RAIRO-Oper. Res. 59 (2025) 3649–3674
https://doi.org/10.1051/ro/2025104

RAIRO Operations Research
www.rairo-ro.org

METAHEURISTICS AND A MATHEURISTIC FOR SOLVING THE FACILITY LAYOUT PROBLEM IN THE PRESENCE OF ALTERNATIVE PROCESS PLANS AND MACHINE REDUNDANCY

Mehdi A. Kamran[1,*], Maghsud Solimanpur[2], Reza Atefi[3], Nooshin Atashfeshan[1] and Yalda Mansouri[2]

Abstract. This research tackles a crucial aspect of manufacturing system design: optimizing the Facility Layout Problem (FLP). We address a specific scenario involving multiple products with flexible processing plans on various machines in a job-shop environment. Redundant machines of each type exist, with known acquisition costs and capacities. Processing times and production volumes for each product are also pre-determined. An integer non-linear mathematical model is formulated to represent the problem. While a linearization technique is applied, the inherent NP-hardness renders exact solution methods impractical for medium to large-scale problems. To address this, three algorithms are proposed: a matheuristic, Iterated Local Search (ILS), and a Genetic Algorithm (GA). These are evaluated based on solution quality, runtime, and robustness across diverse problem instances. Results demonstrate the superiority of the ILS algorithm in terms of solution quality, robustness, and overall effectiveness. These findings offer valuable guidance for decision-makers seeking optimization tools for FLPs. The ILS's consistent delivery of high-quality solutions with minimal variation makes it a reliable choice. Additionally, as many facility layout decisions are tactical or strategic – where computational time is less critical – the matheuristic demonstrates acceptable performance and holds promise for handling problems of varying sizes and complexities. To further validate the effectiveness and demonstrate the practical applicability of our proposed solution methodology, the ILS and matheuristic algorithms were applied to a real-world layout design case adapted from the literature. The results once again confirm the strong performance of both methods in terms of solution quality, computational efficiency, and robustness.

Mathematics Subject Classification. 90B80, 90B06, 90C59, 90C26.

Received May 10, 2024. Accepted July 20, 2025.

Keywords. Facility layout problem, alternative process plans, machine redundancy, matheuristic algorithm, genetic algorithm, iterated local search.

[1] Department of Logistics, Tourism, and Services Management, German University of Technology in Oman, P. O. Box 1816, PC 130, Muscat, Oman.

[2] Faculty of Engineering, Urmia University, Urmia, Iran.

[3] Department of Economics and Management, University of Brescia, Brescia, Italy.

*Corresponding author: mehdi.kamran@gutech.edu.om

Research Highlights

- Studied the FLP with multiple products and machines in a job-shop system.
- Formulated a linearized integer non-linear mathematical programming model.
- Considered alternative process plans, machine redundancy, capacity, and cost.
- Proposed matheuristic, ILS, and GA algorithms for efficient solving.
- ILS and matheuristic excel in solution quality, robustness, and effectiveness.

1. Introduction

A facility layout is an arrangement of facilities needed to produce goods or delivery of services. A facility is an entity that facilitates the performance of a job. It may be a machine tool, a work center, a manufacturing cell, a machine shop, a department, a warehouse, etc. [26]. Appropriate placement of facilities contributes to the overall efficiency of operations and can reduce the total operating expenditures [74].

The facility layout problem (FLP) is investigated in different production systems, namely, classifying into four systems of Job-Shop production, Flow-Shop production, Cellular Manufacturing System (or Group Technology) production, and Fixed-Position production systems [14, 25, 58]. In the job-shop production system, machines and facilities of similar types are arranged together in the layout. This production system is well-suited for low-production, high-variation production systems.

This study explores FLPs in a job-shop production environment, addressing the unique challenges posed by the inherent complexity of alternative process plans, machine redundancy, and capacity constraints. To overcome these challenges, a novel matheuristic approach is proposed, combining mathematical optimization techniques with heuristic methods. Specifically, we leverage Iterated Local Search (ILS) and Genetic Algorithm (GA) to develop efficient and robust solutions. By integrating these methodologies, the study aims to advance optimization strategies for facility layout planning, offering innovative solutions that address both theoretical and practical complexities.

As the literature on the FLPs is quite extensive, in this study, the analysis of the literature has been structured around five main themes: objectives, planning approach, constraints, facility features, and solutions.

The most addressed objective of FLPs in the literature is the minimization of material handling costs. There are different material-handling systems like single-row layout [11, 36, 47, 79], multi-row layout [27, 32, 85, 86], parallel row ordering [2,12,49,80], loop layout [52,75], open-field layout [38,48,51], multi-floor layout [10,37], and handling equipment [16,81] layout that are explored and documented in the FLP literature. Unlike traditional layouts, open-field layouts involve arranging facilities within an open space without predefined constraints, allowing for greater flexibility and innovation in design [69]. However, this layout is the most complex one, as it enables the movement of materials in any order to any machine [28], and this has received comparatively less attention in research. Our focus on addressing open-field layout problems underscores the importance of understanding and addressing these specific challenges within the broader context of facility layout optimization.

Additionally, efficient facility layout design not only aims to minimize the distances traveled during material handling processes but also strives to optimize the allocation and utilization of machines to reduce acquisition costs. An example of this is seen in the work of Kubalík *et al.* [40], who introduced a novel FLP formulation featuring loosely-defined facilities. This formulation is notable in that it not only considers variations in shape and dimensions but also incorporates additional characteristics such as construction costs, operating costs, and more, for facilities. Within the proposed model, our focus on open-field layout problems emphasizes the need to consider both material handling and machine acquisition costs.

From a planning perspective, FLPs can be approached from either a Dynamic [22,77,83] or Static [30,53,54,63] perspective. In the case of Static FLP (SFLP), the material flows between facilities are assumed to be fixed throughout the planning horizon. Conversely, Dynamic FLPs (DFLP) involve multi-period planning, where the material transportation between facilities varies significantly from one period to another. By concentrating on the static perspective, we aim to provide insights and solutions tailored to scenarios where material flows remain constant over time.

In addition to objective functions and planning approach, the discrete or continuous nature of the layout is a significant factor that affects modeling of FLPs [34]. A discrete layout divides the plant site into rectangular blocks. Each block could assign to a facility which is known as the Quadratic Assignment Problem (QAP). The first model of the discrete model as a QAP was proposed by Kusiak and Heragu [44]. Extensive research has historically centered on the QAP and its variations to tackle diverse applications. Our investigation ventures into the domain of integer non-linear mathematical programming QAP models providing a more customized method for facility layout optimization, encompassing alternative processing routes alongside the sequence of operations and the concept of machine redundancy. In contrast to conventional QAP discrete layouts, our model provides a more precise depiction of real-world facility layout scenarios by integrating these realistic features.

In the analysis of FLPs, an important aspect to consider is the characteristics of facility attributes, including their shapes, dimensions, and area [14]. FLP studies mainly focus on two distinct facility shapes: regular, specifically rectangular shapes [1,17,36,46,78] and irregular shapes [8,45]. Moreover, facility dimensions can be categorized as fixed, flexible, or mixed. Fixed dimensions pertain to a facility with constant length and width measurements [17,56]. Conversely, flexible dimensions indicate the ability to alter the width and length within a predetermined range during the arrangement process, which can be regulated by aspect ratios [48,72], area ratios [31,84], minimum area requirements [42,77], or predetermined intervals for department length or width [20]. Nevertheless, most scholars considered equal areas for facilities [63]. Our method focuses on optimizing regular, fixed-shaped facilities with equal sizes. By emphasizing these specific aspects, we aim to offer practical and effective strategies for enhancing facility layout planning.

In context of modeling FLPs, a range of constraints, including but not limited to budgetary limitations, spatial constraints, machine capacity, multiple process plans, overlapping operations, and pick-up/drop-off requirements, sequence-dependent setup times, give rise to the need for incorporating realistic assumptions. In light of these considerations, Solimanpur and Kamran [70] introduced a model that addresses the production of multiple products, each with alternative process plans and specific operation sequences on various types of machines. The model is formulated as a mixed-integer non-linear mathematical programming model. The model aims to determine the optimal process plan for each product and identify the machine locations necessary to produce all products, with the objective of minimizing the total distance traveled by the products. Kumar *et al.* [43] introduced a heuristic approach to finding a shared linear machine locations for multiple products with varying operation sequences. The approach assumed that product flows are only permitted in the forward direction, either in-sequence or by-passing. Ha [23] proposed a new ant colony optimization (ACO) algorithm suitable for integrated process planning and scheduling that optimizes both process planning and scheduling simultaneously considering sequencedependent setups and tool-related capacity constraints.

In terms of solution approach, FLPs are commonly acknowledged to be an NP-hard class of problems, signifying the absence of an exact algorithm capable of providing an optimal solution within a reasonable time frame. As a result, a diverse range of heuristic and metaheuristic algorithms have been developed in the literature to tackle these challenges. Genetic algorithms (GA) [60,61,70,73,82], simulated annealing (SA) [1,33,62,66], Tabu search [5,41,67], variable neighborhood search [19,27,64], and Ant Colony optimization (ACO) [21,37,48]. Among these, GAs have been widely adopted due to their flexibility and historical precedence in tackling optimization problems across diverse domains. In contrast, Iterated Local Search (ILS) has been less frequently applied to FLPs but has demonstrated success in related combinatorial optimization problems, such as route planning [4,55], corridor allocation problem [15] and job sequencing [9]. The strength of ILS for optimization problems lies in its balance between exploration and exploitation, achieved through systematic perturbations and local refinements [6,71]. Despite its advantages, ILS remains underrepresented in FLP studies, although it has recently gained attention in this field. Recent studies, such as Wu *et al.* [76] utilized an iterated local search algorithm based on ejection chains to solve the space-free multi-row facility layout problem. de Almeida *et al.* [13] proposed a hybrid ILS-based matheuristic for large-scale single-source capacitated facility location problems. To the best of our knowledge, there are only a few studies that use multiple solution methodologies including Metaheuristics and Matheuristics to compare their performances. Moreover, this paper pioneers the

TABLE 1. Static facility layout characteristic.

Articles	Characteristics								
	Discrete layout	Regular facility shapes	Alternative process plans	Machine redundancy	Machine capacity constraints	Machine acquisition costs	Exactly known and crisp data	MIP (✘) or MINP including QAP (✓)	Solution approach
Lenin *et al.* [47]	✓	✓		✓		✓	✓	✓	GA
Matai *et al.* [54]	✓	✓†					✓	✓	H
Matai *et al.* [53]	✓	✓†					✓	✓	Modified SA
Hungerlaender [36]	✓	✓†					✓	✓	SDP
Palubeckis [62]	✓	✓†					✓	✓	Modified SA
Izadinia and Eshghi [37]	✓	✓†					✓*	✓✘	RO, ACO
Mallikarjuna *et al.* [52]	✓	✓†	✓	✓			✓	✘	GA, SA
Che *et al.* [10]	✓	✓†					✓	✓✘	ψ-constraint method
Paes *et al.* [60]		✓**					✓	✓	Modified GA
Feng and Che [17]	✓	✓**					✓	✘	Solver
Friedrich *et al.* [19]	✓	✓**					✓	✘	H (VNS, PTH)
Allahyari and Azab [1]		✓†					✓	✓	Modified SA
Feng and Che [17]	✓	✓†					✓	✓	Solver
Jerin Leno *et al.* [38]	✓	✓†					✓	✓✘	GA, SA
García-Hernández *et al.* [20]		✓**					✓	✘	CRO
Liu and Liu [48]	✓	✓**					✓	✘	Muli-objective ACO
Yang *et al.* [79]		✓					✓	✘	Solver
Besbes *et al.* [7]	✓	✓†					✓	✓✘	GA, Sim
Xu *et al.* [78]		✓†					✓	✓✘	GA, SRFA
Ha [23]			✓		✓		✓	✓✘	ACO
Hunagund *et al.* [33]		✓**					✓	✘	SA
McKendall and Hakobyan [56]		✓†					✓	✘	GA
Erik and Kuvvetli [16]		✓**					✓	✓✘	Solver
Lenin and Siva Kumar [46]	✓	✓†					✓	✓	H, HS
Hunagund *et al.* [35]	✓	✓**					✓	✓✘	SA, H
Kubalík *et al.* [40]	✓	✓**				✓	✓	✓	EA
Subulan *et al.* [72]	✓	✓**	✓				✓	✓	H
Yang *et al.* [81]	✓						✓	✓	Sim
Zolfi *et al.* [85]	✓	✓†					✓	✓	Dual memory SA
Wu *et al.* [76]	✓	✓					✓	✓	ILS
de Almeida *et al.* [13]	✓	✓			✓	✓	✓	✓✘	Matheuristic (HILS)
Present work	✓	✓†	✓	✓	✓	✓	✓	✓✘	**Matheuristic, ILS, GA**

Notes. ✓*: Deterministi and (Robust, Stochastic or Fuzzy data); ✓†: Regular and fixed dimension; ✓**: Regular and flexible dimensions, or unequal area; ACO: Ant Colony Optimization; AHP: Analytic Hierarchy Process; BB: Branch & Boun; CRO: Coral Reef Optimizatio; EA: Evolution algorithm; GA: Genetic Algorithm; GT: Game Theory;H: Heuristic; HS: Harmony Search; ILS: Iterated Local Search; MCS: Monte Carlo Simulation; PSO: Particle Swarm Optimizatio; PTH: Parallel Tempering based heuristic; RO: Robust Optimization; SA: Simulated Annealing; SDP: Semi-Definite Programmin; Sim: Simulation; SRFA: Surplus Rectangle Fill Algorithm; TS: Tabu Search; VNS: Variable Neighborhood Search.

application of ILS to FLPs in job-shop environments, comparing its performance with other metaheuristic and matheuristic approaches, and filling a critical gap in the literature.

