Applied Soft Computing Journal 89 (2020) 106052

Contents lists available at ScienceDirect

Applied Soft Computing Journal

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

Configuration space evolutionary algorithm for multi-objective unequal-area facility layout problems with flexible bays

Jingfa Liu [a,b,*], Siyu Liu [c], Zhaoxia Liu [d], Bi Li [a]

[a] School of Information Science and Technology, Guangdong University of Foreign Studies, Guangzhou 510006, China
[b] Guangzhou Key Laboratory of Multilingual Intelligent Processing, Guangdong University of Foreign Studies, Guangzhou 510006, China
[c] School of Computer & Software, Nanjing University of Information Science & Technology, Nanjing 210044, China
[d] Network and Information Center, Guangdong University of Foreign Studies, Guangzhou 510006, China

ARTICLE INFO

Article history:
Received 30 April 2019
Received in revised form 26 December 2019
Accepted 27 December 2019
Available online 13 January 2020

Keywords:
Multi-objective optimization
Configuration space evolutionary algorithm
Flexible bay structure
Pareto-optimal solutions
Facility layout problem

ABSTRACT

Facility layout problem (FLP) which deals with the layout of facilities within a given plant floor is an NP-hard combinatorial optimization problem. This paper studies multi-objective unequal-area facility layout problems (UA-FLPs) with the flexible bay structure (FBS), whose objectives refer to the material handling cost, the closeness relationship, the distance requirement and the aspect ratio of facilities. In recent years, some successes have been achieved by multi-objective evolutionary algorithms (MOEAs) for solving various kinds of optimization problems with multiple conflicting objectives. However, traditional MOEAs face a great challenge in the convergence and diversity of solutions for UA-FLPs. In this paper, a novel MOEA called the configuration space evolutionary (CSE) algorithm is developed to solve the UA-FLPs with multiple objectives. We consider a mating pool called a configuration (solution) bank in the CSE, and use evolutionary operations (selection, novel crossover and mutation) to produce new configurations of the pool. By introducing a measure of the radius d_{space} of the configuration bank, whose value is gradually reduced to narrow the search space, the convergence of solutions in the CSE is controlled. A method of the nearest and farthest candidate solution based on objective function normalization is combined with the fast non-dominated sorting to choose the Pareto-optimal solutions, which is good for the algorithm to keep diversity of the obtained solutions. The main contributions of this study lie in the use of a mechanism of evolution of population in the algorithm based on a configuration bank, and the use of a selection strategy based on the nearest and farthest candidate solution method, in order to improve the convergence and diversity of solutions. Experiments are carried out on eight different representative instances and performance metrics from the literature. Compared with the existing MOEAs, the CSE is able to find the better results and show better performance. The numerical experiments confirm the effectiveness of the CSE for solving multi-objective UA-FLPs.

© 2020 Elsevier B.V. All rights reserved.

1. Introduction

With the growing intense competition in the international market and the rapid development of social economy, an efficient production system is very important to enhance the competitiveness of the enterprises. In order to reduce the production cost and enable quick changes to react to market demands, enterprises need to make full use of existing production workshop resources. Research shows that about 20% ∼ 50% of the processing fees of the enterprise's production system are from the material handling. An excellent layout of facilities within the workshop can enhance the efficiency of material handling between facilities, and reduce material handling costs at least 10% ∼ 30% [1]. Facility layout optimization design is one of the most important and complex problems in modern manufacturing. In general, the facility layout problem (FLP) is the problem of determining of the location of facilities within a given plant floor so as to satisfy some given objectives.

Facility location depends on its center coordinates which are continuous numerical variables. In theory, in the condition of known constraints, the FLP has the optimal solution. But continuous center coordinates of facilities makes facility layout optimization become one of the most complex combinatorial optimization problems. In fact, the FLP belongs to the NP-hard (non-deterministic polynomial-time hard) problem [2] which is difficult to find the optimal solution in reality. Many scholars have widely researched it. General approaches to the FLP mainly address the relative arrangement of equal-area facilities on a

* Corresponding author at: School of Information Science and Technology, Guangdong University of Foreign Studies, Guangzhou 510006, China.
E-mail address: jfliu@gdufs.edu.cn (J. Liu).

https://doi.org/10.1016/j.asoc.2019.106052
1568-4946/© 2020 Elsevier B.V. All rights reserved.

plant floor, which is called equal-area facility layout problem (EA-FLP). The main shortcoming of this method is that equal-area facility is a very poor assumption because few locations can accommodate some of the facilities. In fact, in reality, unequal-area facilities should be considered. This paper researches the unequal-area facility layout problem (UA-FLP). At present, for the UA-FLP, most scholars just focus on reducing the material handling cost between facilities within the workshop. In the actual process of facility layout, however, when the material handling cost declines, it often can make other performances of the workshop equipment decrease [3]. In fact, the closeness relationship, the distance requirement among facilities and some other indicators directly affect the effectiveness of the workshop layout and the investment cost. These factors should also be taken into account when evaluating the best layout scheme. In this paper, we consider simultaneously the material handling, the closeness relationship, the distance requirement and the aspect ratio for all facilities so as to establish a multi-objective optimization (MOO) model and obtain more scientific and reasonable layout scheme.

The weighted sum approach [4] which converts several objectives to a single scalar objective by using weighted coefficients is generally applied to solve the UA-FLPs with multiple objectives. The poor operability of this method attributes to the fact that it is hard to standardize objective functions and determine the weights beforehand. In order to overcome this defect of the classical weighted sum approach, Pareto optimization method [5] characterized by the evolution of the Pareto front which consists of the fitness values of a set of individuals with most profitable trade-offs between objectives can directly find the optimal solutions in multi-objective optimization space and does not need to convert multiple objectives into a single scalar objective, so it can be used in more extensive applications. At present, for multi-objective UA-FLPs, although all kinds of MOO methods [6] based on Pareto optimization have been put forward, they still have some shortcomings, such as slow convergence to the Pareto front and low efficient selection toward diversity of solutions. Therefore, further improvement and development are still awaited to enhance their effectiveness. This study proposes a novel MOEA called the configuration space evolutionary (CSE) algorithm which can effectively exploit and explore the solution space for the UA-FLPs with four optimization objectives. The main contributions of this study are as follows: (1) the use of a mechanism of evolution of population in the algorithm based on a configuration bank, whose radius is gradually reduced to control the convergence of the CSE; (2) the use of a combination of the nearest and farthest candidate solution method based on objective function normalization and fast non-dominated sorting to choose the Pareto-optimal solutions, which is able to get a good spread in the Pareto front and keep the diversity of the solutions obtained. The numerical results show that the proposed CSE algorithm can find effectively the Pareto-optimal solutions of the problem, and has better convergence and diversity than other approaches in the literature.

This paper is structured as follows. Section 2 reviews the related works in the literature. Section 3 describes the problem statement for the multi-objective UA-FLP. To solve this problem, the proposed CSE algorithm is introduced in Section 4 and performances of the suggested algorithm and comparative experiments are presented in Section 5. Conclusions with a short summary are drawn in the last section.

2. Related works

This section first introduces the optimization approaches for UA-FLPs and multi-objective UA-FLPs, and then handling methods of overlapping among facilities are introduced and analyzed. At last, a survey of the related papers in the literature is given.

2.1. Optimization approaches for UA-FLPs

To solve the UA-FLPs, various kinds of exact or approximate approaches have been proposed over the past several decades. Among these approaches, heuristics and meta-heuristics have attracted much attention due to their capability of strong global search. Genetic algorithms (GAs) are computational methods based on the principles of natural selection and have been widely applied to handle the UA-FLPs. Gómez et al. [7] gave a mix-integer programming (MIP) model for optimizing the material handling costs or the closeness relationship between facilities. They applied GAs to solve layout problems in plant floors with aisles. A set of 20 facilities was designed to test the effectiveness of the GAs. Liu and Meller [8] put forward a GA-based heuristic which combined the sequence pair (SP) representation and heuristic strategy based the MIP model to solve the UA-FLP. For a given SP, the corresponding layout was determined by the linear programming relaxation of the MIP model. They showed the effectiveness and efficiency of their algorithm by testing different sized problems from both the literature and industrial applications. Gonçalves and Resende [9] presented a biased random-key GA for determining the order of placement and dimensions of each facility in the UA-FLP whose objective is to minimize the sum of the material handling costs between the centroids of the facilities. In addition, a novel placement strategy was applied to position each facility and a linear programming model was used to fine-tune the solutions. This combination improved the results in most of the cases tested. Paes et al. [10] introduced two alternative heuristics for the UA-FLP: a basic GA, which relied on a solution representation as a permutation of facilities and used a greedy heuristic for their insertion on the floor space, and a GA with decomposition and reconstruction (DRGA). The experimental results showed that the DRGA was more feasible in addressing FLPs for larger-size instances than the basic GA. Approaches based on traditional GAs usually exhibit a typical problem: premature convergence. Moreover, all the individuals in a population normally need to be selected and evaluated in each generation and they can be crossed and mutated, all of which requires additional CPU time when searching for good solutions.

Several GA variants have also been proposed to solve the UA-FLPs. Ulutas and Kulturel-Konak [11] introduced an artificial immune system-based algorithm to solve the UA-FLP with flexible by structure (FBS), where the proposed clonal selection algorithm (CSA) had a new encoding and dummy facilities were introduced to fill the empty space in the plant floor. In the CSA, the hypermutation and the receptor editing operators were distinctive from standard GAs where crossover and mutation are the basic tools for creating new solutions. The algorithm showed consistent performance for the 25 problem cases studied. A new hybrid evolutionary algorithm (HEA) incorporating decision maker's knowledge into the evolutionary algorithm was proposed by García-Hernández, et al. [12] to solve the UA-FLP with FBS. The novel hybrid system included a combination of an interactive GA with two different niching methods to preserve the diversity of solutions. This interactive algorithm might also execute slowly because they required the intervention of a decision maker (generally, a human expert); furthermore, the decision maker could be at risk of fatigue due to the amount of information to be evaluated. Palomo-Romero, et al. [13] presented an island model genetic algorithm (IMGA) which used the parallel evolution of a certain number of populations. The parallel approach helped to avoid premature convergence and excessive execution time. An adaptive penalty function was used to direct the search process to the feasible solution regions. In addition, this method improved search diversity, allowing it to explore a larger search space and obtain better solutions. However, optimizing the number of islands (subpopulations) might be an issue

that should be considered. The procedure of the harmony search (HS) is analogous to an improvement operation for a GA. The main difference between GA and HS is that the HS algorithm generates a new solution after considering all of existing solutions in the harmony memory, while the GA only considers the two parent solutions. In recent years, some researchers have introduced HS to solve the UA-FLPs. For example, Chang and Ku [14] developed a heuristic method by combining the slicing tree representation and a quadratic constraint program (QCP) model with the HS to deal with the UA-FLP. The experimental results showed that this approach was effective and could find optimal or near-optimal solutions for most problems with fewer than 20 facilities. However, one limitation of this study is that the size of the instance is small, which should be expanded. Along the same direction, Kang and Chae [15] proposed a novel HS method by incorporating some heuristic approaches to generate a quality solution of the UA-FLP, where the structure of the slicing tree representation was modified and a re-adjustment operation was added to diversify the possible range of solutions. Their approach used a penalty scheme to produce the feasible solutions more effectively. Furthermore, they provided the results from experiments with a set of large-size problems that demonstrated the robustness of their algorithm. In these proposals, the evolution of the population is guided by implementing subjective scores that evaluate a set of representative layouts, where the problem of becoming trapped in local optima is still easy to happen.

Meta-heuristics have global search ability and broad adaptability. Some meta-heuristics, such as ant colony optimization (ACO), simulated annealing (SA), particle swarm optimization (PSO), and Tabu search (TS), seek to overcome the main disadvantages of sequential GAs. ACO is a bio-inspired optimization algorithm based on simulating the behavior of natural ants that succeed in finding the shortest paths from their nest to food sources. Some hybrid approaches based on ACO have also been developed to handle the UA-FLPs. Komarudin and Wong [16] put forward an ant system (AS) (one of the ACO variants) using the slicing tree structure (STS) to represent the problems for solving UA-FLPs, without too restricting the solution space. An ant solution was given a penalty value that was proportional to the number of infeasible facilities that it contained. Guan and Lin [17] proposed a hybrid algorithm by combining the variable neighborhood search (VNS) and the ACO with a new pheromone rule and a reverse criterion to solve the single row FLP. In their approach, three neighborhood structures were utilized to enhance the exploitation ability of the algorithm. Scholz et al. [18] proposed a TS algorithm based on slicing tree for solving the UA-FLP. They incorporated the possibility to specify various requirements regarding shape and dimensions of each individual facility by using bounding curves. The experimental results showed that the TS algorithm made great improvements compared with previous researches. SA is a probabilistic meta-heuristic to search for global optima in problems with complex search spaces. Kulturel-Konak and Konak [19] focused on solving four different UA-FLPs on the continuous plane for optimizing the sum of material handling costs and rearrangement costs, and proposed a large-scale hybrid simulated annealing (LS-HSA) algorithm based on SA and MIP. Recently, Hunagund et al. [20] presented a SA-based heuristic to solve unequal area dynamic FLPs with FBS. The proposed SA heuristic gave new best solutions or the same solutions as compared to other meta-heuristics for the tested instances. Liu et al. [4] described a model where the sum of the material handling and rearrangement costs was minimized. A hybrid algorithm which combined the Wang–Landau (WL) sampling algorithm and some heuristic strategies (vacant point strategy, pushing strategy, and pressuring strategy) was proposed to solve the UA-FLP. The computational results of the four groups of cases showed the effectiveness of the proposed algorithm. As a representative of bio-inspired algorithm and meta-heuristic algorithm, PSO can also solve continuous optimization problems and has good convergence speed. Asl and Wong [21] developed a modified PSO algorithm combining two local search methods to solve unequal area static and dynamic FLPs where the facilities have fixed shapes and areas throughout the time horizon. The objective of unequal area static FLP is to minimize the sum of the material handling costs while the objective of unequal area dynamic FLP is to minimize the sum of the material handling costs and rearrangement costs. The results showed that the proposed algorithm has created encouraging layouts in comparison with other approaches.

2.2. Optimization approaches of multi-objective UA-FLPs

Although the approaches mentioned above work relatively well for the UA-FLPs, most researches focus only on the single objective optimization of minimizing the material handling costs or the sum of the material handling and rearrangement costs which belongs to a quantitative factor. However, practical UA-FLPs concern generally some competing objectives including both quantitative and qualitative ones. The closeness relationship, distance requirement, safety and so forth are important qualitative factors in UA-FLPs. They have also significant impact on the final location of facilities and should be taken into account simultaneously. So, UA-FLPs generally belong to multi-objective optimization problems (MOPs).

In solving MOPs, an amount of MOO algorithms have been suggested, which mainly include the following categories: EA [12], PSO [21,22], ACO [16,17,23], artificial immune systems (AIS) [11,24], interval multi-objective quantum-inspired cultural algorithms (IMOQCA) [25], brain storm optimization algorithms (BSO) [26] and so on. In these algorithms, a category of the well-known methods based on EA is the so-called multi-objective evolutionary algorithm (MOEA) [27] which is a kind of representative MOO method based on group and is suitable for solving constrained optimization problems with multiple objectives. Based on their selection strategies, these MOEAs can be categorized into three main classes [28]. The first category contains Pareto-dominance-based algorithms, e.g., non-dominated sorting genetic algorithm II (NSGA-II) [29], preference-inspired co-evolutionary algorithm using goal vectors (PICEA-g) [30,31] and grid-based evolutionary algorithm [32,33]. The second contains performance indicator-based algorithms, e.g., approximated hyper volume-based evolutionary algorithm (HypE) [34]. The last contains decomposition-based algorithms, e.g., multi-objective evolutionary algorithms based on decomposition (MOEA/D) [35], and NSGA-III [36]. These algorithms decompose an MOP into a number of optimization problems and solve them using a population-based search in a collaborative manner.

