

University of Windsor

## Scholarship at UWindor

---

Electronic Theses and Dissertations

Theses, Dissertations, and Major Papers

---

3-10-2021

# Dynamic Facility Layout for Cellular and Reconfigurable Manufacturing using Dynamic Programming and Multi-Objective Metaheuristics

Saeideh Salimpour  
*University of Windsor*

Follow this and additional works at: <https://scholar.uwindsor.ca/etd>

---

### Recommended Citation

Salimpour, Saeideh, "Dynamic Facility Layout for Cellular and Reconfigurable Manufacturing using Dynamic Programming and Multi-Objective Metaheuristics" (2021). *Electronic Theses and Dissertations*. 8574.

<https://scholar.uwindsor.ca/etd/8574>

This online database contains the full-text of PhD dissertations and Masters' theses of University of Windsor students from 1954 forward. These documents are made available for personal study and research purposes only, in accordance with the Canadian Copyright Act and the Creative Commons license—CC BY-NC-ND (Attribution, Non-Commercial, No Derivative Works). Under this license, works must always be attributed to the copyright holder (original author), cannot be used for any commercial purposes, and may not be altered. Any other use would require the permission of the copyright holder. Students may inquire about withdrawing their dissertation and/or thesis from this database. For additional inquiries, please contact the repository administrator via email ([scholarship@uwindsor.ca](mailto:scholarship@uwindsor.ca)) or by telephone at 519-253-3000ext. 3208.

**Dynamic Facility Layout for Cellular and Reconfigurable Manufacturing using  
Dynamic Programming and Multi-Objective Metaheuristics**

By

**Saeideh Salimpour**

A Dissertation

Submitted to the Faculty of Graduate Studies  
through the Department of Mechanical, Automotive, and Materials Engineering  
in Partial Fulfillment of the Requirements for  
the Degree of Doctor of Philosophy  
at the University of Windsor

Windsor, Ontario, Canada

2021

© 2021 Saeideh Salimpour

**Dynamic Facility Layout for Cellular and Reconfigurable Manufacturing using  
Dynamic Programming and Multi-Objective Metaheuristics**

by

**Saeideh Salimpour**

APPROVED BY:

---

J. Balakrishnan, External Examiner  
California State University-Sacramento

---

R. Lashkari  
Department of Mechanical, Automotive, and Materials Engineering

---

J. Urbanic  
Department of Mechanical, Automotive, and Materials Engineering

---

E. Abdel-Raheem  
Department of Electrical and Computer Engineering

---

A. Azab, Advisor  
Department of Mechanical, Automotive, and Materials Engineering

February 8, 2021

## **DECLARATION OF CO-AUTHORSHIP / PREVIOUS PUBLICATION**

### ***I. Co-Authorship***

I hereby declare that this dissertation incorporates material that is the result of joint research undertaken under the supervision of and in collaboration with Dr. Ahmed Azab. This collaboration is covered through Chapters 2, 3, and 4 of the dissertation. Chapter 2 of the dissertation was co-authored with Dr. Fazle Baki and Ms. Sophie Viaux, and chapter 4 was co-authored with Mr. Hani Pourvaziri.

In all cases, the key ideas, primary contributions, experimental designs, data analysis, interpretation, and writing were performed by the author, and the contribution of the co-authors was providing feedback on the refinement of ideas and editing of the manuscript.

I am aware of the University of Windsor Senate Policy on Authorship and I certify that I have properly acknowledged the contribution of other researchers to my dissertation, and have obtained written permission from each of the co-authors to include the published, and submitted papers in my thesis.

I certify that, with the above qualification, this dissertation, and the research to which it refers, is the product of my own work.

### ***II. Previous Publication***

This dissertation includes three original papers and a book chapter that have been previously published or submitted for publication, as follows:

Dissertation Chapter	Publication Title/Full Citation	Publication Status
1	Salimpour, S., Viaux, S.C., Azab, A. and Baki, M.F., 2020. A Clustering-Sequencing Approach for the Facility Layout Problem. In <i>Proceedings of the Seventh International Forum on Decision Sciences</i> (pp. 135-142). Springer, Singapore.	Published
2	Salimpour, S., Viaux, S.C., Azab, A. and Baki, M.F., 2020. A Clustering-Sequencing Math-Heuristic Approach For The Unequal-Area Facility Layout Problem. <i>Manufacturing Systems: Recent Progress and Future Directions</i> (ch. 4)	Published
3	Salimpour, S. and Azab, A. A Dynamic Programming Approach to Solve the Facility Layout Problem for Reconfigurable Manufacturing, <i>ASME MSEC 2021</i> .	Accepted (Extension to be Submitted to a Journal)
4	Salimpour, S., Pourvaziri, H. and Azab, A. Semi-Robust Layout Design for Cellular Manufacturing in a Dynamic Environment, <i>Computers and Operations Research</i> .	Revision Requested

I certify that I have obtained written permission from the copyright owners to include the above-published materials in my thesis. I certify that the above material describes work completed during my registration as a graduate student at the University of Windsor.

### ***III. General***

I declare that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or

otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained written permission from the copyright owner(s) to include such material(s) in my thesis.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office, and that this thesis has not been submitted for a higher degree to any other University or Institution.

## ABSTRACT

The facility layout problem is one of the most classical yet influential problems in the planning of production systems. A well-designed layout minimizes the material handling costs (MHC), personnel flow distances, work in process, and improves the performance of these systems in terms of operating costs and time. Because of this importance, facility layout has a rich literature in industrial engineering and operations research. Facility layout problems (FLPs) are generally concerned with positioning a set of facilities to satisfy some criteria or objectives under certain constraints. Traditional FLPs try to put facilities with the high material flow as close as possible to minimize the MHC. In static facility layout problems (SFLP), the product demands and mixes are considered deterministic parameters with constant values. The material flow between facilities is fixed over the planning horizon. However, in today's market, manufacturing systems are constantly facing changes in product demands and mixes. These changes make it necessary to change the layout from one period to the other to be adapted to the changes. Consequently, there is a need for dynamic approaches of FLP that aim to generate layouts with high adaptation concerning changes in product demand and mix. This thesis focuses on studying the layout problems, with an emphasis on the changing environment of manufacturing systems.

Despite the fact that designing layouts within the dynamic environment context is more realistic, the SFLP is observed to have been remained worthy to be analyzed. Hence, a math-heuristic approach is developed to solve an SFLP. To this aim, first, the facilities are grouped into many possible vertical clusters, second, the best combination of the generated clusters to be in the final layout are selected by solving a linear programming model, and finally, the selected clusters are sequenced within the shop floor. Although the presented math-heuristic approach is effective in solving SFLP, applying approaches to cope with the changing manufacturing environment is required.

One of the most well-known approaches to deal with the changing manufacturing environment is the dynamic facility layout problem (DFLP). DFLP suits reconfigurable manufacturing systems since their machinery and material handling devices are reconfigurable to encounter the new necessities for the variations of product mix and demand. In DFLP, the planning horizon is divided into some periods. The goal is to find a layout for each period to minimize the total MHC for all periods and the total

rearrangement costs between the periods. Dynamic programming (DP) has been known as one of the effective methods to optimize DFLP. In the DP method, all the possible layouts for every single period are generated and given to DP as its state-space. However, by increasing the number of facilities, it is impossible to give all the possible layouts to DP and only a restricted number of layouts should be fed to DP. This leads to ignoring some layouts and losing the optimality; to deal with this difficulty, an improved DP approach is proposed. It uses a hybrid metaheuristic algorithm to select the initial layouts for DP that lead to the best solution of DP for DFLP. The proposed approach includes two phases. In the first phase, a large set of layouts are generated through a heuristic method. In the second phase, a genetic algorithm (GA) is applied to search for the best subset of layouts to be given to DP. DP, improved by starting with the most promising initial layouts, is applied to find the multi-period layout. Finally, a tabu search algorithm is utilized for further improvement of the solution obtained by improved DP. Computational experiments show that improved DP provides more efficient solutions than DP approaches in the literature.

The improved DP can efficiently solve DFLP and find the best layout for each period considering both material handling and layout rearrangement costs. However, rearrangement costs may include some unpredictable costs concerning interruption in production or moving of facilities. Therefore, in some cases, managerial decisions tend to avoid any rearrangements.

To this aim, a semi-robust approach is developed to optimize an FLP in a cellular manufacturing system (CMS). In this approach, the pick-up/drop-off (P/D) points of the cells are changed to adapt the layout with changes in product demand and mix. This approach suits more a cellular flexible manufacturing system or a conventional system. A multi-objective nonlinear mixed-integer programming model is proposed to simultaneously search for the optimum number of cells, optimum allocation of facilities to cells, optimum intra- and inter-cellular layout design, and the optimum locations of the P/D points of the cells in each period. A modified non-dominated sorting genetic algorithm (MNSGA-II) enhanced by an improved non-dominated sorting strategy and a modified dynamic crowding distance procedure is used to find Pareto-optimal solutions. The computational experiments are carried out to show the effectiveness of the proposed MNSGA-II against other popular metaheuristic algorithms.



## **DEDICATION**

*I dedicate this dissertation to  
my dear family*

## ACKNOWLEDGEMENTS

The first and infinite gratitude is to Almighty God who gave me existence and guided me through science and knowledge. He was the one who granted me patience in the time of despair and strength to strive in the face of failure. Doubtlessly, this achievement would not have been possible without his help and blessings.

My special thanks goes to my supervisor, Dr. Ahmed Azab, for giving me the opportunity to collaborate with him and for his constant help, support and encouragement. He trusted me and believed in my abilities from the very first day I met him and was always on my side.

I would like to also thank my doctoral committee members, Dr. Reza Lashkari, Dr. Jill Urbanic, and Dr. Esam Abdel-Raheem, for their valuable suggestions, feedback, and insightfulness. I extend a distinctive thanks to my external examiner, Dr. Jaydeep Balakrishnan for his suggestions and help in improving the quality of the dissertation.

In addition, I would like to thank the graduate secretary of the MAME department, Ms. Angela Haskel for all her help and kindness during my studies at the University of Windsor. I am grateful to all my professors, co-authors, friends, and current and former colleagues at the University of Windsor and POM research lab for their help and support.

My deepest and boundless appreciation goes to the first teachers of my life, my lovely mother and father, who instilled in me the love for learning and knowledge and supported me unconditionally throughout my life. I would like to also thank my dear brother and sister for their love, support and inspiration.

My acknowledgements would be incomplete without thanking a very special person, my loving husband, for all his patience, help, and support.

## TABLE OF CONTENTS

DECLARATION OF CO-AUTHORSHIP / PREVIOUS PUBLICATION .....	iii
ABSTRACT.....	vi
DEDICATION.....	viii
ACKNOWLEDGEMENTS.....	ix
LIST OF TABLES.....	xiii
LIST OF FIGURES .....	xiv
CHAPTER 1: INTRODUCTION.....	1
1.1.    Types of Layout based on Production Systems .....	1
1.1.1.    Product Layout.....	2
1.1.2.    Process Layout .....	3
1.1.3.    Fixed Product Layout.....	5
1.1.4.    Cellular Layouts.....	6
1.2.    Equal-Area vs. Unequal-Area FLP.....	7
1.3.    Discrete vs. Continuous FLP .....	8
1.4.    Static vs. Dynamic FLP .....	9
1.5.    Deterministic vs. Stochastic FLP .....	10
1.6.    Robust FLP .....	11
1.7.    Objectives of FLP .....	11
1.8.    Resolution Approaches of FLP.....	12
1.8.1.    Exact Algorithms .....	12
1.8.2.    Heuristic Algorithms.....	12
1.8.3.    Metaheuristic Algorithms .....	13
1.8.4.    Hybrid Approaches .....	13
1.9.    Layout Design Software Packages.....	14
1.10.    Contribution of the Research .....	15
1.11.    Outline of the Dissertation .....	17
Bibliography.....	17
CHAPTER 2: A CLUSTERING-SEQUENCING MATH-HEURISTIC APPROACH FOR THE UNEQUAL-AREA FACILITY LAYOUT PROBLEM.....	28
Nomenclature .....	28
2.1.    Introduction.....	29

2.2.	Problem Statement .....	32
2.3.	Methodology .....	33
2.3.1.	Step 1: Grouping Machines in Different Clusters.....	33
2.3.2.	Step 2: Selecting Between Clusters and Sequencing Them.....	36
2.3.3.	Step 3: Sequencing Machines in Each Cluster .....	38
2.3.4.	Step 4: Calculation of Total Distance Traveled by All Units .....	40
2.4.	Particle Swarm Optimization (PSO).....	40
2.4.1.	PSO Algorithm.....	42
2.4.2.	Solution Representation .....	42
2.4.3.	Objective Function.....	43
2.4.4.	Local Improvement Search .....	44
2.5.	Computational Results .....	44
2.5.1.	Comparison I.....	44
2.5.2.	Comparison II .....	46
2.6.	Conclusions.....	53
	Bibliography.....	54
<b>CHAPTER 3: A DYNAMIC PROGRAMMING APPROACH TO SOLVE THE FACILITY LAYOUT PROBLEM FOR RECONFIGURABLE MANUFACTURING .</b>		
	Nomenclature .....	61
3.1.	Introduction.....	62
3.2.	Problem Formulation .....	66
3.3.	Methodology .....	67
3.3.1.	DP for DFLP .....	67
3.3.2.	Improved DP Algorithms (IDP and IDP-TS) .....	71
3.4.	Computational Results .....	76
3.4.1.	Calibration of Parameters .....	77
3.4.2.	Evaluating the Proposed Algorithms .....	78
3.5.	Conclusions.....	82
	Bibliography.....	82
<b>CHAPTER 4: SEMI-ROBUST LAYOUT DESIGN FOR CELLULAR MANUFACTURING IN A DYNAMIC ENVIRONMENT .....</b>		
	Nomenclature .....	86
4.1.	Introduction.....	88

4.2.	Literature Review.....	90
4.2.1.	Dynamic Cellular Manufacturing System (DCMS) .....	90
4.2.2.	Robust Cellular Manufacturing System (RCMS).....	93
4.2.3.	Conclusions.....	94
4.3.	Problem Description .....	95
4.4.	Mathematical Model .....	97
4.4.1.	Objective Functions .....	97
4.4.2.	Cell Formation Constraints .....	100
4.4.3.	Non-Overlapping Constraints .....	100
4.4.4.	Within-Site Boundaries Constraints .....	102
4.4.5.	Constraints for Determining the Locations of P/D Points .....	103
4.4.6.	Domain Constraints .....	104
4.5.	The Proposed Multi-Objective Optimization Algorithm.....	104
4.5.1.	The Modified Non-Dominated Sorting Procedure .....	105
4.5.2.	Implementation of Modified Non-Dominated Sorting Genetic Algorithm (MNSGA-II) .....	109
4.6.	Computational Results .....	115
4.6.1.	Data Set.....	115
4.6.2.	Evaluation Metrics of the Metaheuristic Algorithms.....	119
4.6.3.	Calibration of Parameters .....	122
4.6.4.	Evaluating the Proposed Metaheuristic .....	125
4.7.	Conclusions.....	130
	Bibliography.....	131
	CHAPTER 5: CONCLUSIONS .....	137
5.1.	Concluding Remarks .....	137
5.2.	Future Work .....	138
	VITA AUCTORIS .....	139

## LIST OF TABLES

Table 2.1. Dimensions of the shop in meter .....	45
Table 2.2. Dimensions of the machines in meter .....	45
Table 2.3. The material handling flow between pairs of machines .....	45
Table 3.1. The parameters of the IDP and IDP-TS algorithms and their optimum values	77
Table 3.2. Computational results for instances with 6 facilities and 5 periods .....	79
Table 3.3. Computational results for instances with 6 facilities and 10 periods .....	79
Table 3.4. Computational results for instances with 15 facilities and 5 periods .....	80
Table 3.5. Computational results for instances with 15 facilities and 10 periods .....	80
Table 3.6. Computational results for instances with 30 facilities and 5 periods .....	81
Table 3.7. Computational results for instances with 30 facilities and 10 periods .....	81
Table 4.1. Summary of the related works .....	95
Table 4.2. The input data for test problems .....	117
Table 4.3. Product demand for problems with 17 products .....	118
Table 4.4. Product demand for problems with 20 products .....	118
Table 4.5. Product demand for problems with 15 products .....	118
Table 4.6. Product demand for problems with 7 products .....	119
Table 4.7. Product demand for problems with 40 products .....	119
Table 4.8. The parameters of the algorithms and their optimum values.....	123
Table 4.9. ANOVA results of the regression model.....	124
Table 4.10. Results of the algorithms for all the test problems .....	126
Table 4.11. ANOVA results for all performance metrics .....	127

## LIST OF FIGURES

Figure 1.1. Types of layout.....	2
Figure 1.2. An example of a product layout.....	3
Figure 1.3. An example of a process layout.....	4
Figure 1.4. An example of a fixed product layout.....	5
Figure 1.5. An example of a cellular layout.....	7
Figure 1.6. (a) An example of EA-FLP (b) An example of UA-FLP.....	8
Figure 1.7. An example of a discrete DFLP with three periods.....	10
Figure 1.8. Summary of the developed models and solution algorithms in this dissertation for different FLP problems considering different manufacturing contexts and settings ..	16
Figure 2.1. Flowchart of the algorithm for generating clusters.....	35
Figure 2.2. Possible positions on the shop floor and their corresponding rankings.....	37
Figure 2.3. A sample layout.....	39
Figure 2.4. Decentralizing machines which are in the endpoint clusters.....	40
Figure 2.5. The best layout.....	46
Figure 2.6. A shop floor with the size of 13 m × 12 m.....	48
Figure 2.7. A shop floor with the size of 13 m × 13 m.....	49
Figure 2.8. A shop floor with the size of 14 m × 8 m.....	51
Figure 2.9. A shop floor with the size of 15 m × 8 m.....	52
Figure 2.10. A shop floor with the size of 14 m × 9 m.....	53
Figure 3.1. All four layouts are symmetric and have the same material handling cost....	68
Figure 3.2. The dynamic facility layout problem. a. There is no initial layout to begin with (starting from scratch) b. There is an initial layout at the beginning (period 0).....	70
Figure 3.3. The flowchart of the proposed IDP algorithms.....	73
Figure 3.4. The structure of chromosome.....	74
Figure 3.5. An example of crossover operator.....	75
Figure 3.6. An example of mutation operator.....	76
Figure 4.1. An example of the SRCMS with 16 facilities and 4 cells.....	97
Figure 4.2. The distance between machine $i$ in cell $k$ and machine $j$ in cell $l$ .....	100
Figure 4.3. An example for the location of P/D point of cell $k$ .....	104
Figure 4.4. Illustration of the drawbacks of the original crowding distance method.....	107
Figure 4.5. An example of a chromosome and its corresponding layout for first period	112
Figure 4.6. The drawback of the original SP metric.....	120

Figure 4.7. Tukey simultaneous confidence intervals of the algorithms. ....	128
Figure 4.8. Convergence graph of the algorithms for problem 19.....	129
Figure 4.9. Convergence graph of the algorithms for problem 25.....	130



# CHAPTER 1

## INTRODUCTION

The facility layout problem (FLP) is concerned with the most efficient placement of departments, cells, or machines (depending on the industry) in a given area. The general factors of efficiency for a layout include material handling cost (MHC), space utilization, and flexibility as well as ergonomic, psychological and technological factors [1]. FLP usually takes into account the first three mentioned factors (MHC, space utilization, and flexibility).

The major part of the assets of a manufacturing system is occupied by its facilities [2]. FLP is a fundamental issue that must be reflected when designing a manufacturing system. It can reduce up to 50 percent of the operating costs and up to 70 percent of manufacturing expenses [3]. It can also indirectly affect the throughput time, overall productivity, quality and inventory levels [4-6]. It is shown that implementing not a proper layout will result in around a 35% reduction in the system's efficiency [7].

Hence, FLP is worthy of being studied and has attracted many researchers in the fields of industrial engineering and operations management in the latest decades [8].

### ***1.1 Types of Layout based on Production Systems***

Considering the type of production system, four basic layout types, including product layout, process layout, fixed product layout, and cellular layout, exist. The most suitable layout for a manufacturing system is selected based on the volume and the variety of products to be produced (Figure 1) [3, 9].

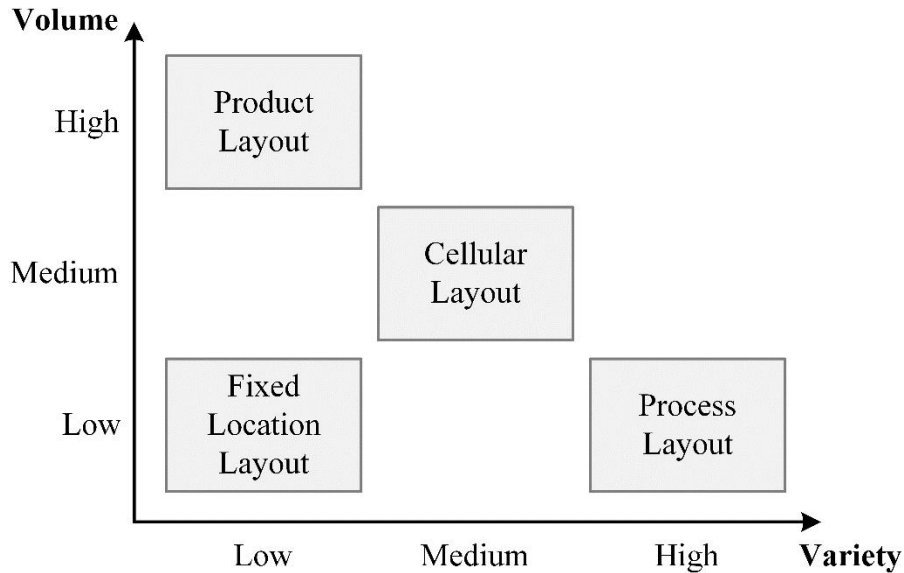


Figure 1.1. Types of layout [3]

### ***1.1.1 Product Layout***

When the machines are placed based on the sequence of the operations, the layout is called the product layout (also known as production layout, flow-line layout, or assembly line layout). A product moves from one machine to another based on its predetermined sequence of operations. Hence, multiple flows may exist in the system if more than one type of product is produced.

An example of a product layout is illustrated in Figure 1.2. Product layout is most suitable for assembly manufacturing firms that produce a single type of product or more than one type yet in high volume [3, 9].