For more details about the different FLP classification and their related scholars in the recent years, we kindly refer readers to reviews done by Hosseini-Nasab *et al.* [30], Hunagund *et al.* [34] and Pérez-Gosende *et al.* [63].

To identify the existing research gap in this specific field, a comprehensive literature review was conducted, focusing on recent studies (published after 2013) that are most relevant to the present investigation. Table 1

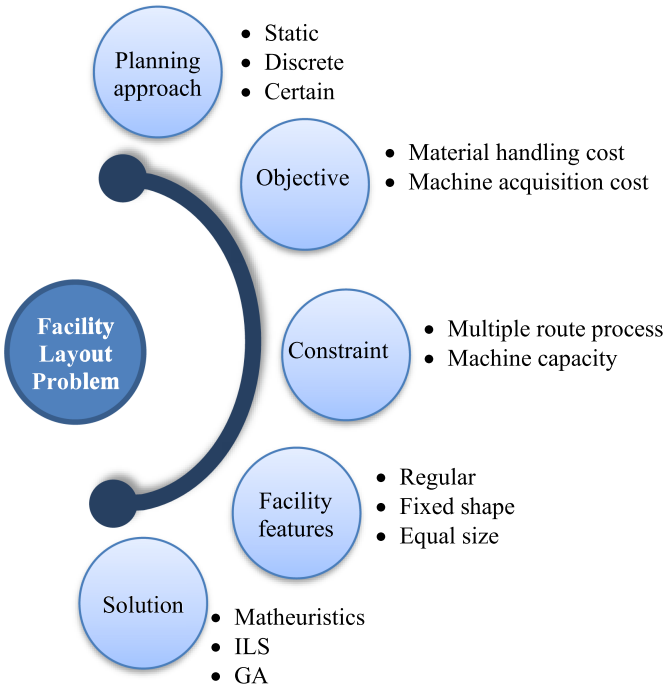


FIGURE 1. The proposed approach.

summarizes the review considering three criteria: workshop characteristics, problem formulation, and solution approach.

It is evident from Table 1 that the concepts of alternative process plans, machine redundancy, and consequently machine capacity and machine acquisition cost are usually neglected for FLPs in job-shop production systems in the literature. It is worth noting that while these concepts are extensively studied in cellular manufacturing systems, *e.g.*, Forghani *et al.* [18], Mahootchi *et al.* [50], Shafiee-Gol *et al.* [68], CMS falls outside the scope of our current study. Following the structure of the literature review, Figure 1 illustrates the problem addressed in this study across the aforementioned streams. Our study addresses a problem that concurrently considers alternative process plans, machine redundancy, machine capacity, and machine acquisition cost within the framework of a static discrete and QAP FLP in a job-shop production system. Facility features include regular, fixed-shape, and equal-sized facilities situated in open-field scenarios, where discrete space allocation and predetermined potential locations are considered. It is important to note that the acquisition cost of machines is represented as an amortized cost over the analysis period, rather than the full purchase cost. For example, if the production plans are per week, the acquisition cost is calculated on a weekly basis; if the plans are annual, the acquisition cost is considered on an annual basis. Proposed solutions entail an innovative matheuristic approach, complemented by two metaheuristic solutions using ILS and GA methodologies. We will compare the results and discuss insights derived from this comparison. Ultimately, our objective is to explore realistic assumptions overlooked in prior literature (see Tab. 1), thereby enhancing the depth of our analysis.

The rest of the paper is organized as follows. In Section 2, the proposed problem is explained in detail and a non-linear mathematical programming model is suggested. Then, a linearization technique is proposed to linearize the model. Section 3 discusses the solution methodology by proposing a new matheuristic and Iterated Local Search (ILS) and Genetic Algorithm (GA) metaheuristics. In Section 4, the performance of the proposed solution algorithms is discussed based on randomly generated problems, and finally, Section 5 includes conclusions and directions for future works.

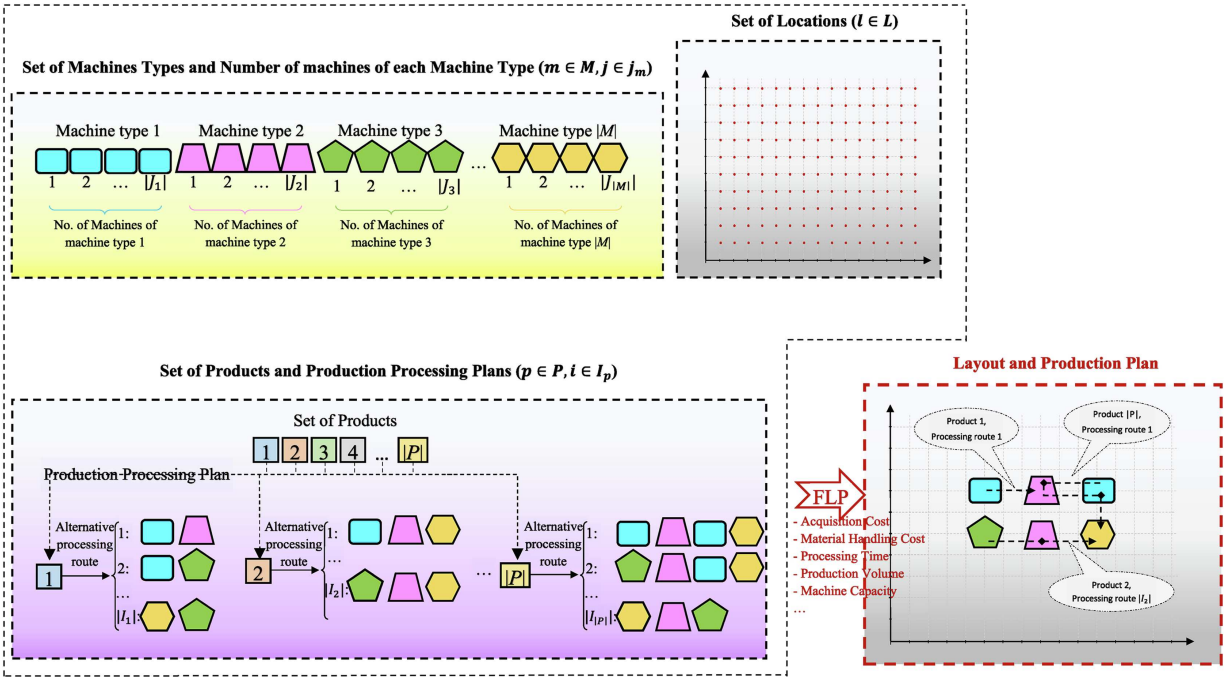


FIGURE 2. Schematic illustration of the proposed problem.

2. PROBLEM STATEMENT

The formulation developed in this paper is an extension of the facility layout problem presented by Solimanpur and Kamran [70] in which a manufacturing system produces $|P|$ different kinds of products in the presence of alternative processing routes. There exist $|M|$ different types of machines to be located in $|L|$ potential locations where the distance between locations l and l' ($D_{ll'}$) is known in advance. In this study, each machine is indexed as (m, j) which indicates machine j of type m. It is assumed that the capacity of machine type $m(C_m)$ is limited and known in advance. Each product can be processed with alternative process plans several machines (I_p). The production volume of each product is known *a priori* (V_p). The model aims to find the optimal number of each machine to be purchased, the layout, and the processing route for each product while minimizing the total costs of material handling and machine acquisition. Figure 2 schematically illustrates the proposed problem. The rest of the parameters and variables applied to formulate the mathematical model are summarized in Table 2.

The mathematical model which minimizes the total material handling and machine acquisition costs can be formulated as follows:

$$\text{Min} \sum_{m} \sum_{\substack{m' \\ m' \neq m}} \sum_{j} \sum_{\substack{j' \\ j' \neq j}} \sum_{l} \sum_{\substack{l' \\ l' \neq l}} \sum_{p} \sum_{i} Fc(z_{mjpi} \cdot z_{m'j'pi} \cdot x_{mjl} \cdot x_{m'j'l'} \cdot y_{pi}) \cdot a_{mm'pi} \cdot v_p \cdot d_{ll'} + \sum_{m} \sum_{j} b_m \cdot u_{mj} \tag{1}$$

Subject to:

$$\sum_{l} x_{mjl} = u_{mj} \qquad \forall m \in M, \quad \forall j \in J_m \tag{2}$$

TABLE 2. Notation used in the paper.

Indices and parameters	
M	Set of machine types ($m, m' \in M$, where $\|M\|$ is the total number of machine types)
L	Set of existing and potential locations ($l, l' \in L$, where $\|L\|$ is the total number of potential locations)
P	Set of products ($p \in P$, where $\|P\|$ is the total number of products)
I_p	Set of alternative processing routes for product p($i \in I_p$, where $\|I_p\|$ is the total number of alternative processing routes for product p)
I	Total number of process plans ($I = \sum_i \|I_p\|$)
J_m	Set of machines of type m ($j, j' \in J_m$, where $\|J_m\|$ is the total number of machines of type m)
S_{pi}	Set of machines needed to produce product p in its process plan i
N	Total number of machines ($N = \sum_m \|J_m\|$)
v_p	Production volume of product p
Fc	Material handling cost per distance unit
b_m	Acquisition cost of machine type m
t_{mpi}	Processing time of product p in its process plan i on machine type m
$d_{ll'}$	Distance between locations l and l' ($l, l' \in L$)
c_m	Capacity of machine type m
$a_{mm'pi}$	$= 1$ if machine type m' is needed just after machine type m in the process plan i of product p; 0 otherwise
ε	A very small positive amount
Decision variables	
x_{mjl}	$= 1$ if machine j of type m is located in location l; 0 otherwise
y_{pi}	$= 1$ if product p has to be produced by its process plan i; 0 otherwise
u_{mj}	$= 1$ if machine $j$of type m is needed for production; 0 otherwise
z_{mjpi}	$= 1$ if machine j of type m is needed to produce product p in process plan i; 0 otherwise

$$\sum_m \sum_j x_{mjl} \leq 1 \qquad \forall l \in L \tag{3}$$

$$\sum_i y_{pi} = 1 \qquad \forall p \in P \tag{4}$$

$$\sum_j z_{mjpi} = y_{pi} \qquad \forall p \in P, \quad \forall i \in I_p, \quad \forall m \in S_{pi} \tag{5}$$

$$\sum_p \sum_i z_{mjpi} \cdot t_{mpi} \cdot v_p \leq c_m \qquad \forall m \in M, \quad \forall j \in J_m \tag{6}$$

$$\sum_p \sum_i \varepsilon \cdot z_{mjpi} \leq u_{mj} \qquad \forall m \in M, \quad \forall j \in J_m \tag{7}$$

$$x_{mjl}, y_{pi}, u_{mj}, z_{mjpi} \in \{0, 1\} \qquad \forall m \in M, \quad \forall j \in J_m, \quad \forall l \in L, \quad \forall p \in P, \quad \forall i \in I_p \tag{8}$$

where constraints (2) and (3) guarantee that each machine of each type is located at only one location and each location is occupied by only one machine at most, respectively. Constraint (4) ensures that only one process plan will be selected for each product. Constraint (5) guarantees that only one machine of each type is being employed in the production of product p in processing plan i. Constraint (6) imposes a capacity constraint on machines such that the amount of production assigned to each machine does not exceed its capacity. Constraint (7) indicates that each machine type can produce more than one product. The constraint is essential to activate (*i.e.*, purchase) machine j for type m, ensuring that the model does not set those variables equal to 0. In other

words, it is a classic "fixed-cost activation constraint". Finally, constraint (8) defines binary domain restrictions of the decision variables.