For multi-objective UA-FLPs, more and more researchers have interested in Pareto optimization method. Liu and Liu [37] proposed an improved version of the multi-objective ant colony optimization (MOACO) algorithm to solve the UA-FLP with two objectives. In the modified MOACO algorithm, they put forward a novel pheromone updating method, and combined the heuristic layout updating strategy, the Pareto optimization and the niche technology to obtain Pareto-optimal solutions of the problem. A combination of the local search based on the adaptive gradient method and the heuristic department deformation strategy (DDS) was applied to deal with the non-overlapping constraint between facilities. The experimental results on ten benchmark instances showed that the proposed MOACO algorithm was an effective method for solving the UA-FLP with two objectives. However, when the sum of the areas of all facilities is close

or equal to the area of the plant floor, the efficiency is low for the MOACO to optimize the objectives and deal with the interference between facilities. Zuo et al. [38] combined multi-objective Tabu search with linear programming (MTS-LP) for an extended double row layout problem where two objectives (material movement and layout area) were considered. Experimental results showed that MTS-LP improved upon NSGA-II and an exact approach for problems of no more than 30 machines. Saraswat et al. [39] presented a hybrid algorithm that combined the SP representation, e-accurate model and multi-objective SA algorithm to solve the UA-FLP with three objectives: flow-distance, the number of required material handling devices and average work-in-process. However, this research neglected intra-departmental queues when calculating the average WIP of a layout. Liu et al. [6] studied a multi-objective UA-FLP where the objectives of the problem aimed to minimize the material handling costs, maximize the total adjacency value and maximize the utilization ratio of the plant floor (i.e. minimize the layout area). They put forward a heuristic configuration mutation operation and subsequent local search to satisfy the non-overlapping constraint, and the multi-objective particle swarm optimization (MOPSO) algorithm to obtain a set of Pareto-optimal solutions of the problem. The numerical results on three sets of different UA-FLPs from the literature with the size of the problem up to 62 facilities showed that the proposed method was effective in solving the multi-objective UA-FLP. Aiello et al. [40] proposed a Pareto based new multi-objective genetic algorithm (MOGA) to solve UA-FLPs with four objective functions (material handling costs, aspect ratio, closeness and distance requests). The suggested MOGA was based upon the STS. Results on an instance of 20 facilities confirmed the effectiveness of MOGA in solving the multi-objective UA-FLP. The main advantage of the proposed approach is its capability to explore a wide space of solutions, preserving the practicability of the design. But, a decision maker cannot choose optimal layouts from the same rank. Li et al. [41] applied NSGA-II to optimize a UA-FLP model referring to three objective functions (material handling costs, closeness and utilization ratio of plant floor). The proposed algorithm adopted the strategy of the line-by-line placement to avoid the overlapping of facilities. However, the resulting layout was not compact enough. Ripon et al. [42] built a mathematic model for optimizing the material handling costs and closeness rating, and proposed an adaptive VNS with a modified 1-opt local search. Unlike conventional local search, the proposed adaptive local search scheme could be used to automatically determine whether the local search should be used in a GA loop or not. This can save the computation time wasted by the traditional local search. García-Hernández et al. [43] proposed an evolutionary neural hybrid system by incorporating human expert knowledge into the UA-FLP where four objectives (material handling costs, closeness and distance requirement, aspect ratio) were considered. A subset of facility designs was generated via the GA and then evaluated by human expert's knowledge learning from an artificial neural network (ANN). The results of the experiments on a real case of 365 facility layout designs validated the proposed approach. Vitayasak et al. [44] presented a modified backtracking search algorithm (BSA) which is a population-based iterative evolutionary algorithm for solving stochastic dynamic FLPs with stochastic demand. The combination of material flow and redesign costs were optimized. These proposals overcome the problems related to the traditional weighted sum approach, where it is difficult to determine the weight coefficients and standardize objective functions. However, the convergence and diversity of solutions are hard to be guaranteed. Some effective optimization techniques need to be explored further.

2.3. Handling methods of overlapping among facilities

For solving the UA-FLPs, another challenging issue except for the optimization approach is how to handle the non-overlapping constraint among facilities. The traditional solution method usually adopts line placement strategy [7,38,41,44] or penalty function method (PFM) [8–10,13,19,21,39]. However, their performances are not always satisfactory because line placement methods limit the number of possible layouts generated and the PFMs are hard to find appropriate penalty factors designed to direct the search toward the constrained optimal solutions. Flexible bay structure (FBS) [11,12,20] is the expression of a continuous facility layout. Based on the FBS, the layout area is divided into several parallel horizontal or vertical bays (columns). Similarly, an alternative decomposition scheme for facility layout design is the slicing tree structure (STS) [14–16,18,40,42]. The STS is an encoding representation that organizes a layout into a tree structure where the relative locations of the facilities on the floor are represented by a location matrix encoded in two chromosomes. A slicing structure is obtained by cutting orthogonally an initial rectangle along horizontal or vertical direction (called guillotine cut) and recursively executing to produce a series of rectangles. In the STS, the plant floor is partitioned both in vertical and horizontal directions simultaneously, while in the FBS, the plant floor is partitioned either in vertical or horizontal direction, but not in both directions. In general, the layout optimization approaches that apply the FBS or the STS to design layouts have good effectiveness for solving the UA-FLPs. The encoding vector of the STS is relatively long because it contains a variety of information about the consecutive operations of the slicing tree. The STS requires a $(3n\text{-}2)$-sized encoding vector [15] to represent a layout with n facilities, whereas the FBS requires only a $(n+B)$-sized vector (where B is the number of bays in the layout). Although the STS can represent wider layouts some of which cannot be constructed in the FBS, the FBS is more suitable for the layout of flexible facilities with fixed area. Furthermore, the blocky structure adopted by the FBS is more suitable for practical application, and corridors can be set in the layout, which also increases the universality of FBS application. Therefore, in this paper, we adopt the FBS method to eliminate overlapping in dealing with UA-FLP with constraints and obtain feasible layouts.

2.4. A survey of the related references

Table 1 gives a survey of the related papers in the literature where the gaps and overlaps are identified with a spotlight on the objective functions of different models, resolution approaches and methods of handling non-overlapping constraint among facilities that have been applied to solve UA-FLPs. Several observations can be made regarding these research works. First, it can be noted that there are more researches focused on a single objective in the mathematical models for the UA-FLPs while relatively less on multiple objectives. Among multi-objective problems, most of them aim to minimize the total material handling costs and maximize the satisfaction of closeness relationship. Second, comparing meta-heuristics together, one finds that the procedures based on GAs and their variants are most popular. Third, for multi-objective UA-FLPs, it can be noted that Pareto-based approaches are relatively successful in exploring the Pareto-front and maintaining diversity. However, the current MOO algorithms still suffer from a great challenge in the convergence to the Pareto optimal set and the maintenance of diversity of the Pareto-optimal solutions [45] when dealing with the UA-FLPs possessing more than three objectives and the large-scale size of instances. Therefore, there is a growing urgent need for developing a high-performance MOO algorithm for multi-objective UA-FLPs.

Table 1
Survey of papers related to UA-FLPs.

Reference	Objective functions	Resolution approaches	Handling overlapping methods
Gómez et al. [7]	Minimize MHC or Maximize CR	GA	Line placement
Liu and Meller [8]	Minimize MHC	GA + Sequence pair representation	Penalty function
Gonçalves and Resende [9]	Minimize MHC	Biased random-key genetic algorithm	Penalty function
Paes et al. [10]	Minimize MHC	Basic GA, GA with decomposition and reconstruction	Penalty function
Ulutas and Kulturel-Konak [11]	Minimize MHC	Clonal selection algorithm	FBS +Penalty function
García-Hernández et al. [12]	Decision maker's preferences	Hybrid evolutionary algorithm	FBS
Palomo-Romero et al. [13]	Minimize MHC	Island model genetic algorithm	Penalty function
Chang and Ku [14]	Minimize MHC	Harmony search-based heuristic	STS
Kang and Chae [15]	Minimize MHC	Harmony search-based heuristic	STS+ Penalty function
Komarudin and Wong [16]	Minimize MHC	Ant system	STS + Penalty function
Guan and Lin [17]	Minimize MHC	ACO + Variable neighborhood search	\
Scholz et al. [18]	Minimize MHC	Tabu search	STS
Kulturel-Konak and Konak [19]	Minimize the sum of MHC and RC	SA	Penalty function
Hunagund et al. [20]	Minimize the sum of MHC and RC	SA	FBS
Liu et al. [4]	Minimize the sum of MHC and RC	Heuristic Wang–Landau	Heuristic approach
Asl and Wong [21]	Minimize MHC or Minimize the sum of MHC and RC	Modified PSO + local search methods	Penalty function
Liu and Liu [37]	Minimize MHC and Maximize CR	MOACO	LS+DDS
Zuo et al. [38]	Minimize MHC and Minimize LA	Multi-objective Tabu search + Linear programming	Line placement
Saraswat et al. [39]	Minimize MHC, Minimize NMHD, and Minimize WIP	Multi-objective SA	Penalty function
Liu et al. [6]	Minimize MHC, Maximize CR, Minimize LA	Multi-objective PSO	LS
Aiello et al. [40]	Minimize MHC, Maximize CR, Maximize DR, and Maximize AR	Multi-objective GA	STS
Li et al. [41]	Minimize MHC and Maximize CR	NSGA-II	Line placement
Ripon et al. [42]	Minimize MHC and Maximize CR	Adaptive variable neighborhood search	STS
García-Hernández et al. [43]	Minimize MHC, Maximize CR, Maximize DR, and Maximize AR	GA + artificial neural networks	\
Vitayasak et al. [44]	Minimize MHC and Minimize RAC	Modified backtracking search algorithm	Line placement
Liu et al. (this paper)	Minimize MHC, Maximize CR, Maximize DR, and Maximize AR	Configuration space evolutionary algorithm	FBS

Notation conventions

MHC: the sum of material handling costs	DR: the satisfaction of distance requirements	RC: the sum of rearrangement costs	WIP: the average work-in process
CR: the satisfaction of closeness relationship	AR: the satisfaction of aspect ratio	NMHD: the number of material handling devices	LA: the layout area

In this paper, four different objectives of the UA-FLPs are taken into account: (1) minimization of material handling costs; (2) maximization of the satisfaction of weighted closeness relationship; (3) maximization of the distance requirements and (4) maximization of the satisfaction of aspect ratio of facilities. The above objectives are chosen because they are frequently cited in the literature and involve quantitative (1) and qualitative (2), (3), and (4) aspects. Along the line of thought of MOGAs (or MOEAs), a novel MOEA, called the CSE algorithm, is proposed. A mechanism of evolution of generations based on a mating pool named configuration (solution) bank is proposed in the CSE algorithm, whose convergence is controlled by a measure of the radius of the configuration bank, whose value is gradually reduced to narrow the search space. Also, a combination of the nearest and farthest candidate solution method based on objective function normalization and non-dominated sorting is used to choose the Pareto-optimal solutions of the problem, which is able to get a good spread in the Pareto front and keep the diversity of the solutions obtained.

It is noteworthy that this paper distinguishes from our previous publications [4,37], and [6] in objective functions, resolution approaches and handling overlapping constraint aspects although these four papers all study the UA-FLPs, as shown in Table 1. In fact, four objectives (minimizing MHC, maximizing CR, maximizing DR, and maximizing AR) in this paper are taken into account, while [4,37], and [6] studied a single objective (minimizing the sum of MHC and RC), two objectives (minimizing MHC and maximizing CR) and three objectives (minimizing MHC, maximizing CR, minimizing LA), respectively. Furthermore, a so-called CSE algorithm is proposed in this paper, which combines a mechanism of evolution of generations based on a mating pool, and a policy of updating the configuration bank based non-dominated sorting and the nearest and farthest candidate solution method, in order to improve the convergence and diversity of solutions. Whereas, in [4], a hybrid algorithm which combined the Wang–Landau sampling algorithm and some heuristic strategies (vacant point strategy, pushing strategy, and pressuring strategy) was proposed to solve the UA-FLP. In [37], an improved version of the MOACO algorithm was proposed to solve the UA-FLPs. In the MOACO, the Pareto optimization based on the local pheromone communication and the global search based on the niche technology were applied to obtain Pareto-optimal solutions of the problem. In [6], a MOPSO algorithm was presented to solve the UA-FLPs. In the MOPSO, an objective space division method was applied to locate addresses of the new configurations of all the particles in the objective space according to their values of objective functions and an external archive was used to store Pareto-optimal solutions of the problem. Finally, the FBS is adopted to satisfy the non-overlapping constraint in this study, while heuristic methods combining a pushing strategy and a pressuring strategy in [4] were used to deal with the non-overlapping constraint. A combination of the local search based on the adaptive gradient method and the heuristic department deformation strategy in [37] was applied to deal with the non-overlapping constraint between facilities. In [6], they put forward a heuristic configuration mutation operation and subsequent local search to satisfy the non-overlapping constraint.

3. The problem statement

Assume that there are n fixed-area facilities and a plant floor, the length and the width of which are L and H, respectively. The multi-objective UA-FLP with FBS is the problem of placing facilities in the given plant floor in the representation of FBS so that facilities do not overlap each other and are satisfied with some given objectives. Because of the complexity of the features that define the UA-FLP configurations (in this problem, a configuration represents a possible layout of all facilities and all configurations are dispersed in the solution space independently), it is preferable to construct a solution for the UA-FLP under the following assumptions: (1) the shapes of facilities are either square or rectangular and their lengths and widths can be changed within a certain range; (2) the area of the plant floor is larger than or equal to the sum of areas of all facilities; (3) facilities must be located within a given plant floor without overlapping each other; (4) The route of handling material is horizontal or vertical, and material flow is from a facility's center to another facility's center; (5) too extreme shape of any facility cannot be accepted.

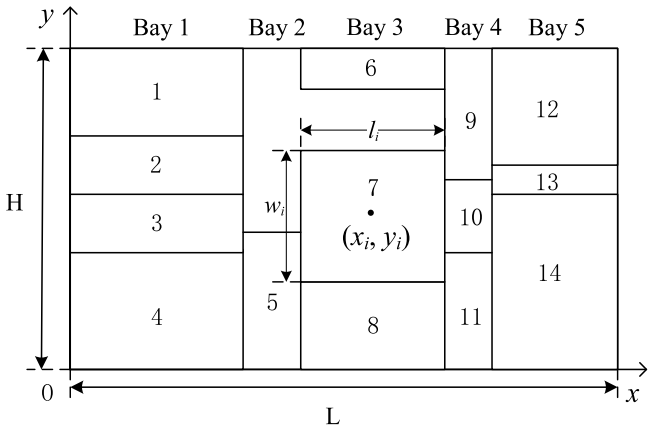


Fig. 1. A schematic diagram of facility layout in a plant floor with five bays.

Considering the characteristics of the facility and the convenience of description, we define a coordinates system (Fig. 1) which is used to describe the positions of n facilities and calculate the center of the configuration. The bottom-left corner of the plant floor is set as the origin and the x-axis coincides with the length edge of the plant floor and the y-axis coincides with the width edge of the plant floor. Let l_i and w_i represent the actual length and the actual width of facility i, respectively. (x_i, y_i) denotes the coordinates of the center of facility i and $\boldsymbol{X}= (x_1, y_1, l_1, w_1, x_2, y_2, l_2, w_2, \ldots, x_n, y_n, l_n, w_n)$ is called as a layout (i.e., a configuration or a solution) of all n facilities. Fig. 1 shows the scheme of a layout with fourteen facilities for a UA-FLP with FBS, where the number of bays is five.

This study refers to the optimization of the following four objectives [40,43]: material handling costs, weighted closeness relationship, distance requirement and aspect ratio requests of facilities. Their formal descriptions are presented as follows:

$$minimize \quad F_1(\boldsymbol{X}) = \sum_i \sum_j (f_{ij} c_{ij}) dm_{ij} \tag{1}$$

$$maximize \quad F_2(\boldsymbol{X}) = \sum_i \sum_j r_{ij} g_{ij} \tag{2}$$

$$maximize \quad F_3(\boldsymbol{X}) = \sum_i \sum_j s_{ij} de_{ij} \tag{3}$$

Formula (1) denotes that the total material handling cost between facilities should be small as far as possible. Formula (2) denotes that the total weighted closeness relationship between facilities should be maximized. The third objective (3) is to maximize the distance requirement between facilities. f_{ij} denotes the material flow between facilities i and j, and c_{ij} is the material handling cost for moving one unit load per unit distance from facility i to facility j. Closeness rating r_{ij} is based on the intimate

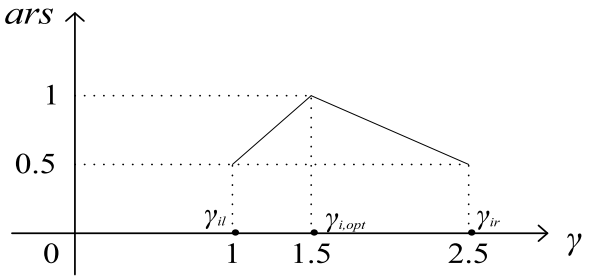


Fig. 2. Satisfaction function of the aspect ratio.

degree between facilities, which describes the connection degree of the layout of facilities, and g_{ij} is the contact perimeter length between facilities$i$ and j. Distance requirement factor s_{ij} is determined by the environment relation between facilities according to a designer's demands which can be the separation of some facilities from others. Reasons for such requirement are on the consideration of environmental issues like vibration, pollution, noise or aspects related to the security of workers or risks of fire or explosion, which requires maintaining a certain degree of distance between some facilities. The distance dm_{ij} between facilities i and j is the Manhattan distance between their centers in formula (1), while the Euclidean distance de_{ij} has been employed in formula (3).