Advantages of product layout include [3]:

- Smoother and more logical flow lines
- Lower amount of work in process (WIP) inventories; work is transferred directly from one process into the next
- Less required space for WIP and temporary storage
- Shorter total production time per unit

- Lower MHC; minimization of distances between machines that perform consecutive operations
- Simpler, shorter and less expensive required training for operators
- Simpler production planning control systems

On the other hand, the disadvantages include [3]:

- Blockage of the entire line because of a breakdown of one machine or absence of one operator
- Lower process flexibility; a change in product design may impose a major change in the layout
- Lower time flexibility; the slowest task of a product reduces the speed of the flow of that product
- Requirement for more general supervision and less specialized; high dependency between stations
- Higher investment; distribution of identical machines (that may not be fully utilized) along the line
- Higher worker boredom and fatigue; tasks are repetitive and simple

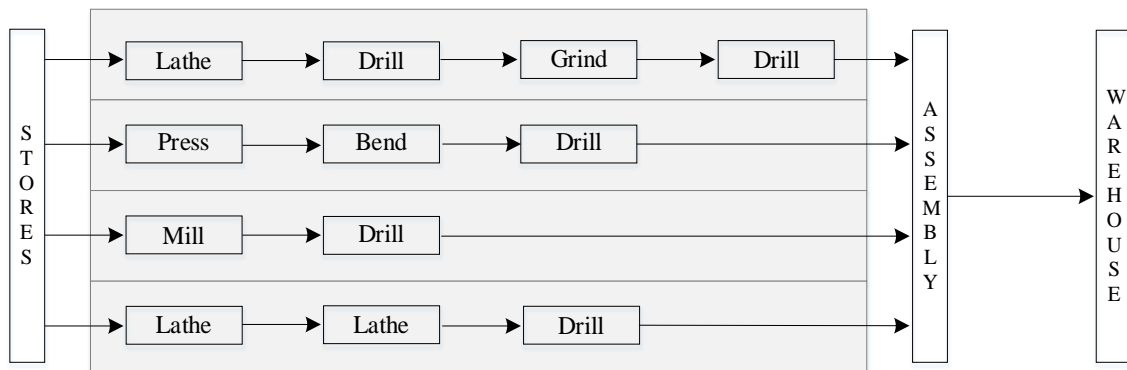


Figure 1.2. An example of a product layout [3]

### 1.1.2 Process Layout

When all the same types of operations are placed together, the layout is called the process layout (also known as functional layout or job shop layout). For instance, in the process layout shown in Figure 1.3, all grinding machines are put together in one department; all assembly machines are placed in another and so on. This layout type is suitable for

manufacturing firms that produce various types of products, yet in small quantities that each product type has a unique sequence of operations [3, 9].

Benefits of this layout include [3]:

- Higher task allocation flexibility to operators and facilities
- Lower investment; identical facilities are duplicated only if the others of the same type are fully utilized
- Lower worker boredom and fatigue; tasks are not repetitive and are more diverse
- Requirement for more specialized supervision

Its disadvantages are [3]:

- Lower process efficiency; long movements for handling the materials and backtrackings
- Lower time efficiency; the waiting time of workers between finishing a task and starting the next task
- Lower productivity; requirement for different setup and training for each job.
- Higher complexity of production planning and control.
- Requirement for more skilled workers
- Higher amount of WIP

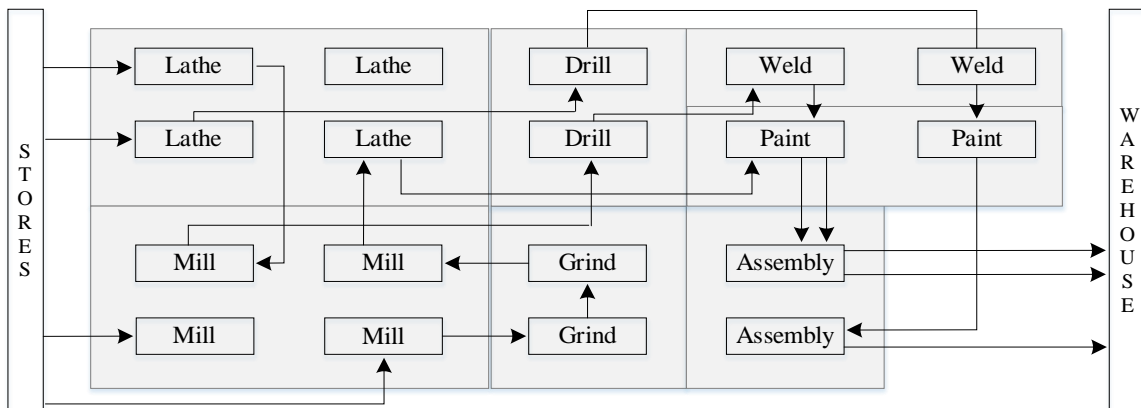


Figure 1.3. An example of a process layout [3]

### 1.1.3 Fixed Product Layout

Sometimes the products are complex, large, or heavy so that it is not possible to move them (e.g., aircraft, shipyards, and heavy constructions such as roads, buildings, and bridges). In such cases, a fixed product layout is suitable to be used, that instead of moving the product, the resources (equipment, operator, machine) are moved and brought to the product's location (Figure 1.4) [3, 9].

Advantages of a fixed product layout include [3]:

- Lower material handling movements and cost
- Lower probability of product damages; no product movement is required
- Higher flexibility to the possible changes of product design, mix, and/or volume
- Lower total production time; the production centers are independent so better scheduling is allowed
- Supports job enlargement; the workers are allowed to operate on the entire job
- Lower amount of re-planning and worker training when starting a new activity

Disadvantages of a fixed product layout include [3]:

- Costs increase due to the increased movement of people and equipment
- Difficulties of combining different types of skills
- Difficulties and the expense of installing equipment
- Lower equipment utilization
- Requirement for general supervision
- Requirement for more skilled workers as they get in charge of more operations
- May Require the duplication of equipment

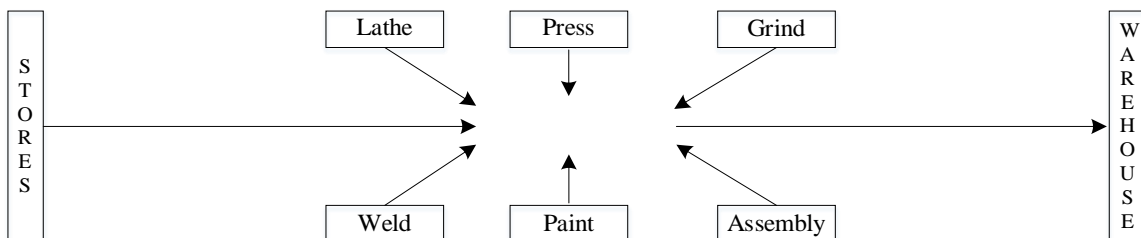


Figure 1.4. An example of a fixed product layout [3]

#### *1.1.4 Cellular Layouts*

Cellular layouts are designed based on the concept of group technology. This type of layouts group the machines into cells that each cell processes a certain part/product family. A set of parts/products that have similar shapes or require the same processing requirements (machine operations, tooling, machinery, jig, or fixtures) are called a part/product family. Typically, all the processes involved in manufacturing a part/product from raw material to a finished part/product are done in one cell. Hence, there is a need for operators that are skilled in several fields.

Cellular layouts divide a manufacturing facility physically into production cells and offer both the efficiency of product layouts and the flexibility of process layouts. The three stages of designing a cellular layout are as follows: (1) assigning machines to different cells (cell formation), (2) the layout of machines within each cell (intra-cellular layout), and (3) the layout of cells within the shop floor (inter-cellular layout). When designing a cellular layout, the most important objective is to minimize the total inter- and intra-cellular MHC [3, 10].

Advantages of the cellular layout include [3]:

- Lower MHC
- Lower setup time
- Lower amount of tooling
- Lower amount of WIP inventory
- Better relationships between workers
- Supports job enlargement
- Requirement for more general-purpose equipment

Disadvantages of the cellular layout include [3]:

- Lower flexibility
- Higher job flow times
- Unbalanced workload

- Lower machine utilization than process layout
- Requirement for buffers in case of imbalanced material flow between product layout and process layout
- Requirement for general supervision
- Requirement of more skilled workers than product layout

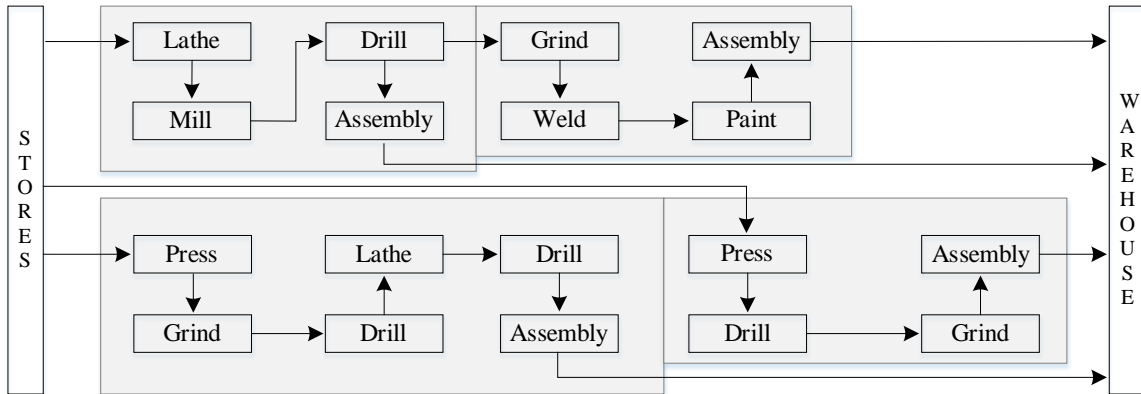


Figure 1.5. An example of a cellular layout [3]

### 1.2 Equal-Area vs. Unequal-Area FLP

In the equal-area facility layout problem (EA-FLP), it is assumed that facilities have equal sizes and are in the same shape (Figure 1.6 (a)). In most EA-FLPs, the problem is to assign facilities to fixed, predetermined locations and lodge any of the facilities [14]. In order to have a feasible assignment, the number of locations should be at least equal to the number of facilities. In case the number of locations is more than the number of facilities, dummy facilities can be considered to make the balance. The flow between these dummy facilities and other facilities is set to zero.

In the unequal-area facility layout problem (UA-FLP), the facilities must not have equal size and shape. The UA-FLP formulation was first introduced by Armour and Buffa [15]. An example of UA-FLP is shown in Figure 1.6 (b). As can be seen, facilities are located anywhere on the available space yet not have overlap with each other. Usually, a facility ( $i$ ) is assumed to have rectangular shape with fixed dimensions; length ( $l_i$ ) and width ( $w_i$ ). But sometimes, the facility can be defined using its area ( $A_i$ ) and aspect ratio ( $\alpha_i$ ) [16].

Aspect ratio is the ratio of the length to the width of a facility ( $\alpha_i = l_i / w_i$ ). The UA-FLP is usually formulated by a mixed-integer programming (MIP) model [17].

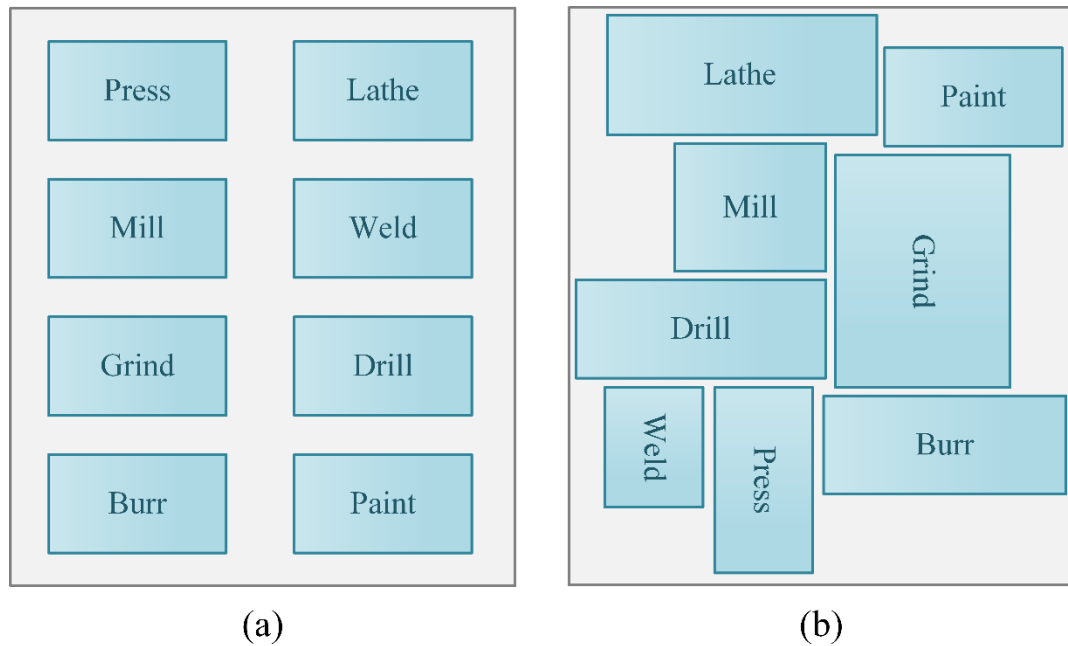


Figure 1.6. (a) An example of EA-FLP (b) An example of UA-FLP

The interested readers are suggested to read quoted articles [18-20] for more information about EA-FLP and UA-FLP and their applications.

### ***1.3 Discrete vs. Continuous FLP***

There are two types of geometric representation for FLP, called discrete and continuous [21]. In a discrete FLP problem, a set of locations is determined in advance, and facilities are assigned to that predetermined locations to search for the optimum assignment subject to some criteria [22]. This is possible only if all the facilities can be accommodated in all locations (called EA-FLP) [23]. Discrete representation is not suitable for UA-FLP since additional constraints are needed for modeling the problem. Discrete FLP problems are often formulated by the quadratic assignment problem (QAP). The QAP model for discrete FLP is as follows [24]:



$$\text{Min} \sum_{i=1}^N \sum_{j=1}^N \sum_{k=1}^N \sum_{l=1}^N F_{ij} D_{kl} X_{ik} X_{jl} \quad (1.1)$$

Subject to:

$$\sum_{k=1}^N X_{ik} = 1 \quad \forall i \quad (1.2)$$

$$\sum_{i=1}^N X_{ik} = 1 \quad \forall k \quad (1.3)$$

$$X_{ik} \in \{0,1\} \quad \forall i, k \quad (1.4)$$

where  $F_{ij}$  is the flow from facility  $i$  to facility  $j$ ,  $D_{kl}$  is the distance from location  $k$  to location  $l$ , and  $X_{ik}$  is a binary decision variable that is equal to one if facility  $i$  is assigned to location  $k$  and zero otherwise.

The objective function (1.1) minimizes the total MHC. Constraint (1.2) is to guarantee that each facility is assigned to one location, and constraint (1.3) is to guarantee that each location has accommodated only one facility. Lastly, constraint (1.4) specifies that the decision variable is binary.

In the continuous representation of FLP, the facilities have different sizes and shapes and can be located in any position, provided that no overlapping happens. Continuous FLPs are usually formulated using the MIP model, as there is a need for auxiliary binary variables in the model [18].

#### ***1.4 Static vs. Dynamic FLP***

The material flow between facilities is assumed as a constant parameter in most of the research in the area of FLP. This type of FLPs is known as static facility layout problems (SFLP). Considering today's competitive and volatile manufacturing environment, this assumption can be unrealistic. In such a volatile environment of manufacturing systems, the product mix and demand are constantly changing, so the current layout loses its efficiency. Therefore, to adapt to these changes, the layout needs to be rearranged from one

period to the other. However, changing the layout at the beginning of each period imposes rearrangement costs to the system (e.g., the costs of moving the machines and loss of production). Therefore, there should be a balance between MHC and rearrangement costs which is the main aim of dynamic facility layout problems (DFLP). In DFLP, the planning horizon is divided into some periods, with deterministic and known demands, and the layout in each period is determined so that total MHC and the total rearrangement costs are minimized [25, 26]. Figure 1.7 illustrates an example of a DFLP with three periods.

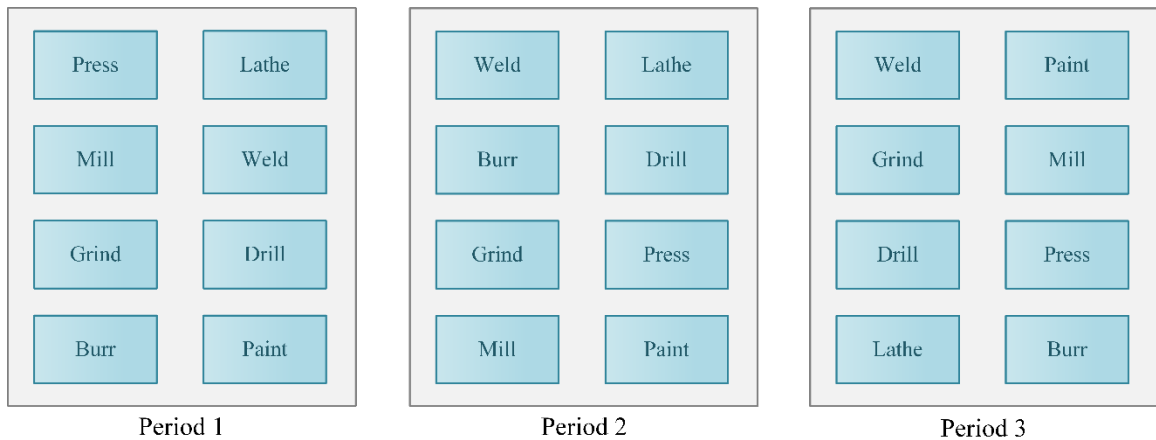


Figure 1.7. An example of a discrete DFLP with three periods

### 1.5 *Deterministic vs. Stochastic FLP*

In both static and dynamic FLPs, the system parameters such as demand and product mix are assumed to be deterministic in each planning horizon (for static, there is only one planning horizon). In real-world applications, it is impossible to forecast the exact amount of some parameters due to the extreme volatility of today's markets and the uncertainty present inherent in some of the manufacturing processes and their processing times. Stochastic static facility layout problem and stochastic dynamic facility layout problem are used to deal with uncertainty in the area of FLP [5, 27-34].

### ***1.6 Robust FLP***

Both stochastic single-period or deterministic multi-period FLPs can be addressed through a robust FLP [35]. A DFLP plan for a multi-period FLP consists of a set of layouts each of which is concerned with a period. If the product mix and demand are known and uncertain, each period's layout needs to be robust with respect to uncertainty in that period. Consequently, a set of robust layouts for all periods is required. A robust layout is not inevitably an optimal layout for any of the scenarios/periods, but it is near-optimal for different demand scenarios [36]. To obtain the performance of a robust layout, which is called the robustness of a layout, first the optimum solutions for different demand scenarios are calculated. Then, the number of times that the robust solution is within a pre-specified percentage of the optimum solutions is calculated [37]. Since the layout is not being changed during the planning horizon, no rearrangement costs are incurred and the total MHC over all the planning periods is minimized.

### ***1.7 Objectives of FLP***

Generally, the FLP tries to obtain the optimum arrangement of facilities within an available space based on a quantitative or a qualitative objective. FLPs with quantitative objectives attempt to determine the optimum layout by minimizing the total travel distance or time by all units, total rearrangement cost, the space occupied by facilities, total work-in-process (WIP), or throughput time [38], to name a few. On the other hand, the ones with qualitative objectives achieve the same by maximizing the adjacency function. However, in real-world scenarios, more than one objective is usually involved when solving the problem. Hence, some researchers try to find the best layout using multi-objective optimization models [39-41].

For multi-objective problems, the objectives can be combined into a single one that leads to a single optimum solution [42]. The Pareto approach can also be employed that result in a set of non-dominated solutions with respect to different objectives. In order to select one solution from this set, the Electre method can be used [39].

## ***1.8 Resolution Approaches of FLP***

There is a lot of research done on the area of FLP focusing on obtaining the optimum solution to this problem. However, due to the NP-hardness and combinatorial nature of the optimization problem in FLP [11-13], the exact algorithms can solve only small-sized problems (with 20 or fewer facilities) in a polynomial time [10]. As a result, near-optimal algorithms (heuristics and metaheuristics) are needed for solving the larger instances of the problem.

### ***1.8.1 Exact Algorithms***

The main character of exact approaches is their ability to reach the global optimum solution. This can be counted as the major draw of exact approaches. To guarantee optimality, the solution space must be comprehensively searched. Some partial/selective enumeration algorithms are applicable to conduct a comprehensive search for small-sized problems. However, by increasing the problem size, the possibility to do a comprehensive search for the solution space will be computationally intractable.

Exact algorithms are demonstrated to be able to solve DFLPs, with MIP formulation, in a timely manner for problems up to the size of twelve facilities and three periods [43]. Branch-and-bound (B&B), branch and cut (B&C), cutting plane algorithms and direct methods are the prominent exact algorithms used to solve FLP optimally [24, 44]. Dynamic programming (DP), if it enumerates all the possible layouts in its state-space, is the classic exact method for solving DFLP [45].

### ***1.8.2 Heuristic Algorithms***

As mentioned in Section 1.8, FLP is NP-hard [12, 13]. Hence, heuristic methods are introduced to solve large instances of the problem. Each heuristics algorithm is tailored to solve a specific optimization problem in a reasonable time [46]. These algorithms are usually based on experiential strategies, and their search process is not dependent on a strict mathematical model. Hence, they can not come up with an optimal solution.

The heuristic solution procedures of the FLP can be categorized into construction algorithms versus improvement algorithms. A construction algorithm generates a layout

from scratch, giving the flow between facilities or the adjacency relationship data. However, an improvement algorithm makes subsequent improvements in an existing (initial) layout to reduce the total cost or increase the total adjacency function. One of the most well-known improvement algorithms is the pair-wise exchange algorithm. This algorithm modifies the existing layout by exchanging two facilities in each iteration and accepts the change if it improves the objective function value. The algorithm continues until all exchange possibilities are considered.

### ***1.8.3 Metaheuristic Algorithms***

Metaheuristics are less greedy, and more problem-independent algorithms compare to the heuristic approaches. As a result, they can perform better in solving combinatorial optimization problems such as FLP, but still, a near-optimal solution can be achieved rather than optimum. This drawback of metaheuristics is because of the random search mechanisms that are employed in these algorithms and that they are less reliant on mathematical relationships between the decision variables [47].

There are numerous metaheuristic algorithms proposed in the literature for solving the FLP. Some of these metaheuristics include genetic algorithms [48-53], simulated annealing [21, 54-56], tabu search [14, 57-59], particle swarm optimization [60, 61], ant colony optimization [62-65], and bee colony algorithm [66-70].

### ***1.8.4 Hybrid Approaches***

Hybrid approaches are the ones that combine two or more exact, heuristic, or metaheuristic approaches. These approaches can be either construction or an improvement method [71, 72]. In hybrid methods, both the practically important factors and the optimization of objectives of designing a layout can be considered by combining knowledge-based systems and analytical algorithms, respectively [73].

Metaheuristic hybrid algorithms have been widely used in the literature. In these algorithms, a population-based evolutionary algorithm is improved by being hybridized with a local search solution-based algorithm [74-85]. Sometimes an exact approach is hybridized by a metaheuristic algorithm [17, 25, 86, 87].