It can be seen that the objective function of the proposed model is non-linear. To linearize the model, a new binary variable ($r_{mm'jj'll'pi}$) is defined and replaced with the multiplication of five binary variables of $z_{mjpi} \cdot z_{m'j'pi} \cdot x_{mjl} \cdot x_{m'j'l'} \cdot y_{pi}$. The following constraint guarantees that variable $r_{mm'jj'll'pi}$ equals 1 only when all the five multiplied variables are 1 and it equals 0, otherwise.

$$(z_{mjpi} + z_{m'j'pi} + x_{mjl} + x_{m'j'l'} + y_{pi}) - 4 \leq r_{mm'jj'll'pi} \leq (z_{mjpi} + z_{m'j'pi} + x_{mjl} + x_{m'j'l'} + y_{pi})/5 \tag{9}$$

$$r_{mm'jj'll'pi} \in \{0,1\} \qquad \forall m, m' \in M, \quad \forall j, j' \in J_m, \quad \forall l, l' \in L, \quad \forall p \in P, \quad \forall i \in I_p, \quad m' \neq m, \quad j' \neq j, \quad l' \neq l. \tag{10}$$

So, the linear equivalent of the non-linear model is as follows:

$$\text{Min} \sum_{m} \sum_{\substack{m' \\ m' \neq m}} \sum_{j} \sum_{\substack{j' \\ j' \neq j}} \sum_{l} \sum_{\substack{l \\ l' \neq l}} \sum_{p} \sum_{i} Fc \cdot a_{mm'pi} \cdot v_p \cdot d_{ll'} \cdot r_{mm'jj'll'pi} + \sum_{m} \sum_{j} b_m \cdot u_{mj} \tag{11}$$

Subject to: (2)–(7),

$$\left(z_{mjpi} + z_{m'j'pi} + x_{mjl} + x_{m'j'l'} + y_{pi}\right) - 4 \leq r_{mm'jj'll'pi} \qquad \forall m, m' \in M, \quad \forall j, j' \in J_m, \quad \forall l, l' \in L, \quad \forall p \in P, \quad \forall i \in I_p, \quad m' \neq m, \quad l' \neq l \tag{12}$$

$$r_{mm'jj'll'pi} \leq (z_{mjpi} + z_{m'j'pi} + x_{mjl} + x_{m'j'l'} + y_{pi})/5 \qquad \forall m, m' \in M, \quad \forall j, j' \in J_m, \quad \forall l, l' \in L, \quad \forall p \in P, \quad \forall i \in I_p, \quad m' \neq m, \quad j' \neq j, \quad l' \neq l \tag{13}$$

$$x_{mjl},\ y_{pi},\ u_{mj},\ z_{mjpi},\ r_{mm'jj'll'pi} \in \{0,1\} \qquad \forall m, m' \in M, \quad \forall j, j' \in J_m, \quad \forall l, l' \in L, \quad \forall p \in P, \quad \forall i \in I_p, \quad m' \neq m, \quad j' \neq j, \quad l' \neq l. \tag{14}$$

Although the second model is linear, this linearizing method considerably increases the number of binary variables and constraints as well in comparison with the non-linear model. However, the linear model can be simplified through the method presented by Solimanpur and Kamran [70]. This would be possible by removing constraint (13) and relaxing variable r to be a non-negative real variable. This simplification is due to the positive coefficients of variables r in the objective function [70].

$$\text{Min} \sum_{m} \sum_{\substack{m' \\ m' \neq m}} \sum_{j} \sum_{\substack{j' \\ j' \neq j}} \sum_{l} \sum_{\substack{l \\ l' \neq l}} \sum_{p} \sum_{i} Fc \cdot a_{mm'pi} \cdot v_p \cdot d_{ll'} \cdot r_{mm'jj'll'pi} + \sum_{m} \sum_{j} b_m \cdot u_{mj} \tag{11}$$

Subject to: (2)–(7), (12)

$$x_{mjl}, y_{pi}, u_{mj}, z_{mjpi} \in \{0,1\} \qquad \forall m \in M, \quad \forall j \in J_m, \quad \forall l \in L, \quad \forall p \in P, \quad \forall i \in I_p \tag{15}$$

$$r_{mm'jj'll'pi} \geq 0 \qquad \forall m, m' \in M, \quad \forall j, j' \in J_m, \quad \forall l, l' \in L, \quad \forall p \in P, \quad \forall i \in I_p, \quad m' \neq m, \quad j' \neq j, \quad l' \neq l. \tag{16}$$

Simplification of the proposed model and reducing the number of constraints and binary variables reduces the computation time of the problem. The total number of constraints will be reduced by $(\sum_m \sum_{\substack{m' \\ m' \neq m}} (|J_m| \cdot |J_{m'}|) \cdot |L| \times (|L| - 1) \times I$. Also, the same reduction is achieved in the number of r binary variables which are now non-negative real variables. For instance, in a problem with size $|m| = 5$, $|J_m| = 3$, $|L| = 15$, and $I = 10$, the number of removed constraints and number of binary variables converted to non-negative real variables will be 189 000. This reduction in the number of constraints and binary variables considerably affects the exploration of the solution space and reduces the computational complexity of the problem.

3. Solution approach

In this study, considering the high computational complexity of the FLPs, *i.e.*, NP-hard [39], three distinct solution approaches have been developed to effectively solve the problem within a reasonable timeframe. These approaches include a matheuristic, ILS (ILS), and Genetic Algorithm (GA). While the matheuristic is specifically tailored for this problem and will be discussed in greater detail in the subsequent subsections, the well-known ILS and GA methods will be outlined using pseudocode.

3.1. Matheuristic algorithm

A matheuristic algorithm is proposed to effectively solve the presented non-linear pure binary FLP. The proposed matheuristic algorithm is composed of two stages as discussed in the following.

Stage 1

Let us define U as the set of process plans of all products, so the number of elements of U is I, ($I = \sum_i |I_p|$). Also, let V denote the set of all existing locations (L). The steps of stage 1 are described in the following.

Step 1. Solve the following facility location problem for each member of set U (Step 1). Notice that in the first stage, all locations are vacant. All parameters are the same as presented in the previous section. x_{ml} is a decision variable, which is 1 if machine m is located in location l; and 0, otherwise.

$$Z^1_{pi} = \text{Min} \sum_{m \in S_{pi}} \sum_{\substack{m' \in S_{pi} \\ m' \neq m}} \sum_{l=1}^{L} \sum_{\substack{l'=1 \\ l' \neq l}}^{L} Fc \cdot x_{ml} \cdot x_{m'l'} \cdot a_{mm'pi} \cdot v_p \cdot d_{ll'} \tag{17}$$

Subject to:

$$\sum_{l} x_{ml} = 1 \qquad \forall m \in S_{pi} \tag{18}$$

$$\sum_{m \in S_{pi}} x_{ml} \leq 1 \qquad \forall l \in L \tag{19}$$

$$x_{ml} \in \{0,1\} \qquad \forall m \in S_{pi}, \quad \forall l \in L. \tag{20}$$

To linearize this model, a new 0–1 variable ($y_{mm'll'}$) is replaced with the multiplication of two 0–1 variables ($x_{ml} \times x_{m'l'}$). Then, the linear model is simplified by relaxing variable y to a non-negative real variable as justified in Section 2. The linear model is given as follows:

$$Z^1_{pi} = \text{Min} \sum_{m \in S_{pi}} \sum_{\substack{m' \in S_{pi} \\ m' \neq m}} \sum_{l} \sum_{\substack{l' \\ l' \neq l}} Fc \cdot a_{mm'pi} \cdot v_p \cdot d_{ll'} \cdot y_{mm'll'} \tag{21}$$

Subject to:

$$\sum_{l} x_{ml} = 1 \qquad \forall m \in S_{pi} \tag{18}$$

$$\sum_{m \in S_{pi}} x_{ml} \leq 1 \qquad \forall l \in L \tag{19}$$

$$(x_{ml} + x_{m'l'} - 1) \leq y_{mm'll'} \qquad \forall m, m' \in S_{pi}, \quad \forall l, l' \in L, \quad m' \neq m, \quad l' \neq l \tag{22}$$

$$x_{ml} \in \{0,1\} \qquad \forall m \in S_{pi}, \quad \forall l \in L \tag{20}$$

$$y_{mm'll'} \geq 0 \qquad \forall m, m' \in S_{pi}, \quad \forall l, l' \in L, \quad m' \neq m, \quad l' \neq l. \tag{23}$$

The objective function (21) calculates the minimum material handling distance induced by process plan i of product p.

Step 2. Compute Z^2_{pi} for each process plan of each product using the following equation:

$$Z^2_{pi} = \sum_{m \in S_{pi}} b_m. \tag{24}$$

This expression computes machine purchasing cost for production of product p by process plan i.

Step 3. Select the best process plan for each product using the following term:

$$Z_{pi^*} = \min\{Z^1_{pi} + Z^2_{pi} | i \in I_p\}. \tag{25}$$

Step 4. Select product p^* that has the maximum impact on material handling and machine acquisition costs. This product and its process plan are selected by the following term:

$$Z_{p^*i^*} = \max\{Z_{pi^*} | p \in P\}. \tag{26}$$

Step 5. Locate machines needed to produce product p^* by process plan i^*, according to the solution obtained for $Z^1_{p^*i^*}$ in Step 1.

Step 6. Remove product p^* and all of its process plans from set U, and also remove the corresponding locations from set V, since these locations have been used to locate machines needed to produce this product.

Stage 2

Step 7. Solve the problem of Step 1 for all products and their process plans being in set U. In the solution process of the problem Z^1_{pi}, the following states may occur:

State 1. If a machine is needed to produce product p by process plan i and no machine of this type has been located so far, such a machine should be purchased and located.

State 2. If a machine is needed to produce product p by process plan i and one machine of this type has already been located, it should be taken into consideration whether the remaining capacity of this machine is sufficient to produce product p in process plan i or not. If the machine's remaining capacity is enough to produce product p by process plan i, there is no need to purchase a new machine of this type. It should be noticed that this machine's location is known, and it has been taken into account in problem Z^1_{pi}. If the remaining capacity is not sufficient, it is necessary to purchase and locate a new machine of this type.

State 3. If a machine is needed to produce product p by process plan i and more than one machine of this type have already been located, we do the following. If only one of these machines has enough capacity to produce product $p$by process plan i, this machine is considered for producing product p using process plan i. If more than one of these machines have enough capacity to produce p, the machine with less remaining capacity is considered for the production of product $p$by process plan i. If the remaining capacity is not sufficient, it is necessary to purchase and locate a new machine of this type.

Step 8. Find the acquisition cost Z^2_{pi} for machines newly located in the problem Z^1_{pi} in Step 7.

Step 9. Iterate steps 3 to 6.

Step 10. Repeat this process until all products are removed from set U. Note that the total material handling and machine acquisition cost is the sum of $Z_{p^*i^*}$ obtained in Step 4.

Figure 3 depicts the architecture of the proposed matheuristic solution algorithm.