For the UA-FLP with FBS, each facility needs to have reasonable aspect ratio. The aspect ratio is considered as an objective (such as the case of Section 5.1) because some ratios are better than others, but the aspect ratio is also a constraint (see the problems of Sections 5.1–5.3) because any solution becomes infeasible if the aspect ratio of any facility surpasses its limit. The aspect ratio of facility i is defined as follows:

$$\gamma_i = \frac{\max\{l_i, w_i\}}{\min\{l_i, w_i\}} \tag{4}$$

For the case of 20 facilities with zero clearance [40] in Section 5.1, if the aspect ratio of the facility is within a certain interval it still is acceptable but its level of satisfaction will decrease as long as it deviates from the optimal shape. If the aspect ratio exceeds the given interval, the shape of facility cannot be accepted because resources for production cannot be placed inside the facility anymore, and at that time, the value of the satisfaction function of the aspect ratio will drop to 0. A zero value implies an unfeasible solution. Such a satisfaction function is given in Fig. 2 [40], where the horizontal coordinate γ denotes the aspect ratio of the facility, the vertical coordinate ars is its corresponding satisfaction function value, and $\gamma_{i,opt}$ denotes the optimal aspect ratio of facility i. γ_{il} and γ_{ir} represent the legal range of upper and lower bounds of aspect ratio for facility i, respectively. If the aspect ratio of each facility is in the interval between γ_{il} =1 and γ_{ir} =2.5, the layout is feasible; otherwise it does not satisfy the aspect ratio constraint of the problem. For a feasible layout $\boldsymbol{X}$, maximization of the satisfaction of the aspect ratio for all facilities is considered a further objective:

$$maximize \quad F_4(\boldsymbol{X}) = \frac{\sum_{i=1}^{n} ars_i}{n} \tag{5}$$

where the satisfaction level of a layout is measured by the mean of the aspect ratio satisfaction score ars of all facilities.

4. Configuration space evolutionary algorithm

MOEAs are widely used for solving optimization problems with multiple conflicting objectives. However, traditional MOEAs have shortcomings, such as slow convergence to the Pareto front and low efficient selection toward diversity of solutions, especially when dealing with the UA-FLPs with three or more objectives. In this section, a novel MOEA called configuration space evolutionary (CSE) algorithm that can search the solution space efficiently and find good spread of solutions is developed to solve the UA-FLP with multiple objectives.

4.1. Framework of configuration space evolutionary algorithm

The configuration space evolutionary (CSE) algorithm (see Algorithm 1) is a novel MOO method. At the beginning of the CSE, we randomly generate $k_1 = 50$ feasible configurations (called an initial configuration bank P_{init} in CSE) in the configuration space (see Section 4.2). On the basis of the center of each configuration in P_{init}, we construct a circular region with the radius $d_{space} = d_{avg}/2$, where d_{avg} equals the average value of Euclidean distances among centers of all configurations in the initial configuration bank P_{init}. The center $o(x, y)$ of each configuration is computed by:

$$o(x, y) = \left(\frac{\sum_{i=1}^{n} x_i}{n}, \frac{\sum_{i=1}^{n} y_i}{n}\right) \tag{6}$$

where n denotes the total number of facilities, and (x_i, y_i) is the coordinates of the center of the ith facility.

Select randomly k_2=10 configurations (called seed configuration bank P_{seed}) from the configuration bank P_{new} (updated in each iteration, and its initial value is P_{init}) to produce $k_3 = 200$ new configurations (called evolution configuration bank P_{evolve}) by using evolutionary operations (selection, crossover and mutation) (see Section 4.3).

Then, we use the fast non-dominated sorting method [29] and the nearest and farthest candidate solution method (see Sections 4.4 and 4.5) to pick out $k_4 = 10$ excellent configurations (called the test configuration bank P_{test}) from the evolution configuration bank P_{evolve}. Subsequently, we apply these 10 new test configurations to update the configuration bank P_{new} (see Section 4.6). Once the configuration bank P_{new} is updated 50 times, we let $d_{space} = d_{space}/2$. If there still exist configurations in P_{new}, which are not used as the seed configurations, we select randomly 10 (or the rest several) unselected configurations as the seed configurations. Repeat the above process until all configurations in the configuration bank P_{new} are used as the seed configurations. Thus, one round of iteration of the CSE is finished. If the maximum number of iterations is not reached, we produce randomly 50 new feasible configurations (denoted by P_{init}) based on the FBS, and reset the new initial configuration bank by selecting 50 configurations with better fitness values from $P_{new} \cup P_{init}$ based on the fast non-dominated sorting method [29] and the nearest and farthest candidate solution method. Repeat the whole procedure until the maximum number of iterations is reached. Afterwards, we use the fast non-dominated sorting method [29] and the multiple attribute decision-making method [46] to select several excellent configurations as the Pareto-optimal solution set of the problem. Obviously, in the CSE, the number of configurations in P_{new} remains unchanged, which is 50 always. Note that k_1, k_2, k_3 and k_4 in the CSE are some changeable positive integer parameters.

4.2. Encoding of solution in flexible bay structure

EAs are a kind of highly parallel, random and adaptive metaheuristic algorithms based on survival fitness in natural evolution. EAs operate with populations, which are named chromosomes and encoded solutions of the problem. Each chromosome is made of fundamental gens. It simulates the collective learning process

Algorithm 1. Framework of the configuration space evolutionary (CSE) algorithm

Generate randomly k_1=50 feasible configurations based on the FBS in the configuration space (see Subsection 4.2), and gain an initial configuration bank P_{init}.
Calculate the average value d_{avg} of Euclidean distances among centers of all configurations in P_{init}, and construct 50 circular regions with the same radius d_{space} =d_{avg}/2. Let P_{new}= P_{init}, and *update*=0.
While (all in P_{new} are not used as seed configurations) **do**
  If (|P_{new}|>10) **then**
    Select randomly k_2=10 unselected configurations from P_{new} as the seed configurations bank P_{seed}.
  Else
    Select the rest several from P_{new} as the seed configurations bank P_{seed}.
  End if
  Apply evolutionary operations (selection, crossover and mutation) (see Subsection 4.3) for P_{seed} to produce k_3=200 configurations as the evolution configuration bank P_{evolve}.
  Choose k_4=10 excellent configurations from P_{evolve} as test configuration bank P_{test} by using the fast non-dominated sorting method and the nearest and farthest candidate solution method (see Subsection 4.4).
  Update the configuration bank P_{new} by replacing dominated configurations in P_{new} with domination configurations in P_{test} (see Subsection 4.6). Let *update*= *update* +1.
  If (*update* ≥ 50) **then**
    Let d_{space} = d_{space} /2, and *update*=0.
  End if
End while
If (maximum number of iterations is reached) **then**
  Execute the fast non-dominated sorting approach to P_{new}.
  Select all configurations in the first level as the Pareto-optimal solutions P_{opt}.
  Use the multiple attribute decision-making selection [46] to pick out the optimal solution set $\boldsymbol{X}^*$ from P_{opt}, and go to step 24.
Else
  Produce randomly 50 feasible configurations (denoted by P_{init}) based on the FBS.
  Select 50 configurations as the new initial configuration bank P_{init} from $P_{new} \cup P_{init}$ by the fast non-dominated sorting and the nearest and farthest candidate solution method, and go to Step 2.
End if
Return $\boldsymbol{X}^*$.

of individual group, and each of these individuals represents a point in solution space of the given problem. For each chromosome, a fitness value determines its survival rate. Usually, randomly generate the initial population and in order to obtain new population, the evolution process performs by executing encoding, selection, crossover, and mutation operators.

In the literature review on the UA-FLPs, some researchers have been using an interesting approach called the flexible bay structure (FBS) [12,20] to represent the scheme of layout. The FBS is a continuous layout representation allowing the facilities to be located only in parallel bays with varying widths. The width of a bay depends on the division result of areas of facilities that it contains, and all facilities in a bay have the same width (Fig. 3). Bays are bounded by straight aisles on both sides, and facilities are allowed to place only in one bay (i.e. no expansion over multiple bays). The number of bays and facilities that are placed in every bay are changeable. Therefore, the FLP is simplified as determining the number of bays and the facilities contained in every bay. In the light of the architecture of the FBS, all facilities in the plant floor are put from down to up, and from left to right. The layout based on the FBS can use simple numbers to describe layout results.

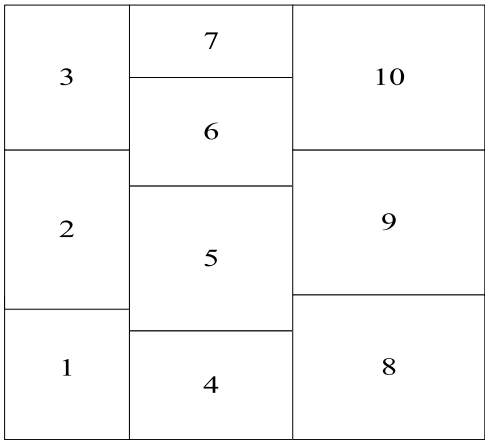


Fig. 3. Bay structure representation.

Optimal layout designs based on the FBS have some limits in layout form, compared with other layout forms such as the slicing structure. However, the FBS has also some desirable features. The

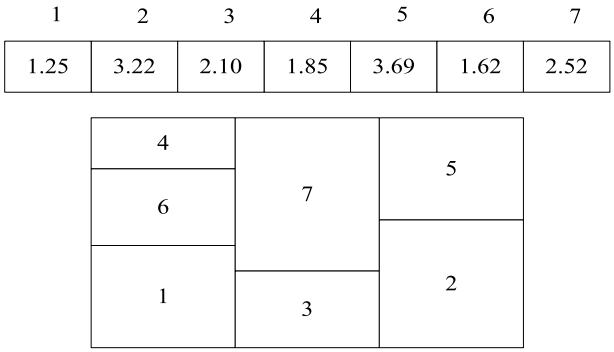


Fig. 4. Layout generated from the chromosome representation.

bay boundaries form the basis of an aisle structure that is in favor of transferring from a block layout to the actual facility layout. Therefore, the difference between the material handling costs of a theoretical block layout and its actual implementation is minimal for an FBS-based layout. In summary, the FBS representation is one of the most important layout representations that researchers and practitioners continue studying and using.

The CSE algorithm uses the FBS to represent the UA-FLP. The scheme of a layout is regarded as a chromosome, which is formed by n genes. Each gene adopts real number encoding z_i ($i = 1,2,\ldots, n$), which is a random number between 1 and $B + 1$ (B is the total number of the bays) whose integer part represents the number of bays and decimal part represents the order of facilities in each bay (smaller numbers have priority). In this paper, the numbering of the bays is from left to right (from $x = 0$) and the order position of the facility is from bottom to top (from $y = 0$). Fig. 4 shows a layout plan for a FLP with seven facilities in the planning horizon. In Fig. 4, the gene related to the sixth facility has 1.62 values. It means this facility will be arranged in the first bay and after comparing with other facilities (the first and fourth facilities, respectively, with 1.25 and 1.85 values) in this bay it has the second rank, in this layout, so it gains the second priority.

At the beginning of the CSE, 50 random configurations based on the FBS are produced according to the following procedures. First, we produce randomly an integer B between 3 and 7, where B is the total number of the bays. Then, n random real numbers z_i ($i = 1,2,\ldots, n$) ($1 \le z_i \le B+1$) are produced, which construct a chromosome. According to these random numbers z_i ($i = 1,2,\ldots, n$), facilities can be put orderly in the plant floor. Because the width of a bay is changeable, all n facilities will be placed within a given plant floor without overlapping each other, resulting in a feasible configuration. But a configuration may become infeasible if the aspect ratio (or the side length) of any facility surpasses its limit, and at that time, we will abandon this generated configuration and produce randomly another chromosome again until a feasible configuration is gained. Repeat the above steps until 50 feasible configurations are produced. Obviously, the coding numbers of genes in a chromosome generate randomly to increase the diversity of solutions. In addition, considering the randomness of the quantity of bays and the quantity of facilities placed in every bay, the UA-FLP can be converted to optimize the number of bays, the number of facilities contained in every bay, and the facilities' placement order.

4.3. Production of the configuration

In this section, we use the selection, crossover and mutation in evolutionary algorithms to produce some new configurations and enhance the diversity of solutions in the configuration space.

Table 2
Crossover operator in the proposed CSE algorithm.

Parent 1	Parent 2	Random values	Child 1	Child 2
1.25	1.42	0.26	1.38	1.29
2.24	2.45	0.12	2.42	2.27
2.62	3.25	0.12	2.05	2.84
3.55	3.83	0.53	3.68	3.70
4.12	4.45	0.22	4.38	4.19

(1) Selection. The elite strategy is used to reserve best individuals of the former population (all configurations in the seed configuration bank P_{seed}) with the elite rate k_{elite} (k_{elite}=0.2 in this paper). The number of individuals selected by the elite strategy is

$$N_{elite} = N_{seed} \times k_{elite} \tag{7}$$

where N_{seed} denotes the amount of individuals in the P_{seed}.

(2) Crossover. Crossover is the process of taking two parent individuals (configurations) to produce two children. After the selection process, the offspring are enriched with better individuals, but selection does not create new ones. Crossover operator is applied to the seed configuration bank P_{seed} with the hope that it creates some better offspring. By crossover, new generated configurations inherit good ingredients of the parents. In this paper, we use a continuous uniform crossover.

In the continuous uniform crossover we choose first two configurations with crossover probability k_{cross} from P_{seed} as parents, then a set of random values $\lambda_i (i = 1, 2, \ldots, n)$ with uniform distribution between 0 and 1 are produced, in same length with existent chromosome, and by using the following Eqs. (8) and (9), two children are created.

$$z'_{1i} = \lambda_i z_{1i} + (1 - \lambda_i) z_{2i}, \quad \forall\, i = 1, 2, \ldots, n \tag{8}$$

$$z'_{2i} = (1 - \lambda_i) z_{1i} + \lambda_i z_{2i}, \quad \forall\, i = 1, 2, \ldots, n \tag{9}$$

where z_{1i} and z_{2i} denote the first and second parents' genes, respectively; z'_{1i} and z'_{2i} are the first and second genes of offspring, respectively. So we can generate a total of 200 new configurations and gain an evolution configuration bank P_{evolve}. The crossover operation for a problem with five facilities is illustrated in Table 2.

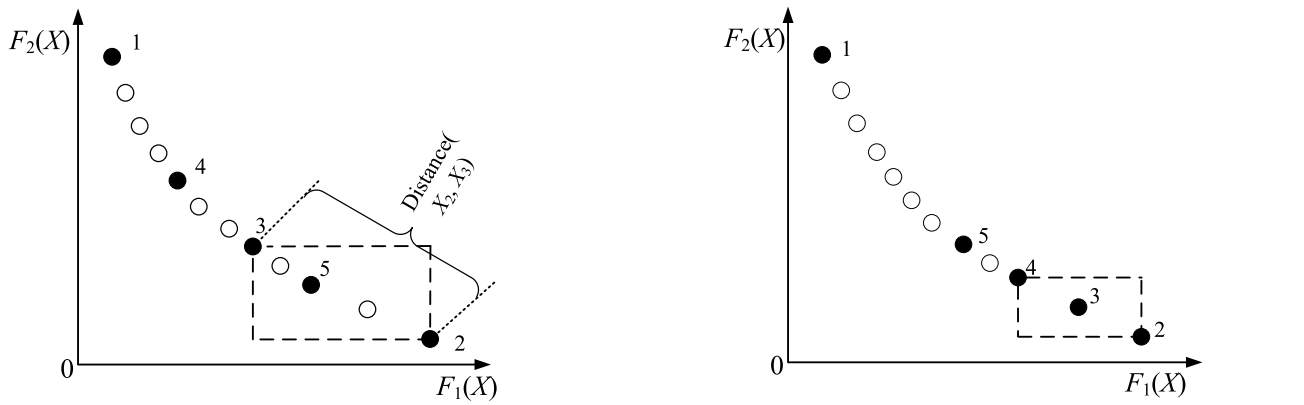
(3) Mutation. After crossover, the genes are subjected to mutation. Mutation is beneficial to increase the diversity of populations, and overcome the defect of premature convergence because adaptation values of offspring that crossover operation produces do not achieve optimum and not evolve any more.