### *1.9 Layout Design Software Packages*

Layout software packages have been used since 1963. Computerized relative allocation of facilities technique (CRAFT) is the first layout software and is originally presented by Armour and Buffa [15]. It is an improvement algorithm and considers exchanging the location of facilities only if they are adjacent or have equal areas. CRAFT can do two-way exchanges, three-way exchanges, two-way exchanges followed by three-way exchanges, and three-way exchanges followed by two-way exchanges [9]. The input data are the dimension of the given area, dimensions of the facilities, the flow between facilities, and an initial layout. In case needed, the user can also put some restrictions on the location of some facilities. Moving a facility physically has a cost that has to be considered when deciding on any change. A relocation proposal should be accepted only if the estimated savings as far as the MHC is concerned is more than the incurred relocation costs. To deal with the fixed and variable rearrangement cost, CRAFT is extended to CRAFT-M by Hicks and Cown [88].

Since then, many more software packages have been developed for optimizing the layout design problem. Selection of material handling equipment and area placement evaluation (SHAPE) [89] is a constructive heuristic algorithm with the objective of minimizing the total flow distance. DISCON [90] and KTM [91] are two hybrid construction and improvement algorithms. TESSA [92] and SPIRAL [93] are graph-theoretic algorithms for solving the layout problem. Multi-floor plant layout evaluation (MULTIPLE) [60] is a CRAFT-based improvement algorithm wherein the number of exchanges is increased due to the usage of space-filling curves and allowance of flexible department areas.

Some other tools ease the layout design and simulation by supporting 3D components programming and assembly such as those present in the current state of the art Digital Manufacturing PLM packages (namely Tecnomatix and DELMIA) as well as in Visual Components, SmartDraw, and CAD Schroer.

### ***1.10 Contribution of the Research***

In this dissertation, new models and solution algorithms are developed for different FLP problems, which entail different manufacturing contexts and settings (see Figure 1.8). This can be summarized as follows:

#### Chapter 2:

- A hybrid math-heuristic approach is presented to solve UA-FLP.
- This constructive algorithm solves the problem by generating many clusters of the facilities, obtaining the optimum set and sequence of clusters in the final layout, and locating them on the shop floor.
- It can be used as an initial solution to the metaheuristic algorithms especially when the area of the available space compare to the total area of the facilities is relatively small as the proposed method ensures feasible solutions.
- By selecting one of the clusters of the final layout randomly, the algorithm is able to generate different near-optimum layouts each time the algorithm is used.
- This randomness characteristic of the algorithm will result in having more than one initial layout which is desirable for initializing a population-based metaheuristic algorithm.

#### Chapter 3:

- A hybrid genetic algorithm-dynamic programming metaheuristic is proposed to solve the discrete DFLP.
- A heuristic procedure is presented to generate a set of good layouts as the state-space of DP.
- GA selects amongst the generated state-space to feed the DP. This way, even though DP does not enumerate all the generated state-space but still gets good results.
- The algorithm is further improved by employing tabu search metaheuristic.
- The system under study is a reconfigurable manufacturing system (RMS).

Chapter 4:

- A semi-robust layout is designed for cellular manufacturing systems (CMS) that are either flexible or conventional manufacturing systems.
- A multi-objective cellular FLP is considered.
- The number of cells, the assignment of machines to the cells, the location of cells and the machines inside the cells are optimized simultaneously.
- Only a few articles in the literature have addressed obtaining the optimum layout design in CMS in a dynamic environment.
- The proposed semi-robust approach catches the benefits of both dynamic and robust approaches and to the best of the author’s knowledge, has not been used previously.

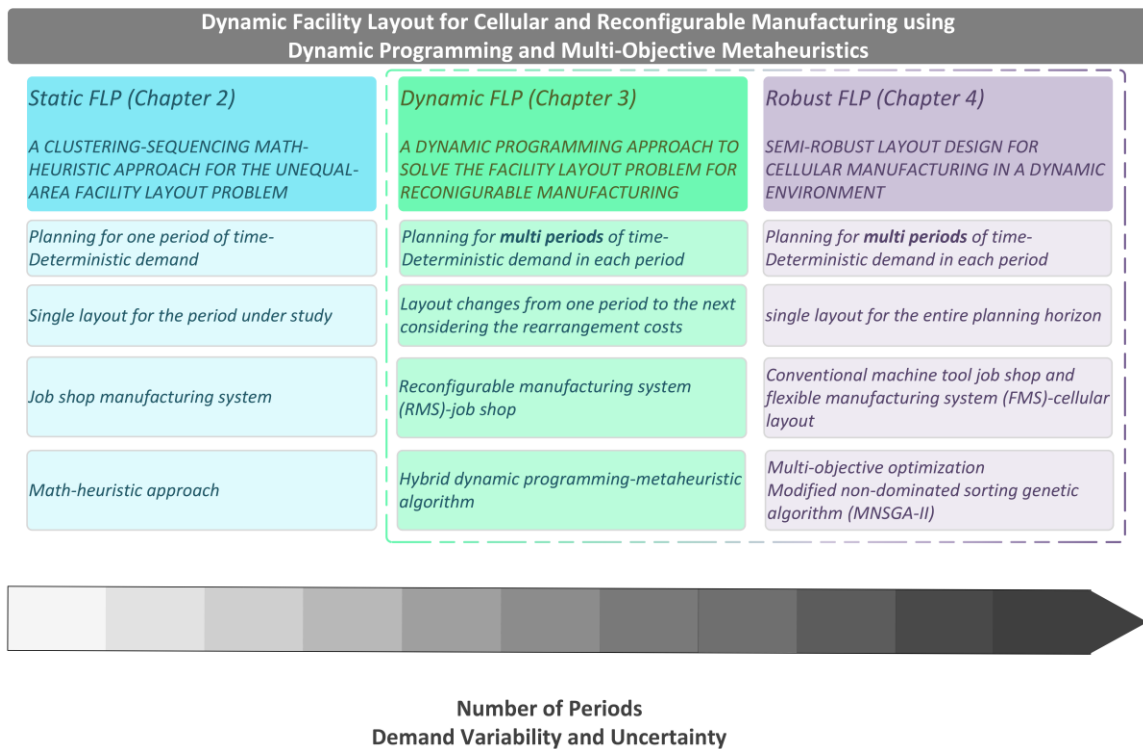


Figure 1.8. Summary of the developed models and solution algorithms in this dissertation for different FLP problems considering different manufacturing contexts and settings



### **1.11 Outline of the Dissertation**

This dissertation is comprised of five chapters. Chapter 1 presents the definition of the FLP along with its different characteristics and variants. Chapter 2 provides an efficient mathematical heuristic algorithm for solving the UA-FLP along with computational results assessing its performance. In Chapter 3, a hybrid genetic algorithm-dynamic programming approach is presented for the DFLP which is improved by tabu search and is validated against a well-known benchmark test set in the literature. Chapter 4 studies the cellular layout problem in a dynamic environment and presents a semi-robust layout. The problem is formulated as a multi-objective mathematical programming model. A modified non-dominated sorting genetic algorithm (MNSGA-II) is then used to obtain Pareto-optimal solutions for the problems. An experimental design and analysis for performance evaluation of the proposed MNSGA-II is also included in this chapter. Finally, the conclusions and some directions for future research are provided in Chapter 5.

### **Bibliography**

1. Hitchings, G., *Control, Redundancy, and Change in Layout Systems I*. AIIE Transactions, 1970. 2(3): p. 253-262.
2. Nordin, N.N., Z.M. Zainuddin, S. Salim, and R.R. Ponnusamy, *Mathematical modeling and hybrid heuristic for unequal size facility layout problem*. Malaysian Journal of Fundamental and Applied Sciences, 2009. 5(1).
3. Tompkins, J.A., J.A. White, Y.A. Bozer, and J.M.A. Tanchoco, *Facilities planning*. 2010: John Wiley & Sons.
4. Pillai, V.M., I.B. Hunagund, and K.K. Krishnan, *Design of robust layout for dynamic plant layout problems*. Computers & Industrial Engineering, 2011. 61(3): p. 813-823.
5. Vitayasak, S., P. Pongcharoen, and C. Hicks, *A tool for solving stochastic dynamic facility layout problems with stochastic demand using either a genetic algorithm or*

- modified backtracking search algorithm*. International Journal of Production Economics, 2017. 190: p. 146-157.
6. Lacksonen, T.A., *Preprocessing for static and dynamic facility layout problems*. International Journal of Production Research, 1997. 35(4): p. 1095-1106.
  7. Izadinia, N. and K. Eshghi, *A robust mathematical model and ACO solution for multi-floor discrete layout problem with uncertain locations and demands*. Computers & Industrial Engineering, 2016. 96: p. 237-248.
  8. Hosseini-Nasab, H., S. Fereidouni, S.M.T.F. Ghomi, and M.B. Fakhrazad, *Classification of facility layout problems: a review study*. The International Journal of Advanced Manufacturing Technology, 2018. 94(1-4): p. 957-977.
  9. Heragu, S.S., *Facilities design*. 2018: Crc Press.
  10. Heragu, S.S., *Facilities design*. 2008: Crc Press.
  11. Said, G.A.E.N.A., A.M. Mahmoud, and E.S.M. El-Horbaty, *A comparative study of meta-heuristic algorithms for solving quadratic assignment problem*. arXiv preprint arXiv:1407.4863, 2014.
  12. Tam, K.Y. and S.G. Li, *A hierarchical approach to the facility layout problem*. The International Journal of Production Research, 1991. 29(1): p. 165-184.
  13. Sahni, S. and T. Gonzalez, *P-complete approximation problems*. Journal of the ACM (JACM), 1976. 23(3): p. 555-565.
  14. Anjos, M.F. and M.V. Vieira, *Mathematical optimization approaches for facility layout problems: The state-of-the-art and future research directions*. European Journal of Operational Research, 2017. 261(1): p. 1-16.
  15. Armour, G.C. and E.S. Buffa, *A heuristic algorithm and simulation approach to relative location of facilities*. Management Science, 1963. 9(2): p. 294-309.
  16. Chwif, L., M.R.P. Barretto, and L.A. Moscato, *A solution to the facility layout problem using simulated annealing*. Computers in industry, 1998. 36(1-2): p. 125-132.

17. Dunker, T., G. Radons, and E. Westkämper, *Combining evolutionary computation and dynamic programming for solving a dynamic facility layout problem*. European Journal of Operational Research, 2005. 165(1): p. 55-69.
18. Drira, A., H. Pierreval, and S. Hajri-Gabouj, *Facility layout problems: A survey*. Annual Reviews in Control, 2007. 31(2): p. 255-267.
19. Kusiak, A. and S.S. Heragu, *The facility layout problem*. European Journal of operational research, 1987. 29(3): p. 229-251.
20. Hosseini-Nasab, H., S. Fereidouni, S.M.T.F. Ghomi, and M.B. Fakhrazad, *Classification of facility layout problems: a review study*. The International Journal of Advanced Manufacturing Technology, 2018. 94(1-4): p. 957-977.
21. Chwif, L., M.R.P. Barretto, and L.A. Moscato, *A solution to the facility layout problem using simulated annealing*. Computers in Industry, 1998. 36(1–2): p. 125-132.
22. Xie, W. and N.V. Sahinidis, *A branch-and-bound algorithm for the continuous facility layout problem*. Computers & Chemical Engineering, 2008. 32(4-5): p. 1016-1028.
23. Li, H. and P.E.D. Love, *Genetic search for solving construction site-level unequal-area facility layout problems*. Automation in construction, 2000. 9(2): p. 217-226.
24. Balakrishnan, J., C.H. Cheng, and K.F. Wong, *FACOPT: A user friendly FACility layout OPTimization system*. Computers & Operations Research, 2003. 30(11): p. 1625-1641.
25. Balakrishnan, J., C.H. Cheng, D.G. Conway, and C.M. Lau, *A hybrid genetic algorithm for the dynamic plant layout problem*. International Journal of Production Economics, 2003. 86(2): p. 107-120.
26. Baykasoglu, A., T. Dereli, and I. Sabuncu, *An ant colony algorithm for solving budget constrained and unconstrained dynamic facility layout problems*. Omega, 2006. 34(4): p. 385-396.

27. Yang, T. and B.A. Peters, *Flexible machine layout design for dynamic and uncertain production environments*. European Journal of Operational Research, 1998. 108(1): p. 49-64.
28. Krishnan, K.K., S.H. Cheraghi, and C.N. Nayak, *Facility layout design for multiple production scenarios in a dynamic environment*. International Journal of Industrial and Systems Engineering, 2008. 3(2): p. 105-133.
29. Ripon, K., K. Glette, M. Hovin, and J. Torresen, *Dynamic facility layout problem under uncertainty: a Pareto-optimality based multi-objective evolutionary approach*. Open Computer Science, 2011. 1(4): p. 375-386.
30. Moslemipour, G. and T. Lee, *Intelligent design of a dynamic machine layout in uncertain environment of flexible manufacturing systems*. Journal of Intelligent Manufacturing, 2012. 23(5): p. 1849-1860.
31. Tayal, A., A. Gunasekaran, S.P. Singh, R. Dubey, and T. Papadopoulos, *Formulating and solving sustainable stochastic dynamic facility layout problem: a key to sustainable operations*. Annals of Operations Research, 2017. 253(1): p. 621-655.
32. Tayal, A. and S.P. Singh, *Integrating big data analytic and hybrid firefly-chaotic simulated annealing approach for facility layout problem*. Annals of Operations Research, 2016: p. 1-26.
33. Raman, D., S.V. Nagalingam, and B.W. Gurd, *A genetic algorithm and queuing theory based methodology for facilities layout problem*. International Journal of Production Research, 2009. 47(20): p. 5611-5635.
34. Moslemipour, G., *A hybrid CS-SA intelligent approach to solve uncertain dynamic facility layout problems considering dependency of demands*. Journal of Industrial Engineering International, 2017: p. 1-14.
35. Meng, G., S.S. Heragu, and H. Zijm, *Reconfigurable layout problem*. International Journal of Production Research, 2004. 42(22): p. 4709-4729.

36. Kouvelis, P., A.A. Kurawarwala, and G.J. Gutierrez, *Algorithms for robust single and multiple period layout planning for manufacturing systems*. European journal of operational research, 1992. 63(2): p. 287-303.
37. Rosenblatt, M.J. and H.L. Lee, *A robustness approach to facilities design*. International journal of production research, 1987. 25(4): p. 479-486.
38. Huang, H., *Facility layout using layout modules*. 2003, The Ohio State University.
39. Aiello, G., M. Enea, and G. Galante, *A multi-objective approach to facility layout problem by genetic search algorithm and Electre method*. Robotics and computer-integrated manufacturing, 2006. 22(5-6): p. 447-455.
40. Mehdizadeh, E. and V. Rahimi, *An integrated mathematical model for solving dynamic cell formation problem considering operator assignment and inter/intra cell layouts*. Applied Soft Computing, 2016. 42: p. 325-341.
41. Jolai, F., R. Tavakkoli-Moghaddam, and M. Taghipour, *A multi-objective particle swarm optimisation algorithm for unequal sized dynamic facility layout problem with pickup/drop-off locations*. International Journal of Production Research, 2012. 50(15): p. 4279-4293.
42. Bagheri, M. and M. Bashiri, *A new mathematical model towards the integration of cell formation with operator assignment and inter-cell layout problems in a dynamic environment*. Applied Mathematical Modelling, 2014. 38(4): p. 1237-1254.
43. Lacksonen, T.A., *Static and dynamic layout problems with varying areas*. Journal of the Operational Research Society, 1994. 45(1): p. 59-69.
44. Amaral, A.R. and A.N. Letchford, *A polyhedral approach to the single row facility layout problem*. Mathematical programming, 2013. 141(1-2): p. 453-477.
45. Rosenblatt, M.J., *The dynamics of plant layout*. Management Science, 1986. 32(1): p. 76-86.
46. Kokash, N., *An introduction to heuristic algorithms*. Department of Informatics and Telecommunications, 2005: p. 1-8.

47. Yang, C.L., 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, 2011. 52(1): p. 303-309.
48. Aiello, G., G. La Scalia, and M. Enea, *A multi objective genetic algorithm for the facility layout problem based upon slicing structure encoding*. Expert Systems with Applications, 2012. 39(12): p. 10352-10358.
49. Lee, K.Y., S.N. Han, and M.I. Roh, *An improved genetic algorithm for facility layout problems having inner structure walls and passages*. Computers & Operations Research, 2003. 30(1): p. 117-138.
50. El-Baz, M.A., *A genetic algorithm for facility layout problems of different manufacturing environments*. Computers & Industrial Engineering, 2004. 47(2-3): p. 233-246.
51. Gonçalves, J.F. and M.G. Resende, *A biased random-key genetic algorithm for the unequal area facility layout problem*. European Journal of Operational Research, 2015. 246(1): p. 86-107.
52. Pourvaziri, H. and B. Naderi, *A hybrid multi-population genetic algorithm for the dynamic facility layout problem*. Applied Soft Computing, 2014. 24: p. 457-469.
53. Palomo-Romero, J.M., L. Salas-Morera, and L. García-Hernández, *An island model genetic algorithm for unequal area facility layout problems*. Expert Systems with Applications, 2017. 68: p. 151-162.
54. Ariafar, S. and N. Ismail, *An improved algorithm for layout design in cellular manufacturing systems*. Journal of Manufacturing Systems, 2009. 28(4): p. 132-139.
55. Meller, R.D. and Y.A. Bozer, *A new simulated annealing algorithm for the facility layout problem*. International Journal of Production Research, 1996. 34(6): p. 1675-1692.

56. McKendall Jr, A.R., J. Shang, and S. Kuppusamy, *Simulated annealing heuristics for the dynamic facility layout problem*. Computers & operations research, 2006. 33(8): p. 2431-2444.
57. Samarghandi, H. and K. Eshghi, *An efficient tabu algorithm for the single row facility layout problem*. European Journal of Operational Research, 2010. 205(1): p. 98-105.
58. Tari, F.G. and K. Ahadi, *Cellular layout design using Tabu search, a case study*. RAIRO-Operations Research, 2019. 53(5): p. 1475-1488.
59. Bozorgi, N., M. Abedzadeh, and M. Zeinali, *Tabu search heuristic for efficiency of dynamic facility layout problem*. The International Journal of Advanced Manufacturing Technology, 2015. 77(1-4): p. 689-703.
60. Bozer, Y.A., R.D. Meller, and S.J. Erlebacher, *An improvement-type layout algorithm for single and multiple-floor facilities*. Management Science, 1994. 40(7): p. 918-932.
61. Liu, J., H. Zhang, K. He, and S. Jiang, *Multi-objective particle swarm optimization algorithm based on objective space division for the unequal-area facility layout problem*. Expert Systems with Applications, 2018. 102: p. 179-192.
62. Hani, Y., L. Amodeo, F. Yalaoui, and H. Chen, *Ant colony optimization for solving an industrial layout problem*. European Journal of Operational Research, 2007. 183(2): p. 633-642.
63. Liu, J. and J. Liu, *Applying multi-objective ant colony optimization algorithm for solving the unequal area facility layout problems*. Applied Soft Computing, 2019. 74: p. 167-189.
64. Hasan, R.A., M.A. Mohammed, N. Țăpuș, and O.A. Hammood, *A comprehensive study: Ant Colony Optimization (ACO) for facility layout problem*. in *2017 16th RoEduNet Conference: Networking in Education and Research (RoEduNet)*. 2017. IEEE.

65. See, P.C. and K.Y. Wong, *Application of ant colony optimisation algorithms in solving facility layout problems formulated as quadratic assignment problems: a review*. International Journal of Industrial and Systems Engineering, 2008. 3(6): p. 644-672.
66. Samanta, S., D. Philip, and S. Chakraborty, *Bi-objective dependent location quadratic assignment problem: Formulation and solution using a modified artificial bee colony algorithm*. Computers & Industrial Engineering, 2018. 121: p. 8-26.
67. Saravanan, M. and P. Arulkumar, *An artificial bee colony algorithm for design and optimize the fixed area layout problems*. The International Journal of Advanced Manufacturing Technology, 2015. 78(9-12): p. 2079-2095.
68. Nagarajan, L., S. K. Mahalingam, S. Gurusamy, and V. K. Dharmaraj, *Solution for bi-objective single row facility layout problem using artificial bee colony algorithm*. European Journal of Industrial Engineering, 2018. 12(2): p. 252-275.
69. Yahya, M. and M. Saka, *Construction site layout planning using multi-objective artificial bee colony algorithm with Levy flights*. Automation in construction, 2014. 38: p. 14-29.
70. Soimart, P. and P. Pongcharoen. *Multi-row machine layout design using artificial bee colony*. in *2011 international conference on economics and business information*. 2011.
71. Welgama, P.S. and P.R. Gibson, *Computer-aided facility layout—a status report*. The International Journal of Advanced Manufacturing Technology, 1995. 10(1): p. 66-77.
72. Heragu, S.S. and A. Kusiak, *Machine layout problem in flexible manufacturing systems*. Operations research, 1988. 36(2): p. 258-268.
73. Welgama, P.S., *Computer aided design of manufacturing facilities*. 1993.



74. Lee, Y.H. and M.H. Lee, *A shape-based block layout approach to facility layout problems using hybrid genetic algorithm*. Computers & Industrial Engineering, 2002. 42(2-4): p. 237-248.
75. Mahdi, A.H., H. Amet, and M.C. Portmann, *Physical layout with minimization of the transport costs*. 1999.
76. Azimi, P. and E. Saberi, *An efficient hybrid algorithm for dynamic facility layout problem using simulation technique and pso*. Economic Computation & Economic Cybernetics Studies & Research, 2013. 47(4).
77. Turanoğlu, B. and G. Akkaya, *A new hybrid heuristic algorithm based on bacterial foraging optimization for the dynamic facility layout problem*. Expert Systems with Applications, 2018. 98: p. 93-104.
78. Yalaoui, N., L. Amodeo, H. Mahdi, and F. Yalaoui, *Hybrid method to solve a facility layout problem: Genetic Algorithm-Particle Swarm Optimization*. IFAC Proceedings Volumes, 2009. 42(4): p. 1251-1255.
79. Zha, S., Y. Guo, S. Huang, Q. Wu, and P. Tang, *A hybrid optimization approach for unequal-sized dynamic facility layout problems under fuzzy random demands*. Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, 2020. 234(3): p. 382-399.
80. Moslemipour, G., T. Lee, and Y. Loong, *Solving stochastic dynamic facility layout problems using proposed hybrid AC-CS-SA meta-heuristic algorithm*. International Journal of Industrial and Systems Engineering, 2018. 28(1): p. 1-31.
81. Guan, J. and G. Lin, *Hybridizing variable neighborhood search with ant colony optimization for solving the single row facility layout problem*. European Journal of Operational Research, 2016. 248(3): p. 899-909.
82. Leno, I.J., S.S. Sankar, and S. Ponnambalam, *MIP model and elitist strategy hybrid GA-SA algorithm for layout design*. Journal of Intelligent Manufacturing, 2018. 29(2): p. 369-387.

