AN INTEGRATED DYNAMIC FACILITY LAYOUT AND JOB SHOP SCHEDULING PROBLEM: A HYBRID NSGA-II AND LOCAL SEARCH ALGORITHM

Behrad Erfani
Sadoullah Ebrahimnejad* and Amirhossein Moosavi

Department of Industrial Engineering
Karaj Branch, Islamic Azad University
Karaj, Iran

(Communicated by Ada Che)

ABSTRACT. The aim of this research is to study the dynamic facility layout and job-shop scheduling problems, simultaneously. In fact, this paper intends to measure the synergy between these two problems. In this paper, a multi-objective mixed integer nonlinear programming model has been proposed where areas of departments are unequal. Using a new approach, this paper calculates the farness rating scores of departments beside their closeness rating scores. Another feature of this paper is the consideration of input and output points for each department, which is crucial for the establishment of practical facility layouts in the real world. In the scheduling problem, transportation delay between departments and machines’ setup time are considered that affect the dynamic facility layout problem. This integrated problem is solved using a hybrid two-phase algorithm. In the first phase, this hybrid algorithm incorporates the non-dominated sorting genetic algorithm. The second phase also applies two local search algorithms. To increase the efficacy of the first phase, we have tuned the parameters of this phase using the Taguchi experimental design method. Then, we have randomly generated 20 instances of different sizes. The numerical results show that the second phase of the hybrid algorithm improves its first phase significantly. The results also demonstrate that the simultaneous optimization of those two problems decreases the mean flow time of jobs by about 10% as compared to their separate optimization.

1. Introduction. These days, companies produce a wide variety of short life cycle products to satisfy the market’s needs. At the same time, they face conflicting uncertainties, such as demand changes, product mix modifications, more advanced production technologies, and stopping production lines. In such an environment, the material flow would change over different periods. Thus, a static design for the arrangement of a company’s facilities, i.e., the Static Facility Layout Planning (SFLP), would not be practical over the long run. To address this issue, the Dynamic Facility Layout Planning (DFLP) appeared as a new version of the SFLP in order to deal with such a dynamic condition.

2010 Mathematics Subject Classification. Primary: 90B50; Secondary: 90B85, 90B35.
Key words and phrases. Dynamic facility layout planning, job shop scheduling, multi-objective optimization, unequal area, NSGA-II, local search algorithm.
Corresponding author: Sadoullah Ebrahimnejad*.
In this paper, we have studied the DFLP problem. Accordingly, we have evaluated three performance metrics relevant to this problem, including material handling cost, machine rearrangement and rotation costs, Closeness and Farness Rating (CFR) scores of departments, and Percentage of Unused Space (PUS). In contrast to many of the previous investigations, e.g., [18], [19], [20], [21], and [24], we have considered inequality of departments’ area. In this problem, we have assumed that each department has its own input and output points. Moreover, we have implemented a new approach to assess the CFR scores of departments. The new approach is more realistic due to the following reason. The traditional approach merely relies on the Closeness Rating (CR) score of departments. However, it is possible that the closeness of two specific departments are not desirable, while their farness is desirable.

This paper also intends to study the Job Shop Scheduling (JSS) problem alongside the DFLP problem. Based on [19], these problems are interconnected and could affect each other; thus, the efficiency of each problem is conditioned on the other one. Thus, we study these two problems, simultaneously. Most of the previous papers have only studied one of these two problems, e.g., [20], [5], [7], and [14]. Even though, there are some studies that addressed both two problems, e.g., [29], [26], [27], and [3]. But these studies treated them as two separate problems. In other words, they tried to optimize the facility layout planning problem in the first stage, and afterward, they optimized the JSS problem at the second stage.

As mentioned by [19], the search space of the combination of discrete DFLP (i.e., Quadratic Assignment Problem (QAP)) and JSS problems with \( T \) periods is equal to \([\binom{N}{M} P^4 P^T]\), where facilities have particular input and output points. Given that the current study investigates the optimization of the continuous DFLP and JSS problems, therefore, this is certainly an NP-hard problem. For this purpose, we have developed a hybrid algorithm, which incorporates the non-dominated sorting genetic algorithm (NSGA-II) and two local search algorithms. In this hybrid algorithm, NSGA-II has been applied in the first phase to find some high-quality initial solutions; then, two local search algorithms have been implemented in the second phase to improve the initial solutions provided by NSGA-II.

In order to better show the research gaps, a number of well-known features studied in the literature are introduced in Table 1. Based on these features, we have compared recently published papers with each other in Table 2.

As shown in Table 2, none of these papers have simultaneously studied the features investigated in this paper. This table shows that our investigation is somehow similar to [19]. But these two studies differ from each other in a number of important respects. First, this investigation studies the DFLP and JSS problems; while [19] studied the combination of SFLP and JSS problems. Second, in contrast to their investigation, we have inequality of departments. Third, we have defined specific input and output points for each department. Fourth, this study considers the setup time of machines. Fifth, this paper evaluates the transportation delays in a more realistic way (explained in Subsection 3.1). Finally, this paper considers that each department might include more than a machine. The main contribution of this paper is threefold:

- The problems of the DFLP and JSS have been studied, simultaneously.
- A new approach for the assessment of CFR scores of departments has been proposed.
Table 1. The features and objectives studied in the literature.

| Problem | Rows | Features | Rows | Objectives |
|---------|------|----------|------|------------|
| FLP     | [F1] | Inequality of departments | [O1] | Material handling cost |
|         | [F2] | Input and output for departments | | Rearrangement cost of departments |
|         | [F3] | Multiple periods | | Desirability of closeness rating scores |
|         | [F4] | Continuous Optimization | | PUS |
|         | | | | Work in process |
| JSS     | [F5] | Setup time | [O6] | Makespan |
|         | [F6] | Transportation delay time | [O7] | Mean Flow Time (MFT) |
|         | [F7] | Multiple periods | [O8] | Earliness |
|         | [F8] | Due date of jobs | [O9] | Lateness |
|         | [F9] | Machine breakdown | | |

- A hybrid algorithm, incorporating NSGA-II and two local search algorithms, has been developed for this problem.

The rest of this paper is organized as follows. In Section 2, we have described the problem under consideration. Then, we have introduced the notations and proposed the mathematical model. The hybrid algorithm has been explained in Section 3. In Section 4, we have conducted a numerical study to validate the model and demonstrate the efficiency of the hybrid algorithm. Finally, Section 5 presents conclusions and future research directions.

2. The mathematical model.

2.1. The new approach for CFR scores. Closeness rating score cannot properly express the intention of facility layout planners. Thus, consideration of farness closeness rating score plays a crucial role for facility layout planners to define their preferences. For instance, if planners prefer to locate two departments neither close nor far from each other, closeness rating score could not perform properly alone. If we suppose two departments like paint (or melting and casting) and semi-manufactured metrical warehouse, the latter department is responsible to provide the required materials for the production line. The closeness of these two departments is undesirable due to fire danger; while farness of them could create difficulties in the provision of required materials for the production line. Therefore, both closeness and farness of these departments are undesirable. To address this issue, we have distinguished between CFR of departments using two different values, i.e., $CR_{ij}$ and $FR_{ij}$. These values are defined as follows.

$CR_{ij} = 1$ Closeness of departments $i$ and $j$ is absolutely important.

$CR_{ij} = 2$ Closeness of departments $i$ and $j$ is especially important.

$CR_{ij} = 3$ Closeness of departments $i$ and $j$ is important.

$CR_{ij} = 4$ Closeness of departments $i$ and $j$ is not important.

$FR_{ij} = 0$ Farness of departments $i$ and $j$ is not important.

$FR_{ij} = 1$ Farness of departments $i$ and $j$ is important.

$FR_{ij} = 2$ Farness of departments $i$ and $j$ is especially important.

$FR_{ij} = 3$ Farness of departments $i$ and $j$ is absolutely important.
Table 2. A summary of the features for a number of studies published recently.

| References | The facility layout problem | The scheduling problem | Solution Approach | Simultaneous optimization |
|------------|----------------------------|------------------------|-------------------|---------------------------|
|            | Problem features          | Objectives             |                   |                           |
| 8          | [F1][F2][F3][F4]          | [O1][O2][O3][O4][O5]  |                   |                           |
| 13         | † † † † †                | † † † † †             | PA, GA, and CSA   |                           |
| 29         | † † † † †                | † † † † †             | GA                |                           |
| 5          | † † † † †                | † † † † †             | CPLEX, NSGA-II, PSA, DEA |                           |
| 22         | † † † † †                | † † † † †             | GA, PSO, and VNS  |                           |
| 24         | † † † † †                | † † † † †             | CPLEX             |                           |
| 7          | † † † † †                | † † † † †             | GA & SA           |                           |
| 1          | † † † † †                | † † † † †             | PSO               |                           |
| 18         | † † † † †                | † † † † †             | Cloud-based SA    |                           |
| 10         | † † † † †                | † † † † †             | WL sampling algorithm |                           |
| 25         | † † † † †                | † † † † †             | GA                |                           |
| 3          | † † † † †                | † † † † †             | Math-heuristic    |                           |
| 6          | † † † † †                | † † † † †             | h-MoEDA           |                           |
| 23         | † † † † †                | † † † † †             | Q-factor          |                           |
| 2          | † † † † †                | † † † † †             | GRASP             |                           |
| 26         | † † † † †                | † † † † †             | GA                |                           |
| 17         | † † † † †                | † † † † †             | GA, and VNS       |                           |
| 19         | † † † † †                | † † † † †             | Barons and a hybrid algorithm |                           |
| Current study | † † † † †                | † † † † †             |                       |                           |

PA: Psychoclonal Algorithm, CSA: Chaotic Simulated Annealing, GA: Genetic Algorithm, TS: Tabu Search, PSA: Pareto Simulated Annealing, DEA: Differential Evolution Algorithm, PSO: Particle Swarm Optimization, VNS: Variable Neighborhood Search, SA: Simulated Annealing, GRASP: Greedy Randomized Adaptive Search Procedure
Then, the following equations determine the closeness and farness violations between departments.