3.2. Iterated local search algorithm

An ILS algorithm (ILS) is also developed in this study to achieve efficient and high-quality solutions for the FLP within limited computational resources. The selection of this metaheuristic algorithm is motivated by its effectiveness, simplicity, and its wide application in similar FLP studies conducted by Ramkumar *et al.* [65] and Durmaz and Şahin [15]. Furthermore, the ILS has demonstrated success in addressing various combinatorial optimization problems *e.g.*, Atefi *et al.* [3], Calmels [9], Öztaş and Tuş [59], Mendes and e Alvelos [57], and Avci [6].

```
Begin
  Generate the U set for all products' process plans;
  Generate the V set for all existing locations;
  Solve Z^1_pi for each member of U;
  Iteration ⟵ 1;
  While U is not null do
    If iteration = 1 then
      Step 1:
        Compute/ Update Z^2_pi for the process plan i of product p;
        Select the best process plan for each product (Z_pi*);
        Select product p* with maximum impact on objective function (Z_p*i*);
        Locate machines needed to produce p* in process plan i*, based on Z^1_p*i*;
        Remove p* and all of its process plans from U;
        Remove the corresponding locations from V;
    Else
        If state1 is true, then
          purchase and locate a new machine of this type based on Z^1_pi;
        Else
          if state2 is true, then
            if the remaining capacity of the machine is enough then
              Update the capacity;
            Else
              Purchase and locate a new machine of this type based on Z^1_pi;
            End if
          Elseif the remaining capacities of the machines are enough then
            if only one of these machines has enough capacity, then
              Update the capacity with less remaining capacity;
            Else
              Choose the machine with less remaining capacity;
            End if
            Update the capacity;
          Else
            Purchase and locate a new machine of this type based on Z^1_pi;
          End if
        End if
        Go to Step 1;
    End if
    Iteration ⟵ Iteration +1;
  End while
End
```

FIGURE 3. Architecture of the proposed matheuristic solution algorithm.

The ILS algorithm developed in this study is strengthened with several heuristics to enhance the exploration and exploitation stages of the ILS framework proposed by Hansen *et al.* [24]. The ILS algorithm commences by initializing a solution and progressively improving it through a combination of local search and perturbation procedures. By leveraging these techniques, the algorithm explores the solution space, continually refining and improving the initial solution. This iterative process allows for the exploitation of promising regions while avoiding entrapment in local optima, ultimately enhancing the chances of obtaining near-optimal solutions within reasonable time limits.

The initialization algorithm begins by randomly selecting one of the products. For every combination of p and i, the inverse of the accumulated product volume values across all machines is computed. Then, using the roulette wheel mechanism, one of the production process plans is chosen (with the lowest process time being more likely to be selected). The first and second machines in the sequence of the selected production plan are placed in locations that minimize their distance from each other. Subsequently, the following machines in the sequence are allocated to locations with progressively smaller distances from the previously allocated machines. Upon the completion of the machine placement process for the initial product, we proceed to the subsequent stage.

In the next step, the product with the lowest facility establishment cost, based on the remaining capacity of the established machines, is chosen from the unselected products. In the event that the remaining capacity of the existing machines exceeds the capacity demands of the subsequent machines relevant to the product production methodology, the corresponding machine is assigned to a location characterized by lower available capacity. In contrast, if the remaining capacity is inadequate, the machine is allocated to a location that imposes the lowest possible transportation cost within the given problem scenario. If the corresponding machine's remaining capacity is less than the required capacity for that machine or if there is no such machine in the layout design, the machine is established in a location that imposes the lowest transportation cost to the problem. This process continues until all products are selected.

The following heuristic algorithms are used to improve the solution.

- LS1: If the remaining capacity of machine j in location l is greater than the required capacity of machine j related to product p in location l' ($l \neq I'$), and the problem cost can be reduced by using machine j in location l' for product p, this move is executed.
- LS2: If the product p on machine j is relocated from location l to location l' ($l \neq I'$), and the cost of the problem is reduced by this change, this relocation is performed.
- LS3: If the production process plan i of product p is replaced by method i', resulting in a cost reduction, this displacement is carried out.
- LS4: If machine j associated with product p is located in location l and machine j related to product p' is situated in location l', and according to the capacity constraints, if the cost of the problem is reduced by changing the production location of these two products, this change is implemented.
- To introduce the perturbation phase of the algorithm, two shaking methods are employed with an equal probability of selection. The first method, shaking I, involves randomly removing a product from the arrangement program and selecting another production method for this product at random, following the same process as the initial solution construction algorithm. The second method, shaking II, randomly selects θ percent of the locations where machines are established and removes them from the layout plan. The removed machines are then reintroduced into the problem and assigned to the most cost-effective locations at each stage. These shaking methods are designed to introduce randomness and prevent the algorithm from becoming trapped in local optima.

The pseudocode of the ILS is presented in Figure 4.

3.3. Genetic Algorithm

The Genetic Algorithm (GA) is one of the prevalent metaheuristic methods for solving NP-hard optimization problems. This computational algorithm was inspired by the evolution of superior individuals through generations [29]. That means individuals with favorable traits survive and reproduce while those with unfavorable traits are eliminated. In GA, the process begins with a population of randomly generated solutions. Each individual within the population is referred to as a "chromosome" and evaluated based on the predefined fitness

```
Begin
    x ← InitialSolution();
    x ← Heuristics(x);
    x* ← x;
    While an improvement is found for max_iter do
        Pertube() ← Rand(Shake I, Shake II);
        x ← Pertube(x);
        x ← Heuristics(x);
        If f(x) < f(x*) then
            x* ← x;
        else
            x ← x*;
        End if
    End while
End
```

FIGURE 4. Pseudocode of the proposed ILS.

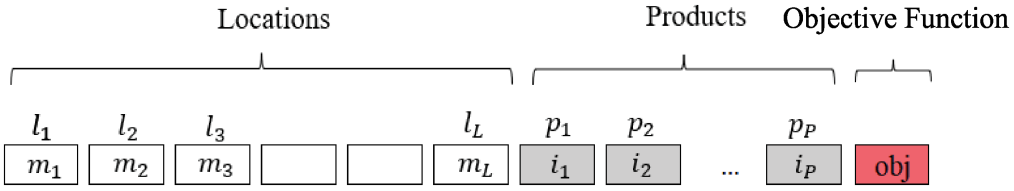
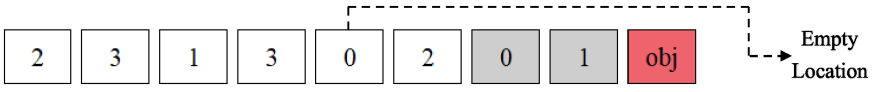


FIGURE 5. Chromosome structure of the proposed GA.

function. which in this case is Function (11). A chromosome consists of a sequence of genes, each of which encodes a specific characteristic or variable of the solution. In this study, the chromosome is designed to be divided into three parts (Fig. 5). The first part includes L genes that show how M machines are placed in L locations. In other words, this part of the chromosome represents variable x_{mjl}. The second part consists of P genes describing the process plan selected for each product, which represents variable p_{pi}. The last part contains the value of objective function.

It is worth mentioning that the encoding scheme remains applicable even in cases where $L > M$, wherein the excess locations (*i.e.*, $L - M$ locations) are considered empty (zero). To further illustrate this concept, let's consider a scenario with two products, each having two process plans, and three types of machines. For Machine 1, there is only one machine at hand, while Machines 2 and 3 have two machines each.



During each iteration, the selection of chromosomes as parents, merging and modifying them creates a new population which might improve the solution. The selection of the chromosomes for producing a new population is based on a fitness function. Recombination of chromosomes, known as "crossover", involves taking two parent individuals from the current population and producing one or more new chromosomes (offspring) by exchanging segments of their genetic information. Modification of existing chromosomes, referred to as "mutation", is a rearrangement of the structure of the chromosomes to produce new ones. This process is utilized for exposing

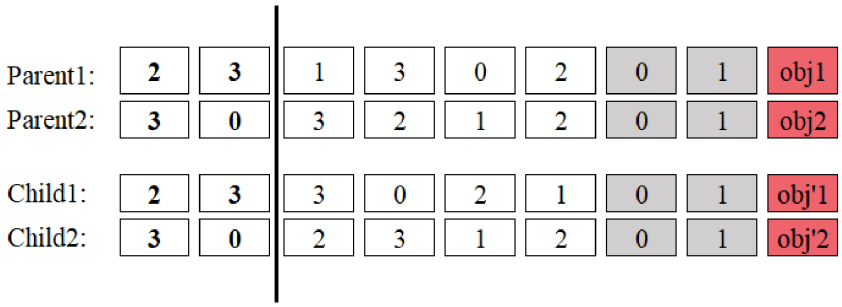


FIGURE 6. Crossover operator in the proposed GA.

unexpected changes in the values of chromosomes. It is noteworthy to mention that in the proposed algorithm, the crossover and mutation are applied only for the first part of the chromosomes, which pertains to the locations of machines. In other words, new populations are created through the crossover and mutation processes for each combination of process plans.

For implementing crossover, a random number (n) is generated from a discrete uniform distribution within the range $(1, L)$. In the resulting offspring chromosome, the genes up to position n remain unchanged, inherited from their respective parent chromosomes. The genes from n to L are filled with the order in which the remaining machines appear in the second parent chromosome. It is worth noting that, during this process, the machine capacities and process plans for each child chromosome are also taken into account. Let's revisit the previous example of two products and three types of machines. Figure 6 illustrates the crossover operator in the proposed algorithm for a specific combination of process plans.

In this study, the well-known swap mutation is utilized for the first part of the chromosome. This mutation involves randomly selecting two genes and swapping their contents. To enhance algorithm diversification, two swap mutations are implemented in the initial portion of the selected parents.

Once the newly generated offspring are created, a new generation is formed by selecting individuals from both the old and new generations. To achieve this, an elitism policy is adopted to strengthen the intensification capability (*i.e.*, exploitation) of the algorithm, while a roulette wheel mechanism is utilized to improve its diversification. The pseudocode for the proposed GA is presented in Figure 7.

4. COMPUTATIONAL STUDY

In this section, we provide an overview of the computational experiments carried out to evaluate the effectiveness of the proposed algorithms. The linearized mathematical model, the matheuristic, the ILS, and the GA are implemented in C++ on a computer with an Intel Core i5 CPU processor 2.50 GHz and 12 GB of RAM, utilizing CPLEX 20.1 as the MILP solver. Section 4.1 describes the sets of randomly generated instances used in our experimentation. Parameter tuning details for the developed ILS and GA are reported in Section 4.2. The results of the three developed algorithms are presented and compared in Section 4.3. Finally, the performance of the two well-performing algorithms is tested once again on a practical case study.

4.1. Randomly generated instances

To assess the effectiveness of the three proposed algorithms under diverse scenarios, we generate a range of random instances. These instances are divided into three distinct sets, each comprising subsets with similar values of $|P|$ and $|M|$. The small-size set contains a total of 15 instances, with $|P|$ values of 4, 5, and 6, and $|M|$ values of 5 and 6. The medium-size set includes 10 instances, with $|P|$ values of 8 and 10, and $|M|$ values of 8. The large-size set consists of 10 instances, with $|P|$ values of 20, and $|M|$ values ranging from 8 to 10. All algorithms are evaluated on these sets, and the results are presented in the subsequent sections. Each row

```
Data: Instance α, size of the population, crossover rate, roulette wheel rate;
Result: best individuals;
Begin
    Generate δ different process plan combinations for products;
    For r =1 to δ
        P (r) ◄— Generate the first population for plan combination r;
        Calculate the value of fitness function for each individual in P(r);
        Popu(r) ◄—Sorted P(r);
    End for
    t ◄— 0;
        While termination conditions are not met do
            For r =1 to δ
                Par_child_matrix (r,t) ◄—Popu(r);
                Select individuals from Par_child_matrix (r,t);
                Recombine individuals;
                Mutate individuals;
                P'(r,t) ◄— newly generated individuals;
                Calculate the value of fitness function for each individual in P'(r,t);
                Add P'(r,t) to Par_child_matrix (r,t);
                Reproduce Par_child_matrix (r,t) based on roulette wheel rate;
                Final_popu (r,t) ◄—Sorted Par_child_matrix (r, t);
            End for
            t ◄— t+1;
        End while
    Return the best individual from Final_popu (r,t);
End
```

FIGURE 7. Pseudocode of the proposed GA.

corresponds to a single problem instance, identified using the format Input-p–m–n, where p denotes the number of products, m the size of the machine set, and n the instance identifier.