We perform the fast non-dominated sorting [29] for the configurations obtained by selection and crossover, and the configurations in the final level after non-dominated sorting will be mutated. Mutation introduces new genetic structures in the population by randomly modifying some of its building blocks. A gene is selected randomly by mutation probability k_{mutate} and then is replaced with a random number between 1 and B+1, which is produced as following:

$$z'_i = \begin{cases} z_i + ((B+1) - z_i)\tanh(ku), & \text{if } \tanh(ku) \ge 0 \\ z_i + (z_i - 1)\tanh(ku), & \text{else} \end{cases} \tag{10}$$

where z_i is the value of the selected randomly gene and z'_i is the replaced value. u is a standard normal distribution random number, and k is its coefficient.

It is worth noting that whenever the crossover or mutation operations produce infeasible configurations (the aspect ratio (or the side length) of any facility surpasses its limit), we will abandon them and the crossover or mutation operations are repeated until new feasible configurations are gained. In the FBS



(a) Optimal solutions by the nearest and farthest candidate solution method (b) Optimal solutions by the crowding degree method

Fig. 5. Five optimal solutions selected by different methods for 12 Pareto-optimal solutions.

representation, because all facilities are put in the plant floor from down to up, and from left to right, it is obvious that configurations are satisfied with the non-overlapping constraint among facilities.

4.4. Nearest and farthest candidate solution method

In MOPs, it is desired that an algorithm maintains a good spread of solutions in the non-dominated solutions as well as the convergence to the Pareto-optimal set. In NSGA-II [29], a crowded degree comparison method is used to keep the spread of solutions. Thereafter, the crowded degree comparison method has been widely used for diversity maintenance of solutions in many MOEAs. However, it has an obvious flaw that the solutions in the high-density space have lower chance to be selected so that the spread of solutions is not good enough. So, some scholars have made improvements on the basis of the original crowded degree method. Kukkonen and Deb [47] proposed an improved pruning of non-dominated solutions for bi-objective optimization problems, which removed the solutions with the smallest crowding distance value one by one. In [48], a fast and effective method for pruning of the non-dominated solutions based on the crowded degree and the nearest neighborhood of solutions was proposed for many-objective problems. However, in some cases, these methods cannot still get a good spread of solutions.

We replaces the crowded degree comparison with the nearest and farthest candidate solution method in this paper, which can solve above difficulty to a certain degree. Given a candidate solution set C, which contains p non-dominated individuals, we need to choose q ($\leq p$) Pareto-optimal solutions from C and obtain an optimal solution set A. At the beginning of computation, let A be an empty set. Firstly, calculate all the objective function values F_i ($\boldsymbol{X}_j$) ($i = 1,2,\ldots, m$) of each solution $\boldsymbol{X}_j$ ($j = 1,2,\ldots, p$) in set C and find the minimal value of each objective, where m is the number of objective functions. There are two cases:

(i) If $q < m$, we randomly select q solutions with the smallest objective function values based on the preferences of different objectives into set A, and at the same time delete them from the candidate solution set C.

(ii) If $q \geq m$, for each objective, we select a solution with the smallest objective value and put it into A; at the same time delete it from C. The remaining u (which equal to q-m) population individuals are chosen from the updated candidate solution set C. For each solution $\boldsymbol{X}_s$ in C ($|C| = u$) at this time, we calculate the nearest distance (about normalized objective function) between $\boldsymbol{X}_s$ and all individuals in the optimal solution set A (see Eq. (11)). Choose solution $\boldsymbol{X}_{far}$ with the farthest distance from C and add it into the optimal solution set A and delete it from set C. Repeat aforementioned operations until the number of solutions in set A reaches q.

In the nearest and farthest candidate solution method, for two solutions $\boldsymbol{X}_s$ and $\boldsymbol{X}_t$, we need to define a distance calculation method based on the objective functions. Due to huge difference between the different objective values, we should normalize these objective functions (see Section 4.5) before calculation. The distance between solutions $\boldsymbol{X}_s$ and $\boldsymbol{X}_t$ is computed by:

$$\text{distance}(X_s,\ X_t) = \sqrt{\sum_{i=1}^{m}(\bar{F}_i(X_s) - \bar{F}_i(X_t))^2} \tag{11}$$

where $\bar{\boldsymbol{F}}_{\boldsymbol{i}}(\boldsymbol{X})$ is the value of objective function $F_i(\boldsymbol{X})$ after normalizing and m is the number of objective functions.

Fig. 5 shows an optimal solution set consisting of five Pareto-optimal solutions selected from twelve Pareto-optimal solutions, where two objectives F_1 and F_2 are considered. The solid circles in Fig. 5(a) is the optimal solution set obtained by the nearest and farthest candidate solution method, and the solid circles in Fig. 5(b) is the optimal solution set obtained by the congestion degree method in NSGA-II [29]. The numbers in the figure indicate the order of selecting solutions. It is not hard to see that the nearest and farthest candidate solution method can obtain a more uniform Pareto front than the traditional congestion degree method in NSGA-II, and the Pareto-optimal solutions obtained is more diverse also.

4.5. Normalization of the objective functions

We use the method of normalization of the objective functions in the NSGA-III [36,49]. First, calculate the least objective value $G_i^{\min}$ of every objective function F_i ($i = 1,2\ldots, m$) in the candidate solution set C, and then construct the ideal point set $G = (G_1^{\min}, G_2^{\min}, \ldots, G_m^{\min})$. The objective value of each solution in C is then translated by subtracting objective value $F_i(\boldsymbol{X})$ by $G_i^{\min}$. This translated objective is denoted by:

$$\boldsymbol{F}_i'(\boldsymbol{X}) = F_i(\boldsymbol{X}) - G_i^{\min} \tag{12}$$

where $\boldsymbol{X}$ is a layout of n facilities, that is, a configuration (or a solution). Therefore, the ideal point of translated C becomes a zero vector.

Second, identify the extreme point $G^{i,\max}$ in each (ith) objective axis by finding solution $\boldsymbol{X}$ that makes the following achievement scalarizing function (ASF) (formed with $F_i'(\boldsymbol{X})$ and a weight vector w close to the ith axis direction) minimum:

$$ASF(\boldsymbol{X}, w) = \max_{i=1}^{m} \frac{F_i'(\boldsymbol{X})}{w_i} \tag{13}$$

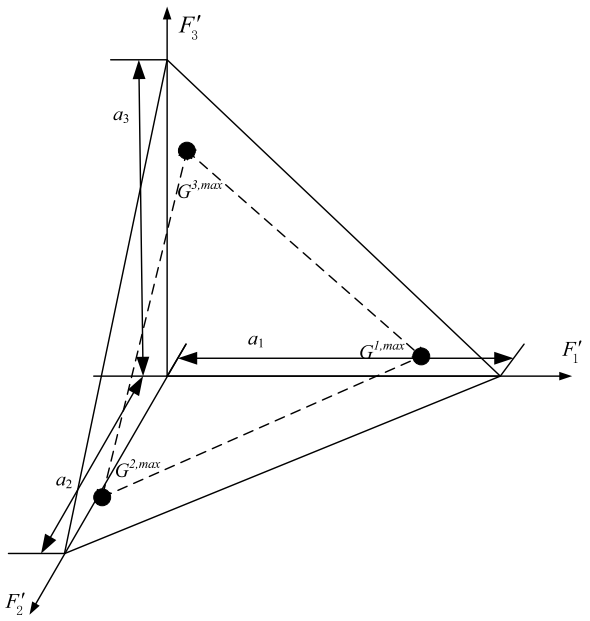


Fig. 6. Procedure of using extreme point to construct a hyper-plane for a three-objective problem.

In the above function, for the ith translated objective direction F_i $(i = l)$, where $l \in \{1, 2, \ldots, m\}$, we set its corresponding weight as 1, and for the rest objective direction $F_i (i \neq l)$, we set $w_i = 0$ (usually we replace it with a small number 10^{-6}).

Finally, according to these m extreme points we construct a m-dimensional hyper-plane. The intercept a_i between this hyper-plane and the ith objective axis can be computed. Note that if the extreme value point cannot construct the hyper-plane, then let $a_i = G^{i,\max}$ (see Fig. 6, where the objective is three-dimensional). The objective functions can then be normalized by the formula (14).

$$\bar{F}_i(\boldsymbol{X}) = \frac{F_i'(\boldsymbol{X})}{a_i}, \quad i = 1, 2, \ldots, m \tag{14}$$

4.6. Updating of the configuration bank

Updating the configuration bank is an essential step in the CSE. For each new test configuration $\boldsymbol{T}$ we calculate the Euclidean distance between it and each configuration in the configuration bank P_{new}. Choose the configuration with the nearest distance, named by $\boldsymbol{Y}$. The distance between configurations $\boldsymbol{T}$ and $\boldsymbol{Y}$ is denoted by $D(\boldsymbol{T}, \boldsymbol{Y})$.

(1) If $D(\boldsymbol{T}, \boldsymbol{Y}) \leq d_{space}$, the test configuration $\boldsymbol{T}$ is contained in the circle centered at the center of $\boldsymbol{Y}$, which means these two configurations are similar. If $\boldsymbol{T}$ dominates $\boldsymbol{Y}$, we add $\boldsymbol{T}$ into the configuration bank P_{new}, and delete configuration $\boldsymbol{Y}$ from P_{new} so as to keep the total number of the configurations in P_{new} constant. The center of circle moves from the center of $\boldsymbol{Y}$ to that of $\boldsymbol{T}$. If $\boldsymbol{Y}$ dominates $\boldsymbol{T}$, we keep $\boldsymbol{Y}$ and abandon configuration $\boldsymbol{T}$. If the configurations $\boldsymbol{T}$ and $\boldsymbol{Y}$ do not dominate each other we randomly select one from $\boldsymbol{T}$ and $\boldsymbol{Y}$ to update configuration bank. When $\boldsymbol{T}$ is selected, the center of circle is moved to that of $\boldsymbol{T}$.

(2) If $D(\boldsymbol{T}, \boldsymbol{Y}) > d_{space}$, the test configuration $\boldsymbol{T}$ falls outside of all existing circles. If $\boldsymbol{T}$ dominates a certain configuration $\boldsymbol{Z}$ in P_{new}, we remove $\boldsymbol{Z}$ from P_{new} and add $\boldsymbol{T}$ to the configuration bank P_{new}. The center of circle moves from the center of $\boldsymbol{Z}$ to that of $\boldsymbol{T}$. If $\boldsymbol{T}$ does not dominate any configuration in P_{new} the new configuration $\boldsymbol{T}$ is removed.

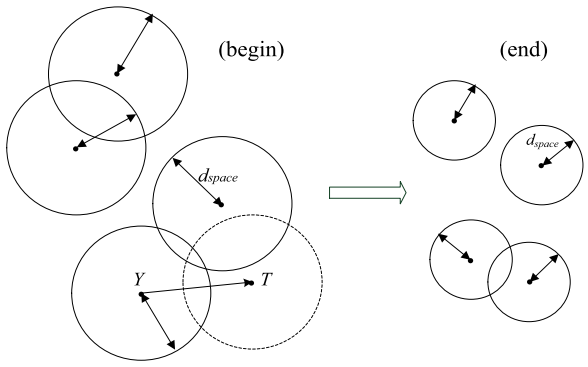


Fig. 7. Search procedure of the CSE in the configuration space.

Table 3
Parameter values of CSE.

Parameter	Value
Initial configuration size (k_1)	50
Seed configuration size (k_2)	10
Evolution configuration size (k_3)	200
Test configuration size (k_4)	10
Elite rate (k_{elite})	0.2
Crossover probability (k_{cross})	0.7
Mutation probability (k_{mutate})	0.2
Update frequency of d_{space}	0.5
Objective axis weight (w_i)	10^{-6}

Obviously, case (1) is more likely to happen when the circle is large, and case (2) happens when the circle is small. Consequently, a larger value of d_{space} produces more diversity of solutions, whereas a smaller value results in quicker convergence toward sub-optimal configurations at the expense of getting trapped in a basin. So, to search efficiently for the configuration space, it is necessary to maintain the diversity of solutions in the early stages and then gradually shift the emphasis toward sub-optimal configurations, which is realized by slowly reducing the value of d_{space} in the CSE (see Fig. 7).

5. Experimental results and analysis

To evaluate the performance of the CSE algorithm, three sets of instances from the literature are performed. The first set contains a case of 20 facilities with zero clearance. The second set contains a case of 12 facilities with vertical and horizontal aisles and the third set consists of six benchmark instances with different number of facilities. The algorithm is compiled by Java language and all instances are solved by a PC with Core 2 Duo processor (2.94 GHz) and 2.0 GB memory. The proposed CSE algorithm contains some parameters that have been tuned experimentally. The parameter values that provide good results are shown in Table 3, and the maximum number of iterations for each instance is listed in the resultant tables of the corresponding problems. To avoid random effects, each instance is executed 10 times independently.

5.1. A case of 20 facilities with zero clearance

The first case is from [40,50]. Facility areas are reported in Table 4. Flows between facilities and unit material handling costs are the same as [50]. Closeness requirements and distance requirements are shown in Table 5 and Table 6, respectively. The satisfaction function of aspect ratio of each facility is shown in Fig. 2.

The feasible and Pareto-optimal solutions obtained by the CSE algorithm, characterized by different non-dominated values of

Table 4
Facility areas in the first case.

Facility	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Area	27	18	27	18	18	18	9	9	9	24	60	42	18	24	27	75	64	41	27	45

Table 5
Closeness requests between two facilities in the first case.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
6							4		4		4									
9												8		8						8
11							2							2						2

Table 6
Distance requests between two facilities in the first case.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1				8	8			8					8		8			8		
2				2	2			2					2		2			2		
3				4	4			4					4		4			4		

Table 7
Best solutions obtained by approaches CSE and MOGA for the first case.

Algorithm	Solution	$F_1(\boldsymbol{X})$ (min)	$F_2(\boldsymbol{X})$ (max)	$F_3(\boldsymbol{X})$ (max)	$F_4(\boldsymbol{X})$ (max)	Iterations	Time (s)
CSE	1	4914	**213**	**3333**	0.76	1500	10739
	2	4979	198	3264	**0.81**		
	3	4879	162	2640	0.71		
	4	**4698**	162	2565	0.72		
MOGA	1*	5128	150	1227	0.71	2500	–
	2*	6211	91.5	2218	0.69		

Table 8
Chromosomes of best solutions by approaches CSE and MOGA for the first case.

Algorithm	Solution	Chromosome
CSE	1	{6.4,2.7,6.0,3.2,2.1,3.9,3.8,3.4,5.4,5.2,4.9,4.6,1.3,3.0,4.2,6.2,1.8,1.5,2.5,5.7}
	2	{6.4,2.6,6.0,3.4,2.0,3.9,2.9,3.6,5.3,5.1,4.8,4.5,1.2,3.1,4.1,6.2,1.7,1.5,2.3,5.7}
	3	{2.8,1.5,1.7,1.2,5.0,3.3,3.1,2.0,3.6,2.6,6.3,6.0,3.9,3.8,5.6,4.7,4.5,5.3,6.1,2.3}
	4	{2.9,1.5,1.7,1.2,4.5,3.3,3.1,2.0,3.6,2.6,6.3,6.1,3.9,3.8,5.3,5.9,4.9,4.6,5.5,2.3}
MOGA	1*	0-0-0-0-0-15-8-7-6-4-1-20-11-19-2-16-9-5-14-17-10-12-3-18-13
	2*	0-0-0-0-0-1-11-20-15-8-7-6-19-9-4-2-17-16-12-5-3-14-10-18-13

the objective functions, are in the number of 28 on a population of 50 individuals. Four of them are selected as optimal solutions by the Electre III [46]. The objective values and the corresponding chromosomes of the four solutions obtained are reported in Table 7 and Table 8, respectively, and the corresponding layouts are shown in Fig. 8. These solutions are compared with those obtained by a non-dominated ranking multi-objective genetic algorithm (MOGA) [50], which began from 50 randomly produced layouts based on the STS. Table 7 shows that all the solutions reported dominate those obtained by the MOGA, verifying the effectiveness of the proposed CSE algorithm. In Fig. 8, the facilities having a closeness and distance requests (Tables 5 and 6) are highlighted in dark gray and light gray areas, respectively. Solutions 1* and 2* (Tables 7 and 8) are optimal solutions reported in [50].