\[
\begin{align*}
V_{ij}^C &= \max(0, -1 + \frac{|x_{it} - x_{jt}|}{C_{R_{ij}, D_{max}^t}/2}) \quad \forall i, j \in \mathcal{D} \text{ and } j > i, t \in \mathcal{T}, \text{ and } C_{R_{ij}} \neq 4 \\
V_{ij}^F &= \max(0, -1 + \frac{|y_{it} - y_{jt}|}{F_{R_{ij}, D_{max}^t}/2}) \quad \forall i, j \in \mathcal{D} \text{ and } j > i, t \in \mathcal{T}, \text{ and } F_{R_{ij}} \neq 0
\end{align*}
\]

(1)

\[
V_{ij}^C = 0, V_{ij}^F = 0 \quad \forall i, j \in \mathcal{D} \text{ and } j > i, t \in \mathcal{T}, \text{ and } C_{R_{ij}} = 4
\]

(2)

\[
\begin{align*}
V_{ij}^C &= \max(0, 1 - \frac{|x_{it} - x_{jt}|}{F_{R_{ij}, D_{max}^t}/3}) \quad \forall i, j \in \mathcal{D} \text{ and } j > i, t \in \mathcal{T}, \text{ and } F_{R_{ij}} \neq 0 \\
V_{ij}^F &= \max(0, 1 - \frac{|y_{it} - y_{jt}|}{F_{R_{ij}, D_{max}^t}/3})
\end{align*}
\]

(3)

\[
V_{ij}^C = 0, V_{ij}^F = 0 \quad \forall i, j \in \mathcal{D} \text{ and } j > i, t \in \mathcal{T}, \text{ and } F_{R_{ij}} = 0
\]

(4)

where \( \mathcal{D} \) and \( \mathcal{T} \) refer to the sets of departments and periods; \( D_{max}^t \) refers to the maximum distance between all departments at period \( t \). \( V_{ij}^C \) and \( V_{ij}^F \) refer to the closeness violation of departments \( i \) and \( j \) regarding x- and y-axes at period \( t \). \( V_{ij}^C \) and \( V_{ij}^F \) also refer to the farness violation of departments \( i \) and \( j \) regarding x- and y-axes at period \( t \). (1) and (2) determine the closeness violation of departments regarding x- and y-axes. To be more precise, (1) determines the closeness violation of departments \( i \) and \( j \), where \( C_{R_{ij}} \leq 3 \). While (2) determines the closeness violation of departments \( i \) and \( j \), where \( C_{R_{ij}} = 4 \). In the same way, (3) and (4) determine the farness violation of departments regarding x- and y-axes. (1) and (3) clarify why \( C_{R_{ij}} = 1 \) and \( F_{R_{ij}} = 3 \) relate to the cases that closeness and farness of departments are absolutely important, respectively. In fact, when closeness of departments is absolutely important, the denominators of the first and second terms in (1) have to be small. For this reason, \( D_{max}^t/3 \) is multiplied by \( C_{R_{ij}} = 1 \) to make the small values of \( |x_{it} - x_{jt}| \) and \( |y_{it} - y_{jt}| \) desirable. With the same reasoning, we can clarify why \( F_{R_{ij}} = 3 \) relates to the case that farness of departments is absolutely important.

2.2. Problem description. As stated, both problems of the DFLP and JSS are studied in this paper. Thus, this investigation aims to design a DFLP for \( I \) departments with unequal areas (where \( i \) and \( j \) are indices of departments, and \( i, j \in \{1, 2, \ldots, I\} \) over \( T \) periods (where \( t \) is the index of period, and \( t \in \{1, 2, \ldots, T\} \)). All these departments must be arranged within a predetermined Plant Floor (PF) that its length and width are equal to \( L \) and \( W \), respectively.

In this study, \( M \) machines (where \( m, n \) and \( l \) are indices of machines, and \( m, n, l \in \{1, 2, \ldots, M\} \)) process \( K \) jobs (where \( k \) and \( v \) are indices of jobs, and \( k, v \in \{1, 2, \ldots, K\} \)). It is already known that machine \( m \) belongs to which department using a parameter called \( \text{Loc}_m \) (\( \text{Loc}_m \in \{1, 2, \ldots, I\} \)). Each job includes \( M \) operations; it means that all jobs are processed on all machines.

The proposed formulation makes two main decisions, such as the arrangement of departments and scheduling of jobs. To be more precise, the first set of decision variables is related to the DFLP problem and determines the coordinate of departments over different periods regarding x- and y-axes, respectively. The second decision variable is related to the JSS problem and specifies the start time of jobs
over different periods. A complimentary table is provided in the following subsection to introduced all notations required to formulate the intended mathematical model.

2.3. Model formulation. To formulate the mathematical model, we introduce the notations as follows.

- **Sets:**
  - \( D \) Set of departments \( D = \{1, 2, \cdots, I\} \)
  - \( K \) Set of jobs \( K = \{1, 2, \cdots, K\} \)
  - \( M \) Set of machines \( M = \{1, 2, \cdots, M\} \)
  - \( T \) Set of periods \( T = \{1, 2, \cdots, T\} \)

- **Parameters:**
  - \( lx_i, ly_i \) Length and width of department \( i \), respectively
  - \( ax_i, ay_i \) Distance from the input point of department \( i \) to its centroid is equal to \( ax_i \) and \( ay_i \) regarding x- and y-axes, respectively
  - \( bx_i, by_i \) Distance from the output point of department \( i \) to its centroid is equal to \( ax_i \) and \( ay_i \) regarding x- and y-axes, respectively
  - \( TA \) Total area of all departments
  - \( FC_i \) Fixed cost of rearrangement and rotation for department \( i \)
  - \( VC_{1i}, VC_{2i} \) Variable costs of rearrangement and rotation for department \( i \), respectively
  - \( Vol_{ijt} \) Material flow between departments \( i \) and \( j \) at period \( t \)
  - \( MC \) Material handling cost (per each unit of material and each unit of distance)
  - \( \eta \) Conversion ratio of time and distance (sec/m)
  - \( SO_{kmnt} \) If job \( k \) is processed on machine \( m \) after machine \( n \) at period \( t \), \( 1 \); otherwise, \( 0 \).
  - \( P_{kmt} \) Process time of job \( k \) on machine \( m \) at period \( t \)
  - \( S_{ikmt} \) Setup time for exchanging jobs \( k \) and \( v \) on machine \( m \) at period \( t \)

- **Positive decision variables:**
  - \( x_{it}, y_{it} \) Coordinate of department \( i \) at period \( t \) regarding x- and y-axes, respectively
  - \( l_{it} \) The length of department \( i \) at period \( t \) with respect to its orientation
  - \( w_{it} \) The width of department \( i \) at period \( t \) with respect to its orientation
  - \( lx_{it}, ly_{it} \) Coordinates of the input of department \( i \) at period \( t \) regarding x- and y-axes, respectively
  - \( Ox_{it}, Oy_{it} \) Coordinates of the output of department \( i \) at period \( t \) regarding x- and y-axes, respectively
  - \( rm_{i(t−1)t} \) The amount of rearrangement for department \( i \) at period \( t \) as compared to period \( t−1 \)
  - \( \theta_{it} \) The orientation of department \( i \) at period \( t \) as compared to its initial orientation \( (\theta_{it} = 1, 2, 3, 4) \)
  - \( d_{ijt} \) The distance between the input of departments \( i \) and \( j \) at period \( t \) (1-norm distance)
  - \( dx'_{ijt}, dy'_{ijt} \) The distance between the center of departments \( i \) and \( j \) at period \( t \) regarding x- and y-axes, respectively
  - \( VC_{iij} \) The closeness violation between departments \( i \) and \( j \) at period \( t \)
  - \( VD_{ij} \) The farness violation between departments \( i \) and \( j \) at period \( t \)
  - \( X_{mint} \) The minimum distance of departments from the edge of the PF at period \( t \) regarding x-axis
  - \( X_{maxt} \) The maximum distance of departments from the edge of the PF at period \( t \) regarding x-axis
  - \( Y_{mint} \) The minimum distance of departments from the edge of the PF at period \( t \) regarding y-axis
  - \( Y_{maxt} \) The maximum distance of departments from the edge of the PF at period \( t \) regarding y-axis
The area of the smallest rectangle that contains all departments at period \( t \)

\[ sr_t \]

start time of job \( k \) on machine \( m \) at period \( t \)

\[ st_{kmt} \]

The transportation delay for moving between machines \( m \) and \( n \) at period \( t \)

\[ td_{mnt} \]

The completion time of job \( k \) at period \( t \)

\[ ct_{kt} \]

- Binary decision variables:
  
  \( Left_{ijt} \) If department \( i \) is on the left-hand side of department \( j \) at period \( t \), 1; otherwise, 0.
  
  \( Below_{ijt} \) If department \( i \) is below of department \( j \) at period \( t \), 1; otherwise, 0.
  
  \( re_{it} \) If department \( i \) has been rearranged at period \( t \), 1; otherwise, 0.
  
  \( O_{it}^G \) If the rotation of department \( i \) at period \( t \) is equal to \((G - 1) \times 90\) as compared to its initial orientation, 1; otherwise, 0. (where \( \forall G \in \{1, 2, 3, 4\} \))
  
  \( hv_{it} \) If department \( i \) has been rotated at period \( t \) and its coordinates have been modified due to rotation (where \( O_{1i}^2, O_{1i}^4, O_{3i}^1, \) or \( O_{4i}^1 = 1 \)), 1; otherwise, 0.
  
  \( ro_{it} \) If department \( i \) has been rotated at period \( t \) (where \( O_{1i}^1, O_{2i}^1, O_{3i}^2, \) or \( O_{4i}^2 = 1 \)), 1; otherwise, 0.
  
  \( sq_{kvmt} \) If job \( k \) precedes job \( v \) on machine \( m \) at period \( t \), 1; otherwise, 0.