The coordinates for each candidate location in every instance are selected randomly from integer values between 0 and 10. The distance between each pair of locations is calculated using the Manhattan distance. Additionally, the processing time for each product in each process plan on every machine type is assigned as a random integer between 1 and 6. Each instance is identified by the tuple $(|P|, |M|, u)$, where u is a numerical index ranging from 1 to 5.

4.2. Parameter tuning

The ILS algorithm developed in this study involves two principal parameters, α and $\max_{iter}$. The values of which are determined through a sequence of experiments conducted on one instance randomly chosen from each category (7 instances in total). The ILS is applied to these instances using all possible combinations of parameter values from the sets $\alpha \in \{0.05, 0.10, 0.15, 0.25\}$ and $\max_{iter} \in \{1000, 5000, 10\,000\}$. Table 3 presents a summary of the results obtained from these experiments, including the average runtime of the ILS algorithm and the

TABLE 3. ILS configuration tuning (Best configuration in Boldface).

	$\max_{iter}$					
	1000		5000		10 000	
α	t(s)	%gap	t(s)	%gap	t(s)	%gap
0.05	1.61	1.96	1.76	1.83	2.09	1.83
0.10	1.55	1.72	**2.10**	**0.00**	2.67	0.00
0.15	1.86	3.89	2.84	2.46	4.03	2.18
0.25	2.13	4.74	4.33	3.04	5.95	2.51

TABLE 4. GA configuration tuning (Best configuration in Boldface).

	Ge							
	40		50		60		70	
Po	t(s)	%gap	t(s)	%gap	t(s)	%gap	t(s)	%gap
40	0.36	0.35	0.461	0.37	0.452	0.15	0.808	0.01
50	0.765	0.12	0.507	0.26	0.405	0.17	0.543	0.01
60	0.497	0.06	0.512	0.17	**0.473**	**0.00**	0.723	0.25
70	0.582	0.24	0.724	0.21	0.722	0.12	0.501	0.13

average gap for each parameter combination across the seven instances. The gap is calculated as the average of $100(z - z^*)/z*$, where z represents the solution value obtained with the specific parameter configuration and z^* denotes the best solution value achieved among all configurations. It is worth noting that Table 3 solely focuses on analyzing the sensitivity of ILS parameters. Knowing in ILS, in contrast with VNS, the intensity of the perturbation remains constant throughout the search.

The parameter combination with $\alpha = 0.10$ and $\max_{iter} = 5000$ prove to be the most effective, consistently achieving the best solution values. Despite a slight increase in computational time, this configuration is employed in all subsequent ILS tests.

Genetic algorithm parameter tuning involves population size (Po), generation size (Ge), percentage of replication of well-performed chromosomes (R), and crossover (R_c) and mutation rates (R_m). Experiments are conducted to determine optimal parameter values. Following the same procedures in the ILS algorithm, the experimental results are expressed in terms of the runtime and average gap of each parameter combination across the sets of experiments. Table 4 indicates suggestions for population and generation size combinations. Increasing these values generally improves solution quality but also increases runtime and cost. The combination of $Po = 60$ and $Ge = 60$ is found to be effective in achieving optimal solutions within a reasonable runtime. The algorithm terminates after producing identical solutions for twenty consecutive iterations.

4.3. Results

In this section, we conduct an analysis and evaluation to assess the effectiveness of the three proposed algorithms. Table 5 presents a comprehensive comparison of the results obtained by the linearized model using the commercial solver CPLEX, and the three developed solution approaches, namely, the matheuristic, the ILS, and the GA for each instance.

In the linearized model solved by CPLEX, the columns "$z^{\text{LB}}_{\text{CPLEX}}$" and "$z^{\text{UB}}_{\text{CPLEX}}$" provide the respective lower and upper bound values. The "gap" column represents the percentage gap, computed as $100(z^{\text{UB}}_{\text{CPLEX}} - z^{\text{LB}}_{\text{CPLEX}})/z^{\text{UB}}_{\text{CPLEX}}$. Additionally, the "$t_{\text{CPLEX}}$" column indicates the run time of the execution. It's important to note that the lower bound values refer to the values obtained at the end of the execution.

TABLE 5. Performance of the three proposed algorithms.

	CPLEX				Matheuristic			ILS						GA					
Data instance	z^{LB}_{CPLEX}	z^{UB}_{CPLEX}	t_{CPLEX}	gap	z_{MAT-H}	t_{MAT-H}	gap	z_{best}	z_{worst}	z_{avg}	std_z	t_{avg}	gap	z_{best}	z_{worst}	z_{avg}	std_z	t_{avg}	gap
Input_4_5_1	5200	1030	3600	49.5	**9650**	4.2	0.0	**9650**	9650	9650	0.0	0.2	0.0	12 200	12 850	12 567	332.9	0.7	26.4
Input_4_5_2	4950	**1045**	3600	52.6	**10 450**	4.1	0.0	**10 450**	10 450	10 450	0.0	0.2	0.0	10 600	11 150	10 833	284.3	0.5	1.4
Input_4_5_3	3950	**6750**	3600	41.5	7700	3.1	14.1	**6750**	6750	6750	0.0	0.6	0.0	7350	7950	7667	301.4	0.7	8.9
Input_4_5_4	4250	7750	3600	45.2	11 350	2.8	52.4	**7450**	7450	7450	0.0	0.3	0.0	7750	8250	8017	251.7	0.4	4.0
Input_4_5_5	2700	**5750**	3600	53.0	6800	3.6	18.3	**5750**	5750	5750	0.0	0.4	0.0	7100	7500	7267	208.2	0.9	23.5
Input_5_6_1	6050	1755	3600	65.5	17 950	4.9	33.5	**13 450**	13 450	13 450	0.0	0.6	0.0	18 450	19 700	18,900	694.6	0.5	37.2
Input_5_6_2	4750	8900	3600	46.6	9900	2.9	17.9	**8400**	8400	8400	0.0	0.3	0.0	9900	10 550	10 150	350.0	0.4	17.9
Input_5_6_3	4700	9500	3600	50.5	10 300	2.5	29.6	**7950**	7950	7950	0.0	0.3	0.0	8600	9800	9000	692.8	0.9	8.2
Input_5_6_4	3950	7650	3600	48.4	8550	2.8	26.7	**6750**	6750	6750	0.0	0.2	0.0	8650	9000	8800	180.3	0.4	28.2
Input_5_6_5	4850	1185	3600	58.2	10 200	2.1	4.1	**9800**	9800	9800	0.0	0.3	0.0	13 150	14 050	13 550	458.3	0.3	34.2
Input_6_6_1	OOM	OOM	–	–	**10 100**	7.3	0.0	**10 100**	10 100	10 100	0.0	0.6	0.0	12 800	14 200	13 567	709.5	0.7	26.7
Input_6_6_2	OOM	OOM	–	–	12 850	8.8	35.3	**9500**	9500	9500	0.0	0.8	0.0	11 350	12 100	11 617	419.3	1.1	19.5
Input_6_6_3	OOM	OOM	–	–	10 150	4.4	5.7	**9600**	9600	9600	0.0	0.5	0.0	12 400	13 450	13 067	579.5	1.1	29.2
Input_6_6_4	OOM	OOM	–	–	10 000	2.1	1.0	**9900**	9900	9900	0.0	0.2	0.0	12 450	13 850	12 983	757.2	0.3	25.8
Input_6_6_5	OOM	OOM	–	–	11 800	5.1	15.7	**10 200**	10 200	10 200	0.0	0.3	0.0	12 600	13 500	12 967	472.6	3.2	23.5
Avg (Small-size)					1584.4	1.4	16.9	1497	1497	1497.3	0.0	0.2	0.0	2295	2509	2406	165.5	0.4	21.0
Input_8_8_1	OOM	OOM	–	–	15 650	23.8	59.7	**9800**	9800	9800	0.0	1.2	0.0	13 600	15 850	14 933	1181.5	18.9	38.8
Input_8_8_2	OOM	OOM	–	–	16 150	19.0	37.5	**11 750**	11 750	11 750	0.0	1.4	0.0	16 200	16 900	16 650	390.5	6.2	37.9
Input_8_8_3	OOM	OOM	–	–	16 750	8.9	30.4	**12 850**	12 850	12 850	0.0	0.9	0.0	18 800	19 100	18 983	160.7	0.9	46.3
Input_8_8_4	OOM	OOM	–	–	17 400	12.5	40.3	**12 400**	12 400	12 400	0.0	1.0	0.0	16 300	18 650	17 217	1257.3	2.1	31.5
Input_8_8_5	OOM	OOM	–	–	15 300	12.6	37.8	**11 100**	11 100	11 100	0.0	1.2	0.0	11 700	13 100	12 533	737.1	1.7	5.4
Input_10_8_1	OOM	OOM	–	–	17 600	64.5	14.7	**15 350**	15 350	15 350	0.0	1.9	0.0	25 300	26 600	25 800	700.0	61.1	64.8
Input_10_8_2	OOM	OOM	–	–	19 100	23.2	48.1	**12 900**	12 900	12 900	0.0	1.3	0.0	19 700	22 300	20 733	1379.6	0.7	52.7
Input_10_8_3	OOM	OOM	–	–	15 750	22.2	43.8	**10 950**	10 950	10 950	0.0	1.3	0.0	19 000	22 400	20 433	1761.6	1.0	73.5
Input_10_8_4	OOM	OOM	–	–	19 600	25.0	55.6	**12 600**	12 600	12 600	0.0	1.7	0.0	16 650	21 650	18 450	2778.5	3.5	32.1
Input_10_8_5	OOM	OOM	–	–	18 750	29.4	25.4	**14 950**	14 950	14 950	0.0	1.9	0.0	27 350	29 150	28 183	907.4	0.9	82.9
Avg (Medium-size)					17 205	24.1	39.3	12 465	12 465	12 465	0.0	1.4	0.0	18 460	20 570	19 392	1125.4	9.7	46.6
Input_20_8_1	OOM	OOM	–	–	41 850	299.1	26.4	**25 450**	25 450	25 450	0.0	5.1	0.0	36 850	41 650	39 217	2400.7	1525	44.8
Input_20_8_2	OOM	OOM	–	–	41 850	299.1	75.8	**23 800**	23 950	23 830	67.1	6.0	0.0	41 100	41 950	41 450	444.4	71.9	72.7
Input_20_8_3	OOM	OOM	–	–	34 400	254.4	75.1	**19 650**	19 650	19 650	0.0	4.1	0.0	45 750	47 200	46 233	837.2	39.2	132.8
Input_20_8_4	OOM	OOM	–	–	35 850	221.2	70.3	**21 050**	21 050	21 050	0.0	4.2	0.0	29 750	32 150	30 983	1201.4	64.7	41.3
Input_20_8_5	OOM	OOM	–	–	42 800	362.1	109.8	**20 400**	20 400	20 400	0.0	6.2	0.0	39 000	41 950	40 617	1495.3	14 316	91.2
Input_20_10_1	OOM	OOM	–	–	42 100	357.5	80.3	**23 350**	23 550	23 390	89.4	4.6	0.0	39 550	41 400	40 217	1027.5	1608	69.4
Input_20_10_2	OOM	OOM	–	–	35 050	347.0	60.8	**21 800**	21 800	21 800	0.0	4.0	0.0	41 500	43 300	42 550	936.8	131.7	90.4
Input_20_10_3	OOM	OOM	–	–	33 500	275.9	27.6	**26 250**	26 400	26 300	70.7	4.1	0.0	39 100	43 150	40 733	2135.6	71.7	49.0
Input_20_10_4	OOM	OOM	–	–	42 950	384.1	90.0	**22 600**	22 600	22 600	0.0	5.5	0.0	41 250	43 100	42 433	1027.5	230.2	82.5
Input_20_10_5	OOM	OOM	–	–	30 250	337.4	31.0	**23 100**	23 100	23 100	0.0	4.9	0.0	30 500	31 750	31 250	661.4	13 765	32.0
Avg (Large-size)					38 060	313.8	64.7	22 745	22 795	22 757	22.7	4.9	0.0	38 435	40 760	39 568	1216.8	3182	70.6
Avg (total)					20 297	98.3	37.0	13 937	13 951	13 941	6.5	1.9	0.0	20 980	22 606	21 730	860.4	912.4	42.5

In the matheuristic method, the objective function values are provided in the "z_{MAT-H}" column, while the average run time is reported in the "t_{MAT-H}" column. The ILS and GA are executed five times for each instance, and we present the best, average, and worst solution values achieved, along with their standard deviation, in the "z_{best}", "z_{avg}", "z_{worst}", and "Std_z" columns, respectively.