83. García-Hernández, L., L. Salas-Morera, C. Carmona-Muñoz, A. Abraham, and S. Salcedo-Sanz, *A Hybrid Coral Reefs Optimization—Variable Neighborhood Search Approach for the Unequal Area Facility Layout Problem*. IEEE Access, 2020. 8: p. 134042-134050.
84. Hosseini, S., A. Al Khaled, and S. Vadlamani, *Hybrid imperialist competitive algorithm, variable neighborhood search, and simulated annealing for dynamic facility layout problem*. Neural Computing and Applications, 2014. 25(7-8): p. 1871-1885.
85. Rodríguez, J. M., F.C. MacPhee, D.J. Bonham, and V.C. Bhavsar, *Solving the Quadratic Assignment and Dynamic Plant Layout Problems Using a New Hybrid Meta-Heuristic Approach*. in *HPCS*. 2004.
86. Mir, M. and M. Imam, *A hybrid optimization approach for layout design of unequal-area facilities*. Computers & Industrial Engineering, 2001. 39(1-2): p. 49-63.
87. Erel, E., J.B. Ghosh, and J.T. Simon, *New heuristic for the dynamic layout problem*. Journal of the Operational Research Society, 2003. 54(12): p. 1275-1282.
88. Hicks, P.E. and T.E. Cowan, *CRAFT-M for layout rearrangement*. Industrial Engineering, 1976. 8(5): p. 30-35.
89. Hassan, M.M., G.L. Hogg, and D.R. Smith, *SHAPE: a construction algorithm for area placement evaluation*. International Journal of Production Research, 1986. 24(5): p. 1283-1295.
90. Drezner, Z., *DISCON: A new method for the layout problem*. Operations Research, 1980. 28(6): p. 1375-1384.
91. Kaku, B.K., G.L. Thompson, and T.E. Morton, *A hybrid heuristic for the facilities layout problem*. Computers & operations research, 1991. 18(3): p. 241-253.
92. Boswell, S.G., *TESSA—A new greedy heuristic for facilities layout planning*. The International Journal Of Production Research, 1992. 30(8): p. 1957-1968.

93. Goetschalckx, M., *An interactive layout heuristic based on hexagonal adjacency graphs*. European Journal of Operational Research, 1992. 63(2): p. 304-321.

## CHAPTER 2

### A CLUSTERING-SEQUENCING MATH-HEURISTIC APPROACH FOR THE UNEQUAL-AREA FACILITY LAYOUT PROBLEM

In this chapter, the unequal-area facility layout problem (UA-FLP) is addressed, which aims at finding the best arrangement of unequal-area rectangular-shaped facilities with fixed dimensions in a given area. As explained in Chapter 1, since FLP is NP-Hard, exact optimization approaches can only solve smaller instances of the problem. For larger instances, heuristic, metaheuristic, and else are used to find sub-optimal solutions. The math-heuristic approach presented in this chapter splits the UA-FLP problem theoretically into three subproblems of clustering, selecting, and sequencing. This constructive algorithm can be used to obtain an initial solution for metaheuristics. A feasible solution is guaranteed even when the available shop floor space is relatively tight. Two computational experiments are done to demonstrate the advantages of using the proposed approach. Particle swarm optimization is implemented and proposed to demonstrate how the developed math heuristic approach can be used to initialize metaheuristics. The resulting hybridized math heuristic metaheuristic approach is validated using test cases from the literature.

#### *Nomenclature*

$i, i'$	Index set of machines $i, i' \in \{1, 2, \dots, M\}$
$j$	Index set of clusters $j \in \{1, 2, \dots, N\}$
$k$	Index set of locations $k \in \{1, 2, \dots, M\}$
$l$	Index set of levels $l \in \{1, 2, \dots, L_j\}$
$r$	Random number $r \in \{1, 2, \dots, N\}$
$H$	The horizontal dimension of the shop floor
$V$	The vertical dimension of the shop floor

$h_i^m$	The horizontal dimension of machine $i$
$v_i^m$	The vertical dimension of machine $i$
$f_{ii'}$	The flow of material between machines $i$ and $i'$
$R_k$	The rank of the position $k$ on the shop floor
$L_j$	The number of levels of cluster $j$
$h_j^c$	The horizontal dimension of cluster $j$
$v_j^c$	The vertical dimension of cluster $j$
$h_{jl}^c$	The horizontal dimension of level $l$ of cluster $j$
$v_{jl}^c$	The vertical dimension of level $l$ of cluster $j$
$f_j^c$	The total flow of material between machines in cluster $j$ and the rest of the machines.
$S_{ij}$	$= \begin{cases} 1 & \text{if machine } i \text{ is in cluster } j \\ 0 & \text{otherwise} \end{cases}$
$S'_{ijl}$	$= \begin{cases} 1 & \text{if machine } i \text{ is in cluster } j \text{ at level } l \\ 0 & \text{otherwise} \end{cases}$
$X_j$	$= \begin{cases} 1 & \text{if cluster } j \text{ is in the final layout} \\ 0 & \text{otherwise} \end{cases}$
$X'_{jk}$	$= \begin{cases} 1 & \text{if cluster } j \text{ is in the final layout at position } k \\ 0 & \text{otherwise} \end{cases}$
$(x_i, y_i)$	The coordination of the centroid of machine $i$

## 2.1. Introduction

The optimal arrangement of facilities in a given area is called the facility layout problem (FLP). According to Tompkins et al. [1], facilities planning and material handling costs account for about 20% to 50% of the operating expenses that these costs can be reduced by 10% to 20% having an efficient layout.

The FLP can be studied discretely or continuously [2]. Quadratic assignment problem (QAP) is well-known in the literature for modeling discrete FLP which assumes that the areas of the facilities are equal and the facilities' locations are fixed [3]. Koopmans and Beckman [4] were the first who formulated the facility layout problem as a QAP. The FLP with unequal-area facilities is too complex to be modeled using a discrete QAP model unless locations are going to be considered large enough to accommodate the largest facility. Of course, this is an approximation that could entail wasting in production floor footage square as well as unnecessary material handling costs. If the problem does not have any of these restrictions, it needs a continuous representation. In the continuous representation, facilities can have any size and shape and can be placed in any position as long as no overlapping occurs. The continuous representation of the facility layout problem usually ends up being a mixed-integer programming (MIP) model due to the need for auxiliary binary variables [5]. Other than QAP and MIP, the facility layout problem is also modeled as a linear integer programming [6], mixed-integer programming [7], the graph-theoretic problem [8], and the quadratic set-covering problem [9].

Sahni and Gonzalez [10] demonstrated that the FLP is an NP-Complete problem and hence the optimum answer is not achievable in polynomial time for the large size problems. There are heuristic and metaheuristic algorithms that can be employed to solve the large instances of the FLP sub-optimally.

Armour and Buffa [11] described the unequal-area facility layout problem (UA-FLP) as a rectangular region with fixed dimensions and a number of facilities, that each has a specific area, to be placed in that region. The sum of the areas of the facilities must be less than or equal to the plant area. In the following, some heuristic and metaheuristic approaches that have been used in the literature to address the UA-FLP are reported.

Systematic layout planning (SLP) introduced by Muther [12] is one of the successful heuristic strategies to solve the facility layout problem. SLP methodology has three steps. In the first step, the relative position of the machines (ignoring their required spaces) is determined by a relationship diagram or an adjacency graph. In the second step, a space relationship diagram is produced by incorporating the area of each machine to the

adjacency graph. Finally, a feasible block layout is produced in the third step by considering all other constraints.

An iterative heuristic and a Branch and Bound algorithm were presented by Bhowmik [13] for solving FLP. In this approach, using clustering methods, departments that are highly related are grouped into levels; using the Branch and Bound method, a mathematical model is solved to obtain the locations of the generated clusters at the different levels; and using the iterative heuristic algorithm, the different departments are located within each level. Park and Seo [14] considered the input and output points of facilities when studying the UA-FLP and proposed a two-step heuristic algorithm for solving the problem. Salimpour et al. [15] firstly proposed a clustering-sequencing approach for solving the UA-FLP.

An iterative heuristic procedure was presented by Taghavi and Murat [16] for solving the integrated layout design and product flow assignment. This procedure is based on the sequential location heuristic, the alternating heuristic of Cooper [17], and the perturbation algorithm. Karray et al. [18] developed an integrated approach for solving the UA-FLP using the fuzzy set theory and a genetic algorithm (GA).

Paes et al. [19] proposed two heuristic methods for solving the UA-FLP that are a basic GA and a GA with decomposition phases and quadrant restrictions. Palomo-Romero et al. [20] examined the use of parallel GA based on the island model to solve UA-FLP with a flexible bay structure and showed that the proposed approach improves the algorithm's exploration and the quality of solutions and prevents premature convergence and unnecessary execution time. An algorithm based on the artificial immune system was introduced by Ulutas and Kulturel-Konak [21] to solve the UA-FLP with a flexible bay structure. Wang et al. [22] solved the multi-objective UA-FLP using GA; the objectives were minimizing the material flow factor cost, minimizing the shape ratio factor, and maximizing the area utilization factor were the objectives.

Allahyari and Azab [23] formulated the UA-FLP by a mixed-integer nonlinear programming model and solved their model using a multi-start search simulated annealing (SA) algorithm. Moreover, the authors proposed a heuristic algorithm for the initialization of the SA. Nordin et al. [24] developed a mixed-integer nonlinear programming model for

the UA-FLP and proposed a hybrid GA-SA metaheuristic algorithm for solving the model. Different orientations of the departments were considered in their model.

Asl and Wong [25] presented a modified particle swarm optimization (PSO) algorithm to solve the UA-FLP in which two local search and department swapping methods were applied to improve the quality of solutions and avoid trapping in the local optima and. Kulturel-Konak and Konak [26] proposed a hybrid PSO and local search approach for UA-FLP. The authors used a relaxed flexible bay structure in which empty spaces in bays were allowed. A multi-objective PSO algorithm was presented by Liu et al. [27] to solve the UA-FLP. The objectives were the minimization of the material handling cost, maximization of the total adjacency value, and the maximization of the utilization ratio of the shop floor. Zhang and Wang [28] solved the construction site UA-FLP using a PSO-based methodology.

In this chapter, a hybrid heuristic- and mathematical programming-based approach is presented to solve UA-FLP. The remainder of the chapter is structured as follows. The problem statement is presented in Section 2.2; Section 2.3 explains the four steps of the math heuristic approach for solving the UA-FLP; the application of the developed constructive approach for initializing a PSO metaheuristic is presented in Section 2.4; two comparative analyses are provided in Section 2.5, and finally, conclusions are provided in Section 2.6.

## ***2.2. Problem Statement***

The problem at hand is to arrange unequal-area facilities in a given area to minimize the total material handling cost, which is proportional to the flow of materials between these facilities and the distances they travel. In this chapter, the arrangement of production facilities, which are mostly machine tools, on the shop floors of manufacturing plants is of interest and one of the most popular applications of FLP. Hence, in this chapter, the facilities are referred to simply as machines for brevity. The following assumptions are taken for the purpose of modeling and solving the problem:



- (a) Machines have an unequal-area rectangular shape with predetermined dimensions.
- (b) The machines' dimensions are determined considering the space required for the operator, material handling equipment, maintenance, etc.
- (c) The shop floor has a rectangular shape with known dimensions.
- (d) The flow of material between machines is known a priori.
- (e) Machines must be located within the shop floor in a way where no overlap occurs between them.
- (f) In calculations of the distances between machines, rectilinear distances between the machines' centroids are used.

### **2.3. Methodology**

As mentioned in Section 2.1, FLP is NP-Hard and hence, there is a need for a heuristic or metaheuristic approach to solve large-sized instances of the problem. This section proposes a math heuristic approach, which is comprised of a combined heuristic- and mathematical-programming technique that has four steps for solving the problem. By definition, math heuristics are algorithms where mathematical programming is used in a heuristic fashion [29].

#### **2.3.1. Step 1: Grouping Machines in Different Clusters**

In this step,  $N$  clusters are generated in a way that machines are selected randomly to form the clusters created from bottom up. Each machine is considered to form a level. When machine  $i$  is selected to be in cluster  $j$  at level  $l$ , the  $S'_{ijl}$  and hence, the  $S_{ij}$  and is set to 1 (Equation 2.1). For any cluster, level  $(l + 1)$  is not to be assigned until that of  $l$  is (Equation 2.2).

$$S_{ij} = \max\{S'_{ijl} | \forall l\} \quad \forall i, j \quad (2.1)$$

$$\sum_{i=1}^M S'_{ij(l+1)} \leq \sum_{i=1}^M S'_{ijl} \quad \forall j, l \quad (2.2)$$

Then, the vertical dimension of level  $l$  in cluster  $j$ ,  $v_{jl}^c$ , is set to be equal to the vertical dimension of the assigned machine within that level,  $v_i^m$  (Equation 2.3).

$$v_{jl}^c = v_i^m S'_{ijl} \quad \forall i, j, l \quad (2.3)$$

The vertical dimension of each cluster is that of the shop floor (Equation 2.4). The sum of the vertical dimensions of all levels in a cluster can not be more than the vertical dimension of that cluster. As the dimensions of machines are different, the number of levels generated in each cluster,  $L_j$ , is also different (Equation 2.5).

$$v_j^c = V \quad \forall j \quad (2.4)$$

$$\sum_{l=1}^{L_j} v_{jl}^c \leq v_j^c \quad \forall j \quad (2.5)$$

The horizontal dimension of a cluster is the maximum of the horizontal dimensions of the machines assigned to that cluster (Equation 2.6). The horizontal dimension of each level in a cluster is equal to the horizontal dimension of that cluster (Equation 2.7).

$$h_j^c = \max\{h_i^m S_{ij} | \forall i\} \quad \forall j \quad (2.6)$$

$$h_{jl}^c = h_j^c \quad \forall j, l \quad (2.7)$$

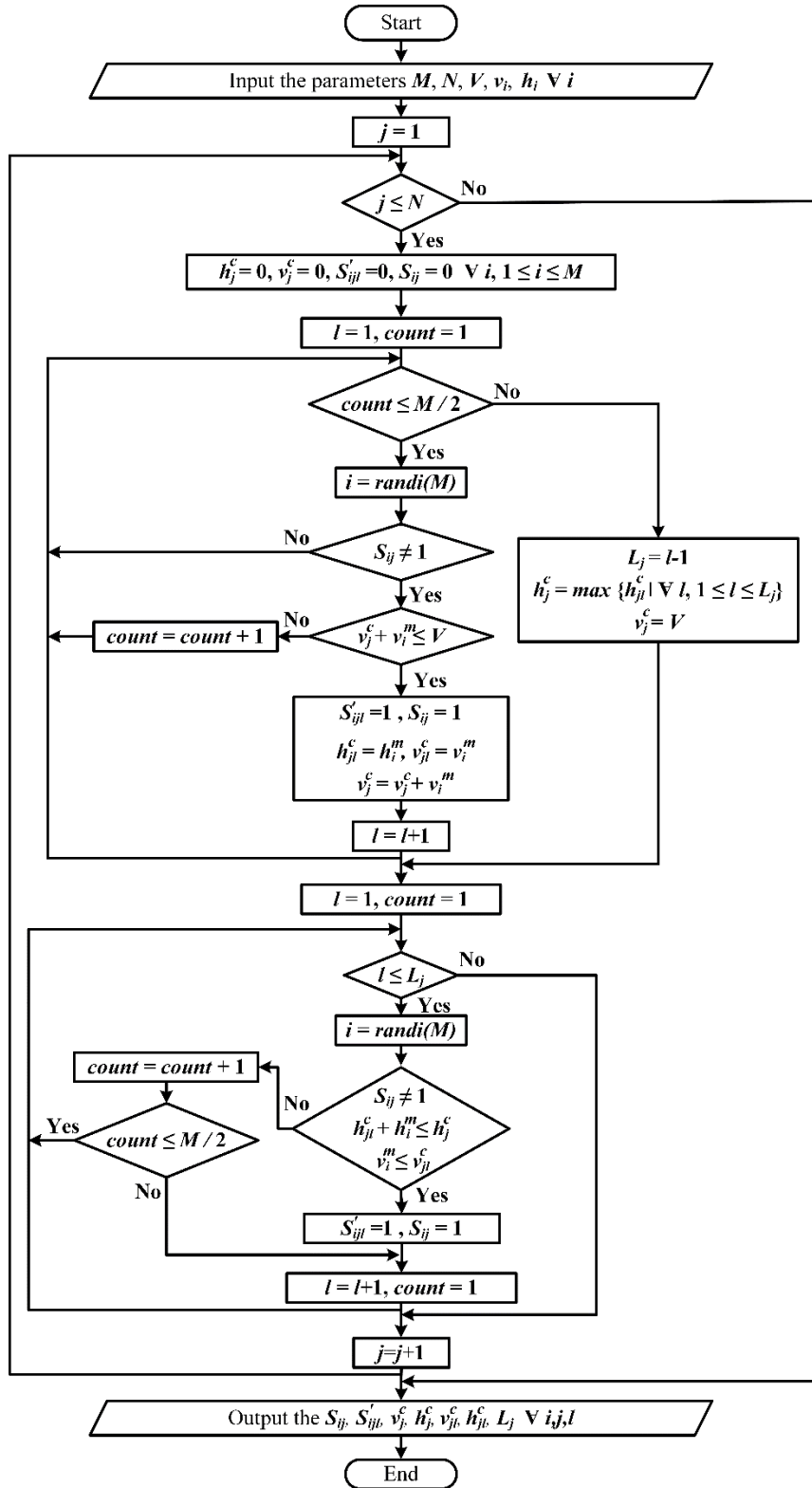


Figure 2.1. Flowchart of the algorithm for generating clusters

Next, it is examined if those machines that have not been selected to be in a cluster can be located horizontally beside the assigned machine at each level and the variables  $S_{ij}$  and  $S'_{ijl}$  are updated accordingly. This happens only if the vertical dimension of a machine is less than that of a cluster's level and the machine's horizontal dimension plus the horizontal dimension of the machine that is already assigned to that level, is less than the horizontal dimension of the cluster's level. This algorithm is repeated  $N$  times in order to provide  $N$  clusters in such a way that it is possible to have the same machine in different clusters. The steps the algorithm undertakes to group the machines into clusters are summarized in the flowchart presented in Figure 2.1.

The parameter  $f_{ii'}$  which represents the flow of material between machines  $i$  and  $i'$ , is used to calculate the flow of material between machines in each cluster with the rest of the machines (Equation 2.8).

$$f_j^c = \sum_{i=1}^M \sum_{i'=1}^M f_{ii'} S_{ij} (1 - S_{i'j}) \quad \forall j \quad (2.8)$$

### 2.3.2. Step 2: Selecting Between Clusters and Sequencing Them

In this section, clusters that are going to be in the final layout are selected optimally and sequenced horizontally with the objective that the clusters that have a higher flow of materials with the rest of the clusters,  $f^c$ , be in positions that have a higher rank on the shop floor. It should be noted that each machine should be naturally assigned to one and only one cluster in the final layout and hence, if a cluster is selected, other clusters that contain the same machines as the chosen cluster can not be selected. In step 1, clusters are generated in a way that their vertical dimensions do not exceed the vertical dimension of the shop floor. In this step, the horizontal dimension of the shop floor should not be violated by controlling the summation of the horizontal dimensions of the selected clusters. To provide different solutions, the first cluster is selected randomly.

Parameter  $R_k$  represents the rank of the position  $k$  on the shop floor (Equation 2.9). It is assumed that the centroid of the shop floor has the maximum rank,  $M$ , which is the number of machines. The rank of a position is decremented by one as you move away from the centroid either way due to the symmetry, as shown in Figure 2.2.

$$R_k = k + \lfloor (M - k)/2 \rfloor \quad \forall k \quad (2.9)$$

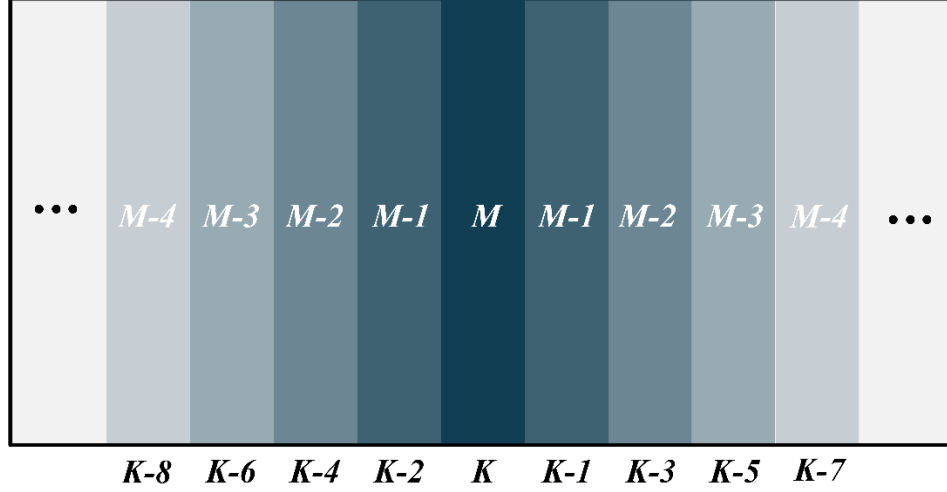


Figure 2.2. Possible positions on the shop floor and their corresponding rankings

To solve this step, the following integer linear programming model is formulated:

$$\text{Maximize } Z = \sum_{j=1}^N \sum_{k=1}^M f_j^c R_k X'_{jk} \quad (2.10)$$

Subject to:

$$\sum_{j=1}^N X_j S_{ij} = 1 \quad \forall i \quad (2.11)$$

$$X_r = 1 \quad (2.12)$$

$$\sum_{k=1}^M X'_{jk} - X_j \leq 0 \quad \forall j \quad (2.13)$$

$$\sum_{j=1}^N X'_{jk} \leq 1 \quad \forall k \quad (2.14)$$

$$\sum_{j=1}^N X_j h_j^c \leq H \quad (2.15)$$

$$X_j, X'_{jk} \in \{0,1\} \quad \forall j, k \quad (2.16)$$

The objective function (2.10) maximizes the sum of the flow of materials of the selected clusters with the rest of the clusters multiplied by the ranks of their assigned positions. The objective function is maximized when the clusters, which have more amount of flow with other clusters, are selected and are assigned to positions with the highest of merit. The cluster, which has the highest amount of flow with other clusters is located in the centroid  $R_k$ , the next two are at positions with the rank of  $R_{k-1}$ , and the next two are at positions with the rank of  $R_{k-2}$  and so forth.

Constraint (2.11) ensures that each machine is in one and only one cluster in the final layout. Constraint (2.12) ensures that the first cluster is chosen randomly, which is key to the overall working of the developed approach; it is necessary to avert having the model being trapped in a local optimum by having it act greedy and assign the best cluster with the most of the interactions to the highest rank position at the centroid of the shop floor. Instead, the model is run a number of times each time having a different set of clusters; this way various clusters are being assigned to the centroid position and tried. The best of these generated layouts and cluster sequences is then chosen. Constraint (2.13) ensures that a cluster is only assigned once if it is chosen. Constraint (2.14) ensures that not more than one cluster is assigned to each position. Constraint (2.15) ensures that the sum of the horizontal dimensions of the selected clusters does not exceed that of the overall given shop. Finally, constraint (2.16) indicates that the decision variables are binary.

### ***2.3.3. Step 3: Sequencing Machines in Each Cluster***

In the previous steps, machines that are horizontally together in each cluster and the clusters that are in the final layout are selected. In this step, first, the coordinates of the selected clusters based on their dimensions and positions on the shop floor are determined. Second, the best arrangement of the machines in each cluster and accordingly the coordinates of the machines are obtained.