The proposed model includes four objective functions; these objective functions are formulated as follows:

\[
\text{Min } Z_1 = \sum_{t \in \mathcal{T}} \sum_{i \in \mathcal{D}} \left( MC \cdot \Delta V_{ijt} \cdot d_{ijt} \right) + \sum_{t \in \mathcal{T}} \sum_{i \in \mathcal{D}} \left( FC_i \cdot re_{it} + VC_{Ai} \cdot rm_{i(it-1)t} \right) + \sum_{t \in \mathcal{T}} \sum_{i \in \mathcal{D}} \left( FC_i \cdot ro_{it} \cdot (1 - re_{it}) + VC_{2i} \cdot ro_{it} \right) 
\]

(5)

\[
\text{Min } Z_2 = \left( \sum_{t \in \mathcal{T}} \sum_{i \in \mathcal{D}} \sum_{j \in \mathcal{D}} V_{ijt} + V_{ijt}^F \right) / T
\]

(6)

\[
\text{Min } Z_3 = \frac{\sum_{t \in \mathcal{T}} sr_t - TA}{sr_t}
\]

(7)

\[
\text{Min } Z_4 = \frac{\sum_{k \in \mathcal{K}} cl_{kt}}{K}
\]

(8)

The first objective function minimizes costs. To be more precise, the first term in (5) refers to the material handling cost, the second term refers to the department rearrangement cost, and the third term refers to the department rotation cost. The second objective function minimizes the violation of closeness and farness of departments (i.e., maximization of the desirability of closeness and farness of departments). The third objective function minimizes the PUS on the PF over all periods. Finally, the last objective function minimizes the value of MFT over all periods. The following constraints limit these objective functions.

\[
\begin{align*}
  x_{it} - x_{i(it-1)} &\leq L \cdot re_{it} \\
x_{i(it-1)} - x_{it} &\leq L \cdot re_{it} \\
y_{it} - y_{i(it-1)} &\leq L \cdot re_{it} \\
y_{i(it-1)} - y_{it} &\leq L \cdot re_{it} \\
O_{it}^1 + O_{it}^2 + O_{it}^3 + O_{it}^4 &\leq 1
\end{align*}
\]

\forall i \in \mathcal{D}; t \in \mathcal{T} \text{ and } t > 1 \quad (9)

\[
\begin{align*}
O_{1i}^1 + O_{1i}^2 + O_{3i}^1 + O_{4i}^1 &\leq 1 \\
\end{align*}
\]

\forall i \in \mathcal{D}; t \in \mathcal{T} \quad (10)
\[ O_{it}^2 + O_{it}^4 = hv_{it} \quad \forall i \in \mathcal{D}; t \in \mathcal{T} \quad (11) \]

\[
\begin{align*}
\theta_{it} &= O_{it}^3 + 2O_{it}^2 \, 2 + 3O_{it}^2 + 4O_{it}^4 \\
\theta_{it} - \theta_{it(t-1)} &\leq 3 \cdot ro_{it} & \forall i \in \mathcal{D}; t \in \mathcal{T} \text{ and } t > 1 \quad (12)
\end{align*}
\]

(9) determines whether department \( i \) has been rearranged at period \( t \) (\( re_{it} = 1 \)) or not (\( re_{it} = 0 \)). In this constraint, the maximum value of the left-hand side could be equal to the length of PF, i.e., \( L \). (10) ensures that department \( i \) is in one of the four possible rotation conditions at period \( t \) (\( O_{it}^1, O_{it}^2, O_{it}^3 \), or; \( O_{it}^4 = 1 \)). (11) determines whether department \( i \) has been rotated at period \( t \) and its coordinates have been modified due to rotation (where \( O_{it}^2 \), or \( O_{it}^3 = 1 \)) or not. (12) determines the orientation of department \( i \) at period \( t \) (the first equation). Furthermore, it determines whether department \( i \) has been rotated at period \( t \) (\( ro_{it} = 1 \)) or not (\( ro_{it} = 0 \)) (the second and third equations). In this constraint, the maximum value of the left-hand side of the second and third equations could be equal to three.

\[
\begin{align*}
|l_{it} - x_{it} + (1 - hv_{it})| &= 1 \quad \forall i \in \mathcal{D}; t \in \mathcal{T} \quad (13)
\end{align*}
\]

(13) determines the actual length and width of department \( i \) at period \( t \) with respect to their rotation. (14) ensures that any two departments \( i \) and \( j \) cannot overlap each other at period \( t \). (15) also guarantees that department \( i \) cannot overstep outside the PF at period \( t \).

\[
\begin{align*}
\begin{cases}
& x_{it} = x_{it} + (O_{it}^1 - O_{it}^2) \cdot ax_i + (O_{it}^2 - O_{it}^3) \cdot ay_i \\
& y_{it} = y_{it} + (O_{it}^2 - O_{it}^3) \cdot ax_i + (O_{it}^1 - O_{it}^2) \cdot ay_i \\
& O_{xit} = x_{it} + (O_{it}^1 - O_{it}^2) \cdot bx_i + (O_{it}^2 - O_{it}^3) \cdot by_i \\
& O_{yit} = y_{it} + (O_{it}^2 - O_{it}^3) \cdot bx_i + (O_{it}^1 - O_{it}^2) \cdot by_i \\
& rm_{it(t-1)} = |x_{it} - x_{it(t-1)}| + |y_{it} - y_{it(t-1)}| \quad \forall i \in \mathcal{D}; t \in \mathcal{T} \text{ and } t > 1 \\
& dx_{ijt} = |x_{ijt} - x_{ij(t-1)}| + |y_{ijt} - y_{ij(t-1)}| \quad \forall i, j \in \mathcal{D} \text{ and } i \neq j; t \in \mathcal{T} \\
& dy_{ijt} = |y_{ijt} - y_{ij(t-1)}| \\
& D_{mx} = \max_{i,j \in \mathcal{D}} (dx_{ijt}, dy_{ijt}) \quad \forall t \in \mathcal{T}
\end{cases}
\end{align*}
\]

(16) calculates the coordinates of the input and output points for department \( i \) at period \( t \). The first two equations of this constraint determine coordinates of
the input for department $i$ at period $t$ regarding x- and y-axes, respectively. The second two equations also determine coordinates for the output of department $i$ at period $t$ regarding x- and y-axes, respectively. (17) determines the rearrangement of department $i$ at period $t$ as compared to period $t-1$. Moreover, (18) determines the distance between the output of departments $i$ and the input of department $j$ at period $t$ based on norm-1 distance. (19) calculates the distance between the center of departments $i$ and $j$ regarding x- and y-axes. (20) determines the closeness and farness violations of departments $i$ and $j$ at period $t$. (21) determines the closeness and farness violations of departments $i$ and $j$ at period $t$.

\[
\begin{align*}
V_{ijt}^C &= \min(V_{xijt}^C, V_{yijt}^C) \\
V_{ijt}^F &= \min(V_{xijt}^F, V_{yijt}^F)
\end{align*}
\quad \forall i, j \in D \text{ and } j > i; \ t \in T
\]  

(1)-(4)

As explained earlier, (1)-(4) determine the closeness and farness violations of departments regarding x- and y-axes. (21) determines the closeness and farness violations of departments $i$ and $j$ at period $t$.

\[
\begin{align*}
X_{\min t} &\leq x_{it} - 0.5l_{it} \\
X_{\max t} &\geq x_{it} + 0.5l_{it} \\
Y_{\min t} &\leq y_{it} - 0.5w_{it} \\
Y_{\max t} &\geq y_{it} + 0.5w_{it}
\end{align*}
\quad \forall i \in D; \ t \in T
\]  

(22)

(23) specifies coordinates of a rectangle that contains all departments at period $t$. (23) also calculates the area of this rectangle at period $t$.

\[
st_{it} = (X_{\max t} - X_{\min t}) \cdot (Y_{\max t} - Y_{\min t})
\quad \forall t \in T
\]  

As explained earlier, (1)-(4) determine the closeness and farness violations of departments regarding x- and y-axes. (21) determines the closeness and farness violations of departments $i$ and $j$ at period $t$.

\[
td_{mnt} = \eta \cdot d_{ijt} \quad \forall i, j \in D; \ m, n \in M; \ t \in T; \ i = \text{Loc}_m \text{ and } j = \text{Loc}_n
\]  

(24)

(25) specifies the start time of job $k$ on machine $m$ at period $t$. For this reason, this constraint must determine which of the first term or second term is greater than the other one. If job $k$ has to be processed on machine $m$ after job $v$ at period $t$. (26) specifies the start time of job $k$ on machine $m$ at period $t$. For this reason, this constraint must determine which of the first term or second term is greater than the other one. If job $k$ has to be processed on machine $m$ after job $v$, the first term refers to the completion time of job $v$ plus the setup time required to setup machine $m$ for job $k$. On the other hand, the second term refers to the finish time of job $k$ plus the transportation delay required to move job $k$ from machine $n$ to machine $m$. Figure 1 clarifies this issue. Figure 1a refers to the case that the first term is greater, and Figure 1b illustrates the case that the second term is greater. (27) determines the completion time of job $k$ at period $t$.

\[
\begin{align*}
Left_{ijt}, Below_{ijt}, re_{it}, ro_{it}, hv_{it}, O_{ijt}^G, sq_{ktm} \in \{0, 1\} \\
\end{align*}
\quad \forall i; j; k; v; m; t
\]  

(28)
Eventually, (28) determines the binary decision variables. (29) also specifies the positive decision variables.

### Figure 1. Two possible cases that could happen to determine the start time of a job

#### 2.4. Big coefficients. In the proposed formulation, two adequately large coefficients, such as $M_1$ and $M_2$, control (25) and (26) activate them in some cases. If small values are chosen for such coefficients, generation of infeasible solutions would be unavoidable. On the other hand, if too large values are chosen for these coefficients, round-off error would be possible ([15]). Therefore, choosing adequately large values is important for these coefficients.

