarding facilities in terms of prioritization of relationships between facilities (i.e., those facilities with higher priority of relationship clustered together).

In comparison with the exact approaches, SSM method performed significantly better in terms computational time performance as well as similar or better objective performance. For the metaheuristics, we improved the application of standard SA and GA approaches by approximating the facility coordinates to the coordinates of the center of the smallest rectangle (CSR) that covers all facilities in the current solution. While this enhancement improved the performance of SA and GA, the SSM outperformed the standard and enhanced metaheuristics in both objective and computational time performances.

Given the promising performance of SSM, it can be effectively employed to solve real life facility layout problems in order to optimize layouts and decrease material handling costs. Future extensions of this approach can consider updating the clustering decisions during the sequential cluster layout optimizations. Given that SSM can converge to a local optimal solution, one strategy to employ in changing the clustering decisions could be to revise the cluster assignments once a local optimal solution is reached. This could potentially allow the SSM escape local optimality for large problem instances.

Acknowledgements We greatly appreciate the editor and anonymous reviewer's constructive feedback and suggestions which improved the papers presentation clarity and content. This research did not receive any specific grant from funding agencies in the public, commercial or not-for-profit sectors.

References

Abdollahi, P., Aslam, M., & Yazdi, A. A. (2019). Choosing the best facility layout using the combinatorial method of Gray relation analysis and nonlinear programming. *Journal of Statistics and Management Systems, 22*(6), 1143–1161.

Amorim Lopes, M., Guimarães, L., Alves, J., & Almada Lobo, B. (2020). Improving picking performance at a large retailer warehouse by combining probabilistic simulation, optimization, and discrete-event simulation. *International Transactions in Operational Research, 28*(2), 687–715.

Balakrishnan, J., & Cheng, C. H. (1998). Dynamic layout algorithms: A state-of the art survey. *Omega-International Journal of Management Science, 26*(4), 507–521.

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

Baykasoglu, A., & Gindy, N. Z. (2001). A simulated annealing algorithm for dynamic layout problem. *Computers & Operations Research, 28*(14), 1403–1426.

Benjaafar, S., Heragu, S. S., & Irani, S. A. (2002). Next generation factory layouts: Research challenges and recent progress. *Interfaces, 32*(6), 58–76.

Braglia, M., Zanoni, S., & Zavanella, L. (2005). Layout design in dynamic environments: Analytical issues. *International Transactions in Operational Research, 12*(1), 1–19.

Chae, J., & Regan, A. C. (2016). Layout design problems with heterogeneous area constraints. *Computers & Industrial Engineering, 102*, 198–207.

Chae, J., & Regan, A. C. (2020). A mixed integer programming model for a double row layout problem. *Computers & Industrial Engineering, 140*, 106244.

Chen, X. (2019). An approximate nondominated sorting genetic algorithm to integrate optimization of production scheduling and accurate maintenance based on reliability intervals. *Journal of Manufacturing Systems*, *54*(31), 227–241.

Chwif, L., Barretto, M. R. P., & Moscato, L. A. (1998). A solution to the facility layout problem using simulated annealing. *Computers in Industry, 36*(1), 125–132.

Covas, M. T., Silva, C. A., & Dias, L. C. (2013). Multicriteria decision analysis for sustainable data centers location. *International Transactions in Operational Research, 20*(3), 269–299.

Crainic, T. G., Perboli, G., & Tadei, R. (2008). Extreme point based heuristics for three dimensional bin packing. *Journal on Computing, 20*, 368–384.

Cravo, G. L., & Amaral, A. R. S. (2019). A grasp algorithm for solving large-scale single row facility layout problem. *Computers & Operations Research, 106*, 49–61.

De Giovanni, L., Massi, G., Pezzella, F., Pfetsch, M. E., Rinaldi, G., & Ventura, P. (2013). A heuristic and an exact method for the gate matrix connection cost minimization problem. *International Transactions in Operational Research, 20*(5), 627–643.

Ebrahimi, A., Jeon, H. W., Lee, S., & Wang, C. (2020). Minimizing total energy cost and tardiness penalty for a scheduling-layout problem in a flexible job shop system: A comparison of four metaheuristic algorithms. *Computers & Industrial Engineering, 141*, 106295.

Friedrich, C., Klausnitzer, A., & Lasch, R. (2018). Integrated slicing tree approach for solving the facility layout problem with input and output locations based on contour distance. *European Journal of Operational Research, 270*(3), 837–851.