From the results in Table 7, one can see that the optimal values of four objectives are 4698, 213, 3333 and 0.81, respectively. Compared with the best results by the MOGA in the literature, material handling costs by the CSE reduces (5128-4698)/5128×100%=8.39% and the closeness, distance and aspect ratio increase (213–150)/150×100%=42%, (3333–2218)/2218×100%=50.27% and (0.81–0.71)/0.71×100%= 14.08%, respectively. About the number of generations of iterations, best solutions are reached by the MOGA in approximately 2500 iterations. The number of iterations for the CSE to find the best solutions are about 1500 interations, which are obviously less than that of the MOGA. Generally, achieving to better solutions can be resulted from more iterations of the algorithm. Whereas the proposed CSE algorithm obtains better solutions than the MOGA in less iterations, demonstrating the effectiveness of the CSE. Because of the differences in the performance of the running computers and the programming languages, the CPU time of other algorithms executed on different computers or programming languages cannot be used in this comparison. The running time by the MOGA has not been reported in [50]. But from Table 7, it is not hard to see that the CSE can find the results in an acceptable time.

5.2. A case of 12 facilities with vertical and horizontal aisles

To verify the validity of the algorithm, we test another case from [51], whose objectives refer to minimization of material handling costs $F_1(\boldsymbol{X})$, maximization of the weighted closeness relationship $F_2(\boldsymbol{X})$ and maximization of satisfaction of distance requirement $F_3(\boldsymbol{X})$. This model is different from the prior one. For the convenience of handling materials, the vertical and horizontal corridors between facilities within the plant floor are set, with a certain width (Fig. 9). This is in accordance with the actual situation.

In this model, the longitudinal aisle width W is 25 m, and the lateral aisle width D between facilities within the same block is 15 m. Total number B of bays is set to 4. The area of the workshop where there are a total number of 12 facilities is 420,000 square meters. The area of each facility is listed in Table 9. Flows, closeness relation and distance requirements for environment and security are shown in Table 10. Unit material handling cost

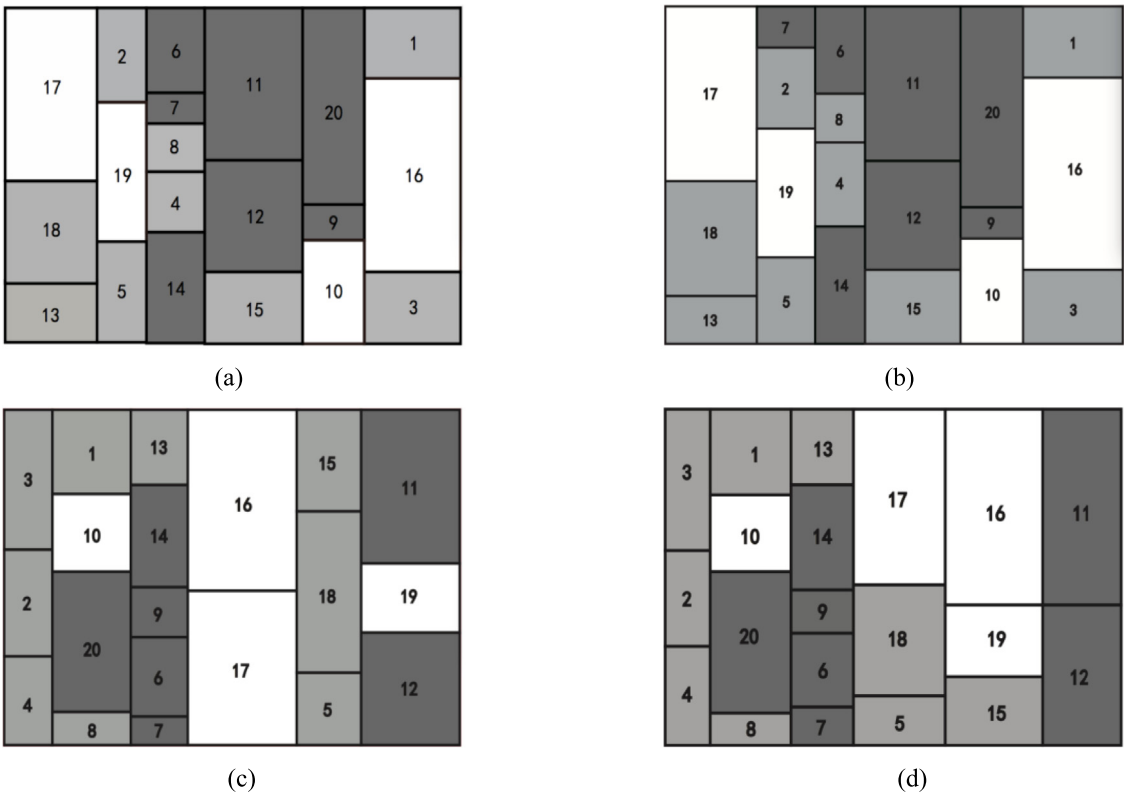


Fig. 8. Block layouts of the optimal solutions obtained by the CSE for the first case. Pictures (a), (b), (c) and (d) are four Pareto-optimal solutions in Table 7 and Table 8, respectively.

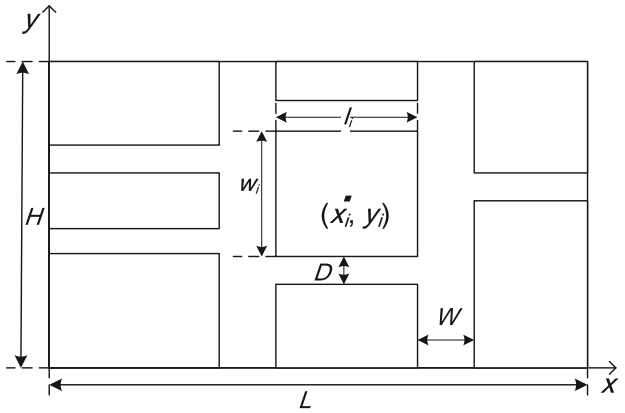


Fig. 9. A layout with vertical and horizontal aisles.

is set to 1 and the aspect ratio range of the facility is between 0.3 and 5.

To apply the CSE to solve the layout problem, we introduce the penalty function method in this experiment. The method is a generic constraint handling method, which adds the degree of constraint violation to punish a candidate solution.

Due to the order of the facility layout is along the direction from bottom to up and from left to right, there will be no facilities beyond layout area in y direction. We need only to determine whether facilities of the last column in the x direction are beyond the boundary of the region. So we use the following penalty function to deal with constraint in the model [51]:

$$A_k = \begin{cases} 1, & if \ \max\limits_{i=1,2,\ldots,n} (x_i + 0.5l_i) \le L \\ \mathrm{R}, & else \end{cases} \tag{15}$$

where A_k is a penalty factor for material handling costs and R is a big positive number. Assume F_{1k} is the objective function value about material flows for solution $\boldsymbol{X}_k$, then the objective about material flows with a penalty term is:

$$F'_{1k} = F_{1k} \times A_k \tag{16}$$

Because the optimization goal of material handling costs in the model is the minimum, in the iterative process, if any facility exceeds the boundary of the region, that is, a solution (layout) does not satisfy the non-overlapping constraint, it will be obliged to eliminate due to large objective value of material handling costs.

The Pareto-optimal solutions obtained by the CSE are in the number of 16 on a population of 50 individuals. Four of them are selected as optimal solutions by the Electre III [46]. The optimal objective values and the corresponding chromosomes by the CSE are given in Tables 11 and 12, respectively, in comparison with those by the NSGA-II that started the search from 100 randomly generated layouts based on the FBS. The corresponding layouts obtained by the CSE are shown in Fig. 10.

From Table 11, it is not hard to see that the results by the CSE can dominate all of the solutions by NSGA-II [51]. The optimal material handling cost obtained by the CSE reduces (1.625–1.569)/1.625×100%=3.446%, the best closeness and the best distance increase (22.7–14.8)/14.8×100%=53.378% and (25846—25330)/25330×100%=2.037%, respectively, compared with the best results obtained by the NSGA-II. Especially, for the objective of closeness, our results are much better than those in the literature. About the number of generations of iterations, optimal solutions are reached by the NSGA-II in approximately 500 iterations. The number of iterations for the CSE to find the optimal solution are about 400 iterations, which are obviously less than that of the NSGA-II. In [51], they did not report the running time of the MOGA. But from Table 11, it is not hard to see that the CSE finds the results in an acceptable time.

Table 9
Facility areas (m^2) in the second case.

Facility	1	2	3	4	5	6	7	8	9	10	11	12
Area	10000	60000	65000	20000	50000	20000	35000	20000	35000	15000	15000	10000

Table 10
Flows, closeness relation and distance requirements in the second case.

Facility	1	2	3	4	5	6	7	8	9	10	11	12
Flows												
2						1200	2200	500	4000			
3						2000	5000	700	10000			
4						1000	2500		3000			
5						1500	4000	300	6000			
6								200	2000			
7								200	6000			
8									500			
Closeness relation												
1											4	
9		3	3			3						
11		4	4	4	4							
12						4				2		
Distance requirements												
1		6	6	2	2			4	4			
10		2	2	2				2	2			

Table 11
Best solutions obtained by approaches CSE and NSGA-II for the second case.

Algorithm	Solution	$F_1(\boldsymbol{X})$ (min)	$F_2(\boldsymbol{X})$ (max)	$F_3(\boldsymbol{X})$ (max)	Iterations	Time (s)
CSE	1	1.581 E+07	21.4	25740	400	6781
	2	**1.569 E+07**	20.6	25441		
	3	1.613 E+07	21.6	**25846**		
	4	1.604 E+07	**22.7**	25535		
NSGA-II	1*	1.625 E+07	7.8	20137	500	–
	2*	2.117 E+07	14.8	17225		
	3*	1.752 E+07	8.6	25330		

Table 12
Chromosomes of best solutions by approaches CSE and NSGA-II for the second case.

Algorithm	Solution	Chromosome
CSE	1	{4.79,1.67,2.97,1.38,3.31,3.62,2.31,1.98,2.60,4.00,4.30,4.60}
	2	{4.77,1.68,2.64,1.38,3.04,3.30,2.08,3.71,2.43,4.03,4.29,4.57}
	3	{4.79,1.95,2.89,1.62,3.42,3.63,2.26,1.35,2.57,4.25,3.95,4.53}
	4	{4.78,1.78,2.02,1.37,3.73,3.37,2.62,3.02,2.39,4.27,3.97,4.55}
NSGA-II	1*	{1,5,3,9,7,12,6,8,4,10,11,2,2,3,4,3}
	2*	{3,9,7,8,2,6,4,11,1,10,12,5,3,2,4,3}
	3*	{4,2,8,7,9,3,12,5,11,10,6,1,3,3,3,3}

5.3. A set of benchmark instances with different number of facilities

In order to investigate further the validity of the CSE, six benchmark instances O8 from [52], Ba14 from [53], Ab20 from [54], SC30 and SC35 from [18], and Du62 from [55] are tested for optimizing objectives $F_1(\boldsymbol{X})$, $F_2(\boldsymbol{X})$ and $F_3(\boldsymbol{X})$. Dimensions of the workshop for this group of benchmark instances are shown in Table 13. The material flows between facilities and the area of each facility for these instances could be found in [16]. Due to very few benchmark problems for the UA-FLPs with multiple objectives can be found in the literature (especially for cases with closeness requests and distance requests), this paper generates test data for closeness relation and distance requirements as [37, 42], that is, some facilities are first chosen randomly and then a set of integers in the interval [1,6] are produced as closeness requests or distance requests of these facilities (details of produced values are shown in Tables A.1–A.6). What is more, these six benchmark instances also need to satisfy specific constraints that the maximum aspect ratio of each facility is 4 for O8, Ab20, SC35 and Du62, and the minimum side length of each facility is 1 for Ba14, and the maximum aspect ratio of each facility is 5 for SC30. The experimental results by the proposed CSE algorithm, the NSGA-II and the MOGA in the literature are shown in Table 14 for contrast, where the CSE and NSGA-II algorithms begin from the same search points based on 50 randomly produced layouts using the FBS while the MOGA starts from 50 randomly produced different initial layouts using the STS.

Table 14 shows that the best values (Best), average values (Avg) and standard deviations (SD) obtained by the CSE algorithm for objective functions $F_1(\boldsymbol{X})$, $F_2(\boldsymbol{X})$ and $F_3(\boldsymbol{X})$ are better than results of the NSGA-II and MOGA for all six instances. For example, the optimal material handling costs by the CSE for instance Du62 reduces (3423555.90–3267707.61)/3423555.90 × 100%= 4.6%, the best closeness and the best distance increase (1044.02–999.77)/

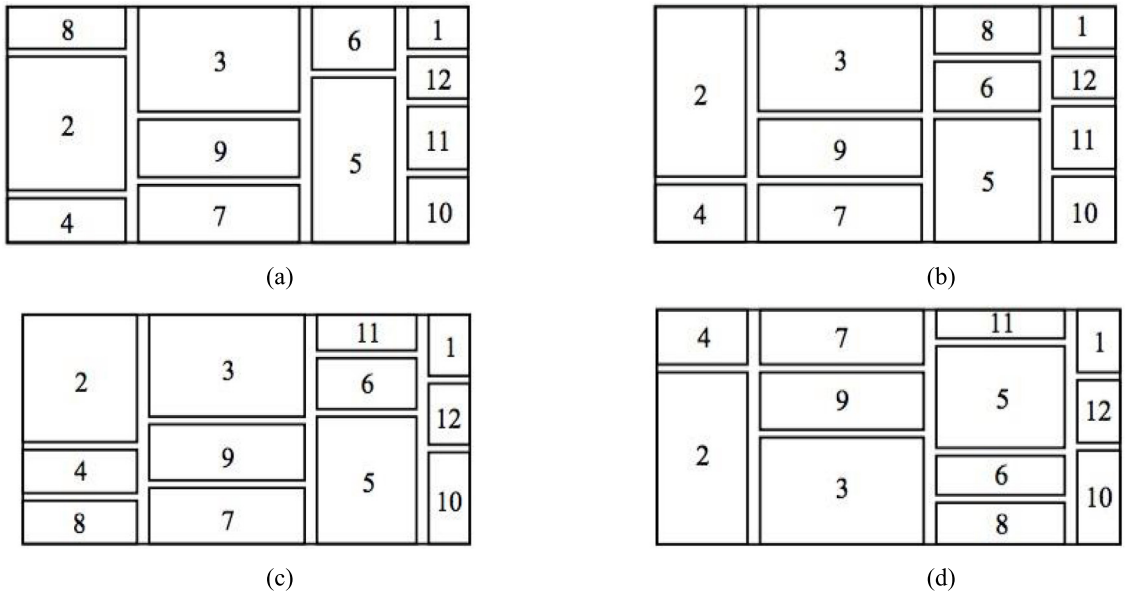


Fig. 10. Block layouts of the optimal solutions obtained by the CSE for the second case. Pictures (a), (b), (c) and (d) are four Pareto-optimal solutions in Table 11 and Table 12, respectively.

Table 13
Dimensions of the workshop for a set of benchmark instances with different number of facilities.

Instance	Number of facilities	Width of the workshop	Length of the workshop
O8	8	13	11.31
Ba14	14	10	6
Ab20	20	3	2
SC30	30	12	15
SC35	35	15	16
Du62	62	137.18	100

Table 14
Comparison of results by CSE, NSGA-II and MOGA for a set of benchmark instances with different number of facilities.