$M_1$ is used in (25), e.g., $st_{kmt} \geq st_{vmt} + P_{vmt} - M_1 \cdot (1 - sq_{vkm})$. When $sq_{vkm} = 0$, we have $st_{kmt} \geq st_{vmt} + P_{vmt} - M_1$. In this equation, $M_1$ has to be greater than or equal to $st_{vmt} + P_{vmt}$. Since $st_{vmt}$ is not already known, there is not a specific value for $M_1$. For this reason, we first determine the maximum values of $td_{nmt}$ and $S_{vkm}$ for each period. Then, we calculate $(\sum_{k \in K} \sum_{m \in M} P_{kmt}) + [L + \frac{W}{\eta} + \max_{v,k,m} S_{vkm}]K(m - 1)$ for period $t$ and call it $worstmakespan_t$. Where $\max_{n,m}(td_{nmt}) = \frac{L + W}{\eta}$ and it refers to the case that the distance between machines $n$ and $m$ is equal to $L + W$. Note that we have multiplied $[L + \frac{W}{\eta} + \max_{v,k,m}(S_{vkm})]$ by $K(m - 1)$ since each of jobs is processed on all machines as mentioned earlier. Thus, each job includes $m - 1$ transportation delays and setup times. Eventually, we determine $\max_t(worstmakespan_t)$ and set $M_1$ equal to it. Using a similar rationale, $M_2$ will be set equal to $\max_t(worstmakespan_t)$.

#### 2.5. Linearization of $\min$ and $\max$ functions. The Baron solver could not handle $\min$ and $\max$ functions used in the proposed mathematical model. Thus, all those $\min$ and $\max$ functions must be linearized. In this model, (1)-(4), (20), (21) and (26) include $\min$ or $\max$ functions. In order to linearize the $\max$ functions of (1)-(4), we replace them with (30)-(33).

\[
\begin{align*}
V_{x_{ij}^C} & \geq -1 + \frac{|x_{ij} - x_{ji}|}{\max_t CR_{ij} \cdot D_{max_t}} & \forall i, j \in D \text{ and } j > i, t \in T, \text{ and } CR_{ij} \neq 4 \quad (30) \\
V_{y_{ij}^C} & \geq -1 + \frac{|y_{ij} - y_{ji}|}{\max_t CR_{ij} \cdot D_{max_t}} & \forall i, j \in D \text{ and } j > i, t \in T, \text{ and } CR_{ij} = 4 \quad (31) \\
V_{x_{ij} t^C} & = 0, V_{ij}^C = 0 & \forall i, j \in D \text{ and } j > i, t \in T, \text{ and } CR_{ij} = 4
\end{align*}
\]
3. The hybrid algorithm. For multi-objective optimization problems and generating Pareto solutions, [4] proposed a new variant of the traditional GA named 

\[
\begin{align*}
V_{ijt}^F & \geq 1 - \frac{|x_{ijt}^v - x_{ijt}^o|}{\max_{i,j} D_{ijt}} & \forall i, j \in D \text{ and } j > i, \ t \in T, \text{ and } FR_{ij} \neq 0 \\
V_{ijt}^F & \geq 1 - \frac{|y_{ijt}^v - y_{ijt}^o|}{\max_{i,j} D_{ijt}}
\end{align*}
\]

\[V_{ijt}^F = 0, \ V_{ijt}^F = 0 \quad \forall i, j \in D \text{ and } j > i, \ t \in T, \text{ and } FR_{ij} = 0 \] (33)

(20) can be linearized as (34). Using this equation, it is possible that \(D_{ijt}\) gets greater values than \(\max_i \ \{dx_{ijt}', dy_{ijt}'\}\). Consequently, we add \(D_{ijt} \cdot PQ\) to the third objective function to prevent this issue; \(PQ\) is a large penalty coefficient that forces \(D_{ijt}\) to get its least possible value, i.e., the real maximum value of \(dx_{ijt}'\) and \(dy_{ijt}'\). To report the final value of the third objective function, however, we have to obliterate \(D_{ijt} \cdot PQ\) from this equation and make sure that the reported value only relates to the closeness and farness violations. (21) also includes two \(\min\) functions. To linearize this constraint, we should use (35) and (36). Like (34), it is possible that \(V_{ijt}^C\) and \(V_{ijt}^F\) get less values than \(\min(V_{ijt}^C, V_{ijt}^F)\) and \(\min(V_{ijt}^C, V_{ijt}^F)\), respectively. To resolve this difficulty, we can apply a similar rationale introduced for (34).

\[
\begin{align*}
D_{ijt} & \geq dx_{ijt}' \\
D_{ijt} & \geq dy_{ijt}' \\
V_{ijt}^C & \leq V_{ijt}^C \\
V_{ijt}^C & \leq V_{ijt}^C \\
V_{ijt}^F & \leq V_{ijt}^F \\
V_{ijt}^F & \leq V_{ijt}^F
\end{align*}
\]

\(\forall i, j \in D \text{ and } j > i, \ t \in T\) (34)

\(\forall i, j \in D \text{ and } j > i, \ t \in T\) (35)

\(\forall i, j \in D \text{ and } j > i, \ t \in T\) (36)

Finally, (26) can be linearized by (37) and (38); where \(dv_{kmtn}\) is an auxiliary binary variable. When \(dv_{kmtn} = 1, (st_{vmt} + P_{vmt} + S_{vkmtn}) - M_2(1 - sq_{vkmtn})\) is greater than the other term. Otherwise, \(\sum_{n \in M} (SO_{knmt} \cdot (st_{knt} + P_{knt} + td_{nnt}))\) is greater.

\[
\frac{1}{2} \left( [(st_{vmt} + P_{vmt} + S_{vkmtn}) - M_2(1 - sq_{vkmtn})] - \left[ \sum_{n \in M} (SO_{knmt} \cdot (st_{knt} + P_{knt} + td_{nnt})) \right] \right) + \frac{1}{2} \left( [(st_{vmt} + P_{vmt} + S_{vkmtn}) - M_2(1 - sq_{vkmtn})] + \left[ \sum_{n \in M} (SO_{knmt} \cdot (st_{knt} + P_{knt} + td_{nnt})) \right] \right) \]

\[= dv_{kmtn} \cdot [(st_{vmt} + P_{vmt} + S_{vkmtn}) - M_2(1 - sq_{vkmtn})] + (1 - dv_{kmtn}) \cdot \left[ \sum_{n \in M} (SO_{knmt} \cdot (st_{knt} + P_{knt} + td_{nnt})) \right] \]

\[\forall k, v \in K \text{ and } k \neq v; \ m \in M; \ t \in T\] (37)

\[st_{kmtn} \geq dv_{kmtn} \cdot ((st_{vmt} + P_{vmt} + S_{vkmtn}) - M_2(1 - sq_{vkmtn})) + (1 - dv_{kmtn}) \sum_{n \in M} (SO_{knmt} \cdot (st_{knt} + P_{knt} + td_{nnt})) \]

\[\forall k, v \in K \text{ and } k \neq v; \ m \in M; \ t \in T\] (38)

3. The hybrid algorithm. For multi-objective optimization problems and generating Pareto solutions, [4] proposed a new variant of the traditional GA named
NSGA-II. This algorithm optimizes the solutions found for a problem based on the non-dominated sorting technique. NSGA-II is a posteriori multi-objective optimization method, which provides Pareto fronts. In such a posteriori algorithm, a set of efficient solutions for the problem are found; then, the decision-maker selects a solution that satisfies his preferences ([11]). On the other hand, local search algorithms apply a sequence of local modifications to an initial solution. These algorithms are implemented in an iterative manner and continue until no further modifications cannot improve the solution ([13]).

We have developed a new hybrid algorithm consisting of two phases. NSGA-II is applied in the first phase; two local search algorithms are deployed in the second phase. The main intention of the first phase is to provide high-quality initial solutions for the second phase of this algorithm in order to decrease the CPU time.

3.1. Solution representation. We have used a matrix-based solution representation including two main parts. The first part represents the solution relevant to the DFLP problem; while the second part represents the solution of the JSS problem. In this matrix-based representation, each row denotes the solution of a period. In the first part, an integer number in each cell indicates the location of a department on all candidate locations. This part includes \( A \) columns; where \( A \) is equal to the number of departments. Thus, the cell located on the first row and second column determines the location of the second department at the first period. Beside each number in each cell, an integer number (between one and four) also determines the rotation of departments. In the second part, a permutation of real numbers between zero and one determines the sequence of jobs for each period. We convert these numbers to integer values in order to make an order of them using the random key method. Further details about this method are provided in [28]. This part includes \( B \) columns; where \( B \) is equal to the number of jobs. Figure 2 shows an illustrative example of a solution representation for a problem with 12 departments, 20 jobs, and four periods.

| Dynamic Facility Layout Planning | Job Shop Scheduling |
|---------------------------------|---------------------|
| 8 | 4 | 5 | 2 | 7 | 4 | ... | 2 | 3 | 6 | 1 | 0.65 | 0.12 | 0.48 | ... | 0.69 |
| 2 | 2 | 4 | 1 | 16 | 2 | ... | 13 | 2 | 6 | 4 | 0.06 | 0.49 | 0.60 | ... | 0.81 |
| 12 | 1 | 13 | 4 | 15 | 4 | ... | 8 | 3 | 16 | 2 | 0.28 | 0.63 | 0.34 | ... | 0.16 |
| 12 | 1 | 10 | 2 | 8 | 3 | ... | 16 | 1 | 7 | 3 | 0.91 | 0.80 | 0.72 | ... | 0.59 |

*Figure 2.* An illustrative example of a solution with 12 departments and 20 jobs

To decode the values in the second part of the chromosome, a schedule builder is required. Thus, we have incorporated the schedule builders used by [25] and [19] and adapted them to the problem of this study. This schedule builder is capable of providing active schedules; a schedule is called active when no permissible left
**ALGORITHM 1:** Psudeo code of the schedule builder

```plaintext
input A chromosome C and a problem instance P
output The schedule built from chromosome C for instance P

01: A = \{k, 1 \leq k \leq K\} where A only includes operations with no prerequisite

while A ≠ ∅, do

02: ∀k ∈ A, let st_{km} = \text{max}(st_{vm} + P_{vm} + S_{vkm}, st_{kn} + P_{kn} + TD_{nm}); if v is the last job allocated to the machine m, and job k is processed on machine m after machine n, td_{nm} refers to the transportation delay between machines n and m

03: Determine the operation v ∈ A with the earliest completion time if it was scheduled in the current state, i.e., st_v + P_v ≤ st_k + P_k, ∀k ∈ A

04: Let m be the machine required by v, and C_s the subset of A whose operations require m

05: Remove from C_s every operation k that st_k ≥ st_v + P_v + S_{vkm}

06: Select k* ∈ C_s so that it is the leftmost operation of C_s in the chromosome sequence

07: Schedule k* as early as possible to build the partial schedule corresponding to the next state, i.e., r_{θ*} = st_{θ*}

08: Remove k* from A and insert the next operation of the job to A if k* is not the last operation of its job

end
```

shift can be applied to it ([9]). ALGORITHM 1 illustrates the pseudo code of the integrated schedule builder.