Funke, J., Hougardy, S., & Schneider, J. (2016). An exact algorithm for wirelength optimal placements in VLSI design. *Integration, 52*, 355–366.

García-Hernández, L., Palomo-Romero, J. M., Salas-Morera, L., Arauzo-Azofra, A., & Pierreval, H. (2015). A novel hybrid evolutionary approach for capturing decision maker knowledge into the unequal area facility layout problem. *Expert Systems with Applications, 42*(10), 4697–4708.

Garcia-Hernandez, L., Pierreval, H., Salas-Moreraa, L., & Arauzo-Azofra, A. (2019). Handling qualitative aspects in Unequal Area Facility Layout Problem: An Interactive Genetic Algorithm. *Applied Soft Computing, 13*(4), 1718–1727.

Guan, J., & Lin, G. (2016). Hybridizing variable neighborhood search with ant colony optimization for solving the single row facility layout problem. *European Journal of Operational Research, 248*(3), 899–909.

Hahn, P., & Krarup, J. (2001). A hospital facility layout problem finally solved. *Journal of Intelligent Manufacturing, 12*(5), 487–496.

Hahn, P. M., Zhu, Y. R., Guignard, M., & Smith., J.M. (2010). Exact solution of emerging quadratic assignment problems. *International Transactions in Operational Research, 17*(5), 525–552.

Hosseini-Nasab, H., Fereidouni, S., Ghomi, S. M. T. F., & Fakhrzad, M. B. (2018). Classification of facility layout problems: A review study. *The International Journal of Advanced Manufacturing Technology, 94*(1–4), 957–977.

Hu, G., Chen, Y., Zhou, Z., & Fang, H. (2007). A genetic algorithm for the inter-cell layout and material handling system design. *The International Journal of Advanced Manufacturing Technology, 34*(11), 1153–1163.

Jeroslow, R. G. (1989). *Logic based decision support: Mixed integer model formulation*. North-Holland.

Kang, S., & Chae, J. (2017). Harmony search for the layout design of an unequal area facility. *Expert Systems with Applications, 79*, 269–281.

Kang, S., Kim, M., & Chae, J. (2018). A closed loop based facility layout design using a cuckoo search algorithm. *Expert Systems with Applications, 93*, 322–335.

Kellegöz, T., Toklu, B., & Wilson, J. (2008). Comparing efficiencies of genetic crossover operators for one machine total weighted tardiness problem. *Applied Mathematics and Computation, 199*, 590–598.

Keller, B., & Buscher, U. (2015). Single row layout models. *European Journal of Operational Research, 8*, 1–12.

Kim, J. G., & Kim, Y. D. (2000). Layout planning for facilities with fixed shapes and input and output points. *International Journal of Production Research, 38*(18), 4635–4653.

Kim, M., & Chae, J. (2019). Monarch butterfly optimization for facility layout design based on a single loop material handling path. *Mathematics, 7*(2), 154.

Klausnitzer, A., & Lasch, R. (2019). Optimal facility layout and material handling network design. *Computers & Operations Research, 103*, 237–251.

Koopmans, T. C., & Beckman, M. (1957). Assignment problems and the location of economic activities. *Econometrica, 25*, 53–76.

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

Kumar, S., & Cheng, J. (2015). A BIM-based automated site layout planning framework for congested construction sites. *Automation in Construction, 59*, 24–37.

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

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

Li, X., Wang, Z. X., Chan, F. T. S., & Chung, S. H. (2019). A genetic algorithm for optimizing space utilization in aircraft hangar shop. *International Transactions in Operational Research, 26*(5), 1655–1675.

Lipowski, A., & Lipowska, D. (2012). Roulette-wheel selection via stochastic acceptance. *Physica, 391*, 2193–2196.

Liu, J. (2019). Applying multi-objective ant colony optimization algorithm for solving the unequal area facility layout problems. *Applied Soft Computing, 74*, 167–189.

Lodi, A., Martello, S., & Monaco, M. (2002a). Two dimensional packing problems: A survey. *European Journal of Operational Research, 141*, 241–252.

Lodi, A., Martello, S., & Vigo, D. (2002b). Recent advances on two dimensional bin packing problems. *Discrete Applied Mathematics, 123*, 379–396.

Maganha, I., Silva, C., & Ferreira, L. M. D. (2019). The layout design in reconfigurable manufacturing systems: A literature review. *The International Journal of Advanced Manufacturing Technology, 105*(1), 683–700.