Upon analyzing Table 8, we can observe that CPLEX only finds the best solutions for some of small-sized instances. However, when it comes to instances from the medium- and large-size sets, the computer runs out of memory (OOM) due to the significant size of the model.

The ILS algorithm, in the case of small-sized instances, consistently demonstrates superior performance compared to the GA and matheuristic algorithm (with no proof of optimality). Additionally, the remarkably low standard deviation across runs highlights the robustness of the ILS algorithm on these simple instances, whereas the large standard deviation of the GA implies its inability to ensure convergence and robustness. When analyzing average run times, the ILS achieves an average run time of just 0.2 s, whereas the GA requires 0.4 s, and the matheuristic algorithm needed an average of 1.4 s.

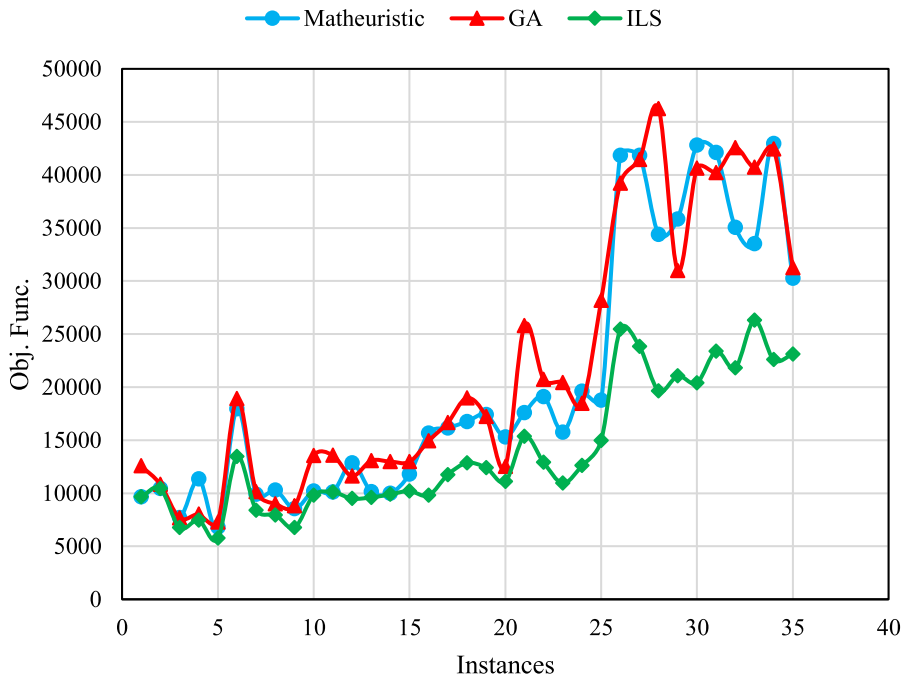


FIGURE 8. Performance of the three proposed algorithms in terms of objective function.

The ILS algorithm demonstrates exceptional robustness on medium and large-sized instances with an average standard deviation of 0.0 and 22.7, respectively. In contrast, the GA displays inferior robustness with an average standard deviation exceeding 1000 for both medium and large-sized instances.

The comparative analysis of the ILS, GA, and matheuristic algorithm in terms of objective value, gap to the best solution, and run time across a range of instances has revealed significant insights. The ILS algorithm achieves the best objective value across all instances, while the matheuristic algorithm tied for the best solution in three instances. On the other hand, the GA exhibits a substantial average gap to the best solution, which is 42.5% across all instances. The matheuristic algorithm performs better than the GA in terms of gap to the best solution, with an average value of 37.% across all instances.

Furthermore, the run time comparison of the three algorithms also reveals notable differences. The ILS has the lowest average run time of 1.9 s to solve all instances, while the GA exhibits the longest average run time of 912.4 s to solve all instances. The matheuristic algorithm falls in between, with an average run time of 98.3 s across all instances. As the performance of the commercial solver to solve the linearized model is not acceptable at all in terms of solution quality and run time, in Figures 8 and 9 only the performance of the three proposed algorithms is graphically depicted in terms of objective function and run time, respectively.

4.4. Practical case study

After evaluating the efficiency and effectiveness of the algorithms in the previous section – and confirming that both the ILS and matheuristic approaches outperform the GA – this section aims to further demonstrate the practical applicability of our proposed solution methodology by testing the performance of the ILS and matheuristic algorithms on a practical layout design case adapted from the benchmark introduced by Lenin *et al.* [47]. To align the original benchmark with our problem context, several modifications were applied. These include the addition of new attributes for machines and products, incorporation of multiple alternative

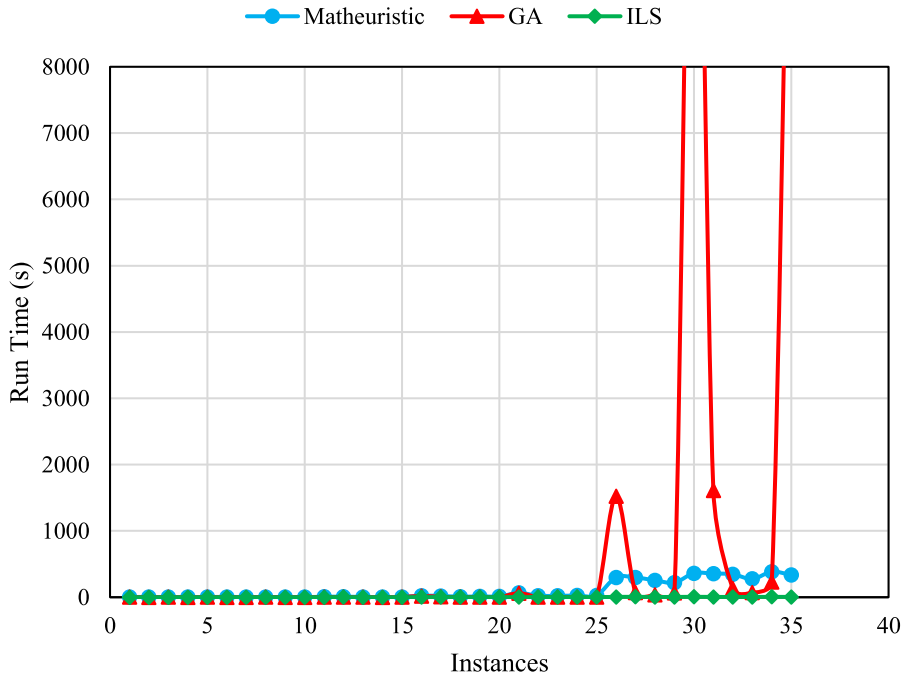


FIGURE 9. Performance of the three proposed algorithms in terms of run time.

process plans per product, and the introduction of stochastic processing parameters to better reflect real-world manufacturing environments.

A key adjustment was the update to the cost structure. The original investment cost for each machine was replaced with an acquisition cost, calculated by multiplying the original value by 1.25, to better represent real-life procurement costs. Additionally, to enhance realism and complexity, a range of machine and product attributes were randomly generated. Machine capacities were set between 100 and 350 units. Volume requirements per product were derived from their specific process plans, while machine-product processing times were also randomly assigned.

Each of the five products was associated with three alternative process plans, each defining a specific sequence of machines required for production. These process plans, along with the corresponding processing times, are summarized in Table 6. Table 7 presents the acquisition cost and capacity of the twelve machines used in the system, which were integrated into the optimization models for feasibility and cost evaluation. Product volumes, which directly affect machine utilization, are reported in Table 8.

To provide a comprehensive evaluation of algorithm performance, a diverse set of test cases was generated. The instances were grouped into three categories based on the number of products: three, four, or five. For each category, five distinct instances were created by varying product combinations and associated process plans, resulting in a total of 15 problem instances.

Both the ILS algorithm and the matheuristic were applied to all instances. The ILS was executed five times per instance to account for stochastic behavior, and its performance was assessed using best, average, and worst objective values, standard deviation, and average runtime. In contrast, the matheuristic was executed once per instance, and its objective value and runtime were recorded for comparison.

Table 9 and Figure 10 presents the computational results, divided into two sections – one for the matheuristic and one for the ILS. The findings consistently show that the ILS outperforms the matheuristic across all key performance indicators. In every case, the ILS delivered better best, average, and worst objective values. For

TABLE 6. Process plans and sequence (processing time) of each process plan for manufacturing each product.

Product	Process plan	Sequence	Product	Process plan	Sequence
1	1	8(1)–2(6)–10(4)–9(3)–6(8)	4	1	12(4)–3(5)–7(3)–1(8)
	2	4(1)–6(8)–8(4)–1(8)		2	3(4)–5(4)–1(8)–8(5)
	3	1(1)–3(8)–5(8)–7(7)		3	4(4)–2(4)–3(6)–7(3)
2	1	4(2)–8(7)–6(4)–5(4)	5	1	10(5)–9(4)–5(7)–7(4)–2(8)
	2	7(2)–1(7)–8(8)–2(7)		2	5(5)–9(5)–8(3)–1(7)
	3	2(2)–4(3)–6(4)–7(3)		3	7(5)–2(7)–5(7)–6(7)
3	1	1(3)–11(5)–4(3)–5(3)			
	2	5(3)–6(5)–9(4)–8(5)–3(8)			
	3	3(3)–5(5)–7(5)–1(5)			

TABLE 7. Configuration of each machine.

	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12
Acquisition cost	20 831	12 380	22 658	24 658	17 230	16 660	12 557	6088	10 912	27 943	24 234	8132
Set of machine	2	2	2	2	2	2	2	2	2	2	2	2
Capacity	350	130	210	220	210	270	190	110	140	150	100	250

TABLE 8. Production volume of each product.

	Product 1	Product 2	Product 3	Product 4	Product 5
Volume	800	600	1000	750	700

example, in instance Input-3–6–1, the matheuristic yielded an objective value of 229 026, while the best ILS result was 186 604 – a cost reduction of 18.5%. Similar improvements were observed across all instances, with the average improvement (calculated as the relative difference between the matheuristic and the best ILS result) reaching approximately 24.8%.

The robustness of the ILS is further demonstrated by its minimal variability across repeated runs. In fact, the standard deviation was zero for all instances, indicating consistent convergence to high-quality solutions. This level of repeatability is particularly important in practical applications where reliability is critical.

As expected, increasing the number of products led to higher objective values due to greater production demands, more complex process plans, and tighter capacity constraints. For example, the average objective value for three-product instances was around 205 000, while it exceeded 340 000 in the matheuristic results and 210 000 in the ILS results for five-product cases. Despite the added complexity, the ILS maintained strong performance, producing solutions over 40% better than those of the matheuristic in the largest problem sets.

From a computational standpoint, the ILS also demonstrated excellent efficiency. Across all test cases, the average runtime was just 1.09 s, and even for the most complex scenarios – those involving five products and up to 12 machines – it never exceeded 1.5 s. This highlights the algorithm's potential for real-time or near-real-time industrial applications. In contrast, the matheuristic, although executed only once per instance, generally required longer runtimes for small problems and scaled less efficiently with problem size.

TABLE 9. Performance of the ILS and Matheuristic algorithms on practical layout design case.