In contrast to [19], we have considered a more realistic form of the transportation delay between departments. They defined a threshold distance. If the distance between two departments is greater than or equal to the threshold distance, they considered a fixed delay time. But if the distance is less than the threshold distance, they considered no delay time. This is not realistic since the delay time between different departments placed in different locations could vary. For example, we consider that the distance between department A and B is longer than the distance between department A and C, respectively. Therefore, the transportation delay of the former case is greater. In this study, we have determined the transportation delay based on the average speed of the transportation and the distance between departments.

3.2. First phase: NSGA-II. To implement NSGA-II, we have applied the single-point, two-point, and uniform crossover operators to both the DFLP and JSS sections in the chromosome. These operators have been randomly applied based on the Uniform distribution. As stated previously, we have used the matrix-based solution representation; where each row refers to the solution of a distinct period. Accordingly, we have separately applied the crossover operator to each period. On the other hand, we have randomly applied the insertion, swap and reverse operators for both of the DFLP and JSS sections in the chromosome. As the same as the crossover operator, we have separately applied the mutation operator for each of periods.

To improve the performance of NSGA-II, we have embedded three local search algorithms. The first algorithm is called the department swapping local search. This
algorithm initially generates a random permutation for the pairs of departments. According to this permutation, the algorithm changes the location of two departments for each period separately. If the new solution dominates the incumbent one, it replaces the incumbent solution with the new one. Otherwise, it continues to evaluate other periods for the same pair of departments or other pairs of departments. For more details, interested readers are referred to [1]. The second algorithm is called the rotation local search algorithm. First, this algorithm creates a random permutation of departments. Then, it starts with the first department on the list and evaluates all its possible rotations to generate a better solution. This procedure will be applied to all other departments on the list. Like the previous local search algorithm, if the new solution dominates the incumbent one, the algorithm replaces the incumbent solution with the new one. The third algorithm is called the period local search. This algorithm changes the solution of two periods with each other. If the new solution dominates the incumbent one, it replaces the incumbent solution with the new one. Otherwise, it continues to evaluate other pairs of periods. For more details, interested readers are referred to [1]. Since all of these algorithms evaluate all possible changes (like a whole-numeration algorithm) and are costly in terms of the CPU time, we only apply them in the last iteration of NSGA-II.

In the first phase, we have considered this problem as a QAP and solved it using NSGA-II. Thus, this phase should optimize the problem according to three of the objective functions, i.e., material handling, machines rearrangement and rotation costs, closeness and farness violations, and MFT of jobs. It is evident that the minimization of the PUS is not practical for this phase since the candidate locations of departments are determined as discrete points. To apply the QAP, the following three steps have to be implemented.

Step 1: Determine the square root of a square number greater than or equal to the number of departments, and call it as $SR$.
Step 2: Divide the width and length of PF to $SR$ equal segments.
Step 3: Specify the candidate locations of departments to the center of each segment.

Figure 3a shows an illustrative example of the candidate locations for a problem with 12 departments. For this example, $SR$ is equal to four ($16 \geq 12 \Rightarrow SR = \sqrt{16} = 4$). Furthermore, Figure 3b also demonstrates the arrangement of departments for the same illustrative example.

3.3. Second phase: Local search algorithms. In contrast to the first phase, the second phase is the continuous optimization of the DFLP problem. Consequently, the minimization of PUS is considered in this phase. Since the departments can overlap each other in the continuous optimization of the DFLP, we have considered a penalty function beside other objective functions as follows:

$$\text{Min} \ Z = w_1 \cdot \left( \frac{Z_1^l - Z_1^u}{Z_1^l - Z_1^u} \right) + w_2 \cdot \left( \frac{Z_2^l - Z_2^u}{Z_2^l - Z_2^u} \right) + w_3 \cdot \left( \frac{Z_3^l - Z_3^u}{Z_3^l - Z_3^u} \right) + w_4 \cdot \left( \frac{Z_4^l - Z_4^u}{Z_4^l - Z_4^u} \right) + Z_{\text{pen}} \quad (39)$$

where $w_i$ denotes the importance of objective function $i$, $Z_i$ refers to the value of objective function $i$, $Z_i^l$ refers to a lower bound value for objective function $i$, and $Z_i^u$ denotes an upper bound value for objective function $i$.

As mentioned, we have considered the minimization of PUS in the second phase. To calculate the PUS, we have proposed a new method. The new method draws multiple rectangles; then, it minimizes the weighted PUS of all rectangles. But the
traditional method only draws the smallest rectangle, which contains all departments. Then, it simply calculates the area of the smallest rectangle and compares it with the total area of all departments (like (7)). Using the new method leads to decrement in the PUS. This method has been explained in the following:

Step 1: Determine the coordinates of departments’ sides, e.g., Figure 4a.
Step 2: Sort the coordinates of sides. To do this, sort the coordinates of the left, right, up, and downside in ascending and descending orders, respectively, e.g., Figure 4b.
Step 3: If the total number of departments is odd, \((I + 1)/2\) first values of the coordinates sorted previously will be selected. Otherwise, \(I/2\) first values will be selected, e.g., Figure 4b.
Step 4: Draw rectangles for the set of coordinates to determine the PUS, e.g., Figure 4c.

Local search for layout. To search the neighborhood of a solution with respect to the DFLP problem, we have defined nine possible movements and four possible rotations. In the case of movement, it is possible to modify the coordinates of any of the departments by \(\delta\) units according to Figure 5a. As the algorithm goes to the next iteration and finds a better solution, the value of \(\delta\) decreases. Otherwise, we do not change the value of \(\delta\). Since we expect the departments get closer to each other in greater iterations, decreasing the value of \(\delta\) is practical to generate newer feasible solutions. Moreover, in the case of rotation, it is possible to rotate departments as shown in Figure 5b.

This algorithm first assigns a distinct number to each department in each period. For example, if a problem includes four departments and three periods, 1 to 4 refers to the number of departments at the first period. 5 to 8 refers to the number of departments at the second period. 9 to 12 refers to the number of departments at the third period. Second, we review each of departments and recognize departments
(A) The equivalent coordinate of sides of departments on the side of PF

|   | 1   | 3.5 | 6   | 10  |
|---|-----|-----|-----|-----|
| Left side | 1   |     |     |     |
| Right side |     |     | 21  | 20.6|
| Down side | 29.2| 27.2| 24.2| 18.1|
| Up side | 4   | 5   | 12  | 12.5|

(b) Sorted values of aforementioned coordinates

(c) Rectangles required for calculation of PUS

Figure 4. An illustrative example of the calculation of PUS
that have not rearranged in consecutive periods. We put all such departments in identical groups. For instance, if the first department of each period (1, 5 and 9) has an identical location in the second and third periods, we put the number 1 in a group and numbers 5 and 9 in one another group. The algorithm generates a random permutation of these groups. Then, it starts with the first group of this permutation to generate new solutions by applying all possible movements and rotations introduced to each department. Given that there are nine possible movements and four possible rotations, the algorithm generates 36 solutions for each department in each period (the incumbent solution is one of them); the algorithm stores all these solutions and compares them. In fact, it recognizes the best solution and replaces the incumbent solution with the best one. Grouping here allows us to evaluate such departments in consecutive iterations of the algorithm and have the chance to keep their location identical. This means that the algorithm will be more efficient to avoid unnecessary costs of rearrangement. The termination condition is defined based on the number of consecutive iterations that the algorithm could not find a better solution. These steps are summarized in ALGORITHM 2.

**Local search for scheduling.** To search the neighborhood of a solution with respect to the JSS problem, we have developed a local search algorithm for the scheduling problem. This algorithm takes the solution provided by the local search of the layout and uses two operators to improve it in an iterative manner. One of these operators is the reverse mutation operator, and another one is a new operator named the additive operator. To apply these operators, we select one of them randomly based on the Uniform distribution. The additive operator randomly selects two points for each period in the scheduling part of the chromosome. Then, this operator generates a random number between zero and one and adds it to all values between those two points. Moreover, if the additive operator is applied, it is possible that some numbers in the chromosome become greater than one. Since we should keep all values between zero and one, therefore, we subtract one from those values. The pseudo code for this algorithm is explained in ALGORITHM 3.