Instance	Algorithm	$F_1(\boldsymbol{X})$ (min)			$F_2(\boldsymbol{X})$ (max)			$F_3(\boldsymbol{X})$ (max)			Iterations (Scanning times)
		Best	Avg	SD	Best	Avg	SD	Best	Avg	SD	
O8	CSE	**221.28**	**231.61**	**6.05**	**108.73**	**86.35**	**11.59**	**52.11**	**42.78**	**10.68**	**400 (2 400 784)**
	NSGA-II	225.50	235.87	6.74	105.48	84.19	11.86	51.99	40.87	11.46	750 (4 102 015)
	MOGA	225.73	237.96	7.22	107.29	80.59	13.21	**52.11**	41.68	12.77	900 (5 793 380)
Ba14	CSE	**5134.05**	**5300.88**	**99.06**	**78.09**	**68.07**	**6.54**	**110.30**	**95.13**	**7.09**	**600 (4 676 996)**
	NSGA-II	5442.18	5604.76	115.00	72.75	59.99	9.56	103.33	89.80	8.07	1100 (6 831 310)
	MOGA	5587.63	5680.19	128.20	77.24	62.52	11.29	94.25	88.64	8.93	1800 (9 521 682)
Ab20	CSE	**3770.04**	**4303.35**	**284.88**	**13.57**	**9.24**	**2.29**	**36.14**	**31.48**	**3.04**	**1500 (10 959 006)**
	NSGA-II	4332.90	4898.75	318.55	11.42	6.77	2.30	32.05	28.64	3.84	3000 (16 680 015)
	MOGA	4412.32	4942.93	372.24	13.23	8.27	2.33	31.41	26.84	4.02	2500 (14 000 316)
SC30	CSE	**5295.77**	**6193.97**	**489.17**	**128.66**	**98.90**	**13.80**	**468.87**	**398.90**	**35.78**	**1800 (13 048 194)**
	NSGA-II	6483.01	7999.44	929.44	107.61	81.70	14.72	434.77	371.13	39.81	4500 (31 363 401)
	MOGA	6281.60	7343.72	832.43	110.32	80.14	15.79	446.25	368.78	39.92	5000 (35 261 297)
SC35	CSE	**6964.75**	**7929.88**	**1120.93**	**100.02**	**94.81**	**3.01**	**327.75**	**306.62**	**8.63**	**2100 (18 115 973)**
	NSGA-II	8645.38	11039.86	1178.10	89.89	63.78	15.96	322.88	280.68	21.13	6000 (33 806 821)
	MOGA	8827.41	9313.58	1145.88	91.42	68.35	16.32	318.37	273.35	19.35	8000 (41 241 394)
Du62	CSE	**3267707.61**	**3441469.94**	**77796.23**	**1044.02**	**872.85**	**106.25**	**10907.6**	**9605.36**	**620.36**	**3000 (24 110 602)**
	NSGA-II	3423555.90	3533971.02	84201.77	999.77	781.82	120.00	10514.8	9304.43	716.73	8000 (57 477 519)
	MOGA	3482942.48	3724182.37	100832.42	835.39	639.13	130.11	9990.53	9142.25	724.84	12000 (80 372 726)

$999.77 \times 100\%=4.4\%$, and $(10907.6-10514.8)/10514.8 \times 100\%=3.7\%$, respectively, compared with the best results obtained by the NSGA-II. For two larger instances SC35 and Du62, two layouts with the smallest $F_1(\boldsymbol{X})$ as the optimal solutions for preference which are found by the proposed CSE algorithm are shown in Fig. 11(a) and (b), respectively. In addition, the number of iterations and times of scanning objective functions to find the Pareto-optimal solutions in the CSE, NSGA-II and MOGA algorithms for each instance are also listed for comparison. It is obvious that the CSE can find optimal solutions with less scanning times of objective functions within less iterations than NSGA-II and MOGA. In Table 14, the best results are shown in **bold** font.

5.4. Performance metrics

Convergence to the true Pareto front and uniformity (or diversity) of the Pareto-optimal solutions are generally used as distinct goals for testing the performance of MOO approaches in

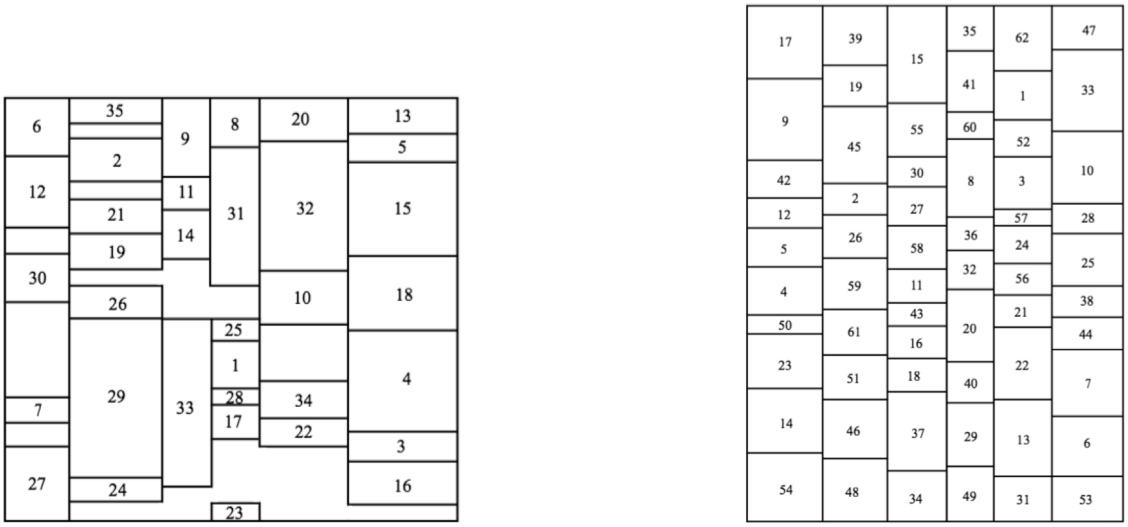


Fig. 11. Layouts with the smallest $F_1(\boldsymbol{X})$ as the optimal solutions for preference by the CSE algorithm for two larger instances. (a) Instance SC35 with $F_1(\boldsymbol{X})$=6964.75, $F_2(\boldsymbol{X})$=96.16, and $F_3(\boldsymbol{X})$=304.81. (b) Instance Du62 with $F_1(\boldsymbol{X})$=3267707.61, $F_2(\boldsymbol{X})$=964.87, and $F_3(\boldsymbol{X})$=8082.639.

MOPs. Generally speaking, some statistical analysis tools should be applied to analyze the quality of MOO approaches. In this paper, four commonly used performance metrics including Pareto Ratio (PR) [56], Generational distance (GD) [57], Space (SP) [58] and Overall Pareto Spread (OPS) [59] are first used to measure the convergence and uniformity of the CSE algorithm. Then, the non-parametric statistical analysis based on the Wilcoxon signed ranks test [60] is adopted to compare the performance of the proposed algorithm with those of the other algorithms.

PR metric [56] is used to compute the rate of production of non-dominated solutions at a given iteration of an optimization algorithm. PR is computed by the Eq. (17).

$$PR = \frac{|PPF(S)|}{|S|} \tag{17}$$

Here *PPF* is the set of the practical Pareto frontier that is close but different from the actual Pareto frontier of a problem. $|PPF(S)|$ represents the number of solutions of the *PPF* in the solutions set *S*. *S* is the set of all the solutions computed by the multi-objective optimization method at a given iteration, and $|S|$ represents the total number of solutions of *S*. The value of PR varies from 0 to 1 and the ideal value of a population is 1.

The GD proposed by Van et al. [57] is generally applied to evaluate the average distance between the Pareto-optimal solution set Q obtained by an algorithm and the actual Pareto-optimal solution set P^*. This distance is computed by Eq. (18).

$$GD = \frac{\sqrt{\sum_{i=1}^{|Q|} d_i^2}}{|Q|} \tag{18}$$

Here $d_i = \min_{j=1}^{|P^*|} \sqrt{\sum_{k=1}^{m}(\boldsymbol{F_k}(\boldsymbol{X_i}) - \boldsymbol{F_k^*}(\boldsymbol{X_j}))}$, $i = 1, 2, \ldots, |Q|$. $\boldsymbol{F_k^*}(\boldsymbol{X_j})$ means that the kth objective value of solution $\boldsymbol{X_j}$ in the Pareto-optimal solution set P^*. m is the number of objective functions and d_i denotes the Euclidean distance between each solution and the nearest one in the Pareto optimal set. The value of GD reflects the relative distance between the obtained solutions and the actual Pareto frontier. A smaller value of GD reveals a better convergence to the Pareto-optimal solution set and when GD = 0 all obtained solutions are located in the Pareto frontier.

The SP proposed by Schott [58] is another important metric tool which is generally used to evaluate the uniformity of the spread of the solutions obtained by an algorithm. The distance variance of each of the obtained solutions to its closest neighbor is computed by Eq. (19).

$$SP = \sqrt{\frac{1}{p-1}\sum_{i=1}^{p}(\bar{D} - D_i)^2} \tag{19}$$

where $D_i = \min_j \left(\sum_{k=1}^{m} \left|\boldsymbol{F}_k(\boldsymbol{X}_i) - \boldsymbol{F}_k(\boldsymbol{X}_j)\right|\right)$, $i, j = 1, 2, \ldots, p$, and p denotes the number of Pareto-optimal solutions by the optimization algorithm and $\bar{D}$ is the average of all D_i. The closer the found solutions are to the uniform distribution, the smaller the corresponding value of SP is. When SP = 0, all non-dominated solutions found are evenly distributed.

The OPS [59] is computed by Eq. (20). When the obtained solutions are distributed more widely, the value of OPS becomes larger.

$$OPS = \prod_{t=1}^{m} OPS_t \tag{20}$$

$$OPS_t = \frac{|\max_{\boldsymbol{X}\in P^*} F_t(\boldsymbol{X}) - \min_{\boldsymbol{X}\in P^*} F_t(\boldsymbol{X})|}{|F_t(P_B) - F_t(P_G)|} \tag{21}$$

Here P_B and P_G are the Utopia and Nadir design points [61], which are composed of the best and worst values of the objectives in the set of all the solutions, respectively. P^* is the Pareto-optimal solution set found.

Table 15 lists the average value of each performance measure by the CSE over 10 independent runs for case 1 and case 2, and it is clear that the convergence, uniformity and diversity distribution of the obtained solutions are satisfying. Table 16 lists the statistical results for these two test cases, including the best, the worst, the mean values, and the standard deviation for each objective function over 10 independent runs. For all objectives in these two cases, the standard deviations are small, indicating that the objective values obtained by the CSE are not scattered.

Table 15
The average value of each performance measure by the CSE over 10 independent runs for case 1 and case 2.

Test case	PR value	GD value	SP value	OPS value
Case 1	0.833	17.03	5.017	0.694
Case 2	0.698	112.59	9.163	0.581

Table 16
Statistics of results by the CSE for each objective function for case 1 and case 2.

Test case	Objective	Best value	Worst value	Average value	Standard deviation
Case 1	$F_1(X)$	4698	5258	4868.14	190.220
	$F_2(X)$	213	142	178.06	12.764
	$F_3(X)$	3333	2545	2754	175.872
	$F_4(X)$	0.81	0.65	0.69	0.021
Case 2	$F_1(X)$	1.569E+07	1.892 E+07	1.660E+07	0.112 E+07
	$F_2(X)$	22.7	18.3	19.667	1.063
	$F_3(X)$	25846	22184	24189.31	394.201

Table 17
The average value of each performance measure by the CSE, the NSGA-II and the MOGA over 10 independent runs for a set of benchmark instances with different number of facilities.

Instance	Algorithm	PR value	GD value	SP value	OPS value
O8	CSE	**0.822**	**2.55**	**1.216**	**0.795**
	NSGA-II	0.788	3.71	1.931	0.743
	MOGA	0.708	3.84	1.985	0.714
Ba14	CSE	**0.732**	**5.42**	**1.802**	**0.676**
	NSGA-II	0.690	8.63	3.874	0.669
	MOGA	0.633	8.67	4.051	0.639
Ab20	CSE	**0.810**	**13.11**	**4.118**	**0.775**
	NSGA-II	0.790	14.32	4.471	0.761
	MOGA	0.714	14.97	5.579	0.529
SC30	CSE	**0.697**	**16.41**	**6.028**	**0.615**
	NSGA-II	0.632	17.63	6.357	0.547
	MOGA	0.552	16.88	6.085	0.551
SC35	CSE	**0.616**	**16.25**	**5.131**	**0.521**
	NSGA-II	0.570	19.02	5.859	0.505
	MOGA	0.418	24.65	7.820	0.500
Du62	CSE	**0.608**	**77.03**	**6.183**	**0.571**
	NSGA-II	0.590	86.28	7.307	0.530
	MOGA	0.474	101.58	9.576	0.570

For a set of benchmark instances with different number of facilities, Table 17 lists the average value of the each performance measure by the proposed CSE algorithm, NSGA-II and MOGA in the literature over 10 independent runs for comparison. From Table 17 where the best results are shown in **bold** font, it is obvious that Pareto-optimal solutions found by the CSE have better convergence, uniformity and diversity than the results found by the NSGA-II and the MOGA.

The details of the final solutions obtained by the CSE in a running for each of two typical cases and a set of benchmark instances with different number of facilities are illustrated in Figs. 12 and 13, respectively, where all circles denote the gained feasible solutions and the asterisk points indicate the Pareto-optimal solutions obtained. From these figures, one can see that the final solutions by the CSE are well spread.

Besides the above performance measures, the non-parametric Wilcoxon signed ranks test [60] is also used to analyze the performances of the proposed algorithm and the other algorithms statistically. It is a pairwise test, which aims to detect a significant difference between the behaviors of two algorithms. From the statistical point of view, the test is safer since it does not assume normal distributions of samples. Also, the outliers (exceptionally good/bad performances of a few problems) have less effect on the Wilcoxon test than on the t-test. The Wilcoxon signed ranks test is described briefly in the following.

Assume that there are h test problems, let d_i denote the difference between the performance scores of the two algorithms on the ith test problem. Those ranks of $d_i = 0$ are neglected. Then, the rest of those differences are ranked according to their absolute value in ascending order. The sum of positive ranks and negative ranks are denoted as R^+ and R^-, respectively. If there exist equal ranks, those ranks are split evenly. Let h' be the number of test problems whose difference is not equal to 0, and $T = min(R^+, R^-)$. The null hypothesis will be rejected if the test statistic T is not greater than the value of the distribution of Wilcoxon for h' degrees of freedom. The normal approximation for the Wilcoxon T statistics is used for getting p-value. We use the R software packages to compute the p-value in this study. The level of significance α is assigned as 0.05, which indicates that if the p-value is smaller than α, there is a significant difference between the two algorithms.

In order to analyze the non-parametric statistical test of the performances of the proposed CSE algorithm and other two algorithms in the literature, NSGA-II is run more ten times for each of instances Case 2, O8, Ba14, Ab20, SC30, SC35, and Du62, MOGA for each of instances Case 1, O8, Ba14, Ab20, SC30, SC35, and Du62, and CSE for each of all eight instances. Statistical results by the Wilcoxon signed ranks test for CSE, NSGA-II and MOGA are given in Table 18, where CSE versus NSGA-II considers the average results of 20 independent runs for each of four performance measures on Case 2, O8, Ba14, Ab20, SC30, SC35, and Du62, while CSE versus MOGA on Case 1, O8, Ba14, Ab20, SC30, SC35, and Du62. When using the Wilcoxon's test in our experiments, the first step is to compute the R^+ and R^- related to the comparisons between CSE and the other two algorithms. Once they have been obtained, their associated p-values can be computed by the R software packages. It is obvious that for every comparison, there is the property $R^+ + R^- = h'*(h'+1)/2$. As Table 18 states, the CSE algorithm shows a significant improvement over the NSGA-II and MOGA algorithms, with a level of significance $\alpha = 0.05$.

To summarize the experimental results of three groups of instances from the literature for the UA-FLPs, the proposed CSE algorithm successfully optimizes four objectives including material handling costs (quantitative aspect), closeness requests, distance requests and aspect ratio satisfactions of facilities (qualitative aspects). The all properties (convergence and diversity) of the CSE algorithm overmatch those of other algorithms in the literature. Furthermore, the obtained Pareto-optimal layout set provides a wide range of choices for the decision makers who could express their preferences based on the production requirements or market circumstances.

5.5. Numerical experiments for the CSE under different parameters

In addition, the contrast experiments are designed for testing and verifying the effects of the parameters in the CSE. The testing parameters in the CSE include the initial configuration number k_1, the objective axis weight w_i, and the update frequency of d_{space}. In order to test the effects of parameters on the results of the experiments, we select some representative values to calculate the range of each parameter. The initial configuration number k_1 is set to 20, 50, 100 and 200, respectively, while w_i is fixed to 10^{-6} and the update frequency of d_{space} is fixed to 1/2 ($d_{space} = d_{space}/2$). Similarly, the objective axis weight w_i is set to 10^{-5}, 10^{-6}, 10^{-7} and 10^{-8}, respectively, while k_1 is fixed to 50 and the update frequency of d_{space} is fixed to 1/2. The update frequency of

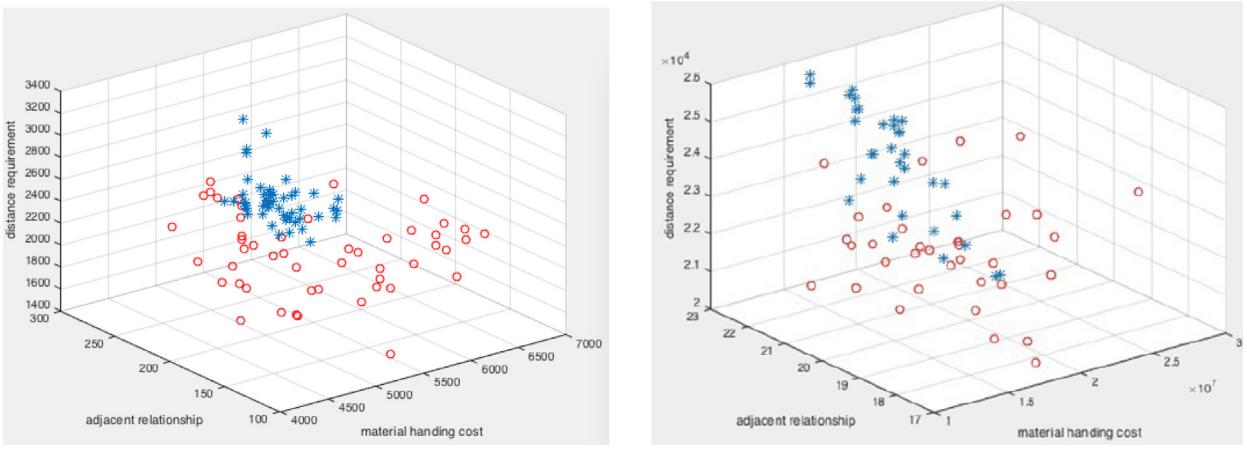


Fig. 12. Details of final solutions obtained by the CSE in a running for each of two typical cases. (a) Case 1. (b) Case 2.