For finding the coordinates of the clusters, it should be noted that centering a cluster on the shop floor does not mean that the cluster is exactly in the middle of the shop floor in the final layout. Sometimes after placing the cluster that has the most amount of flow with the rest of the clusters at the centroid of the shop floor and the rest of the clusters one after the other next to it in their positions as shown in Figure 2.3 (a), the layout may exceed the dimensions of the shop floor; see Figure 2.3 (a) for an illustration of this. To fix this issue, all the clusters are treated as a batch and placed in the centroid of the shop floor as depicted in Figure 2.3 (b).

The basic idea for sequencing the machines vertically in each cluster is the same as sequencing the clusters horizontally on the shop floor. This means that the centroid of each cluster is assumed to have the maximum position rank and the machine that has the most flow of material with the rest of the machines is assigned to that position and so forth. It

should be noted that if two machines are in one level, they are treated as a single entity (practically one single machine) in regards to the flow of material calculations, which are the summation of the flow of material of both of these machines with the rest of the machines. The arrangement of machines should be in a way that machines should not have any overlap with each other and they should not exceed their clusters' boundaries. Locating a machine at the centroid of a cluster follows the same scheme explained in Figure 2.3.

The machines that are located at the endpoint clusters only have interaction between themselves and the machines that are located on one side of them. Hence, to lower the total material handling cost, these machines are not horizontally centralized and are moved toward the centroid of the shop floor as long as the boundaries of their cluster are not violated (see Figure 2.4).

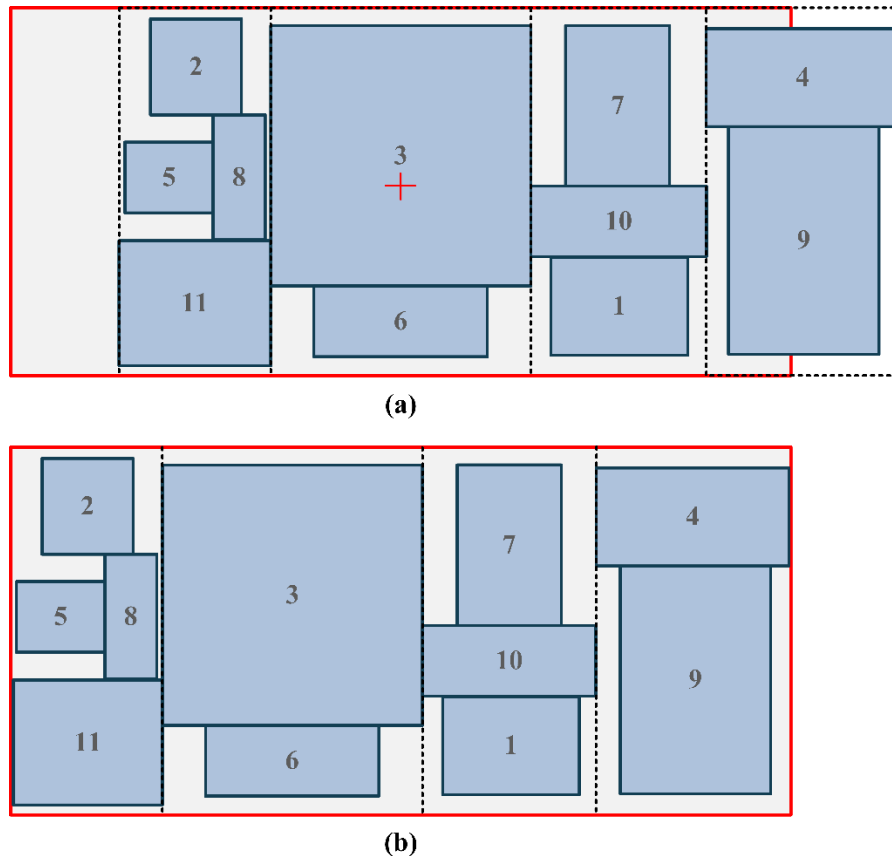


Figure 2.3. A sample layout (a) The cluster which contains machines 3 and 6 is centralized and the configuration of the clusters exceeds the horizontal dimension of the shop floor. (b) All the clusters are treated as a batch and placed in the centroid of the shop floor.

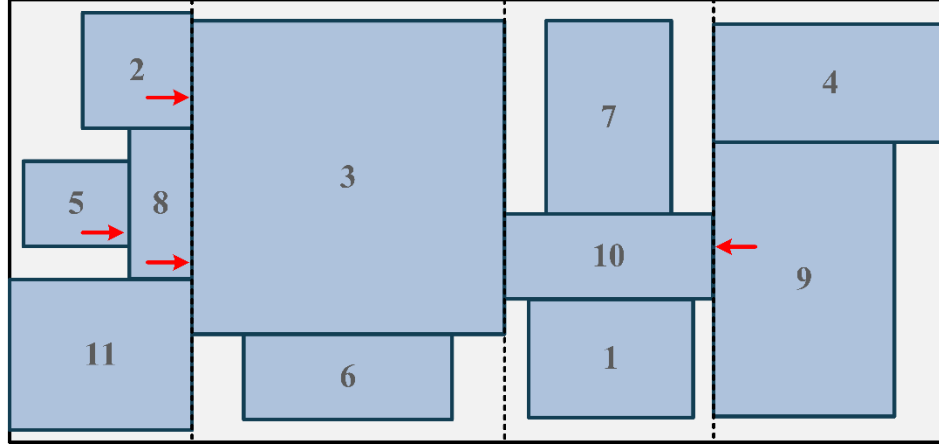


Figure 2.4. Decentralizing machines which are in the endpoint clusters

#### 2.3.4. Step 4: Calculation of Total Distance Traveled by All Units

In this step, the coordinates of the machines are known. Hence, the distance between them can be easily obtained. The total material handling cost ( $TC$ ) is the sum of the product of the flow of materials between each pair of machines and the rectilinear distance between them. This cost is calculated using Equation 2.17.

$$TC = \sum_{i=1}^{M-1} \sum_{i'=i+1}^M f_{ii'} (|x_i - x_{i'}| + |y_i - y_{i'}|) \quad (2.17)$$

#### 2.4. Particle Swarm Optimization (PSO)

As mentioned in Section 2.1, FLP is NP-Hard, and hence no exact solutions can be produced in polynomial time solving larger instances of the problem (more than 15 facilities). To overcome this issue, a hybridized math heuristic metaheuristic approach is used. Generally speaking, metaheuristics perform better than heuristics. And, hence combining a powerful math-heuristic initializing a metaheuristic is proving to provide better results.

One of these metaheuristic methods is PSO that is used here as an example. Kennedy and Eberhart [30] are the first who developed the particle swarm optimization, which is an evolutionary computation algorithm based on the social behavior of swarms (swarm intelligence) [31].



PSO is initialized by creating a population of random solutions, named particles that each is assigned a random velocity. Each particle represents a candidate solution that is located somewhere in the solution space with a fitness value. In each iteration, the best location a particle has experienced up to the current iteration as well as the best location of all the particles (the whole population) thus far is saved in the memory. In every iteration, a particle moves towards both its own best location and the global best location so far using its velocity [32]. The basic PSO algorithm is modified by introducing an inertia weight (also named as damping weight) to the model [33]. This weight controls the effect of the previous velocities on the current velocity. A large inertia weight (closer to one) results in a better exploration, while smaller inertia (less than 0.4) leads to better exploitation [27, 34]. The particles are maneuvered according to Equations (2.18) and (2.19):

$$V_{it} = w \times V_{i(t-1)} + c_1 \times rand_{1t} \cdot (Xbest_{i(t-1)} - X_{i(t-1)}) + c_2 \times rand_{2t} \cdot (Xgbest_{(t-1)} - X_{i(t-1)}) \quad (2.18)$$

$$X_{it} = X_{i(t-1)} + V_{it} \quad (2.19)$$

where  $t$  is the index for iterations,  $i$  is the index for particles,  $w$  is the inertia weight,  $X$  is the position vector and  $V$  is the velocity vector of a particle in an iteration,  $Xbest$  is the best position vector that a particle has experienced so far before the current iteration and the  $Xgbest$  is the global best position vector that all the particle have experienced so far before the current iteration. To prevent a particle from moving faraway outside the boundaries of the search space, the velocity of particles should be anywhere between the predefined upper limit,  $V_{max}$ , and the lower limit,  $V_{min}$ . The cognition coefficient,  $c_1$ , and the social coefficient,  $c_2$ , facilitate the convergence of the PSO algorithm and take a value between 0 to 2 [27].  $rand_{1t}$  and  $rand_{2t}$  are vectors of uniform random numbers between 0 and 1 in iteration  $t$ . The size of all the above-mentioned vectors are equal to the total number of decision variables.

PSO is a population-based search method that has the ability to search for the optimum solution simultaneously in many directions [35]. PSO is shown to be successful when dealing with static optimization problems [36].

### **2.4.1. PSO Algorithm**

The PSO algorithm has the following steps [37]:

1. Particles are created, their position vector and velocity vector are initialized and the corresponding cost of the position of each particle is evaluated
2. The best position of each particle is set to be equal to the initial position of that particle and the one that has the best fitness function value (minimum cost or maximum profit) is set as the global best position.
3. The steps “a” to “f” are repeated until the stopping criteria is met. The stopping criteria can be reaching the maximum number of iterations, or no improvement for a certain number of iterations.
  - (a) Each particle’s velocity is updated using Equation (2.18).
  - (b) If the updated velocity is less than the lower limit, then it is set to the lower limit and if it is higher than the upper limit, it is set to the upper limit.
  - (c) Each particle’s position is updated using Equation (2.19).
  - (d) The fitness of the updated position of each particle is evaluated
  - (e) The local improvement is applied
  - (f) If the current position of a particle is better than the particle’s best position, then the best position of the particle is updated and then if his position is better than the global best position as well, the global best position is updated, too.
4. The global best position and cost is reported.

### **2.4.2. Solution Representation**

The decision variables of the UA-FLP are the  $x$  and  $y$  coordinates of the centroid of all the machines.  $x_i^{min}$  and  $x_i^{max}$  are the minimum and maximum possible values for the  $x$ -coordinate of the centroid of machine  $i$ , and  $y_i^{min}$  and  $y_i^{max}$  are the minimum and maximum possible values for the  $y$ -coordinate of the centroid of machine  $i$  which are calculated using Equations (2.20)-(2.25).

$$x_i^{min} = \frac{h_i^m}{2} \quad \forall i \quad (2.20)$$

$$x_i^{max} = H - \frac{h_i^m}{2} \quad \forall i \quad (2.21)$$

$$x_i^{min} \leq x_i \leq x_i^{max} \quad \forall i \quad (2.22)$$

$$y_i^{min} = \frac{v_i^m}{2} \quad \forall i \quad (2.23)$$

$$y_i^{max} = V - \frac{v_i^m}{2} \quad \forall i \quad (2.24)$$

$$y_i^{min} \leq y_i \leq y_i^{max} \quad \forall i \quad (2.25)$$

The usual practice encoding solutions in PSO is using normalized variables,  $\hat{x}_i$  and  $\hat{y}_i$ , the values of which ranges from 0 to 1 [25]. However, for evaluating the solutions, the values of  $x_i$  and  $y_i$  are needed that can be found using Equations (2.26)-(2.28).

$$x_i = x_i^{min} + (x_i^{max} - x_i^{min})\hat{x}_i \quad \forall i \quad (2.26)$$

$$y_i = y_i^{min} + (y_i^{max} - y_i^{min})\hat{y}_i \quad \forall i \quad (2.27)$$

$$0 \leq \hat{x}_i, \hat{y}_i \leq 1 \quad \forall i \quad (2.28)$$

### 2.4.3. Objective Function

The objective function,  $Z$ , is the summation of  $TC$  calculated by Equation (2.17) as well as the total penalty function to prevent overlapping between machines which can be obtained using Equations (2.29)-(2.32).

$$Z = TC + \sum_{i=1}^{M-1} \sum_{i'=i+1}^M pI_{ii'} \quad (2.29)$$

$$I_{ii'}^x = \frac{h_i^m + h_{i'}^m}{2} - |x_i - x_{i'}| \quad \forall i' > i \quad (2.30)$$

$$I_{ii'}^y = \frac{v_i^m + v_{i'}^m}{2} - |y_i - y_{i'}| \quad \forall i' > i \quad (2.31)$$

$$I_{ii'} = \begin{cases} I_{ii'}^x \times I_{ii'}^y & \text{if } I_{ii'}^x, I_{ii'}^y > 0 \\ 0 & \text{Otherwise} \end{cases} \quad \forall i' > i \quad (2.32)$$

where  $p$  is the penalty cost per square unit of overlapping between a pair of machines. The calculation of the amount of Infraction between a pair of machines,  $I_{ii'}$ , is similar but not completely the same as the one used by Chwif et al. [38]. Moreover, the way the total penalty function is involved in the objective function is different.

#### **2.4.4. Local Improvement Search**

The local improvement method — similar to the one proposed by Asl and Wong [25] — is used to improve the updated solution of each particle in each iteration. For each machine, the algorithm checks whether a move with a length *delta* to the right, left, up, or down can provide a better feasible solution. The value of *delta* is selected randomly by the algorithm but it is restricted to a prespecified range.

### **2.5. Computational Results**

In this section, first, the performance of the proposed math heuristic is compared with one of the benchmarks in the literature (Comparison I). And next, the proposed approach is used to generate the initial solution for the PSO metaheuristic and the results are compared with the case where the initial solutions are generated randomly (Comparison II).

Steps 1, 3, and 4 of the algorithm, as well as the PSO algorithm, are deployed using MATLAB, while Step 2 is using Xpress Mosel algebraic modeling language and solvers. The algorithms are tested on a 64-bit architecture with an Intel Xeon processor, the clock speed of which is 3.07 GHz, and a 6 GB of memory personal computer.

#### **2.5.1. Comparison I**

In Comparison I, the computational results of the proposed math heuristic are presented and compared with those obtained by Karray et al. [39]. The input data of the problem, presented in Tables 2.1, 2.2 and 2.3, are the same as the data they have used; however, with two small differences. The first difference is that facility 6 in that example [39] is a building consists of two attached buildings, the total area of which is given. For comparison and as the facilities in the proposed approach are assumed to have a rectangular shape, facility 6 is split into two separate ones (rectangular shape and equal in the area) and named 6 and 7. However, in order to make sure that they stick together in the final layout, the flow of material between them is set to a very high amount. The second difference is the terminology used; to maintain consistency the term “facility” is replaced by “machine.”

The values in Table 2.3 are not integer because Karray et al. [39] have assumed that there are three different types of flow between facilities which are material flow, information flow, and equipment flow; each of these variables is assumed to have five membership functions and also a weight factor. After applying fuzzification, which results in closeness relationship values being generated for each pair of facilities, and after establishing the decision-making rules, the min-max composition rule of the interface and the centroid of the area are used for the defuzzification process and generation of the values of the facility relationship chart which are not integer values. To be consistent with the terminology used in this chapter, Table 2.3 is named the material handling flow between pairs of machines.

The final layout obtained solving the problem using the proposed math heuristic is presented in Figure 2.5. (a) Having this layout, the total material handling cost is \$9976.3. The best layout presented by Karray et al. [39] is depicted in Figure 2.5. (b). To compare the two layouts, the total material handling cost of this layout is calculated using Equation 16. The obtained result is \$9990.5.

Table 2.1. Dimensions of the shop in meter

Horizontal Dimension ( $H$ )	Vertical Dimension ( $V$ )
27	16

Table 2.2. Dimensions of the machines in meter

Machine's Number	1	2	3	4	5	6	7
Horizontal Dimension ( $h^m$ )	4	3	5	3	3	12	6
Vertical Dimension ( $v^m$ )	2	4	5	2	1	6	12

Table 2.3. The material handling flow between pairs of machines

Machine's Number	7	6	5	4	3	2	1
1	81.7	81.7	48.4	11.2	10.6	10.5	-
2	36.8	36.8	12.3	15.3	52.9	-	
3	36.5	36.5	46.9	29.8	-		
4	10.8	10.8	10.8	-			
5	11.3	11.3	-				
6	100	-					
7	-						

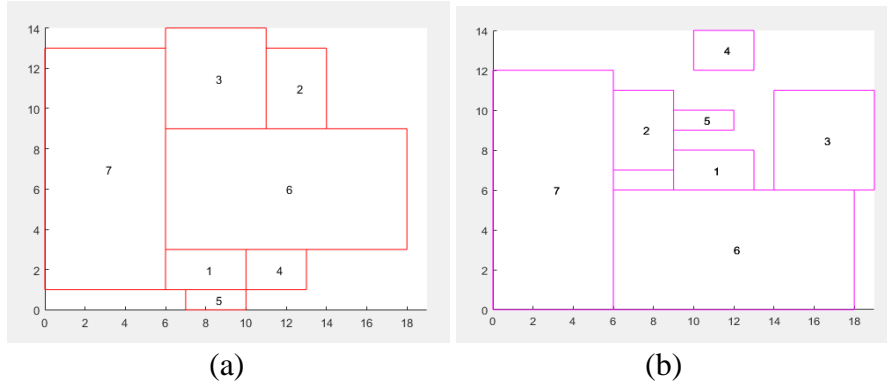


Figure 2.5. The best layout (a) result of solving by proposed hybrid heuristic- and mathematical-programming-based approach (b) obtained by Karray et al. [39] using genetic algorithms.

It can be concluded that the proposed approach obtained a better solution even though the solutions are quite close (0.14% better). Moreover, this solution makes better use of the available space; and hence, the resulting unutilized spaces can be used to make room for larger restrooms, cafeterias, or secondary functions.

### 2.5.2. Comparison II

In Comparison II, the proposed math heuristic is used to generate an initial solution for the PSO metaheuristic approach to solve UA-FLP and the results are compared with the case where PSO starts with completely random initial solutions.

Some of the benchmarks on UA-FLP have not restricted the floor space's dimensions and considered that the dimensions of the shop floor are allowed to be decided by the algorithm [19, 23, 40-44] or have considered that the area of the shop floor is far larger than that of the total machines combined [25, 27]. However, this is not the case in real manufacturing systems; the shop floor has specific dimensions and its area may not be that relatively large.

The proposed constructive approach performs well and results in feasible solutions in the cases where the area of the shop is restricted and the difference in required vs available areas/aspect-ratio is tight. When this layout is given to PSO as one of the initial solutions, PSO improves this solution and results in a better feasible layout. Nevertheless, PSO, when fed with random initial solutions, does not come up with feasible solutions in such cases.

Two data sets, one with 11 machines and the other one with 20 machines, are used to test the benefit of this constructive algorithm. The data sets are available in the original references.

The PSO parameters  $w$ ,  $c_1$ , and  $c_2$  are set to 0.4, 2, and 2, respectively. The size of the population is 50 and the stopping criteria is reaching the maximum number of iterations that is set to 500.

#### *2.5.2.1. Data Set 1*

The first data set with 11 facilities is first introduced by Imam and Mir [43]; the area of the shop floor in the test case is not restricted. The total area of the facilities is 124.4 m<sup>2</sup>.

Figure 2.6 displays the result for this problem considering a shop floor with an area of 156 m<sup>2</sup> (13 m × 12 m). As can be seen in Figure 2.6 (a), the proposed math heuristic algorithm generates a feasible solution with the total material handling cost of \$1567. By using this layout as one of the initial solutions for PSO, an improved layout shown in Figure 2.6 (b) is obtained with the total material handling cost of \$1520.11. However, when the PSO is initialized all randomly, the generated layout is not feasible (Figure 2.6. (c)). It can be concluded that the proposed algorithm can find a feasible solution for the problem even if the empty space is only 20%.

Figure 2.7 shows the solution of the problem considering a shop floor with an area of 169 m<sup>2</sup> (13 m × 13 m). Figure 2.7 (a) depicts that the proposed math heuristic algorithm generates a feasible solution with the total material handling cost of \$1495.3. As this layout is used as one of the initial solutions of PSO, an improved layout shown in Figure 2.7 (b) is obtained with the total material handling cost of \$1478.90. However, by randomly initializing the PSO, the resulted layout is not feasible (Figure 2.7. (c)). It can be concluded that the proposed algorithm can find a feasible solution for the problem even if the empty space is only 26%.

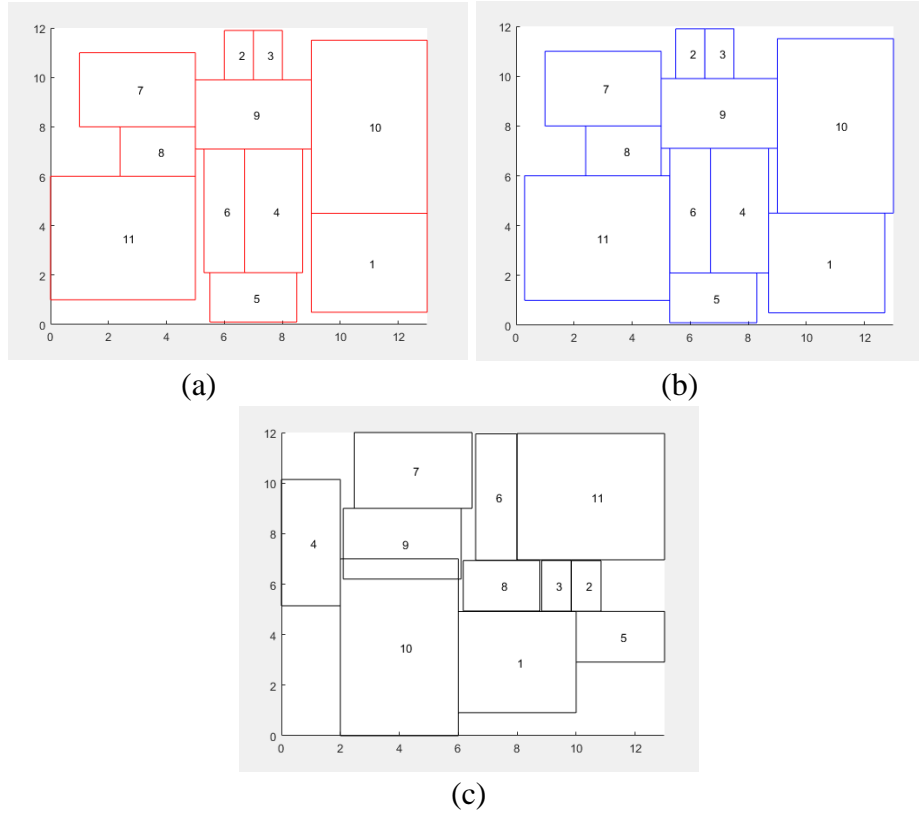


Figure 2.6. A shop floor with the size of 13 m × 12 m (a) The layout resulted from the proposed math heuristic (b) The layout resulted applying PSO when one of the initial solutions is obtained by the math heuristic algorithm (c) The resulting layout obtained by PSO when all the initial solutions are generated randomly (infeasible).