Figure 6 illustrates a flowchart for the hybrid algorithm.
Figure 6. The flowchart of the second phase of the hybrid algorithm (local search algorithms)
**ALGORITHM 2:** Psudeo code of the local search for layout

**input** The incumbent solution \((IS_l)\) and the initial value that a department can move \((x)\)

1. Determine the objective function of \(IS_l\) and call it \(F_{IS_l}\)
2. Recognize departments that have not rearranged in consecutive periods and put all such departments in identical groups (where \(NG\) refers to the number of groups)
3. Create a random permutation of these groups
4. \(it=1\)
   - **while** \(it \leq N1\) (\(N1\) is a constant defined by the user)
     - **for** \(i = 1 : NG\) (to evaluate possible rearrangements and rotations for departments over all periods)
       - **for** \(j = 1 : 9\) (to evaluate nine possible rearrangements)
         - **for** \(k = 1 : 4\) (to evaluate four possible rotations)
           - Rearrange department \(i\) based on movement \(j\)
           - Rotate department \(i\) based on rotation \(k\\)
           - \((Sol_{jk})\)
     - **end**
   - **end**
   - Determine the best solution among \(Sol_{jk}\) and name it \(BS\)
   - **if** \(F_{BS} < F_{IS_l}\)
     - Apply the local search algorithm for scheduling to \(BS\)
     - \(x = x - 0.5/N1\)
   - **else**
     - \(it \leftarrow it + 1\)
   - **end**
   - **end**

3.4. **Penalty function.** In the DFLP problem, the overlap violation among departments has to be controlled. To handle the possible violations of this constraint, we have added a penalty function, (40), to the aggregated objective function as follows.

\[
\text{Min } Z_{\text{pen}} = \frac{1}{T} \left( \sum_{t=1}^{T} \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{OV_{ijt}}{N} \right) \quad (40)
\]

subject to:

\[
\begin{align*}
OV_{x_{ijt}} &= \max(0, 1 - \frac{|x_{it} - x_{jt}|}{(l_{it} + l_{jt})/2}) & \forall i, j > i, t \\
OV_{y_{ijt}} &= \max(0, 1 - \frac{|y_{it} - y_{jt}|}{(w_{it} + w_{jt})/2}) & \forall i, j > i, t \\
OV_{ijt} &= \min(OV_{x_{ijt}}, OV_{y_{ijt}}) & \forall i, j > i, t 
\end{align*}
\] (41) (42)

where \(OV_{x_{ijt}}\) and \(OV_{y_{ijt}}\) refer to the overlap violation of departments \(i\) and \(j\) at period \(t\) with respect to x- and y-axes, respectively. \(OV_{ijt}\) also refers to the
Algorithm 3: Pseudo code of the local search for scheduling

```
input  The solution provided by the local search for layout (BS)
01: Determine the objective function of BS and call it $F_{BS}$
02: $it = 1$
   while $it \leq N2$ (N2 is a constant defined by the user)
      for $j = 1 : T$ (to apply operators on the schedule of each of periods)
         03: Select one of the operators randomly
         04: Apply the operator selected in the last step on the schedule of period $j$
      end
      05: Determine the objective function of the new solution and call it $F_{new}$
      If $F_{new} < F_{BS}$
         $BS \leftarrow S_{new}$
         06: $it \leftarrow 1$
      else
         07: $it \leftarrow it + 1$
      end
   end
```

Overlap violation of departments $i$ and $j$ at period $t$. In (42), we have selected the minimum value of violation occurred regarding x- and y-axes due to two reasons. First, if departments $i$ and $j$ do not overlap each other regarding x- and y-axes, simultaneously ($OV_{ijt} = 0$ or $OV_{jyt} = 0$), they do not overlap each other, i.e., Scenario †. Figure 7a clarifies this statement. Second, selecting the minimum value of violation regarding x- and y-axes helps the algorithm to implement slight changes and make an infeasible solution feasible, i.e., Scenario ‡. Figure 7b also clarifies this issue.

![Figure 7](image-url)
To evaluate the proposed formulation, we have studied some randomly generated instances in the following section.

4. Numerical example. As noted by [19], there is not a real case study in the literature for this problem. Based on [12], [7], and OR-Library\(^1\), we have randomly generated 20 instances of different sizes. The general specifications of these instances have been summarized in Table 3.

In all of these instances, we have considered that the demand for products follows a normal distribution or a triangular distribution. Table 4 provides the demand for products, which have been extracted from [7]. This is evident that the demand for products determines material flow between departments. For the sake of brevity, we do not provide the rest of the input data here. Thus, we stored them on the web at the address www.bitbucket.org/Pro_Data/instances_2. The interested readers can use these data for future comparisons.

### Table 3. Specifications of randomly generated instances.

| Size of instances | Instance (No. of periods) | No. of departments | No. of machines | No. of jobs |
|-------------------|---------------------------|--------------------|-----------------|-------------|
| Small             | 1 (2), 11 (3)             | 3                  | 3               | 3           |
|                   | 2 (2), 12 (3)             | 4                  | 5               | 5           |
|                   | 3 (2), 13 (3)             | 5                  | 7               | 7           |
| Medium            | 4 (2), 14 (3)             | 6                  | 9               | 9           |
|                   | 5 (2), 15 (3)             | 8                  | 11              | 11          |
|                   | 6 (2), 16 (3)             | 10                 | 13              | 13          |
| Large-scale       | 7 (2), 17 (3)             | 12                 | 16              | 16          |
|                   | 8 (2), 18 (3)             | 14                 | 19              | 19          |
|                   | 9 (2), 19 (3)             | 16                 | 21              | 21          |
|                   | 10 (2), 20 (3)            | 18                 | 23              | 23          |

### Table 4. The demand for products over different periods ([7])

| Product | Period      | 1 (*10)       | 2 (*10)       | 3 (*10)       |
|---------|-------------|---------------|---------------|---------------|
| 1       | T(250, 280, 300) | T(40, 50, 60) | T(40, 50, 60) |
| 2       | T(70, 75, 90)   | T(350, 400, 430)| T(110, 125, 135)|
| 3       | N(5, 56)      | N(2, 55)      | N(20, 550)    |
| 4       | N(4, 40)      | N(4, 50)      | N(4, 70)      |

4.1. Parameter tuning. The performance of meta-heuristic algorithms depends on the tuning of their parameters significantly. Therefore, we have used the Taguchi experimental design method to tune the parameters of NSGA-II. To do this, we have evaluated four parameters, such as the maximum number of iteration, initial population size, crossover probability, and mutation probability. For each of these parameters, we have defined three possible levels. These levels have been provided in Table 5. To increase the accuracy of experiments, we have distinguished between the sizes of instances. Consequently, we have designed three different experiments

\(^1\)http://people.brunel.ac.uk/~mastjjb/jeb/info.html
for each of small, medium, and large-scale instances. The optimal setting for the parameters of NSGA-II has been provided in Table 6.

| Parameter          | Level of parameters | Small size | Medium size | Large-scale |
|--------------------|---------------------|------------|-------------|-------------|
|                    | I       | II      | III         | I   | II     | III     | I        | II     | III     |
| Iteration          | 60   | 80     | 100        | 80  | 100    | 120     | 100      | 150    | 200     |
| Initial population | 10   | 20     | 30         | 30  | 40     | 50      | 80       | 100    | 120     |
| $C_p$              | 0.7   | 0.8    | 0.9        | 0.7 | 0.8    | 0.9     | 0.7      | 0.8    | 0.9     |
| $M_p$              | 0.1   | 0.2    | 0.3        | 0.1 | 0.2    | 0.3     | 0.1      | 0.2    | 0.3     |

Table 5. The levels of parameters defined for experiments

Table 6. The optimal setting for the parameters of NSGA-II

| Parameter | Size of instances |
|-----------|------------------|
|           | Small | Medium | Large-scale |
| Iteration | 60   | 100    | 150        |
| Initial population | 20   | 30     | 100        |
| $C_p$     | 0.7   | 0.7    | 0.8        |
| $M_p$     | 0.2   | 0.3    | 0.3        |

4.2. Impact of the new proposed method for the PUS. In Subsection 3.3, we proposed a new method to calculate the PUS of the PF. To show the efficacy of this method, this subsection aims to compare the traditional and proposed methods of the PUS. For this reason, we adapt the hybrid algorithm to solve instances based on a single objective function like the PUS. Afterward, we optimize Instances 4, 7, 10, 13, 16 and 19 using both methods. Since the objective functions of both methods are not comparable, we first solve instances based on both types of objective functions. Second, we convert one of them to the other one and make them comparable. It is worth mentioning that both the traditional and proposed methods are explained in Subsection 4.1. The results obtained for these two methods are provided in Table 7.

Table 7. The comparison of the traditional and proposed method of the PUS

| Method                  | Instance | 4    | 7    | 10   | 13   | 16   | 19   |
|-------------------------|----------|------|------|------|------|------|------|
| Traditional method (%)  |          | 35.2 | 48.8 | 57.3 | 34   | 55.6 | 51.6 |
| Proposed method (%)     |          | 38.9 | 41.1 | 49.8 | 35.3 | 42   | 37.6 |
| Gap (%)                 |          | -10.6| 15.9 | 13.1 | -3.8 | 24.3 | 27   |

Table 7 shows that the proposed method outperforms the traditional one over the majority of instances. To be more precise, except Instances 4 and 13, the performance of the proposed method is better. Since Instances 7, 10, 16 and 19 includes more departments as compared to Instances 4 and 13, it can be argued that the proposed method outperforms the traditional one over larger size instances.
This table also indicates the difference between these two methods is considerable. In fact, the proposed method could potentially reduce the PUS by about 11% on average as compared to the traditional method. This is considerable and shows the efficacy of the proposed method.

4.3. Computational results.

**Validation of the hybrid algorithm.** To evaluate the quality of solutions of the proposed model, we have coded the model in GAMS 24.1.2 and used the Baron solver to solve instances. In this paper, we have used the epsilon constraint method. This method optimizes the problem based on merely one of the objective functions while considering the rest of the objective functions as constraints. For more details, interested readers are referred to Mavrotas (2009). In the case of the hybrid algorithm, we have coded this algorithm in Matlab 2016. Then, we have used a laptop with a 2.3 GHz CPU and 4GB of RAM to solve all instances.

First, we have solved all instances by the Baron solver. Based on the computational results, the Baron solver could only solve Instance 1 in less than 200 minutes. Then, we have compared the Pareto front found by the Baron solver and the hybrid algorithm for Instance 1. This comparison aims to show the efficiency of the hybrid algorithm. The results are provided in Table 8.

The Baron solver and the hybrid algorithm generated 11 Pareto solutions. If we more focus on the first objective function, based on Table 8, the Baron solver could generate four values of about 227,700 for the first objective function (rows 4, 6, 8 and 9). The hybrid algorithm could also generate five values of about 223,500 for the first objective function (rows 1, 2, 5, 7 and 11). This shows that the hybrid algorithm outperforms the Baron solver by about 4,200 units with respect to the first objective function. However, regarding the rest of objective functions, the Baron solver outperformed the hybrid algorithm by about 0.3 units in the violation of the closeness and farness, 0.04 and 1.3 units in the second, third and fourth objective functions, respectively. Based on the first objective function, the hybrid algorithm outperforms the Baron solver by about 17% on average. This is while the Baron solver outperforms the hybrid algorithm by about 50% on average with respect to the second objective function. Comparison of other objective functions for both solution approaches demonstrates negligible difference.