Mak, K. L., Wong, Y. S., & Chan, F. T. S. (1998). A genetic algorithm for facility layout problems. *Computer Integrated Manufacturing Systems, 11*(1), 113–127.

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

Mendes, A. B., & Themido, I. H. (2004). Multi-outlet retail site location assessment. *International Transactions in Operational Research, 11*(1), 1–18.

M'Hallah, R., & Bouziri, A. (2016). Heuristics for the combined cut order planning two dimensional layout problem in the apparel industry. *International Transactions in Operational Research, 23*(1–2), 321–353.

Montreuil, B., Ouazzani, N., Brotherton, E., Nourelfath, M., (2004b). Location matrix based design methodology for the facility layout problem including aisles and door locations. In *Proceedings of the 8th international material handling research colloquium,* (pp. 155–182). Graz, Austria.

Montreuil, B., Ouazzani, N., Brotherton, E., Nourelfath, M., (2004a). Coupling zone-based layout optimization, and colony system and domain knowledge. In *Proceedings of the 8th international material handling research colloquium*, (pp. 303–334). Graz, Austria.

Montreuil, B. (1991). A modelling framework for integrating layout design and flow network design. *Material handling'90* (pp. 95–115). Berlin: Springer.

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

Opasanon, S., & Lertsanti, P. (2013). Impact analysis of logistics facility relocation using the analytic hierarchy process (AHP). *International Transactions in Operational Research, 20*(3), 325–339.

Ou-Yang, C., & Utamima, A. (2013). Hybrid estimation of distribution algorithm for solving single row facility layout problem. *Computers & Industrial Engineering, 66*(1), 95–103.

Ozyurt, D. B., & Realff, M. J. (1999). Geographic and process information for chemical plant layout problems. *AIChE Journal, 45*(10), 2161–2174.

Pagès Bernaus, A., Ramalhinho, H., Juan, A. A., & Calvet, L. (2019). Designing ecommerce supply chains: A stochastic facility–location approach. *International Transactions in Operational Research, 26*(2), 507–528.

Palomo-Romero, M. J., Salas-Morera, L., & Garc´ıa-Herna´ndez, L. (2017). An island model genetic algorithm for unequal area facility layout problems. *Expert Systems with Applications, 68*, 151–162.

Palubeckis, G. (2017). Single row facility layout using multi-start simulated annealing. *Computers & Industrial Engineering, 103*, 1–16.

Papageorgiou, L. G., & Rotstein, G. E. (1998). Continuous-domain mathematical models for optimal process plant layout. *Industrial & Engineering Chemistry Research, 37*, 3631–3639.

Pérez-Gosende, P., Mula, J., & Díaz-Madroñero, M. (2021). Facility layout planning. An extended literature review. *International Journal of Production Research, 59*(12), 3777–3816.

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*, 1–12.

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

Scalia, G., Micale, R., Giallanza, A., & Marannano, G. (2019). Firefly algorithm based upon slicing structure encoding for unequal facility layout problem. *International Journal of Industrial Engineering Computations, 10*(3), 349–360.

Schneuwly, P., & Widmer, M. (2003). Layout modeling and construction procedure for the arrangement of exhibition spaces in a fair. *International Transactions in Operational Research, 10*(4), 311–338.

Scholz, D., Petrick, A., & Domschke, W. (2009). 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. P., & Meller, R. D. (2003). Enhanced model formulations for optimal facility layout. *Operations Research, 51*(4), 692–644.

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

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

Tavakkoli-Moghaddam R., Panahi, H., (2007). Solving a new mathematical model of a closed-loop layout problem with unequal-sized facilities by a genetic algorithm. In *Proceedings of the Industrial Engineering and Engineering Management, (*pp. 327–331). Singapore.

Turanoğlu, B., & Akkaya, G. (2018). A new hybrid heuristic algorithm based on bacterial foraging optimization for the dynamic facility layout problem. *Expert Systems with Applications, 98*, 93–104.

Ulutas, B. H., & Kulturel, K. S. (2012). An artificial immune system based algorithm to solve unequal area facility layout problem. *Expert Systems with Applications, 39*(5), 5384–5395.

Xie, W., & Sahinidis, N. V. (2008). A branch-and-bound algorithm for the continuous facility layout problem. *Computers & Chemical Engineering, 32*, 1016–1028.

Yang, T., & Peters, B. (1998). Flexible machine layout design for dynamic and uncertain production environments. *European Journal of Operational Research, 108*, 49–64.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.