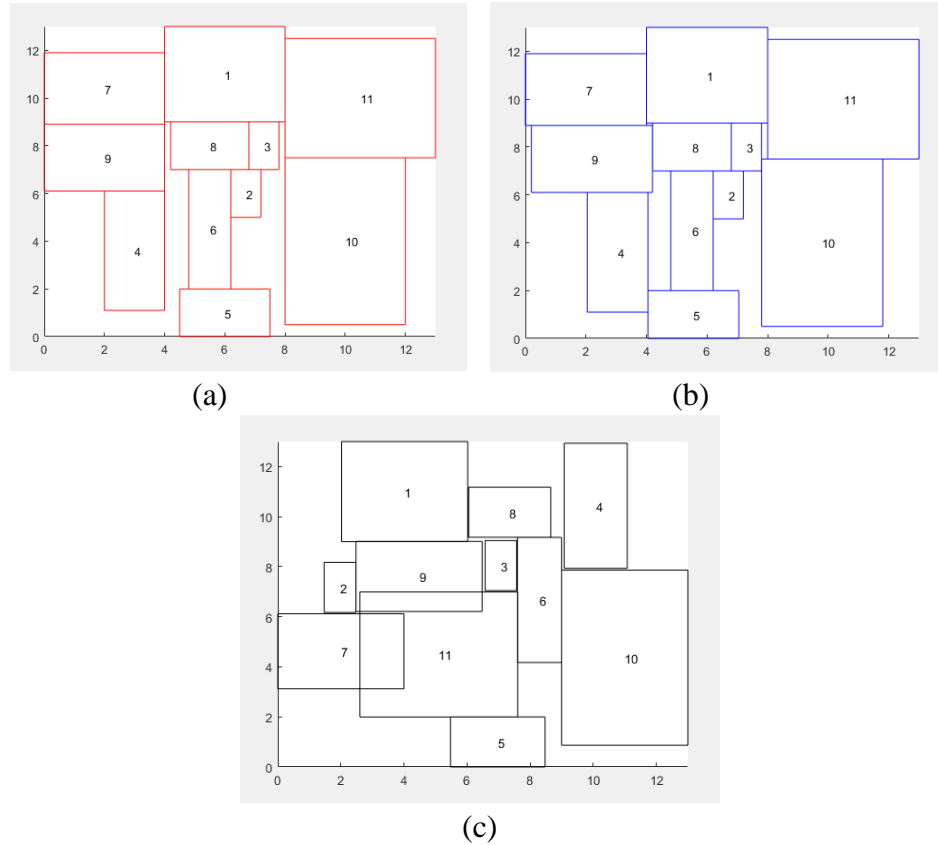


Figure 2.7. A shop floor with the size of 13 m × 13 m (a) The layout resulted from the proposed math heuristic (b) The layout resulted applying PSO when one of the initial solutions is obtained by the math heuristic algorithm (c) The resulting layout obtained by PSO when all the initial solutions are generated randomly (infeasible).

#### 2.5.2.2. Data Set 2

The second data set with 20 facilities is first introduced by Mir and Imam [40]; the area of the shop floor in the test case is not restricted. The total area of the facilities is 101 m<sup>2</sup>.

Figure 2.8 presents the result of solving the problem considering a shop floor with an area of 112 m<sup>2</sup> (14 m × 8 m). As shown in Figure 2.8 (a), the proposed math heuristic algorithm generates a feasible solution with the total material handling cost of \$3421. When this layout is used as one of the initial solutions of the PSO algorithm, PSO improves the layout and results in the layout presented in Figure 2.8 (b) with the total material handling cost of \$3418.04. However, when PSO is initialized randomly, the generated layout is not feasible (Figure 2.8. (c)). Thus, it can be concluded that the proposed algorithm can find a feasible solution for the problem even if the empty space is only 10%.

Figure 2.9 shows the result of solving the problem considering a shop floor with an area of  $120 \text{ m}^2$  ( $15 \text{ m} \times 8 \text{ m}$ ). As shown in Figure 2.9 (a), the proposed math heuristic algorithm generates a feasible solution with the total material handling cost of \$3948. When this layout is used as one of the initial solutions of the PSO algorithm, PSO improves the layout and results in the layout presented in Figure 2.9 (b) with the total material handling cost of \$3770. However, when PSO is initialized all randomly, the generated layout is not feasible (Figure 2.9. (c)). It can be concluded that the proposed algorithm can obtain a feasible solution for the problem even if the surplus in the area is only 16% of the total shop area.

Considering a shop floor with an area of  $126 \text{ m}^2$  ( $14 \text{ m} \times 9 \text{ m}$ ), the solution to the problem is presented in Figure 2.10. The proposed math heuristic algorithm generates a feasible solution with the total material handling cost of \$3557, which is shown in Figure 2.10 (a). By utilizing this as an initial solution for PSO, the PSO algorithm improves the layout and reduces the total material handling cost to \$3432.32. This layout is presented in Figure 2.10 (b). When the PSO is initialized all randomly, the generated layout is not feasible which is depicted in Figure 2.10 (c). It can be concluded that the proposed algorithm can find a feasible solution for the problem even if the empty space is only 20%.

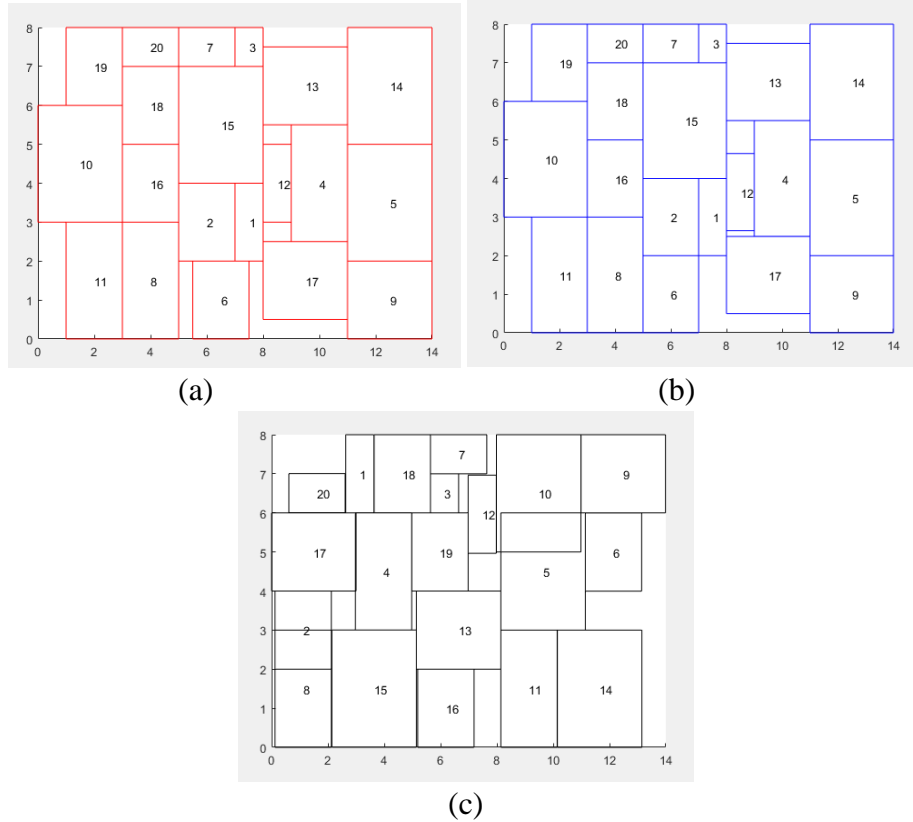


Figure 2.8. A shop floor with the size of 14 m × 8 m (a) The layout resulted from the proposed math heuristic (b) The layout resulted applying PSO when one of the initial solutions is obtained by the math heuristic algorithm (c) The resulting layout obtained by PSO when all the initial solutions are generated randomly (infeasible).

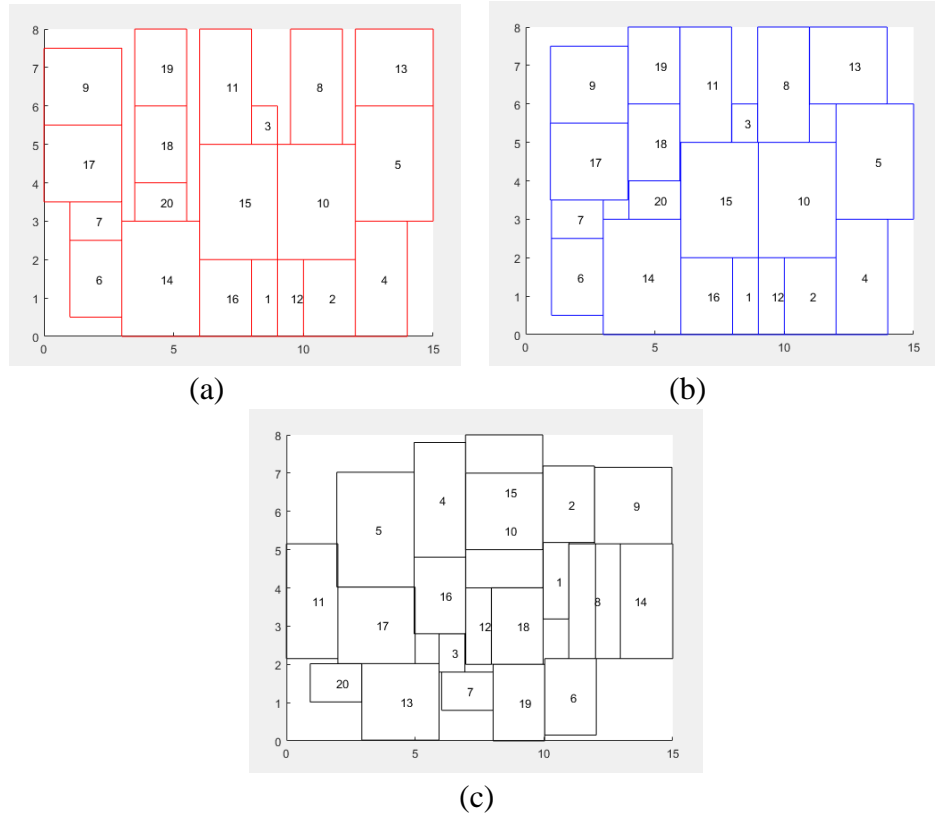


Figure 2.9. A shop floor with the size of 15 m × 8 m (a) The layout resulted from the proposed math heuristic (b) The layout resulted applying PSO when one of the initial solutions is obtained by the math heuristic algorithm (c) The resulting layout obtained by PSO when all the initial solutions are generated randomly (infeasible).

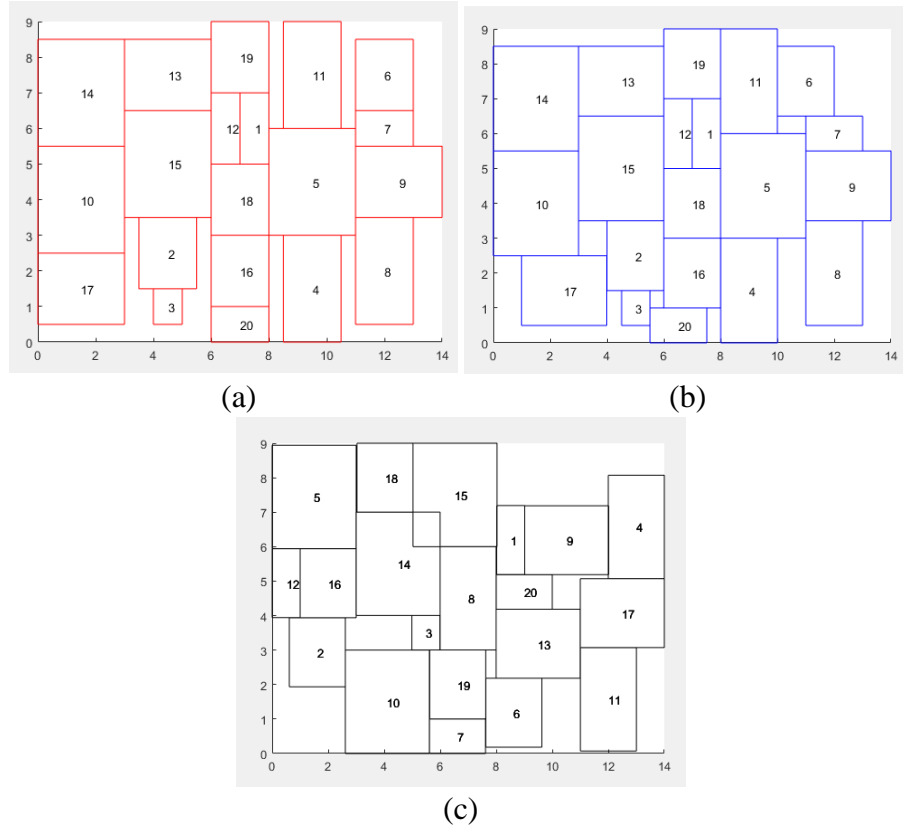


Figure 2.10. A shop floor with the size of 14 m × 9 m (a) The layout resulted from the proposed math heuristic (b) The layout resulted applying PSO when one of the initial solutions is obtained by the math heuristic algorithm (c) The resulting layout obtained by PSO when all the initial solutions are generated randomly (infeasible).

## 2.6. Conclusions

FLP is one of the most critical functions in production and manufacturing system design. There is a lot of interest in the facility layout problem; and due to the NP-hardness of the problem, near-optimal algorithms and heuristics have been developed as substitution of optimal exact algorithms solving large instances of FLP.

A math heuristic approach is introduced in this chapter for the UA-FLP with continuous representation. This constructive algorithm solves the problem by clustering the facilities, which in this chapter are referred to as machines, selecting and sequencing the clusters and then locating them on the shop floor. Consequently, facilities in each cluster are sequenced and placed. The common objective of the FLP is to minimize the material handling cost. In this algorithm, this goal is reached by using the developed alternate objective function

based on the proposed ranking scheme. By maximizing the objective function, clusters that have more flow of materials with other clusters are assigned closer to the centroid of the shop. Hence, the inter-distances between facilities that have a higher amount of flow of materials among each other are minimized and as a result, the total material handling cost is minimized.

One of the prominent aspects of this method is that it can be used as an initial solution to the metaheuristic algorithms such as PSO, which was explained in this chapter. The experimental results showed that if the total area of all the facilities compares to the area of the shop floor is not relatively small, PSO is not able to come up with a feasible solution. Whereas, if the proposed constructive layout is given as one of the initial solutions, the result is always feasible and PSO improves the given layout. Moreover, the randomness by which different facilities are assigned to different clusters and the selection of the first final layout's cluster provides the ability to generate different sub-optimum layouts each time the proposed algorithm is used. As this constructive algorithm can be used for initializing metaheuristic algorithms, the randomness improves not only the exploration capability of the algorithm but also makes it possible to have more than one initial layout for the population-based metaheuristic algorithms such as PSO and GA.

Moreover, even when the total area of the machines compare to the area of the shop floor is relatively large, the proposed method guarantees feasible solutions when initializing metaheuristics. However, it is not possible to advocate a generalized percentage for the area of the shop floor relative to the total area of the machines since this really depends on the problem at hand. PSO is implemented and two comparisons with available benchmark problems are made to validate and prove this feature of the presented hybridized math heuristic metaheuristic approach; the results show the benefits of using the developed algorithm.

### ***Bibliography***

1. Tompkins, J.A., J.A. White, Y.A. Bozer, and J.M.A. Tanchoco, *Facilities planning*. 2010: John Wiley & Sons.

2. Chwif, L., M.R.P. Barretto, and L.A. Moscato, *A solution to the facility layout problem using simulated annealing*. Computers in Industry, 1998. 36(1-2): p. 125-132.
3. Kusiak, A. and S.S. Heragu, *The facility layout problem*. European Journal of operational research, 1987. 29(3): p. 229-251.
4. Koopmans, T.C. and M. Beckmann, *Assignment problems and the location of economic activities*. Econometrica: journal of the Econometric Society, 1957: p. 53-76.
5. Drira, A., H. Pierreval, and S. Hajri-Gabouj, *Facility layout problems: A survey*. Annual Reviews in Control, 2007. 31(2): p. 255-267.
6. Lawler, E.L., *The quadratic assignment problem*. Management science, 1963. 9(4): p. 586-599.
7. Bazaraa, M.S. and H.D. Sherali, *Benders' partitioning scheme applied to a new formulation of the quadratic assignment problem*. Naval Research Logistics Quarterly, 1980. 27(1): p. 29-41.
8. Foulds, L. and D.F. Robinson, *Graph theoretic heuristics for the plant layout problem*. The International Journal of Production Research, 1978. 16(1): p. 27-37.
9. Bazaraa, M.S., *Computerized layout design: a branch and bound approach*. AIIE transactions, 1975. 7(4): p. 432-438.
10. Sahni, S. and T. Gonzalez, *P-complete approximation problems*. Journal of the ACM (JACM), 1976. 23(3): p. 555-565.
11. Armour, G.C. and E.S. Buffa, *A heuristic algorithm and simulation approach to relative location of facilities*. Management Science, 1963. 9(2): p. 294-309.
12. Muther, R., *Systematic Layout Planning/by Richard Muther*. 1973.
13. Bhowmik, R., *An approach to the facility layout design optimization*. International Journal of Computer Science and Network Security, 2008. 8(4): p. 212-220.

14. Park, H. and Y. Seo, *An efficient algorithm for unequal area facilities layout planning with input and output points*. INFOR: Information Systems and Operational Research, 2019. 57(1): p. 56-74.
15. Salimpour, S., S.C. Viaux, A. Azab, and M.F. Baki, *A Clustering-Sequencing Approach for the Facility Layout Problem*. In Proceedings of the Seventh International Forum on Decision Sciences, 2020 (pp. 135-142). Springer, Singapore.
16. Taghavi, A. and A. Murat, *A heuristic procedure for the integrated facility layout design and flow assignment problem*. Computers & Industrial Engineering, 2011. 61(1): p. 55-63.
17. Cooper, L., *Location-allocation problems*. Operations research, 1963. 11(3): p. 331-343.
18. Karray, F., E. Zanelidin, T. Hegazy, A. Shabeeb, and E. Elbeltagi, *Computational intelligence tools for solving the facilities layout planning problem*. in *American Control Conference, 2000. Proceedings of the 2000*. 2000.
19. Paes, F.G., A.A. Pessoa, and T. Vidal, *A hybrid genetic algorithm with decomposition phases for the unequal area facility layout problem*. European Journal of Operational Research, 2017. 256(3): p. 742-756.
20. Palomo-Romero, J.M., L. Salas-Morera, and L. García-Hernández, *An island model genetic algorithm for unequal area facility layout problems*. Expert Systems with Applications, 2017. 68: p. 151-162.
21. Ulutas, B.H. and S. Kulturel-Konak, *An artificial immune system based algorithm to solve unequal area facility layout problem*. Expert Systems with Applications, 2012. 39(5): p. 5384-5395.
22. Wang, M.J., M.H. Hu, and M.Y. Ku, *A solution to the unequal area facilities layout problem by genetic algorithm*. Computers in Industry, 2005. 56(2): p. 207-220.



23. Allahyari, M.Z. and A. Azab, *Mathematical modeling and multi-start search simulated annealing for unequal-area facility layout problem*. Expert Systems with Applications, 2018. 91: p. 46-62.
24. Nordin, N.N., Z.M. Zainuddin, S. Salim, and R.R. Ponnusamy, *Mathematical modeling and hybrid heuristic for unequal size facility layout problem*. Malaysian Journal of Fundamental and Applied Sciences, 2009. 5(1).
25. Asl, A.D. and K.Y. Wong, *Solving unequal-area static and dynamic facility layout problems using modified particle swarm optimization*. Journal of Intelligent Manufacturing, 2017. 28(6): p. 1317-1336.
26. Kulturel-Konak, S. and A. Konak, *A new relaxed flexible bay structure representation and particle swarm optimization for the unequal area facility layout problem*. Engineering Optimization, 2011. 43(12): p. 1263-1287.
27. Liu, J., H. Zhang, K. He, and S. Jiang, *Multi-objective particle swarm optimization algorithm based on objective space division for the unequal-area facility layout problem*. Expert Systems with Applications, 2018. 102: p. 179-192.
28. Zhang, H. and J.Y. Wang, *Particle swarm optimization for construction site unequal-area layout*. Journal of construction engineering and management, 2008. 134(9): p. 739-748.
29. Caserta, M. and S. Voß, *A math-heuristic Dantzig-Wolfe algorithm for capacitated lot sizing*. Annals of Mathematics and Artificial Intelligence, 2013. 69(2): p. 207-224.
30. Kennedy, J. and R. Eberhart. *Particle swarm optimization (PSO)*. in *Proc. IEEE International Conference on Neural Networks, Perth, Australia*. 1995.
31. Parsopoulos, K.E., D.K. Tasoulis, and M.N. Vrahatis. *Multiobjective optimization using parallel vector evaluated particle swarm optimization*. in *Proceedings of the IASTED international conference on artificial intelligence and applications*. 2004. Acta Press.

32. Trelea, I.C., *The particle swarm optimization algorithm: convergence analysis and parameter selection*. Information processing letters, 2003. 85(6): p. 317-325.
33. Shi, Y. and R. Eberhart. *A modified particle swarm optimizer*. in *1998 IEEE international conference on evolutionary computation proceedings. IEEE world congress on computational intelligence (Cat. No. 98TH8360)*. 1998. IEEE.
34. Eberhart, R.C. and Y. Shi. *Comparison between genetic algorithms and particle swarm optimization*. in *International conference on evolutionary programming*. 1998. Springer.
35. Paul, R.C., P. Asokan, and V. Prabhakar, *A solution to the facility layout problem having passages and inner structure walls using particle swarm optimization*. The International Journal of Advanced Manufacturing Technology, 2006. 29(7-8): p. 766-771.
36. Parsopoulos, K. and M. Vrahatis, *Particle swarm optimization for imprecise problems*, in *Scattering and Biomedical Engineering: Modeling and Applications*. 2002, World Scientific. p. 254-264.
37. Marini, F. and B. Walczak, *Particle swarm optimization (PSO). A tutorial*. Chemometrics and Intelligent Laboratory Systems, 2015. 149: p. 153-165.
38. Heragu, S.S. and A.S. Alfa, *Experimental analysis of simulated annealing based algorithms for the layout problem*. European Journal of Operational Research, 1992. 57(2): p. 190-202.
39. Karray, F., et al. *Computational intelligence tools for solving the facilities layout planning problem*. in *Proceedings of the 2000 American Control Conference. ACC (IEEE Cat. No. 00CH36334)*. 2000. IEEE.
40. Imam, M.H. and M. Mir, *Automated layout of facilities of unequal areas*. Computers & industrial engineering, 1993. 24(3): p. 355-366.
41. Gonçalves, J.F. and M.G. Resende, *A biased random-key genetic algorithm for the unequal area facility layout problem*. European Journal of Operational Research, 2015. 246(1): p. 86-107.

42. Dunker, T., G. Radons, and E. Westkämper, *A coevolutionary algorithm for a facility layout problem*. International Journal of Production Research, 2003. 41(15): p. 3479-3500.
43. Imam, M. and M. Mir, *Nonlinear programming approach to automated topology optimization*. Computer-aided design, 1989. 21(2): p. 107-115.
44. Tam, K.Y. and S.G. Li, *A hierarchical approach to the facility layout problem*. The International Journal of Production Research, 1991. 29(1): p. 165-184.