**Discrete and continuous optimization.** As stated, the hybrid algorithm includes two phases. In the first phase, the problem is the discrete optimization of the DFLP. In the second phase, the problem is the continuous optimization of the DFLP.

In order to demonstrate the effectiveness of the second phase, Instances 18, 19 and 20 have been reviewed. Based on the results, the second phase improves the first one by about 40% regarding the first objective function, 20% regarding the second objective function, and 11% regarding the fourth objective function. Due to discrete layout, distances between departments are long in the first phase; while the layout of departments in the second phase is continuous and it decreases the distance between departments. Thus, the material handling costs, farness violation between departments and transportation delay between departments reduce. This leads to the improvement of the objective functions. For more clarification, Figures 8-11 illustrate this issue.
Table 8. Pareto solutions found by the Baron solver and the hybrid algorithm for Instance 1

| Row | Baron solver | Hybrid algorithm |
|-----|--------------|------------------|
|     | Obj. 1       | Obj. 2 | Obj. 3 (%) | Obj. 4 | Obj. 1 | Obj. 2 | Obj. 3 (%) | Obj. 4 |
| 1   | 596,320.6    | 0.2777 | 0.68        | 23.3751 | 223,500 | 0.58   | 24.9        | 22.851 |
| 2   | 441,669.9    | 0      | 3.33        | 22.1017 | 224,616.6 | 0.57   | 24.7        | 22.798 |
| 3   | 482,948.3    | 0.2148 | 3.33        | 22.1017 | 236,688.6 | 0.555  | 24.2        | 22.764 |
| 4   | 227,695.7    | 0.2838 | 20.44       | 21.5836 | 249,431.5 | 0.54   | 23.1        | 22.693 |
| 5   | 269,847.4    | 0.2532 | 20.93       | 21.3166 | 223,500 | 0.6    | 25.7        | 21.854 |
| 6   | 227,756.6    | 0.28528| 20.44       | 21.5832 | 236,688.6 | 0.555  | 24.2        | 22.764 |
| 7   | 351,791.2    | 0.40051| 9.68        | 21.9565 | 223,500 | 0.6    | 20.3        | 22.379 |
| 8   | 268,928.5    | 0.25386| 20.76       | 21.3141 | 375,100 | 0.3    | 25.3        | 22.903 |
| 9   | 228,763.6    | 0.2868 | 20.45       | 21.5826 | 300,388.8 | 0.494  | 20.6        | 21.854 |
| 10  | 228,571.3    | 0.2871 | 20.38       | 21.5841 | 379,873.8 | 0    | 20.6        | 21.64 |
| 11  | 360,249.7    | 0.1717 | 40.14       | 22.5674 | 223,500 | 0.786  | 20          | 21.645 |
Separate and simultaneous optimization of the DFLP and JSS. As stated, one of the main contributions of this paper is the simultaneous optimization of the DFLP and JSS problems. We aim to show that the optimization of these two problems at the same time increases the quality of solutions as compared to their separate optimization. For the separate optimization, we first solve all 20 instances for the DFLP problem. Then, we use these results to solve the JSS problem for all instances. In the following, we have compared the results found by the separate optimization and simultaneous optimization. For this reason, we use four popular assessment metrics for the comparison of Pareto fronts:

**Quality Metric (QM):** This metric creates a pool of Pareto solutions by combining the entire non-dominated solutions of two Pareto fronts. Then, QM determines the non-dominated solutions of the pool and calculates the percentage of non-dominated solutions for each Pareto front. For this metric, higher values are better.

**Mean Ideal Distance (MID):** This metrics calculates the distance between the best possible solutions and the Pareto solutions. For this metric, lower values are better.
Figure 9. Illustration of the solution found for the continuous facility layout of Instance 15

**Diversification Metric (DM):** This metric determines the diversity of solutions in a Pareto front. For this metric, higher values are better.

**Spacing Metric (SM):** This metric calculates the uniformity of the spread of non-dominated solutions in a Pareto front. For this metric, smaller values are better.

The interested reader is referred to [16] for more details. The comparison of these two cases has been provided in Table 9.

Table 9 shows that simultaneous optimization provides better solutions in terms of the QM and MID. In fact, this type of optimization could potentially outperform the separate optimization by about 36% regarding the QM and 14% regarding the MID. Whereas the separate optimization provides better solutions regarding the DM and SM. According to the table, the separate optimization could potentially outperform the simultaneous optimization by about 3% regarding the DM and 53% regarding the SM. We apply the 1-Sample t test in order to increase the reliability of our conclusion. The results of this hypothesis test evaluate the meaningfulness of difference between each assessment metric for both cases. The results have been illustrated in Figure 12.
Figure 10. Illustration of the solution found for the scheduling of Instance 15 at period 1 (Phase 1).
Figure 11. Illustration of the solution found for the scheduling of Instance 15 at period 1 (Phase 2).
Table 9. The comparison of the solutions’ quality for both the separate optimization and simultaneous optimization

| Instance | QM | MID | DM | SM |
|----------|----|-----|----|----|
|          | Sep. | Sim. | $d_1$ | Sep. | Sim. | $d_2$ | Sep. | Sim. | $d_3$ | Sep. | Sim. | $d_4$ |
| 1        | 0.600 | 1 | 0.400 | 0.975 | 0.781 | 0.194 | 1.310 | 1.933 | 0.623 | 0.655 | 1.151 | -0.496 |
| 2        | 0.600 | 1 | 0.400 | 1.010 | 0.586 | 0.424 | 1.906 | 0.739 | -1.167 | 1.454 | 1.730 | -0.276 |
| 3        | 0.500 | 0.750 | 0.250 | 0.994 | 1.269 | -0.275 | 1.213 | 1.967 | 0.754 | 0.464 | 0.548 | -0.084 |
| 4        | 0.555 | 0.888 | 0.333 | 1.241 | 1.141 | 0.100 | 1.337 | 1.479 | 0.142 | 0.610 | 0.861 | -0.251 |
| 5        | 0.500 | 1 | 0.500 | 1.902 | 1.432 | 0.470 | 1.666 | 1.479 | -0.187 | 1.037 | 1.524 | -0.487 |
| 6        | 0.428 | 0.714 | 0.286 | 1.342 | 1.286 | 0.056 | 1.555 | 1.294 | -0.261 | 0.504 | 0.677 | -0.173 |
| 7        | 0.875 | 0.375 | -0.500 | 1.107 | 1.287 | -0.180 | 1.576 | 1.461 | -0.115 | 0.415 | 0.985 | -0.570 |
| 8        | 0.500 | 1 | 0.500 | 0.893 | 0.624 | 0.269 | 1.324 | 1.636 | 0.312 | 0.602 | 0.854 | -0.252 |
| 9        | 0.600 | 1 | 0.400 | 1.365 | 1.017 | 0.348 | 1.521 | 1.241 | -0.280 | 0.439 | 0.950 | -0.511 |
| 10       | 0.555 | 0.875 | 0.320 | 1.698 | 1.205 | 0.493 | 1.722 | 1.625 | -0.097 | 0.520 | 0.991 | -0.471 |
| 11       | 0.666 | 1 | 0.334 | 1.031 | 0.743 | 0.288 | 0.883 | 1.397 | 0.514 | 0.999 | 1.986 | -0.987 |
| 12       | 1 | 1 | 0 | 0.863 | 1.041 | -0.178 | 1.068 | 0.883 | -0.185 | 0.080 | 1.278 | -1.198 |
| 13       | 0.500 | 1 | 0.500 | 2.631 | 2.115 | 0.516 | 1.536 | 1.625 | 0.089 | 0.268 | 0.790 | -0.522 |
| 14       | 0.666 | 1 | 0.334 | 1.656 | 1.328 | 0.328 | 1.658 | 1.031 | -0.627 | 0.771 | 0.790 | -0.019 |
| 15       | 0.666 | 0.666 | 0 | 1.246 | 1.101 | 0.145 | 1.521 | 1.677 | 0.156 | 0.950 | 0.721 | 0.229 |
| 16       | 0.666 | 0.500 | -0.166 | 1.462 | 1.482 | -0.020 | 1.409 | 1.324 | -0.085 | 0.537 | 0.746 | -0.020 |
| 17       | 0.666 | 0.666 | 0 | 0.877 | 1.077 | -0.200 | 1.624 | 1.359 | -0.265 | 0.357 | 0.472 | -0.115 |
| 18       | 0.500 | 0.777 | 0.277 | 0.645 | 0.639 | 0.006 | 1.446 | 1.365 | -0.081 | 0.698 | 0.685 | 0.013 |
| 19       | 0.833 | 1 | 0.167 | 1.791 | 1.450 | 0.341 | 1.552 | 1.701 | 0.149 | 0.578 | 0.773 | -0.195 |
| 20       | 0.666 | 0.888 | 0.222 | 1.308 | 0.812 | 0.496 | 1.291 | 1.105 | -0.186 | 0.661 | 0.808 | -0.147 |
| Average  | 0.626 | 0.854 | 0.227 | 1.301 | 1.120 | 0.181 | 1.455 | 1.415 | -0.039 | 0.629 | 0.960 | -0.336 |
| Gap (%)  | 36.42 | 13.91 | -2.74 | -52.62 |
According to Figure 12a, the simultaneous optimization meaningfully outperforms the separate optimization regarding the QM. The same explanation applies to the MID (Figure 12b). Figure 12d demonstrates the separate optimization significantly outperforms the simultaneous optimization regarding the SM. Finally, Figure 12c illustrates that there is no significant difference between the performances of two types of optimization with respect to the DM. For conclusion, the simultaneous optimization provides better solutions regarding two assessment metrics, including QM and MID; whereas the separate optimization only provides better solutions with respect to one assessment metrics, i.e., SM. Since MID is an important metric that shows the quality of Pareto solutions in terms of the objective function values, it can be argued that the overall performance of the simultaneous optimization is better as compared to the separate optimization.