Data instance	Matheuristic		ILS				
	Objective Value	Time (s)	Best Obj.	Worst Obj.	Average Obj.	STD.	Time (s)
Input-3–6–1	229 026	1.85	186 604	186 604	186 604	0.00	0.74
Input-3–8–2	200 573	1.95	164337	164 337	164 337	0.00	0.82
Input-3–9–3	215 869	1.67	187689	187 689	187 689	0.00	1.07
Input-3–9–4	217 368	3.56	179674	179 674	179 674	0.00	0.86
Input-3–10–5	201 873	3.93	163033	163 033	163 033	0.00	1.11
Input-4–10–1	261 030	8.64	197887	197 887	197 887	0.00	1.07
Input-4–10–2	269 619	4.65	220133	220 133	220 133	0.00	1.11
Input-4–10–3	256 750	6.12	198974	198 974	198 974	0.00	1.15
Input-4–11–4	298 613	4.38	257784	257 784	257 784	0.00	1.00
Input-4–11–5	309 888	5.23	243107	243 107	243 107	0.00	1.17
Input-5–10–1	352 178	7.24	186172	186 172	186 172	0.00	1.24
Input-5–11–2	334 124	10.12	244 522	244 522	244522	0.00	1.47
Input-5–11–3	343 717	9.23	138 951	138 951	138 951	0.00	1.36
Input-5–12–4	352 690	9.4	189 303	189 303	189 303	0.00	1.18
Input-5–12–5	336 813	8.61	217 884	217 884	217 884	0.00	0.99
Average		5.78				0.00	1.09

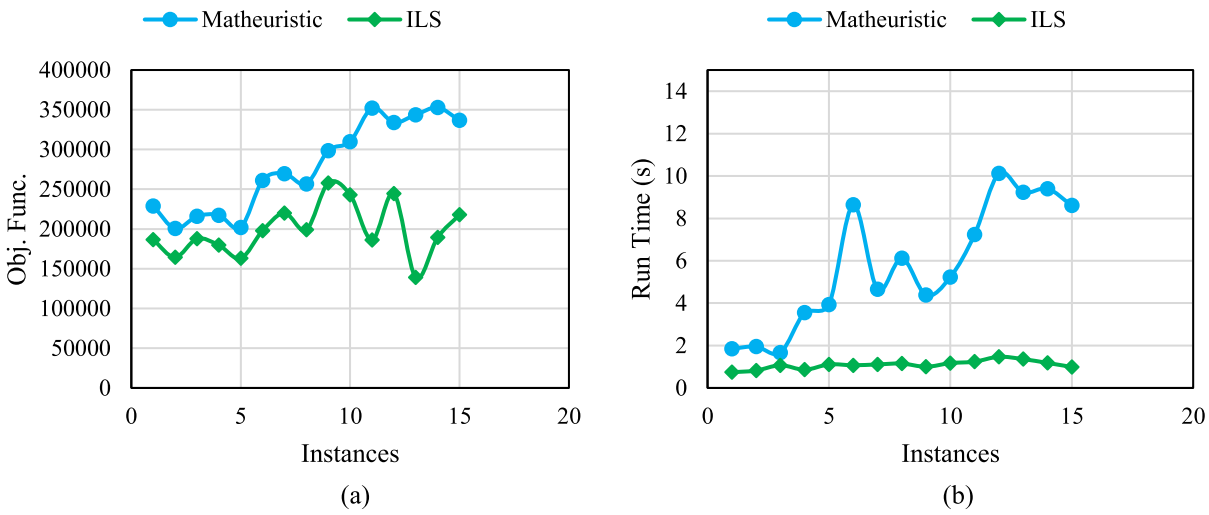


FIGURE 10. Performance of the ILS and Matheuristic algorithms on practical layout design case in terms of (a) objective function and (b) run time.

5. CONCLUSION

In this paper, a non-linear integer mathematical programming model is proposed for the facility location problem to minimize the total material handling and machine acquisition cost. It is assumed that there are multiple products, and alternative process plans for each product. Also, A technique is used to linearize the formulated non-linear model but still, the model is computationally intractable and CPLEX commercial solver was only able to solve some of the small-sized instances but not the medium- and large sized ones. We conducted a comprehensive analysis and evaluation of three proposed algorithms, namely the matheuristic method, the

ILS, and the GA, for solving optimization problems. The experiments encompassed diverse instances of varying sizes and complexities, and the algorithms were compared based on solution quality, robustness, and run time.

The ILS algorithm consistently outperforms the GA and matheuristic algorithm, particularly on small-sized instances, demonstrating superior solution quality and robustness. It maintains its performance on medium and large-sized instances, showcasing stability and minimal solution variations. The ILS algorithm is also the most efficient in terms of run time. The matheuristic algorithm also exhibited efficient performance in solving diverse instances of different sizes FLPs, specifically medium- and large-sized ones. To further validate the effectiveness and demonstrate the practical applicability of our proposed solution methodology, the ILS and matheuristic algorithms were applied to a practical layout design case adapted from the literature. The results reaffirm their strong performance across realistic scenarios, with the ILS continuing to deliver superior solution quality and robustness, and the matheuristic demonstrating acceptable performance and scalability. These outcomes further support their applicability to real-world facility layout problems of varying sizes and complexities.

Although the current implementation of the matheuristic does not outperform the ILS in this study, we believe that it can serve as a foundational exploration of alternative approaches for solving the FLP. One promising direction for enhancing the matheuristic could involve modifying the Stage 1 model to be a Total Unimodular model, which might significantly improve its computational efficiency. Additionally, creating a feedback loop between Stage 1 and Stage 2 of the algorithm represents another avenue for refinement, potentially enhancing solution quality. Such adjustments would make the matheuristic particularly effective in more restrictive facility layout environments, where its adaptability might lead to superior performance compared to existing metaheuristics.

These findings provide valuable insights for decision-makers, highlighting the suitability of the ILS algorithm for NP-hard optimization problems, especially for small to medium-sized instances. It offers reliable and high-quality solutions with minimal variations, enabling informed managerial decision-making. These findings provide valuable insights for decision-makers seeking optimization algorithms for facility layout domain problems.

Finally, in the following, some directions for future work are recommended.

- In this paper, it is assumed that if a machine is needed to produce a product, one machine of this type is sufficient to do so. This assumption can be relaxed in future work so that more than one machine of each type might be needed to produce some products.
- The demand for products may be assumed to vary over time. This assumption necessitates revising the proposed model for the dynamic layout of machines.
- Consideration of stochastic processing time seems a useful extension.
- As the proposed matheuristic is developed based on decomposing the problem and the solution algorithm into two phases of iterative assignment phase and capacity checking phase, big improvements could be obtained by exploiting benefits of this decomposition. Benefits could range from small improvements in each stage (for example applying the idea of Minimax Regret Approach or Vogle Approximation Method instead of using the *maximum impact* strategy used in step 4 of the heuristic) to fundamental and structured matheuristic improvements.
- Furthermore, compare the effectiveness of the developed solution approaches to successful matheuristics and metaheuristics used in the literature.

Data availability statement

No new data/codes were created or analyzed in this study.

References

[1] M.Z. Allahyari and A. Azab, Mathematical modeling and multi-start search simulated annealing for unequal-area facility layout problem. *Expert Syst. Appl.* **91** (2018) 46–62.

[2] A.R. Amaral, A parallel ordering problem in facilities layout. *Comput. Oper. Res.* **40** (2013) 2930–2939.

[3] R. Atefi, M. Salari, L.C. Coelho and J. Renaud, The open vehicle routing problem with decoupling points. *Eur. J. Oper. Res.* **265** (2018) 316–327.
[4] R. Atefi, M. Iori, M. Salari and D. Vezzali, Solution of a practical vehicle routing problem for monitoring water distribution networks. *J. Oper. Res. Soc.* **75** (2024) 1989–2007.
[5] P. Aurich, M. Speckmann, C. Böning and M. Stonis, A two-stage Tabu Search for multi-objective facility layout problem (2021). ESSN: 2701-6277.
[6] M. Avci, An effective iterated local search algorithm for the distributed no-wait flowshop scheduling problem. *Eng. App. Artif. Intell.* **120** (2023) 105921.
[7] M. Besbes, M. Zolghadri, R. Costa Affonso, F. Masmoudi and M. Haddar, A methodology for solving facility layout problem considering barriers: genetic algorithm coupled with A^* search. *J. Intell. Manuf.* **31** (2020) 615–640.
[8] M. Besbes, Y.I. Mahjoub, T. Bonte, T. Berger, Y. Sallez and M. Zolghadri, Solving facility layout problem with safety consideration of reconfigurable manufacturing and assembly systems. *Proc. CIRP* **104** (2021) 1942–1947.
[9] D. Calmels, An iterated local search procedure for the job sequencing and tool switching problem with non-identical parallel machines. *Eur. J. Oper. Res.* **297** (2022) 66–85.
[10] A. Che, Y. Zhang and J. Feng, Bi-objective optimization for multi-floor facility layout problem with fixed inner configuration and room adjacency constraints. *Comput. Ind. Eng.* **105** (2017) 265–276.
[11] V. Coppé, X. Gillard and P. Schaus, Solving the constrained single-row facility layout problem with decision diagrams, in 28th International Conference on Principles and Practice of Constraint Programming (CP 2022) (2022).
[12] G.L. Cravo and A.R.S. Amaral, Adaptive iterated local search for the parallel row ordering problem. *Expert Syst. App.* **208** (2022) 118033.
[13] G.B. de Almeida, E.M. de Sá, S.R. de Souza and M.J.F. Souza, A hybrid iterated local search matheuristic for large-scale single source capacitated facility location problems. *J. Heuristics* **30** (2024) 145–172.
[14] A. Drira, H. Pierreval and S. Hajri-Gabouj, Facility layout problems: a survey. *Ann. Rev. Control* **31** (2007) 255–267.
[15] E.D. Durmaz and R. Şahin, An efficient iterated local search algorithm for the corridor allocation problem. *Expert Syst. App.* **212** (2023) 118804.
[16] A. Erik and Y. Kuvvetli, Integration of material handling devices assignment and facility layout problems. *J. Manuf. Syst.* **58** (2021) 59–74.
[17] J. Feng and A. Che, Novel integer linear programming models for the facility layout problem with fixed-size rectangular departments. *Comput. Oper. Res.* **95** (2018) 163–171.
[18] K. Forghani, M. Mohammadi and V. Ghezavati, Two-stage method for solving cell formation and layout problems with alternative routings and machine capacity constraints. *Int. J. Manage. Sci. Eng. Manage.* **8** (2013) 292–301.
[19] C. Friedrich, A. Klausnitzer and R. Lasch, Integrated slicing tree approach for solving the facility layout problem with input and output locations based on contour distance. *Eur. J. Oper. Res.* **270** (2018) 837–851.
[20] L. García-Hernández, L. Salas-Morera, J. Garcia-Hernandez, S. Salcedo-Sanz and J.V. de Oliveira, Applying the coral reefs optimization algorithm for solving unequal area facility layout problems. *Expert Syst. App.* **138** (2019) 112819.
[21] J. Guan and G. Lin, Hybridizing variable neighborhood search with ant colony optimization for solving the single row facility layout problem. *Eur. J. Oper. Res.* **248** (2016) 899–909.
[22] C. Guan, Z. Zhang, L. Zhu and S. Liu, Mathematical formulation and a hybrid evolution algorithm for solving an extended row facility layout problem of a dynamic manufacturing system. *Robot. Comput.-Integr. Manuf.* **78** (2022) 102379.
[23] C. Ha, Evolving ant colony system for large-sized integrated process planning and scheduling problem considering sequence-dependent setup times. *Flexible Serv. Manuf. J.* **32** (2020) 523–560.
[24] P. Hansen, N. Mladenović, R. Todosijević and S. Hanafi, Variable neighborhood search: basics and variants. *Eur. J. Comput. Optim.* **5** (2017) 423–454.
[25] J. Heizer, B. Render and C. Munson, Operations Management. Prentice-Hall (2008).
[26] S.S. Heragu, Facilities Design. CRC Press (2008).
[27] A. Herrán, J.M. Colmenar and A. Duarte, An efficient variable neighborhood search for the space-free multi-row facility layout problem. *Eur. J. Oper. Res.* **295** (2021) 893–907.
[28] M. Hirvikorpi, On the Tactical Level Production Planning in Flexible Manufacturing Systems. Citeseer (2005).
[29] J.H. Holland, Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. MIT Press (1992).
[30] H. Hosseini-Nasab, S. Fereidouni, S.M.T. Fatemi Ghomi and M.B. Fakhrzad, Classification of facility layout problems: a review study. *Int. J. Adv. Manuf. Technol.* **94** (2018) 957–977.