## CHAPTER 3

### A DYNAMIC PROGRAMMING APPROACH TO SOLVE THE FACILITY LAYOUT PROBLEM FOR RECONFIGURABLE MANUFACTURING

Since in today's competitive and volatile manufacturing environment, the products' life cycles are short and the product mix and demand changes constantly, the layout needs to be designed in such a way that these changes are considered or can be able to be reconfigured. Reconfigurable manufacturing systems (RMS) which are characterized by being rapid and cost-effective in response to market changes, are a good alternative to cope with such events. From the layout point of view, in an RMS, the layout of facilities needs to be changeable and able to be redesigned easily. Dynamic facility layout problem (DFLP) is a good approach to develop layouts that are capable to be changed and redesigned. Dynamic programming (DP) has been known as one of the effective methods to deal with DFLP. To optimize DFLP by DP, the set of possible layouts for every single period which is called the state-space is given to DP and the best multi-period layout is found. Since the number of possible layouts increases rapidly with the increase in the number of facilities, considering all these layouts encounters two major difficulties; memory requirements and computer time requirements. This chapter proposes a method that has two main phases. In the first phase, the set of layouts to be considered in each period are determined using a heuristic approach. These layouts are the states in the DP approach where the periods constituted the decomposition stages. The recursive formulation of DP is solved in the second phase using a hybridized metaheuristic approach. The proposed approach restricts the DP to a good subset of the state-space. A genetic algorithm is applied to search for the best subset of layouts where each chromosome represents one subset of layouts. This subset is given to DP to be solved and the result is considered as the fitness of the chromosome. By the evolution of the chromosomes, the best subset of layouts that leads to the best multi-period layout plan is found. Finally, the resulted layout plan is improved by a tabu search metaheuristic. The proposed approach is evaluated against DP benchmarks in the literature. Computational results show that the proposed approach is able to provide more efficient solutions, especially for large-sized problems.

### *Nomenclature*

$i, j$	Indices for the machines $i, j \in \{1, 2, \dots, N\}$
$k, l$	Indices for the locations $k, l \in \{1, 2, \dots, N\}$
$t$	Index for the periods $t \in \{1, 2, \dots, T\}$
$F_{tij}$	Flow of material between machines $i$ and $j$ in period $t$
$D_{kl}$	Rectilinear distance between the centroids of locations $k$ and $l$
$C_{tij}^M$	Material handling cost per unit distance from machine $i$ to machine $j$ in period $t$
$C_{tikl}^R$	Rearrangement cost of machine $i$ from location $k$ to location $l$ at the beginning of period $t$
$X_{tik}$	$\begin{cases} 1 & \text{if machine } i \text{ is at location } k \text{ in period } t \\ 0 & \text{otherwise} \end{cases}$
$Y_{tikl}$	$\begin{cases} 1 & \text{if machine } i \text{ is rearranged from location } k \text{ to } l \text{ at beginning of period } t \\ 0 & \text{otherwise} \end{cases}$
$TC$	Total material handling and rearrangement cost
$L$	Set of all layouts to be considered in each period
$Z_{t,m}$	Material handling costs for layout $L_m$ in period $t$
$C_{k,m}$	Rearrangement cost from layout $L_k$ to layout $L_m$
$U_{t,m}^*$	Minimum total material handling and rearrangement costs for all periods up to period $t$ , where the layout $L_m$ is used in period $t$ (optimal-value function)
$U_{t-1,m}^*$	Minimum total material handling and rearrangement costs for all periods from after period $t - 1$ to the end, where the layout $L_m$ is used in period $t - 1$ (optimal-value function)

### ***3.1. Introduction***

Finding the most efficient placement of machines, cells, or departments in a facility is called the facility layout problem (FLP). The term to be used depends on the industry; for the purpose of this chapter, it is going to be referred to as machines. One of the most popular applications of FLP is in the manufacturing systems; the arrangement of production facilities (machines) on the shop floor. In this chapter, the facility layout design for reconfigurable manufacturing systems (RMS) is being tackled and discussed. RMS offers a customizable type of flexibility to reconfigure both capacity and functionality to adapt to the ongoing evolving demand. RMS machinery and material handling devices are reconfigurable to meet the new requirements for the new product mix. Thus, it must be able to handle configuration changes to be more robust, flexible, and reconfigurable while adapting to diverse production process requirements. One of the enablers for RMS is the layout of facilities [1].

Layout for RMS needs to be designed considering the variability in product demand and material flow between the machines. Most of the research in the area of the facility layout problem assume that the material flow between machines is constant over all planning horizons, which is called the static facility layout problem (SFLP). However, considering the fluctuations in demand, this is not always the case and the layout needs to be changed from one period to the other to adapt to the changes in product demand. Otherwise, there is an increase in material handling costs. Nevertheless, there is also a rearrangement cost related to changing the layout at the beginning of each period (e.g., the equipment and labor cost for moving the machines and also the loss of production cost). Dynamic facility layout problem (DFLP) is all about finding a layout for each period in a planning horizon to minimize the total material handling costs, for all periods and the total rearrangement costs between the periods [2]. FLP, and hence the DFLP are NP-hard, a class of problems for which there is no polynomial-time algorithm [3]. Hence, there is a need to use heuristic or metaheuristic methods to solve large instances of the problem.

Hitchings [4] is the first who recognized the need for change in the layout of facilities in a continuous review and reports on the dynamic nature of the facility layout problem. The author suggests that when the rearrangement cost is less than the incurred savings resulting

from the change due to higher efficiency, a change in a layout system is appropriate. Rosenblatt [5] proposes a dynamic programming (DP) formulation to solve the deterministic multi-period plant layout problem in an exact or heuristic manner. In his model, each layout to be considered in each period is a state and the stages are the periods. The number of states to be considered for each period and the solution method used to solve SFLP determines whether the procedure is optimal or heuristic. Enumerating all possible layouts for each period will result in an optimum solution, but it is computationally prohibitive for large instance problems.

Hence, Rosenblatt [5] suggests a simplifying procedure based on Sweeney and Tatham [6] theorem. In this procedure, a branch and cut algorithm is used to generate a number of best layouts for each period. The author's computational results show that if the optimal solution is desired, there is not a large number of layouts (states) that can be ignored for the DP procedure. Therefore, to solve a DFLP practically with DP, only a subset of all possible layouts should be selected by a heuristic procedure and as a result, two heuristic approaches were developed. In the first approach, which is similar to the heuristic procedure proposed by Ballou [7] for the warehouse location problem, the SFLP is solved optimally for each period and the set of all the best layouts of all periods is considered as the states of the DP. The second approach is appropriate when it is not computationally feasible to solve SFLP optimally. In such cases, either the SFLP can be solved using heuristic procedures such as CRAFT [8] or different layouts can be generated randomly. As the number of random layouts generated increases, the accuracy will also increase.

Lacksonen and Ensore [9] propose a heuristic procedure to find the set of layouts to be considered as states of the DP method proposed by Rosenblatt [5] for solving the DFLP. Urban [10] presents a steepest-decent pairwise-interchange heuristic procedure for solving DFLP. This method is similar to the CRAFT method yet different by including the concept of forecast window and rearrangement cost for a DFLP. The forecast window is the number of successive periods whose flow data is aggregated to generate the best layout of each period, from a given initial layout, using the steepest-decent pairwise-interchange (CRAFT) method. The forecast window starts from one and is increased to the total number of time periods. Urban [10] assumes that there are two types of rearrangement costs in this

problem: one is variable that is incurred when a particular machine is relocated in a period and the other is the fixed rearrangement cost that has to be considered if there are any rearrangements in a period. The initial layout of the first period is generated randomly but for the rest of the periods, the initial layout of a period is the final layout of its previous period. The process of interchanging machines continuous until all the possibilities are evaluated. For each interchange, the difference between the material handling costs of the initial layout after doing the exchange and the initial layout is calculated and subtracted from the sum of variable rearrangement costs of the exchanged machines. The exchange that has the maximum improvement is then selected and the amount of improvement is compared with the fixed rearrangement cost. If the improvement is more, the exchange will be accepted and the new layout is generated for the current period otherwise the layout of the previous period will be used for the current period as well. When for each forecast window, a layout plan for DFLP is generated, as explained, the sum of the material handling and rearrangement costs is calculated. Finally, the best layout plan with the minimum total cost is chosen as the solution to the DFLP problem. This model seems to be suitable for DP under rolling planning horizon as it uses different forecast windows.

Urban's model [10] is modified and improved by Balakrishnan et al. [11] using two heuristic procedures. The first heuristic improves the layout plan of each forecast window by using a backward-pass pairwise-exchange procedure. The exchange starts at the penultimate period and moves backward until the first period. In each period, if a pairwise-exchange makes an improvement, the exchange is accepted and results in a new layout for that period. This improvement is measured based on both material handling and rearrangement costs. Having the forecast window in the backward pass equal to one signifies that for each period only product flow of its own is considered to calculate the material handling cost. The rearrangement cost is the cost of shifting the exchanged machines from their locations in the previous period's layout to their new locations in the current period and those of moving them again to their locations in the next period's layout. But in the forward path exchange procedure, the rearrangement cost is only the cost of shifting the exchanged machines from their locations in the previous period's layout to their new locations in the generated layout for the current period as that for the next periods are not known. The second heuristic improves the model by combining it with DP. The set



of all layouts generated for the entire forecast windows in the forward pass pairwise-exchange procedure are used as the states of the DP. The solution of these two heuristic procedures introduced by Balakrishnan et al. [11] is at least as good as the Urban [10] since the solution of the forward pass pairwise-exchange method is embedded in them.

Balakrishnan et al. [12] use a hybrid genetic algorithm (GA) to solve DFLP. They propose a crossover operator that is based on DP. In their model, each gene is represented by one single string consisting of layouts in each period from period one to the end. The crossover operator engages many strings and crossover points. The crossover points are all the places that two consecutive periods of each string meet, for all the strings. The result of doing the crossover cuts on strings is that many feasible static layouts will become available to be used as the states of DP approach. DP finds the best combination of those layouts for the DFLP and hence, generates a new string as an offspring. The proposed crossover operator always generates feasible solutions. As for mutations, the authors choose one period randomly and apply CRAFT to the static layout of that period. The mutant replaces the original layout.

Erel et al. [13, 14] propose a different heuristic procedure for generating a subset of all possible layouts to have as the states of DP to solve the DFLP. They consider viable layouts that are defined as the layouts that are more likely to be in the optimal DFLP solution. These layouts have the lowest material handling cost solving SFLP for the flow data of a single period or the combination of the flow data from two or more consecutive periods using a weighting scheme. After solving the DP, the solution of DFLP is improved by applying a simulated annealing procedure.

Dunker et al. [15] combine DP and genetic search to solve the DFLP with unequal-area departments. In their proposed method, a GA is used to generate a population for each period where each individual in the generated population denotes a specific layout. The fitness of a layout for a particular period is evaluated considering the previous and next periods by DP. Udomsakdigool and Bangsaranthip [16] utilize the ant colony metaheuristic procedure to obtain the best-ranked solutions for each period in order to restrict the state-space of DP when solving DFLP.

In this chapter, a hybrid algorithm based on DP is proposed to find the optimum/near optimum multi-period facility layout plan in an RMS. This approach has two main phases in alignment with the previous work to solve DFLP using DP [5, 13, 16]. A heuristic approach is used in the first phase to generate a large set of good layouts that can be used as the state-space of DP. Then, a hybrid genetic algorithm-dynamic programming approach is developed that feeds DP with the best subset of the generated layouts. The obtained layout plan is further improved by a tabu search metaheuristic.

This chapter is organized as follows; the methodology starts in Section 3.2 where the QAP formulation for DFLP is presented. Section 3.3 proposes a heuristic procedure, for determining the state-space of each period, and a hybrid genetic algorithm-dynamic programming approach, which is called the improved DP (IDP) algorithm, to solve the problem. The IDP algorithm is then additionally improved using a tabu search metaheuristic (IDP-TS algorithm). The methods presented are validated and compared against a well-known benchmark test set in Section 3.4. Finally, the conclusions and future work are presented in Section 3.5.

### 3.2. Problem Formulation

Considering the layout to be discrete with equal-sized machines, the DFLP can be formulated as modified QAP [12, 16-20]:

$$\begin{aligned}
 (P) \text{ Min } TC = & \sum_{t=2}^T \sum_{i=1}^N \sum_{k=1}^N \sum_{l=1}^N C_{tikl}^R Y_{tikl} \\
 & + \sum_{t=1}^T \sum_{i=1}^N \sum_{j=1}^N \sum_{k=1}^N \sum_{l=1}^N F_{tij} C_{tij}^M D_{kl} X_{tik} X_{tjl}
 \end{aligned} \tag{3.1}$$

Subject to:

$$\sum_{k=1}^N X_{tik} = 1 \quad \forall i, t \tag{3.2}$$

$$\sum_{i=1}^N X_{tik} = 1 \quad \forall k, t \tag{3.3}$$

$$Y_{tikl} = X_{(t-1)ik}X_{til} \quad \forall i, k, l \text{ \& } t > 1 \quad (3.4)$$

$$X_{tik}, Y_{tikl} \in \{0,1\} \quad \forall i, k, l, t \quad (3.5)$$

The objective function (3.1) minimizes the total material handling and rearrangement costs during the planning horizon. Constraint (3.2) ensures that each machine is assigned to one location in each period. Constraint (3.3) ensures that each location has one machine assigned to it in each period. Constraint (3.4) states that the 0–1 machine rearrangement variable,  $Y_{tikl}$ , takes on a value of 1 only if a machine is rearranged from one location to another in two consecutive periods. Constraint (3.5) specifies that the decision variables are binary.

### **3.3. Methodology**

Considering all possible layouts for each period in the DP procedure will ensure an optimum solution. However, for a discrete DFLP with  $N$  machines and  $T$  time periods, there are  $(N!)^T$  different possible solutions. Hence, for the large problems enumerating all of the possible solutions is computationally prohibitive, and researchers propose heuristic methods for selecting a set of layouts among all to consider as states of the DP [5, 13, 16].

#### **3.3.1. DP for DFLP**

In the following, the general procedure of DP to solve a discrete DFLP is described. It consists of two phases: In the first phase, a heuristic procedure is presented to determine a set of layouts to be considered as states of the DP, and in the second phase, the DP recursive formulation is solved.

##### *3.3.1.1. State-Space Determination*

To obtain these layouts, first, the multi-period problem (P) is decomposed to  $T$  single period FLPs. Then these SFLPs are solved, and the best layouts of each period are determined. For small-sized problems, a mathematical programming approach can be used to solve SFLP exact, but for larger-sized problems, because of its intractable nature, a heuristic or metaheuristic approach should be used.

If there is a rectangular layout like the example in Figure 3.1, then there exist three other layouts that are symmetric, matching the best layout in each period and having the same material handling cost. Considering all four symmetric layouts is trivial for discrete one period FLP. However, for DFLP, the rearrangement cost contributes to the total cost and carries a significant impact.

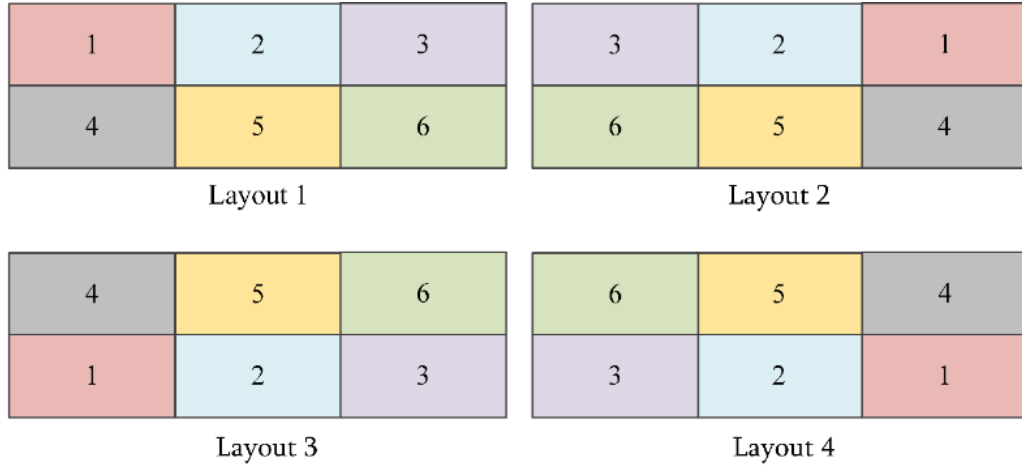


Figure 3.1. All four layouts are symmetric and have the same material handling cost

For each period, the best layout is obtained and its other three symmetric layouts are generated. It should be noted that for larger-sized problems that the exact optimum solution cannot be obtained, more than one near-optimum layout, depending on the size of the problem, is found. Then,  $l$  neighborhood solutions for each of the optimum or near optimum solutions and their symmetric layouts are generated in such a way that  $L$  good layouts are created in total for all the periods.

### 3.3.1.2. DP Recursive Formulation

In this phase, using the recursive formulations (3.6) and (3.7), the state-spaces at each stage are searched, and hence DFLP is solved. The recursive formulations for this DP approach, as presented by Rosenblatt [5] are as follows:

$$U_{t,m}^* = \min_k \{U_{t-1,k}^* + C_{k,m}\} + Z_{t,m} \quad \forall t, m \quad (3.6)$$

$$U_{0,k}^* = 0 \quad \forall k \quad (3.7)$$

This method of computation used in Equation (3.6) is forward induction. If the calculations start from the last period and move backward, one stage at a time, then is called backward induction. The recursive relationship for backward induction is:

$$U_{t-1,m}^* = \min_k \{U_{t,k}^* + C_{k,m}\} + Z_{t-1,m} \quad \forall t, m \quad (3.8)$$

$$U_{T,k}^* = 0 \quad \forall k \quad (3.9)$$

Although the nature of both calculations is the same, the interpretation of the optimal value function is changed.

In Figure 3.2, the stages represent the time periods, while the nodes at each stage represent the material handling cost for that particular layout in the stage. The arcs between the nodes in two consecutive stages represent the rearrangement cost from one layout to another.

Figure 3.2 is adapted from the constrained shortest-path problem proposed by Balakrishnan et al. [21]. They show that if all possible layouts are considered, the discrete DFLP can be modeled by the shortest-path problem. In their model, the arcs represent the total material handling and rearrangement costs. The nodes in each period are the static layouts. The number of nodes (layouts) considered at each period is constant across periods. By using dynamic or network programming, this SP can be solved exact [13].

As depicted in Figure 3.2, there are two situations: a. There is no initial layout at the beginning of the planning horizon. In this situation, Equation (3.6) for  $t = 1$  becomes:

$$U_{1,m}^* = Z_{1,m} \quad \forall m \quad (3.10)$$

b. There exists an initial layout in period 0. In this situation, Equation (3.6) for  $t = 1$  becomes:

$$U_{1,m}^* = C_{0,m} + Z_{1,m} \quad \forall m \quad (3.11)$$

In the literature, the state-space considered in each stage of DP is identical [5, 13, 16]. In this chapter, a heuristic approach is proposed to find the state-space of the problem.

However, not all of these layouts are going to be given to DP, and GA is used to select the best layouts from this state-space to be used by DP.

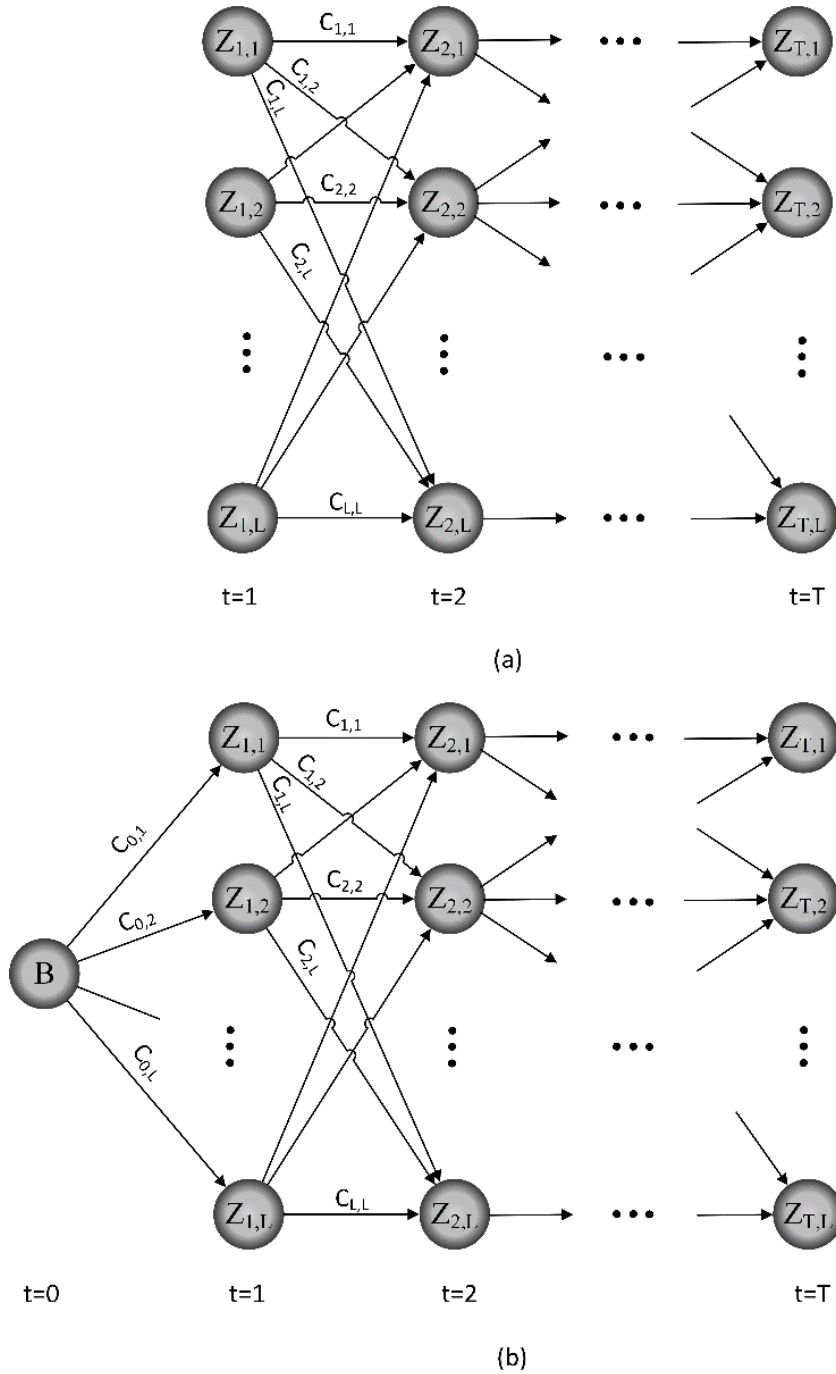


Figure 3.2. The dynamic facility layout problem adapted from [21]. a. There is no initial layout to begin with (starting from scratch) b. There is an initial layout at the beginning (period 0)

### 3.3.2. Improved DP Algorithms (IDP and IDP-TS)

The main challenge of DP is computation time and the required memory that increases rapidly with the increment in the size of state-space. Because of this limitation, a restricted number of layouts is fed to DP, but that may lead to ignoring some good layouts, and so optimality is lost. Therefore, to preserve the optimality, the initial layouts should be intelligently given to DP.