In the following, we intend to better analyze how the simultaneous optimization affects the quality of solutions, particularly in terms of the MFT. As stated earlier, the separate optimization includes two separate steps. The first step optimizes the
DFLP problem, and the second step optimizes the JSS problem. Because all objective functions of the DFLP problem are considered in the first step of the separate optimization, we do not compare the simultaneous optimization and separate optimization with respect to these objective functions. But the MFT is the merely objective function of the second step in the separate optimization which is curbed to the solution of the first step. Consequently, it is important to realize how the separate optimization deteriorates the MFT. Table 10 compares these two types of optimization cases with respect to the MFT.

Table 10 shows that the simultaneous optimization could improve the average value of the MFT over all instances. This table shows that the simultaneous optimization could potentially improve the separate optimization by about 10% percent on average regarding the MFT. In the separate optimization, the scheduling of jobs is determined based on a predetermined layout of departments. If the material flow between departments varies based on the variation of demand, the replacement of departments is not possible in the second step. Under this circumstance, the transportation delay between departments would become long; this leads to the increment in the MFT.

5. **Conclusions and future research directions.** In this paper, we tried to study the dynamic essence of manufacturing systems. For this purpose, we proposed a multi-objective model to optimize the DFLP and JSS problems, simultaneously. This model evaluates both closeness and farness rating scores of departments. The results showed that the consideration of these two scores affects the layout of department. This paper studied the input and output points of each department, which is crucial for the establishment of practical facility layouts in the real world. To solve the proposed model, an exact method (Baron solver) was used for small size instances. Also, a hybrid algorithm incorporating NSGA-II and two local search algorithms was proposed for large-scale instances. The results indicated that the hybrid algorithm provides comparable solutions in comparison to the Baron solver. In the comparison of these solutions approaches, the hybrid algorithm outperformed the Baron solver in terms of the first objective function (material handling, rearrangement and rotation costs). On the other hand, the Baron solver provided better solutions with respect to the second objective function (violation of closeness and farness). Moreover, this study proposed a new approach to determine the PUS. The results showed that the proposed method improves the traditional method up to 25%. The results also revealed the impact of variation in the demand over different periods on both DFLP and JSS. According to our analysis, these two problems are interconnected, and the synergy between them is considerable. In order to prove this, first, we applied a 1-Sample t test with 95% of confidence level on the quality of solutions provided by the separate optimization and simultaneous optimization of those two problems. The results showed that the simultaneous optimization outperforms in terms of QM and MID; while the separate optimization provides better solutions only regarding SM. Second, further analysis revealed that the simultaneous optimization of these problems reduces the MFT by 10%. For the future research, the authors believe that the consideration of uncertainty in demands of products over different periods and setup time of machines or process time of jobs would be interesting.
Table 10. The comparison of the average unit of the MFT for both the separate optimization and simultaneous optimization

| Type of optimization | Instance | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9   | 10  |
|----------------------|----------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Separate             | 23.2     | 42.2| 80.1| 94.7| 124 | 185 | 238.3| 253.8| 295.7| 317.6|
| Simultaneous         | 22.3     | 40.6| 74.2| 90.3| 118.9| 161.8| 211.1| 212.7| 250.1| 265.6|
| Gap (%)              | 3.7      | 3.9 | 7.4 | 4.6 | 4.1 | 12.5| 11.4 | 16.2 | 15.4 | 16.4 |

| Type of optimization | Instance | 11  | 12  | 13  | 14  | 15  | 16  | 17  | 18  | 19  | 20  |
|----------------------|----------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Separate             | 23.1     | 42.3| 73  | 94.6| 126.6| 197.4| 245.6| 266.4| 299.9| 324.5|
| Simultaneous         | 21.8     | 41  | 70.6| 70.6| 118.8| 163.6| 212.9| 223.7| 254.7| 285.6|
| Gap (%)              | 5.5      | 3.1 | 3.3 | 25.4| 6.1 | 17.1| 13.3 | 16   | 15   | 11.98|
REFERENCES

[1] A. D. Asl and K. Y. Wong, Solving unequal-area static and dynamic facility layout problems using modified particle swarm optimization, *Journal of Intelligent Manufacturing*, 28 (2017), 1317–1336.

[2] C. Bierwirth and J. Kulhfahl, Extended GRASP for the job shop scheduling problem with total weighted tardiness objective, *European Journal of Operational Research*, 261 (2017), 835–848.

[3] H. X. Chen and H. C. Lau, A math-heuristic approach for integrated resource scheduling in a maritime logistics facility, *2011 IEEE International Conference on Industrial Engineering and Engineering Management*, (2011).

[4] K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, *IEEE transactions on Evolutionary Computation*, 6 (2002), 182–197.

[5] S. Emami and A. S. Nookabadi, Managing a new multi-objective model for the dynamic facility layout problem, *The International Journal of Advanced Manufacturing Technology*, 68 (2013), 2215–2228.

[6] X. Hao, M. Gen, L. Lin and G. A. Suer, Effective multiobjective EDA for bi-criteria stochastic job-shop scheduling problem, *Journal of Intelligent Manufacturing*, 28 (2017), 833–845.

[7] M. Kaveh, V. M. Dalfard and S. Amiri, A new intelligent algorithm for dynamic facility layout problem in state of fuzzy constraints, *Neural Computing and Applications*, 24 (2014), 1179–1190.

[8] N. Khilwani, R. Shankar and M. Tiwari, Facility layout problem: An approach based on a group decision-making system and psychoclonal algorithm, *International Journal of Production Research*, 46 (2008), 895–927.

[9] R. Kolisch, A. Sprecher and A. Drexel, Characterization and generation of a general class of resource-constrained project scheduling problems, *Management Science*, 41 (1995), 1693–1703.

[10] J. Liu, D. Wang, K. He and Y. Xue, Combining Wang-Landau sampling algorithm and heuristics for solving the unequal-area dynamic facility layout problem, *European Journal of Operational Research*, 262 (2017), 1052–1063.

[11] G. Mavrotas, Effective implementation of the $\epsilon$-constraint method in multi-objective mathematical programming problems, *Applied Mathematics and Computation*, 213 (2009), 455–465.

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

[13] N. Mladenović and P. Hansen, Variable neighborhood search, *Computers & Operations Research*, 11 (1997), 1097–1100.

[14] A. Mohamadi, S. Ebrahimihejaz and R. Tavakkoli-Moghaddam, A novel two-stage approach for solving a bi-objective facility layout problem, *International Journal of Operational Research*, 31 (2018), 49–87.

[15] A. Moosavi and S. Ebrahimihejaz, Scheduling of elective patients considering upstream and downstream units and emergency demand using robust optimization, *Computers & Industrial Engineering*, 120 (2018), 216–233.

[16] N. Nekooghadirli, R. Tavakkoli-Moghaddam, Y. R. Ghezavati and S. Javanmard, Solving a new bi-objective location-routing-inventory problem in a distribution network by metaheuristics, *Computers & Industrial Engineering*, 76 (2014), 204–221.

[17] M. Pirayesh and S. Poormoaid, Location and job shop scheduling problem in fuzzy environment, 5th Int. Conference of the Iranian Society of Operations Research, Azarbaijan, Iran, (2012).

[18] H. Pourvaziri and H. Pierreval, Dynamic facility layout problem based on open queuing network theory, *European Journal of Operational Research*, 259 (2017), 538–553.

[19] K. S. N. Ripon and J. Torresen, Integrated job shop scheduling and layout planning: a hybrid evolutionary method for optimizing multiple objectives, *Evolving Systems*, 5 (2014), 121–132.

[20] K. S. N. Ripon, C. H. Tsang and S. Kwong, An evolutionary approach for solving the multi-objective job-shop scheduling problem, *In Evolutorial Scheduling*, 2007, 165–195, Springer, Berlin, Heidelberg.

[21] M. H. Salmani, K. Eshghi and H. Neghabi, A bi-objective MIP model for facility layout problem in uncertain environment, *The International Journal of Advanced Manufacturing Technology*, 81 (2015), 1563–1575.
[22] H. Samarghandi, P. Taabayan and M. Behroozi, Metaheuristics for fuzzy dynamic facility layout problem with unequal area constraints and closeness ratings, *The International Journal of Advanced Manufacturing Technology*, 67 (2013), 2701–2715.

[23] J. Shahrabi, M. A. Adibi and M. Mahmootchi, A reinforcement learning approach to parameter estimation in dynamic job shop scheduling, *Computers & Industrial Engineering*, 110 (2017), 75–82.

[24] A. Srinivasan, *Integrating Block Layout Design and Location of Input and Output Points in Facility Layout Problems*, M.Sc. thesis, Concordia University in Canada, 2014.

[25] C. R. Vela, R. Varela and M. A. González, Local search and genetic algorithm for the job shop scheduling problem with sequence dependent setup times, *Journal of Heuristics*, 16 (2010), 139–165.

[26] L. Wang, Combining facility layout redesign and dynamic routing for job-shop assembly operations, *2011 IEEE International Symposium on Assembly and Manufacturing*, Tampere, Finland, (2011).

[27] L. Wang, S. Keshavarzimanesh and H. Y. Feng, A hybrid approach for dynamic assembly shop floor layout, *2010 IEEE International Conference on Automation Science and Engineering*, Toronto, ON, Canada, (2010).

[28] L. Wang, H. Wu, F. Tang and D. Z. Zheng, A hybrid quantum-inspired genetic algorithm for flow shop scheduling, *International Conference on Intelligent Computing*, Berlin, Heidelberg, (2005), 636–644.

[29] C. L. Yang, S. P. Chuang and T. S. Hsu, A genetic algorithm for dynamic facility planning in job shop manufacturing, *The International Journal of Advanced Manufacturing Technology*, 52 (2011), 303–309.

Received February 2018; revised October 2018.

E-mail address: behrad.erfani@yahoo.com
E-mail address: ibrahimnejad@kiau.ac.ir
E-mail address: moosavi.amir@outlook.com