Table 18
Statistical results by the Wilcoxon signed ranks test for CSE, NSGA-II and MOGA, where CSE versus NSGA-II considers the average results of 20 independent runs on Case 2, O8, Ba14, Ab20, SC30, SC35, and Du62 for each of four performance measures, while CSE versus MOGA on Case 1, O8, Ba14, Ab20, SC30, SC35, and Du62.

Metric name	PR-metric			GD-metric			SP-metric			OPS-metric		
CSE vs.	R^+	R^-	p-value	R^+	R^-	p-value	R^+	R^-	p-value	R^+	R^-	p-value
NSGA-II	28	0	0.01563	28	0	0.01563	27	1	0.03125	26	2	0.04688
MOGA	28	0	0.01563	28	0	0.01563	26	2	0.04688	28	0	0.01563

Table 19
Computational results obtained by the CSE algorithm with different initial configuration number k_1 when w_i is fixed to 10^{-6} and the update frequency of d_{space} is fixed to 1/2 ($d_{space} = d_{space}/2$).

Instance	k_1	$F_1(X)$ (min)			$F_2(X)$ (max)			$F_3(X)$ (max)			Time (s)
		Best	Avg	SD	Best	Avg	SD	Best	Avg	SD	
O8	20	223.39	237.45	7.38	103.18	83.11	12.04	52.11	40.05	11.39	429
	50	221.28	232.07	6.01	108.73	86.74	11.43	52.11	42.63	10.61	731
	100	221.28	233.91	7.02	108.73	85.39	11.81	52.11	42.61	10.72	1255
	200	221.28	233.74	6.67	108.73	84.88	12.64	52.11	41.36	11.12	1807
SC30	20	5337.90	6272.57	511.92	112.73	97.40	15.42	462.97	396.64	38.20	5480
	50	5295.77	6199.24	488.58	128.66	99.34	13.72	468.87	396.51	35.34	8316
	100	5295.77	6198.06	493.84	128.66	95.55	14.36	468.87	393.18	35.81	8675
	200	5295.77	6184.48	507.29	128.66	97.74	15.10	468.87	394.15	36.62	10082
Du62	20	3267980.66	3442095.68	82037.16	997.04	854.00	109.25	10864.46	9569.17	632.69	12400
	50	3267707.61	3441469.94	77796.23	1044.02	872.85	106.25	10907.60	9605.36	620.36	15280
	100	3267822.71	3441486.76	77852.29	1037.33	869.95	112.77	10874.03	9535.53	632.98	18965
	200	3267831.73	3441535.20	78069.96	1042.58	861.64	107.32	10891.23	9592.36	626.85	21114

Table 20
Computational results obtained by the CSE algorithm with different objective axis weight w_i when k_1 is fixed to 50 and the update frequency of d_{space} is fixed to 1/2 ($d_{space} = d_{space}/2$).

Instance	w_i	$F_1(X)$ (min)			$F_2(X)$ (max)			$F_3(X)$ (max)			Time (s)
		Best	Avg	SD	Best	Avg	SD	Best	Avg	SD	
O8	10^{-5}	221.28	235.58	6.81	108.73	85.12	12.62	52.11	40.89	11.85	722
	10^{-6}	221.28	232.07	6.01	108.73	86.74	11.43	52.11	42.63	10.61	731
	10^{-7}	221.28	233.39	6.53	108.73	86.41	11.68	52.11	41.43	11.34	728
	10^{-8}	221.28	231.42	6.27	108.73	85.94	11.44	52.11	42.01	11.26	735
SC30	10^{-5}	5323.65	6208.17	491.39	126.45	97.10	14.16	463.68	384.20	39.38	7503
	10^{-6}	5295.77	6199.24	488.58	128.66	99.34	13.72	468.87	396.51	35.34	8316
	10^{-7}	5301.90	6196.29	486.15	127.05	97.67	15.01	465.37	393.38	37.58	9575
	10^{-8}	5362.67	6213.06	503.61	121.58	96.14	15.21	462.43	387.78	41.04	10506
Du62	10^{-5}	3267860.41	3441538.85	77946.69	1021.90	867.96	109.60	10886.16	9538.91	622.56	15138
	10^{-6}	3267707.61	3441469.94	77796.23	1044.02	872.85	106.25	10907.60	9605.36	620.36	15280
	10^{-7}	3267810.91	3441470.23	77801.60	1030.13	867.70	119.14	10896.63	9524.73	631.69	16486
	10^{-8}	3267831.73	3441511.49	78963.05	1028.75	870.30	125.96	10821.23	9584.11	631.98	17782

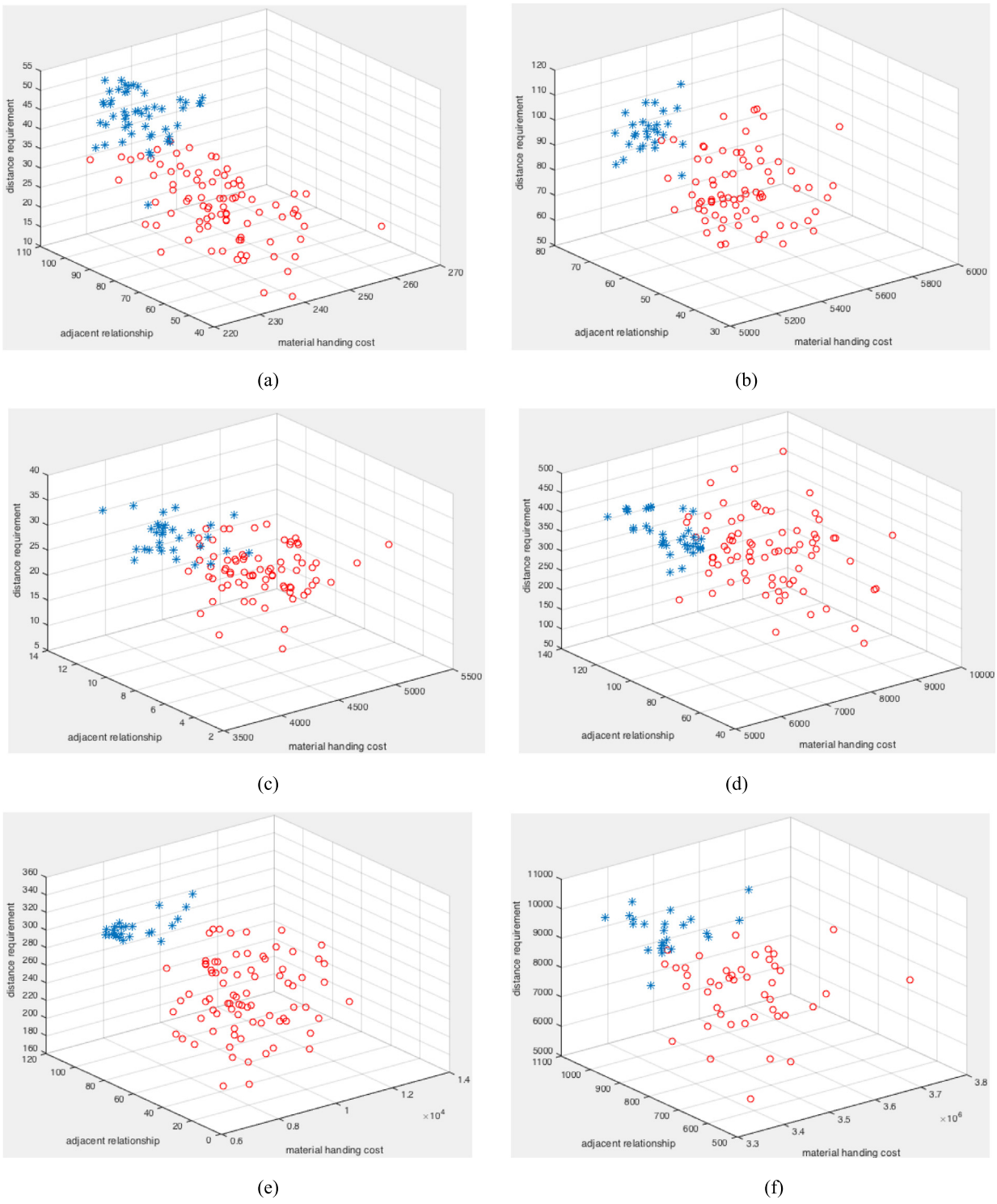


Fig. 13. Details of final solutions obtained by the CSE algorithm in a running for each of a set of benchmark instances with different number of facilities. (a) Instance O8. (b) Instance Ba14. (c) Instance Ab20. (d) Instance SC30. (e) Instance SC35. (f) Instance Du62.

d_{space} is set to 2/3, 1/2, 1/3 and 1/5, respectively, while k_1 is fixed to 50 and w_i is fixed to 10^{-6}. The maximum number of iterations is set as 3000 to ensure that the algorithm reaches convergence and the optimal solution of each objective can be obtained. The algorithm is run 10 times independently for each parameter of three representative instances O8, SC30 and Du62. The best values (Best), the average values (Avg), the standard deviations (SD) and the average of running time of the algorithm with different parameters k_1, w_i and the update frequency of d_{space} are listed in Table 19, Table 20 and Table 21, respectively.

Table 19 shows that for instances O8, SC30 and Du62, the best values, the average values, the standard deviations for all objectives are slightly different when the number k_1 of initial configurations is set to 50, 100 and 200, respectively. But with the increase of k_1, the average running time will also increase, especially for instances SC30 and Du62. While k_1 is set to 20,

Table 21
Computational results obtained by the CSE algorithm with different the update frequency of d_{space} when k_1 is fixed to 50 and w_i is fixed to 10^{-6}.

Instance	d_{space}	$F_1(X)$ (min)			$F_2(X)$ (max)			$F_3(X)$ (max)			Time (s)
		Best	Avg	SD	Best	Avg	SD	Best	Avg	SD	
O8	2/3	221.28	232.57	7.04	108.73	87.19	11.88	52.11	41.31	11.17	789
	1/2	221.28	232.07	6.01	108.73	86.74	11.43	52.11	42.63	10.61	731
	1/3	221.28	233.14	6.96	108.73	84.27	12.35	52.11	43.07	11.06	728
	1/5	221.28	237.50	7.82	108.73	81.03	12.69	52.11	40.92	13.26	722
SC30	2/3	5295.77	6225.01	561.58	128.66	97.45	19.00	468.87	380.00	38.02	10778
	1/2	5295.77	6199.24	488.58	128.66	99.34	13.72	468.87	396.51	35.34	8316
	1/3	5313.36	6262.55	532.15	127.45	97.83	18.86	461.78	396.45	36.15	8037
	1/5	5382.34	6214.33	698.05	114.88	93.06	14.91	452.36	397.94	46.89	7337
Du62	2/3	3267822.71	3441561.10	82820.41	1036.88	877.04	137.05	10783.42	9547.41	637.01	17308
	1/2	3267707.61	3441469.94	77796.23	1044.02	872.85	106.25	10907.60	9605.36	620.36	15280
	1/3	3267884.56	3441632.85	79731.53	1027.59	849.19	117.44	10880.58	9578.45	651.62	14709
	1/5	3268761.69	3441668.66	93069.13	1033.91	834.34	162.30	10708.27	9594.75	702.05	14157

Table A.1
Closeness relation and distance requirements between two facilities for instance O8.

Facility	1	2	4	5	6	7	8
Closeness relation							
1			5	6			2
2			4	3			1
5							3
Distance requirements							
6						4	

Table A.2
Closeness relation and distance requirements between two facilities for instance Ba14.

Facility	1	2	3	4	5	7	8	10	12	13	14
Closeness relation											
1		3	5	4	4						
2			3	3			2				
3				3	4		1				
5							2				
Distance requirements											
7										2	1
10										1	3
12											3
13											3

Table A.3
Closeness relation and distance requirements between two facilities for instance Ab20.

Facility	2	3	4	5	6	7	8	9	10	13	14	18	19
Closeness relation													
2			5			3							
3											3		4
4													3
7							5		2				
8													3
10											4		
Distance requirements													
5					2			1		3		2	
6								2		1			
9										1		1	
13												2	

Table A.4
Closeness relation and distance requirements between two facilities for instance SC30.

Facility	1	2	3	4	7	8	9	10	11	12	13	14	15	17	18	25	29	30
Closeness relation																		
3				6														
4													4		5		5	
8							3											
9										4								
10										4								
12													5					
15															5			
25																	2	
29																		5
Distance requirements																		
1		4			1				3		2	1		2				
2					1				2		3	1		1				
7									1		1	3		2				
11											2	1		3				
13												3		2				
14														1				

the results are worse than results of other values. For the evolutionary algorithm, the production of the initial configurations is important, because the subsequent crossover and mutation are all based on the initial configurations for evolution and updation. When k_1 is set to 20, it is easy to result in the premature convergence of the algorithm and the probability of obtaining the optimal solution is lower because of the smaller number of initial configurations. When k_1 is 100 or 200, although the diversity of initial configurations is increased the results do not turn better, furthermore the running time is greatly increased. Therefore, on the whole, when k_1 is set to 50, the results of the algorithm are slightly better than those of other k_1 values.

As can be seen from Table 20, the change of objective axis weight w_i has little influence on the results of instances O8, SC30 and Du62. But, on the whole, when w_i is set to 10^{-6}, the results of the algorithm are slightly better than those of the algorithm with other w_i values. For instance O8, the optimal solution of each objective can be obtained under the four values of w_i. For instances SC30 and Du62, one can see that although the best values, the average values, and the standard deviations for all objectives of these two instances have little difference for different parameters of w_i, as the decrease of w_i, because the accuracy increases gradually, the running time of the algorithm is longer accordingly.

From Table 21, for the small instance O8, when the update frequency of d_{space} is 2/3, 1/2, 1/3 and 1/5, the best values, the average values, the standard deviations of the three objectives and the average running time are all similar. For two larger instances SC30 and Du62, when the update frequency of d_{space} is 2/3, 1/2 and 1/3, the optimal values of the three objectives obtained are similar. However, when the update frequency of d_{space} is 2/3, the running time of the algorithm is relatively longer. When it becomes 1/5, the results of the algorithm for instances SC30 and Du62 are slightly worse than those by the algorithm

Table A.6
Closeness relation and distance requirements between two facilities for instance Du62.

Facility	4	5	13	16	18	19	20	21	22	29	31	34	35	37	38	50	53	56	57	59	60	61	62
Closeness relation																							
4		5	5	4	3		3		2	3						2		4		1			
5				3	2			2	1	3				5	4			3		5			
13				4			5		5	5					2			3				4	
16					5		3	6	2	4				2	4	4		4		3		3	
18							5	4	3	4				4				6		5		5	
20								5	3	2				1	6	1		4		2		1	
21									6	1				2	2	5		2		3		5	
22										1				2	4	4		3		4		3	
29														6	1	4		3		5		2	
37																3		4		2		5	
38																		1		3		5	
50																		1		2		4	
56																				3		6	
59																						5	
Distance requirements																							
19											4	3	5				6				2		4
31												5	4				4		5		3		1
34													6				4		4		5		5
35																			4				
53																			5		4		4
57																					6		
60																							5

Table A.5
Closeness relation and distance requirements between two facilities for instance SC35.