[31] M.H. Hu and M.-J. Wang, Using genetic algorithms on facilities layout problems. *Int. J. Adv. Manuf. Technol.* **23** (2004) 301–310.

[32] B. Hu and B. Yang, A particle swarm optimization algorithm for multi-row facility layout problem in semiconductor fabrication. *J. Ambient Intell. Human. Comput.* **10** (2019) 3201–3210.

[33] I.B. Hunagund, V.M. Pillai and U.N. Kempaiah, Solving unequal area facility layout problems with flexible Bay structure by simulated annealing algorithm, in Advances in Production and Industrial Engineering: Select Proceedings of ICETMIE 2019. Springer Singapore, Singapore (2021).

[34] I.B. Hunagund, V.M. Pillai and U.N. Kempaiah, A survey on discrete space and continuous space facility layout problems. *J. Facilities Manage.* **20** (2022) 235.

[35] I.B. Hunagund, M. Pillai and U.N. Kempaiah, A two-stage solution approach for unequal area facility layout problem with bi-directional relaxed flexible bay structure. *J. Facilities Manage.* **21** (2023) 751–779.

[36] P. Hungerlaender, Single-row equidistant facility layout as a special case of single-row facility layout. *Int. J. Prod. Res.* **52** (2014) 1257–1268.

[37] N. Izadinia and K. Eshghi, A robust mathematical model and ACO solution for multi-floor discrete layout problem with uncertain locations and demands. *Comput. Ind. Eng.* **96** (2016) 237–248.

[38] I. Jerin Leno, S. Saravana Sankar and S. Ponnambalam, MIP model and elitist strategy hybrid GA–SA algorithm for layout design. *J. Intell. Manuf.* **29** (2018) 369–387.

[39] D.S. Johnson and M.R. Garey, Computers and Intractability: A Guide to the Theory of NP-completeness. WH Freeman (1979).

[40] J. Kubalík, L. Kurilla and P. Kadera, Facility layout problem with alternative facility variants. *Appl. Sci.* **13** (2023) 5032.

[41] S. Kulturel-Konak, A linear programming embedded probabilistic tabu search for the unequal-area facility layout problem with flexible bays. *Eur. J. Oper. Res.* **223** (2012) 614–625.

[42] S. Kulturel-Konak and A. Konak, A large-scale hybrid simulated annealing algorithm for cyclic facility layout problems. *Eng. Optim.* **47** (2015) 963–978.

[43] M.S. Kumar, M.N. Islam, N. Lenin, D. Vignesh Kumar and D. Ravindran, A simple heuristic for linear sequencing of machines in layout design. *Int. J. Prod. Res.* **49** (2011) 6749–6768.

[44] A. Kusiak and S.S. Heragu, The facility layout problem. *Eur. J. Oper. Res.* **29** (1987) 229–251.

[45] Y.H. Lee and M.H. Lee, A shape-based block layout approach to facility layout problems using hybrid genetic algorithm. *Comput. Ind. Eng.* **42** (2002) 237–248.

[46] N. Lenin and M. Siva Kumar, Harmony search algorithm for simultaneous minimization of bi-objectives in multi-row parallel machine layout problem. *Evol. Intell.* **14** (2021) 1495–1522.

[47] N. Lenin, M. Siva Kumar, M.N. Islam and D. Ravindran, Multi-objective optimization in single-row layout design using a genetic algorithm. *Int. J. Adv. Manuf. Technol.* **67** (2013) 1777–1790.

[48] J. Liu and J. Liu, Applying multi-objective ant colony optimization algorithm for solving the unequal area facility layout problems. *Appl. Soft Comput.* **74** (2019) 167–189.

[49] S.K. Mahalingam and L. Nagarajan, Strategy to reduce floor area and flow distance of products in multi-row parallel machine layout design using cuckoo search algorithm. *J. Adv. Manuf. Syst.* **20** (2021) 273–315.

[50] M. Mahootchi, K. Forghani and M. Abdollahi Kamran, A two-stage stochastic model for designing cellular manufacturing systems with simultaneous multiple processing routes and subcontracting. *Sci. Iran.* **25** (2018) 2824–2837.

[51] K. Maier and V. Taferner, Solving the constrained single-row facility layout problem with integer linear programming. *Int. J. Prod. Res.* **61** (2023) 1882–1897.

[52] K. Mallikarjuna, V. Veeranna and K.H. Reddy, A new meta-heuristics for optimum design of loop layout in flexible manufacturing system with integrated scheduling. *Int. J. Adv. Manuf. Technol.* **84** (2016) 1841–1860.

[53] R. Matai, S. Singh and M. Mittal, Modified simulated annealing based approach for multi objective facility layout problem. *Int. J. Prod. Res.* **51** (2013) 4273–4288.

[54] R. Matai, S. Singh and M. Mittal, A non-greedy systematic neighbourhood search heuristic for solving facility layout problem. *Int. J. Adv. Manuf. Technol.* **68** (2013) 1665–1675.

[55] V.R. Maximo, J.F. Cordeau and M.C. Nascimento, A hybrid adaptive iterated local search heuristic for the maximal covering location problem. *Int. Trans. Oper. Res.* **32** (2025) 176–193.

[56] A. McKendall and A. Hakobyan, An application of an unequal-area facilities layout problem with fixed-shape facilities. *Algorithms* **14** (2021) 306.

[57] A.B. Mendes and F.P. e Alvelos, Iterated local search for the placement of wildland fire suppression resources. *Eur. J. Oper. Res.* **304** (2023) 887–900.

[58] S. Nahmias and Y. Cheng, Production and Operations Analysis (Vol. 6). McGraw-hill New York (2009).
[59] T. Öztaş and A. Tuş, A hybrid metaheuristic algorithm based on iterated local search for vehicle routing problem with simultaneous pickup and delivery. *Expert Syst. App.* **202** (2022) 117401.
[60] F.G. Paes, A.A. Pessoa and T. Vidal, A hybrid genetic algorithm with decomposition phases for the unequal area facility layout problem. *Eur. J. Oper. Res.* **256** (2017) 742–756.
[61] J.M. Palomo-Romero, L. Salas-Morera and L. García-Hernández, An island model genetic algorithm for unequal area facility layout problems. *Expert Syst. App.* **68** (2017) 151–162.
[62] G. Palubeckis, Fast simulated annealing for single-row equidistant facility layout. *Appl. Math. Comput.* **263** (2015) 287–301.
[63] P. Pérez-Gosende, J. Mula and M. Díaz-Madroñero, Facility layout planning. An extended literature review. *Int. J. Prod. Res.* **59** (2021) 3777–3816.
[64] R. Phanden, H. Demir and R. Gupta, Application of genetic algorithm and variable neighborhood search to solve the facility layout planning problem in job shop production system, in 2018 7th International Conference on Industrial Technology and Management (ICITM) (2018).
[65] A. Ramkumar, S. Ponnambalam and N. Jawahar, A new iterated fast local search heuristic for solving QAP formulation in facility layout design. *Robot. Comput.-Integr. Manuf.* **25** (2009) 620–629.
[66] R. Şahin, A simulated annealing algorithm for solving the bi-objective facility layout problem. *Expert Syst. App.* **38** (2011) 4460–4465.
[67] D. Scholz, A. Petrick and W. Domschke, STaTS: a slicing tree and tabu search based heuristic for the unequal area facility layout problem. *Eur. J. Oper. Res.* **197** (2009) 166–178.
[68] S. Shafiee-Gol, R. Kia, R. Tavakkoli-Moghaddam, M. Kazemi and M.A. Kamran, Integration of parts scheduling, MRP, production planning and generalized fixed-charge transportation planning in the design of a dynamic cellular manufacturing system. *RAIRO-Oper. Res.* **55** (2021) S1875–S1912.
[69] G. Sini, J. Daaboul and J. Le Duigou, Layout and scheduling optimization problem for a reconfigurable manufacturing system. *Int. J. Ind. Eng. Manage.* **12** (2021) 174–186.
[70] M. Solimanpur and M.A. Kamran, Solving facilities location problem in the presence of alternative processing routes using a genetic algorithm. *Comput. Ind. Eng.* **59** (2010) 830–839.
[71] T. Stützle, Iterated local search for the quadratic assignment problem. *Eur. J. Oper. Res.* **174** (2006) 1519–1539.
[72] K. Subulan, B. Varol and A. Baykasoğlu, Unequal-area capability-based facility layout design problem with a heuristic decomposition-based iterative mathematical programming approach. *Expert Syst. App.* **214** (2023) 119199.
[73] X. Sun, L.-F. Lai, P. Chou, L.-R. Chen and C.-C. Wu, On GPU implementation of the island model genetic algorithm for solving the unequal area facility layout problem. *Appl. Sci.* **8** (2018) 1604.
[74] J.A. Tompkins, J.A. White, Y.A. Bozer and J.M.A. Tanchoco, Facilities Planning. John Wiley & Sons (2010).
[75] X. Wan, X. Zuo and X. Zhao, A differential evolution algorithm combined with linear programming for solving a closed loop facility layout problem. *Appl. Soft Comput.* **121** (2022) 108725.
[76] S. Wu, W. Yang, S. Hanafi, C. Wilbaut and Y. Wang, Iterated local search with ejection chains for the space-free multi-row facility layout problem. *Eur. J. Oper. Res.* **316** (2024) 873–886.
[77] Y. Xiao, Y. Xie, S. Kulturel-Konak and A. Konak, A problem evolution algorithm with linear programming for the dynamic facility layout problem – A general layout formulation. *Comput. Oper. Res.* **88** (2017) 187–207.
[78] S. Xu, Y. Wang and X. Feng, Plant layout optimization for chemical industry considering inner frame structure design. *Sustainability* **12** (2020) 2476.
[79] X. Yang, W. Cheng, P. Guo and Q. He, Mixed integer programming formulations for single row facility layout problems with asymmetric material flow and corridor width. *Arab. J. Sci. Eng.* **44** (2019) 7261–7276.
[80] X. Yang, W. Cheng, A.E. Smith and A.R. Amaral, An improved model for the parallel row ordering problem. *J. Oper. Res. Soc.* **71** (2020) 475–490.
[81] C. Yang, S. Liu and Z. Xu, A simulation-based optimization method for facility layout considering the AGV path. *J. Phys. Conf. Ser.* **2430** (2023) 012019.
[82] J.A. Zapata-Cortés, M.D. Arango-Serna and S. Cáceres-Gelvez, Genetic algorithm for the optimization of the unequal-area facility layout problem, in Handbook on Decision Making: Volume 3: Trends and Challenges in Intelligent Decision Support Systems. Springer (2022) 399–418.
[83] S. Zha, Y. Guo, S. Huang, Q. Wu and P. Tang, A hybrid optimization approach for unequal-sized dynamic facility layout problems under fuzzy random demands. *Proc. Inst. Mech. Eng. Part B: J. Eng. Manuf.* **234** (2020) 382–399.
[84] J.-L. Zhou, J.-S. Wang, Y.-X. Zhang, Q.-S. Guo, H. Li and Y.-X. Lu, Particle swarm optimization algorithm with variety inertia weights to solve unequal area facility layout problem, in 2020 Chinese Control And Decision Conference (CCDC) (2020).

[85] K. Zolfi, J. Jouzdani and H. Shirouyehzad, A novel memory-based simulated annealing algorithm to solve multi-line facility layout problem. *Decis. Sci. Lett.* **12** (2023) 69–88.

[86] X. Zuo, S. Gao, M. Zhou, X. Yang and X. Zhao, A three-stage approach to a multirow parallel machine layout problem. *IEEE Trans. Autom. Sci. Eng.* **16** (2018) 433–447.