Facility	1	3	4	7	9	13	15	18	23	24	25	26	27	28	29	32	33
Closeness relation																	
1										1	2	1		1			
4							4	5									
15								5								4	
24															2		
25															3		
26															1		
27															2		
28															2		
29																	5
Distance requirements																	
3				2	3	1			2				1				
7					1	2			1				1				
9						1			4								
13									2				3				

with other three update frequencies. When the update frequency of d_{space} is 2/3, the solution space shrinks slowly, and in turn the algorithm converges slowly so that the running time of the algorithm is longer. On the contrary, when the update frequency of d_{space} is 1/5, the solution space shrinks quickly without full update so that the algorithm is easy to premature converges, causing the running time becomes shorter and the results become worse accordingly.

The experimental results of test instances illustrate the effectiveness of the settings of parameters in the CSE. In order to make the algorithm find the Pareto-optimal solutions efficiently, we generally set the parameters in a certain range, for example, initial configuration number $k_1 \in [50, 200]$, the objective axis weight $w_i \in [10^{-5}, 10^{-7}]$, the update frequency of $d_{space} \in [1/3, 2/3]$. In this paper, the parameters k_1, w_i and the update frequency of d_{space} are set to 50, 10^{-6} and 1/2, respectively.

6. Conclusions

Reasonable facilities layout of enterprises can effectively improve operational efficiency and reduce operating costs. Nowadays, most of classical methods focus on the equal-area FLPs. But, in real-world situations, the layout with unequal-area facilities is more practical because of the geometric constraints of the equal-area ones. In addition, most of the existing approaches study either the quantitative single objective FLPs or multi-objective FLPs which are converted into single objective problem by using weighted coefficients. In comparison, this paper involves the investigation of static multi-objective UA-FLPs based on Pareto optimization method. Three sets of multi-objective layout instances whose objectives refer to material handling costs (quantitative aspect), closeness requests, distance requests and satisfaction of aspect ratio of the facility (qualitative aspects) are introduced.

For multi-objective UA-FLPs, the topic of MOO algorithms approached by GAs (such as MOGA, NSGA-II) is nowadays one of the most promising and widely investigated research field. These approaches are capable of finding a set of Pareto-optimal layouts that optimizes the multiple objective functions simultaneously throughout the entire evolutionary process, giving the decision maker a restricted number of solutions among which one can choose that he considers the best. However, the convergence and diversity of traditional MOO algorithms approached by GAs largely depend upon the genetic operators. They have generally shortcomings, such as slow convergence to the Pareto front and low efficient selection toward diversity of solutions, especially when dealing with the UA-FLPs with three or more objectives. In this paper, a novel MOO algorithm based on GA(EA), called configuration space evolutionary (CSE) algorithm which can search the solution space efficiently and find good spread of solutions, is developed to solve the static UA-FLPs with multiple objectives. In the CSE, we introduce a measure of the radius of the configuration bank, the value of which is gradually reduced until the algorithm attains the convergence. Considering the special characteristics of the problem, new encoding, selection, crossover, and mutation operators are used to produce new individuals in the course of evolution. Also, to get a good spread to Pareto front and maintain diversity of the obtained solutions, we adopt a method of the nearest and farthest candidate solution based on objective function normalization and combine it with the fast non-dominated sorting to choose the Pareto-optimal solutions. The experimental results on eight typical cases show that the proposed CSE algorithm is an effective MOO algorithm for solving the UA-FLPs.

From the executing process of the CSE, it is not hard to see that the proposed CSE algorithm is easy to be extended to investigate other multi-objective or even many-objective optimization problems in various real-world scenarios. For further work, this study can be expanded in many ways. It can be extended to add the input/output points for facilities. It is also suggested that other MOO algorithms can be utilized to solve the same problems. Furthermore, in today's market, with increasing global competition and short life cycle of production, material flows among facilities change during the planning horizon. Hence, future research should be directed toward dynamic FLPs rather than static FLPs. Along this research line, the idea of the proposed algorithm will be applied to solve the multi-objective or even many-objective unequal-area dynamic FLPs with variable shapes and areas of facilities throughout the time horizon.

Declaration of competing interest

No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.asoc.2019.106052.

Acknowledgments

This work is supported by the Natural Science Foundation of Jiangsu Province (Grant No. BK20181409), the Major Program of the National Social Science Foundation of China (Grant No. 16ZDA047), the Special Foundation of Guangzhou Key Laboratory of Multilingual Intelligent Processing, China (Grant No. 201905010008), and the Program of Basic and Applied Basic Research of Guangzhou, China.

Appendix

See Tables A.1–A.6.

References

[1] B. Marcello, Z. Simone, Z. Lucio, Layout design in dynamic environments: Strategies and quantitative indices, Int. J. Prod. Res. 41 (5) (2003) 995–1016.
[2] H. Pourvaziri, B. Naderi, A hybrid multi-population genetic algorithm for the dynamic facility layout problem, Appl. Soft Comput. 24 (24) (2014) 457–469.
[3] J.A. Tompkins, J.A. White, Y.A. Bozer, E.H. Frazelle, J.M.A. Tanchoco, J. Trevino, Facilities Planning, Wiley, 1996.
[4] J.F. Liu, D.W. Wang, K. He, Y. Xue, Combining Wang-Landau sampling algorithm and heuristics for solving the unequal-area dynamic facility layout problem, European J. Oper. Res. 262 (1) (2017) 1052–1063.
[5] K.S.N. Ripon, K. Glette, O. Mirmotahari, M. Hovin, J. Torresen, Pareto optimal based evolutionary approach for solving multi-objective facility layout problem, in: Proceedings of the 16th International Conference on Neural Information Processing, ICONIP, Berlin, Heidelberg, 2009, pp. 159–168.
[6] J.F. Liu, H.Y. Zhang, K. He, S.Y. Jiang, Multi-objective particle swarm optimization algorithm based on objective space division for the unequal-area facility layout problem, Expert Syst. Appl. 102 (2018) 179–192.
[7] A. Gómez, Q.I. Fernández, D.D.L.F. Garcıa, P.J. Garcıa, Using genetic algorithms to resolve layout problems in facilities where there are aisles, Int. J. Prod. Econ. 84 (3) (2003) 271–282.
[8] Q. Liu, R.D. Meller, A sequence-pair representation and MIP-model-based heuristic for the facility layout problem with rectangular departments, IIE Trans. 39 (4) (2007) 377–394.
[9] J.F. Gonçalves, M.G.C. Resende, A biased random-key genetic algorithm for the unequal area facility layout problem, European J. Oper. Res. 246 (1) (2015) 86–107.
[10] F.G. Paes, A.A. Pessoa, T. Vidal, A hybrid genetic algorithm with decomposition phases for the unequal area facility layout problem, European J. Oper. Res. 256 (3) (2017) 742–756.
[11] B.H. Ulutas, S. Kulturel-Konak, An artificial immune system based algorithm to solve unequal area facility layout problem, Expert Syst. Appl. 39 (5) (2012) 5384–5395.
[12] L. García-Hernández, J.M. Palomo-Romero, L. Salas-Morera, A. Arauzo-Azofra, H. Pierreval, A novel hybrid evolutionary approach for capturing decision maker knowledge into the unequal area facility layout problem, Expert Syst. Appl. 42 (10) (2015) 4697–4708.
[13] J.M. Palomo-Romero, L. Salas-Morera, L. García-Hernández, An island model genetic algorithm for unequal area facility layout problems, Expert Syst. Appl. 68 (2017) 151–162.
[14] M.S. Chang, T.C. Ku, A slicing tree representation and QCP-model-based heuristic algorithm for the unequal-area block facility layout problem, Math. Probl. Eng. 2013 (4) (2013) 1–19.
[15] S. Kang, J. Chae, Harmony search for the layout design of an unequal area facility, Expert Syst. Appl. 79 (2017) 268–281.
[16] K.Y. Komarudin, Applying ant system for solving unequal area facility layout problems, European J. Oper. Res. 202 (3) (2010) 730–746.
[17] J. Guan, G. Lin, Hybridizing variable neighborhood search with ant colony optimization for solving the single row facility layout problem, European J. Oper. Res. 248 (3) (2016) 899–909.
[18] D. Scholz, A. Petrick, W. Domschke, Stats: A slicing tree and tabu search based heuristic for the unequal area facility layout problem, European J. Oper. Res. 197 (1) (2009) 166–178.
[19] S. Kulturel-Konak, A. Konak, A large-scale hybrid simulated annealing algorithm for cyclic facility layout problems, Eng. Optim. 47 (7) (2015) 963–978.
[20] I.B. Hunagund, V.M. Pillai, U.N. Kempaiah, A simulated annealing algorithm for unequal area dynamic facility layout problems with flexible bay structure, Int. J. Ind. Eng. Comput. 9 (3) (2018) 307–330.
[21] A.D. Asl, K.Y. Wong, Solving unequal-area static and dynamic facility layout problems using modified particle swarm optimization, J. Intell. Manuf. 28 (2015) 1317–1336.
[22] H.L. Wei, N.A.M. Isa, Bidirectional teaching and peer-learning particle swarm optimization, Inform. Sci. 280 (4) (2014) 111–134.
[23] Y. Luo, Y.P. Waden, The improved ant colony optimization algorithm for MLP considering the advantage from relationship, Math. Probl. Eng. 2017 (2017) 1–11.
[24] Z.H. Hu, A multi-objective immune algorithm based on a multiple-affinity model, European J. Oper. Res. 202 (1) (2010) 60–72.
[25] Y.N. Guo, P. Zhang, J. Cheng, C. Wang, D.W. Gong, Interval multi-objective quantum-inspired cultural algorithms, Neural Comput. Appl. 30 (3) (2018) 709–722.
[26] J. Cheng, J.J. Chen, Y.N. Guo, S. Cheng, L.K. Yang, P. Zhang, Adaptive CCR-ELM with variable-length brain storm optimization algorithm for class-imbalance learning, Nat. Comput. (2019) http://dx.doi.org/10.1007/s11047-019-09735-9.
[27] C.M. Fonseca, P.J. Fleming, An overview of evolutionary algorithms in multi-objective optimization, Evol. Comput. 3 (1) (2014) 1–16.
[28] R. Wang, P.J. Fleming, R.C. Purshouse, General framework for localised multi-objective evolutionary algorithms, Inform. Sci. 258 (3) (2014) 29–53.
[29] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multi-objective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6 (2) (2002) 182–197.
[30] R. Wang, R.C. Purshouse, P.J. Fleming, Preference-inspired co-evolutionary algorithms for many-objective optimization, IEEE Trans. Evol. Comput. 17 (4) (2013) 474–494.
[31] R. Wang, M.M. Mansor, R.C. Purshouse, P.J. Fleming, An analysis of parameter sensitivities of preference-inspired co-evolutionary algorithms, Internat. J. Systems Sci. 46 (13) (2015) 2407–2420.
[32] S. Yang, M. Li, X. Liu, J. Zheng, A grid-based evolutionary algorithm for many-objective optimization, IEEE Trans. Evol. Comput. 17 (5) (2013) 721–736.
[33] M. Li, S. Yang, X. Liu, Shift-based density estimation for pareto-based algorithms in many-objective optimization, IEEE Trans. Evol. Comput. 18 (3) (2014) 348–365.
[34] J. Bader, E. Zitzler, Hype: An algorithm for fast hypervolume-based many-objective optimization, Evol. Comput. 19 (1) (2014) 45–76.
[35] T.C. Wang, R.T. Liaw, C.K. Ting, MOEA/D using covariance matrix adaptation evolution strategy for complex multi-objective optimization problems, in: Proceedings of 2016 IEEE Congress on Evolutionary Computation, CEC, IEEE, Vancouver, BC, Canada, 2016;, pp. 983–990.
[36] K. Deb, H. Jain, An evolutionary many-objective optimization algorithm using reference-point-based non-dominated sorting approach, part I: Solving problems with box constraints, IEEE Trans. Evol. Comput. 18 (4) (2014) 577–601.
[37] J.F. Liu, J. Liu, Applying multi-objective ant colony optimization algorithm for solving the unequal area facility layout problems, Appl. Soft Comput. 74 (2019) 167–189.
[38] X.Q. Zuo, C.S. Murray, A.E. Smith, Solving an extended double row layout problem using multiobjective tabu search and linear programming, IEEE Trans. Autom. Sci. Eng. 11 (2014) 1122–1132.
[39] A. Saraswat, U. Venkatadri, I. Castillo, A framework for multi-objective facility layout design, Comput. Ind. Eng. 90 (2015) 167–176.

[40] G. Aiello, G.L. Scalia, M. Enea, A multi objective genetic algorithm for the facility layout problem based upon slicing structure encoding, Expert Syst. Appl. 39 (12) (2012) 10352–10358.
[41] A.P. Li, Z.Y. Yan, N. Xie, J.Z. Huang, A research of multi-objective facility layout based on NSGA-? Mach. Des. Research 28 (3) (2012) 90–95.
[42] K.S.N. Ripon, K. Glette, K.N. Khan, M. Hovin, J. Torresen, Adaptive variable neighborhood search for solving multi-objective facility layout problems with unequal area facilities, Swarm Evol. Comput. 8 (1) (2013) 1–12.
[43] L. García-Hernández, M. Pérez-Ortiz, A. Araúzo-Azofra, L. Salas-Morera, C. Hervás-Martínez, An evolutionary neural system for incorporating expert knowledge into the UA-FLP, Neurocomputing 135 (2014) 69–78.
[44] S. Vitayasak, P. Pongcharoen, C. Hicks, A tool for solving stochastic dynamic facility layout problems with stochastic demand using either a genetic algorithm or modified backtracking search algorithm, Int. J. Prod. Econ. 190 (2017) 146–157.
[45] T. Irohara, H. Yamashita, Y. Ishizuka, Facility layout problem with buffer space allocation for throughput and material handling cost, J. Japan Ind. Manage. Assoc. 58 (2) (2017) 87–96.
[46] C. Giannoulis, A. Ishizaka, A web-based decision support system with electre III for a personalised ranking of British universities, Decis. Support Syst. 48 (3) (2010) 488–497.
[47] S. Kukkonen, K. Deb, Improved pruning of non-dominated solutions based on crowding distance for bi-objective optimization problems, in: Proceedings of 2006 IEEE Congress on Evolutionary Computation, 2006, pp. 1179–1186.
[48] S. Kukkonen, K. Deb, A fast and effective method for pruning of non-dominated solutions in many-objective problems, in: Proceedings of International Conference on Parallel Problem Solving from Nature, Springer-Verlag, 2006, pp. 553–562.
[49] H. Jain, K. Deb, An evolutionary many-objective optimization algorithm using reference-point based non-dominated sorting approach, Part II: Handling constraints and extending to an adaptive approach, IEEE Trans. Evol. Comput. 18 (4) (2014) 602–622.
[50] G. Aiello, G.L. Scalia, M. Enea, A non dominated ranking multi objective genetic algorithm and electre method for unequal area facility layout problems, Expert Syst. Appl. 40 (12) (2013) 4812–4819.
[51] Z. Chang, J. Lu, Analysis of multi-objective facility layout problem using flexible bay structure, Oper. Res. Manage. Sci. 24 (2) (2015) 128–134.
[52] R.D. Meller, V. Narayanan, P.H. Vance, Optimal facility layout design, Oper. Res. Lett. 23 (3–5) (1998) 117–127.
[53] D.J. Van Camp, A Nonlinear Optimization Approach for Solving Facility Layout Problem (thesis), University of Toronto, Canada, 1989.
[54] G.C. Armour, E.S. Buffa, A heuristic algorithm and simulation approach to relative allocation of facilities, Manage. Sci. 9 (2) (1963) 294–309.
[55] T. Dunker, G. Radons, E. Westkämper, A coevolutionary algorithm for a facility layout problem, Int. J. Prod. Res. 41 (15) (2003) 3479–3500.
[56] Y. Collette, P. Siarry, Three new metrics to measure the convergence of metaheuristics towards the Pareto frontier and the aesthetic of a set of solutions in bi-objective optimization, Comput. Oper. Res. 32 (2005) 773–792.
[57] D.A. Van, V. Gary, B. Lamont, Multi-objective evolutionary algorithm research: A history and analysis, Evolutionary Computation 8 (2) (1998) 125–147.
[58] J.R. Schott, Fault tolerant design using single and multi-criteria genetic algorithm optimization, Cell. Immunol. 37 (1) (1995) 1–13.
[59] K. Zheng, R.-J. Yang, H.Y. Xu, J. Hu, A new distribution metric for comparing pareto optimal solutions, Struct. Multidiscip. Optim. 55 (2017) 53–62.
[60] D.J. Sheskin, Handbook of Parametric and Nonparametric Statistical Procedures, third ed., Chapman & Hall/CRC, 2003, pp. 609–632.
[61] M. Makowski, Multi-objective decision support including sensitivity analysis, Encycl. Life Support Syst. (2004) 1–24.