In this section, an approach that can select the most promising initial layouts that lead to the best solution of DP for DFLP is proposed. GAs are population-based optimization algorithms that are proved to be efficient for solving FLPs [12, 15, 18, 22]. In this approach, a GA is used to solve the problem and find the best initial layouts for DP. This algorithm is called IDP.

In IDP, first, according to the method explained in Section 3.3.1.1, for each single period FLP a large set of good layouts,  $L$ , is generated. The layouts of each period are partitioned into  $G$  groups. These groups of layouts are considered as the initial input of the GA. Then,  $N_{pop}$  number of chromosomes are generated as the initial population called  $pop$ . Each chromosome determines the amount of contribution of each group of a period to create the state-space of DP. The state-space corresponding to each chromosome is given to DP; DP is executed, and the result is considered as the fitness function of the chromosome. Then better chromosomes in terms of fitness function value are selected as parents, and GA operators such as crossover and mutation are applied on  $pc$  and  $pm$  percentage of the population to generate new populations of chromosomes,  $popc$  and  $popm$ , respectively. The generated populations are merged with the original population,  $pop = pop \cup popc \cup popm$ , which results in having a  $pop$  with more than  $N_{pop}$  chromosomes. Hence, the  $pop$  is sorted in ascending order and the extra chromosomes are truncated and got rid of,  $pop = pop(1:N_{pop})$ . The process will be continued until the stopping criteria is satisfied which here is reaching the maximum number of GA iterations,  $MaxIt$ .

For further improvement, a tabu search (TS) is used to improve the IDP. This algorithm is called IDP-TS. The best multi-period layout plan obtained by GA,  $BestSol$ , is given to TS as the initial solution,  $Sol_{TS} = BestSol$ , and in each iteration TS performs a neighborhood

search on the current solution. The actions that can be applied on a solution to generate a neighborhood solution are stored in an *ActionList* with the size of  $N_{action}$ . All the actions except the ones that are in the tabu list are being tried and the one that results in a better solution is accepted. Then the current solution is updated, that corresponding action becomes a tabu action and is being stored and remained in the tabu list for *TL* (tabu list length) number of times. TS accepts non-improving moves and to prevent being trapped in local optima. The process will be continued until the termination criteria is met which here is also reaching the maximum number of TS iterations, *MaxSubIt*. The flowchart of the proposed algorithm is shown in Figure 3.3.



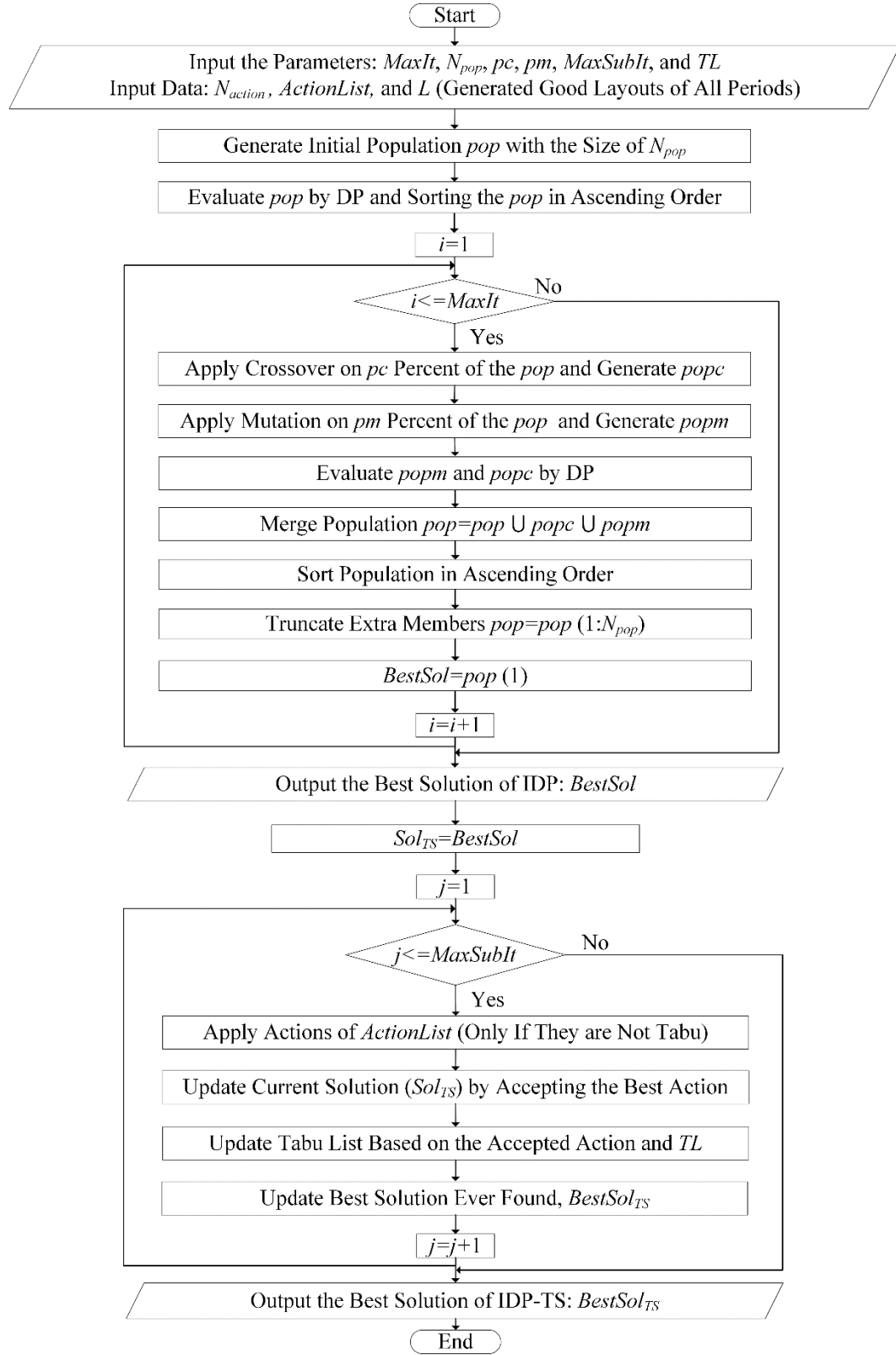


Figure 3.3. The flowchart of the proposed IDP algorithms

### 3.3.2.1. Chromosome Representation

The proposed chromosome is concerned with determining the state-space of DP. It is shown by a  $T \times G$  matrix in which  $T$  is the number of time periods and  $G$  is the number of groups. The value in each gene shows the percentage of layouts of that group that should participate in DP. It is generated in such a way that the sum of all the genes is equal to one, and so the feasibility of the chromosomes is satisfied. For example, Figure 3.4 illustrates the form of a chromosome in six periods of time. The generated layouts for each have been partitioned into five groups. According to this chromosome, 0.10 of the DP state-space are the layouts that belong to group one of period 1.

	Group 1	Group 2	Group 3	Group 4	Group 5
Period 1	0.1	0.03	0.04	0.02	0.02
Period 2	0.06	0.02	0.04	0.01	0.01
Period 3	0.13	0.04	0.03	0.03	0.01
Period 4	0.08	0.02	0.04	0.02	0.01
Period 5	0.05	0.02	0.03	0.01	0.02
Period 6	0.06	0.02	0.01	0.01	0.01

Figure 3.4. The structure of chromosome

### 3.3.2.2. Roulette Wheel Selection

Roulette wheel selection strategy is the main selection strategy used in GAs. In this selection strategy, the probability that a chromosome is being selected is proportional to its fitness in such a way that better chromosomes have a higher chance to be selected and survive.

### 3.3.2.3. Crossover Operator

The crossover is designed to enhance the quality of the population in each generation. In the proposed algorithm, a uniform crossover operator is designed that is able to preserve the feasibility of the chromosomes; two parents are selected using a roulette wheel selection strategy, and the value of each offspring gene is set equal to the average value of

corresponding genes in parents. An example of a crossover operator is illustrated in Figure 3.5.

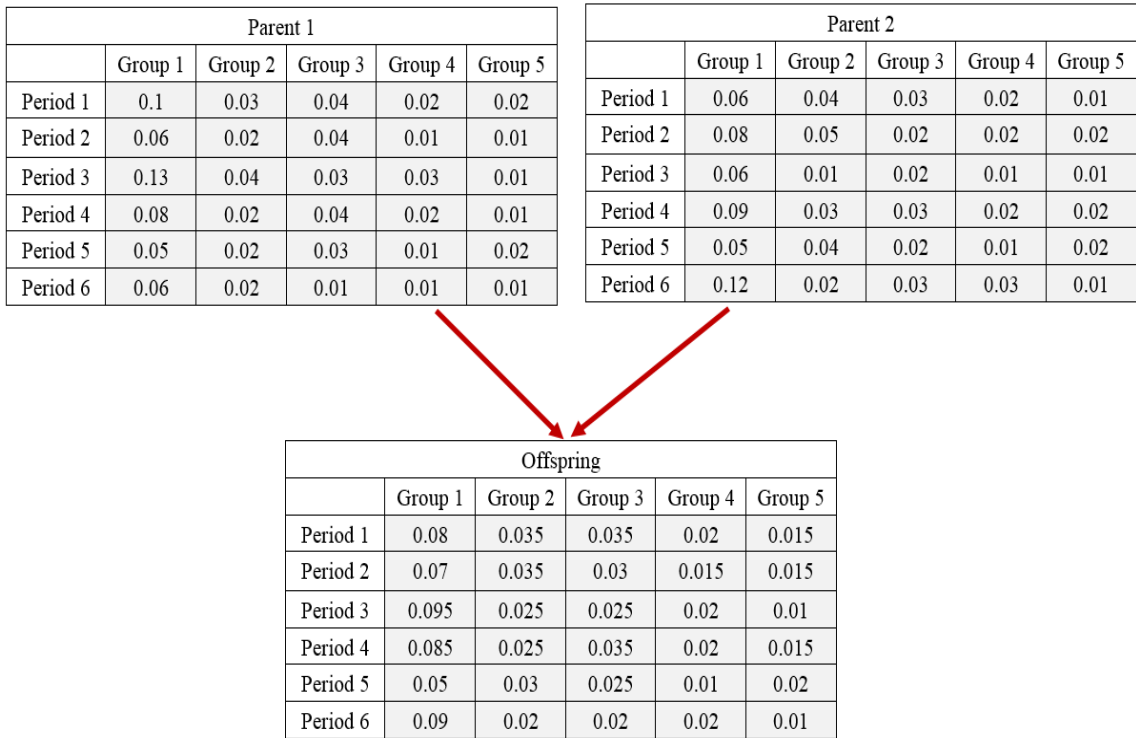



Figure 3.5. An example of crossover operator

#### 3.3.2.4. Mutation Operator

After crossover, the chromosomes are subjected to mutation. The mutation operator prevents premature convergence to local optima and maintains the diversity in the population. Therefore, it should be designed in such a way that it creates a dramatic change in the chromosome. To this aim, in each row, all the genes are inversely rearranged (Figure 3.6).

Parent					
	Group 1	Group 2	Group 3	Group 4	Group 5
Period 1	0.1	0.03	0.04	0.02	0.02
Period 2	0.06	0.02	0.04	0.01	0.01
Period 3	0.13	0.04	0.03	0.03	0.01
Period 4	0.08	0.02	0.04	0.02	0.01
Period 5	0.05	0.02	0.03	0.01	0.02
Period 6	0.06	0.02	0.01	0.01	0.01



Muted offspring					
	Group 1	Group 2	Group 3	Group 4	Group 5
Period 1	0.02	0.02	0.04	0.03	0.1
Period 2	0.01	0.01	0.04	0.02	0.06
Period 3	0.01	0.03	0.03	0.04	0.13
Period 4	0.01	0.02	0.04	0.02	0.08
Period 5	0.02	0.01	0.03	0.02	0.05
Period 6	0.01	0.01	0.01	0.02	0.06

Figure 3.6. An example of mutation operator

### 3.4. Computational Results

In this section, the effectiveness of two algorithms: 1. DP improved by being hybridized with GA (IDP) and 2. DP improved by being hybridized with GA and TS (IDP-TS) are evaluated. To this aim, a data set of Balakrishnan and Cheng [22] is used. This data set consists of 48 test instances with 6, 15, and 30 facilities, each with 5 and 10 periods. The proposed method is compared against the following DP benchmarks for discrete DFLP problem:

- GADP (R) and GADP (U) [12]
- DP\_10L, DP\_10LI, DP\_5L, DP\_5LI, DP\_10S, DP\_10SI, DP\_5S, DP\_5SI, SA\_EG\_1 and SA\_EG\_2 [13, 14]

The details of each method can be found in Section 3.1. Notably, the correct results obtained by Erel et al. [13] are used in this comparison. These are not published in their

erratum [14] but by Rodríguez et al. [23] who published them for the 15 and 30 facilities instances.

The proposed algorithms are programmed in Matlab (release 2018a) and are tested on a 64-bit architecture with an Intel Xeon processor, the clock speed of which is 3.07 GHz, and a 6 GB of memory personal computer. The parameter tuning is done using Design Expert 7.

### 3.4.1. Calibration of Parameters

Tuning the parameters of metaheuristics will ensure higher quality solutions that can be done using experimental design. To this aim, all the parameters of the proposed algorithms are tuned using the response surface method. Preliminary experiments are executed to determine the levels of the parameters of each algorithm which are presented in Table 3.1. Subsequently, experiments are designed and executed. Lastly, to obtain the relationship between the objective function value and the parameters of the algorithms, a stepwise regression method with  $\alpha = 0.1$  is used and the quadratic Equations (3.13) and (3.14) are developed.

It should be noted that as explained in Section 3.3.2, TS has two parameters; *MaxSubIt* and *TL* which  $TL = (\beta \times N_{action})$ . Since  $N_{action}$  is the input data of the problem, only  $\beta$  needs to be tuned.

Table 3.1. The parameters of the IDP and IDP-TS algorithms and their optimum values

Algorithm	Parameter	Description	Low level (-1)	High level (+1)	Optimum
IDP	$N_{pop}$	Initial pop size	30	70	<b>54</b>
	$pc$	Percent of crossover	0.4	0.8	<b>0.79</b>
	$pm$	Percent of mutation	0.2	0.4	<b>0.35</b>
	<i>MaxIt</i>	Number of iteration	50	150	<b>149</b>
IDP-TS	$N_{pop}$	Initial pop size	30	70	<b>69</b>
	$pc$	Percent of crossover	0.4	0.8	<b>0.4</b>
	$pm$	Percent of mutation	0.2	0.4	<b>0.2</b>
	<i>MaxIt</i>	Number of iteration	50	150	<b>150</b>
	$\beta$	TL Coefficient	0.2	0.7	<b>0.69</b>
	<i>MaxSubIt</i>	Number of iteration	800	1000	<b>999</b>

$$\begin{aligned}
OF_{IDP} = & 1174143.886048 - 792.04062500004 \times Npop & (3.13) \\
& - 133758.55902778 \times pc - 199167.07070708 \times pm \\
& - 333.43618055557 \times MaxIt + 460.95312500003 \\
& \times Npop \times pc + 1.6594375000001 \times Npop \times MaxIt \\
& + 154.68125000001 \times pc \times MaxIt + 247419.1919192 \\
& \times pm^2
\end{aligned}$$

$$\begin{aligned}
OF_{IDP-TS} = & 1016860 - 153.83049 \times Npop - 6512.27273 \times pc & (3.14) \\
& - 7051.64773 \times pm - 159.37905 \times MaxIt - 4495.15152 \\
& \times \beta - 10.07152 \times MaxSubIt + 0.245812 \times Npop \\
& \times MaxIt + 50.55 \times pc \times MaxIt + 69.59375 \times pm \times MaxIt
\end{aligned}$$

By optimizing the Equations (3.13) and (3.14), the optimum values for the parameters of the algorithms are obtained which are bolded in Table 3.1.

### 3.4.2. Evaluating the Proposed Algorithms

The comparative results of all the algorithms are given in Tables 3.2-3.7. In these tables, the best solution value obtained for each test instance is bolded. In Tables 3.2 and 3.3, the results are shown for the test instances with five facilities. For the test instances with five periods (Instance 1– Instance 8), SA\_EG\_2 and IDP-TS obtain the optimal solutions for all eight test instances, and so these algorithms are the preferred choices for this set of eight instances. However, for the test instances with ten periods (Instance 9– Instance 16), IDP and IDP-TS are the only algorithms that find the best solutions for all instances (0.0992% and 0.0180% improvement for instances 12 and 13, respectively). Therefore, it can be said that for test problems with five facilities, IDP-TS performs slightly better than all other algorithms.

In Tables 3.4 and 3.5, the results obtained for the instances consisting of 15 facilities can be seen. Clearly, the two algorithms SA\_EG\_1 and IDP-TS outperform all the other algorithms. SA\_EG\_1 algorithm obtains the best results for ten instances, and IDP-TS finds the best solutions for five instances (0.1245%, 0.2555%, 0.1365%, 0.0972%, and 0.1228% improvement for instances 17, 21, 25, 28, and 32, respectively).

The results for the test instances with 30 facilities are presented in Tables 3.6 and 3.7. As shown in these tables, the proposed IDP-TS algorithm finds the best solutions for six out

of eight instances with five periods (0.2564%, 0.1604%, 0.1942%, 0.1304%, 0.2576%, and 0.2815% improvement for instances 33, 35, 36, 37, 38, and 40 respectively) and achieves the best results for six out of eight instances with ten periods (0.1590%, 0.1987%, 0.1474%, 0.0749%, 0.0930%, and 0.1379% improvement for instances 41-46, respectively). Hence, it can be concluded that IDP-TS outperforms other methods in solving large-sized problems. In summary, the proposed IDP-TS obtained the best solutions for 33 out of 48 instances. This is due to the intelligent GA-based procedure applied in the proposed IDP-TS algorithm that feeds the DP with the best possible set of layouts from the state-space.

Table 3.2. Computational results for instances with 6 facilities and 5 periods

Approach	Instance 1	Instance 2	Instance 3	Instance 4	Instance 5	Instance 6	Instance 7	Instance 8
<i>GADP</i>	<b>106,419</b>	<b>104,834</b>	104,529	106,583	<b>105,628</b>	104,315	106,447	<b>103,771</b>
<i>DP_10</i>	<b>106,419</b>	<b>104,834</b>	<b>104,320</b>	106,509	<b>105,628</b>	<b>103,985</b>	106,447	<b>103,771</b>
<i>DP_10I</i>	<b>106,419</b>	<b>104,834</b>	<b>104,320</b>	106,509	<b>105,628</b>	<b>103,985</b>	106,447	<b>103,771</b>
<i>DP_5</i>	<b>106,419</b>	<b>104,834</b>	<b>104,320</b>	106,885	105,737	104,053	106,447	104,185
<i>DP_5I</i>	<b>106,419</b>	<b>104,834</b>	<b>104,320</b>	106,515	105,737	104,053	106,447	104,185
<i>SA_EG_1</i>	<b>106,419</b>	<b>104,834</b>	104,520	<b>106,399</b>	105,737	<b>103,985</b>	<b>106,439</b>	<b>103,771</b>
<i>SA_EG_2</i>	<b>106,419</b>	<b>104,834</b>	<b>104,320</b>	<b>106,399</b>	<b>105,628</b>	<b>103,985</b>	<b>106,439</b>	<b>103,771</b>
<i>IDP</i>	<b>106,419</b>	<b>104,834</b>	<b>104,320</b>	106,509	<b>105,628</b>	104,987	<b>106,439</b>	<b>103,771</b>
<i>IDP-TS</i>	<b>106,419</b>	<b>104,834</b>	<b>104,320</b>	<b>106,399</b>	<b>105,628</b>	<b>103,985</b>	<b>106,439</b>	<b>103,771</b>

Table 3.3. Computational results for instances with 6 facilities and 10 periods

Approach	Instance 9	Instance 10	Instance 11	Instance 12	Instance 13	Instance 14	Instance 15	Instance 16
<i>GADP</i>	<b>214,313</b>	<b>212,134</b>	<b>207,987</b>	212,741	210,944	210,000	215,452	<b>212,588</b>
<i>DP_10</i>	<b>214,313</b>	<b>212,134</b>	<b>207,987</b>	212,741	211,022	<b>209,932</b>	<b>214,252</b>	<b>212,588</b>
<i>DP_10I</i>	<b>214,313</b>	<b>212,134</b>	<b>207,987</b>	212,741	211,022	<b>209,932</b>	<b>214,252</b>	<b>212,588</b>
<i>DP_5</i>	<b>214,313</b>	212,138	208,246	213,117	211,022	210,000	<b>214,252</b>	213,002
<i>DP_5I</i>	<b>214,313</b>	212,138	208,060	212,747	211,022	210,000	<b>214,252</b>	213,002
<i>SA_EG_1</i>	<b>214,313</b>	<b>212,134</b>	<b>207,987</b>	212,747	211,076	210,000	214,823	<b>212,588</b>
<i>SA_EG_2</i>	<b>214,313</b>	213,015	208,351	212,747	211,072	<b>209,932</b>	214,438	<b>212,588</b>
<i>IDP</i>	<b>214,313</b>	<b>212,134</b>	<b>207,987</b>	<b>212,530</b>	<b>210,906</b>	<b>209,932</b>	<b>214,252</b>	<b>212,588</b>
<i>IDP-TS</i>	<b>214,313</b>	<b>212,134</b>	<b>207,987</b>	<b>212,530</b>	<b>210,906</b>	<b>209,932</b>	<b>214,252</b>	<b>212,588</b>

Table 3.4. Computational results for instances with 15 facilities and 5 periods

Approach	Instance 17	Instance 18	Instance 19	Instance 20	Instance 21	Instance 22	Instance 23	Instance 24
<i>GADP (R)</i>	493,707	494,476	506,684	500,826	502,409	497,382	494,316	500,779
<i>GADP (U)</i>	484,090	485,352	489,898	484,625	489,885	488,640	489,378	500,779
<i>DP_10L</i>	484,054	489,322	491,310	487,884	491,617	490,205	490,544	494,994
<i>DP_10LI</i>	483,568	489,322	491,310	487,275	491,346	489,847	490,051	493,577
<i>DP_5L</i>	484,972	491,102	493,632	489,929	494,040	490,782	491,984	496,841
<i>DP_5LI</i>	482,123	488,840	493,632	489,480	494,040	490,782	490,251	496,672
<i>DP_10S</i>	484,369	487,274	491,790	487,956	491,178	490,305	490,161	494,954
<i>DP_10SI</i>	483,708	485,702	491,790	486,851	491,178	489,947	489,583	494,534
<i>DP_5S</i>	484,369	489,819	493,224	489,698	493,097	492,275	492,430	496,990
<i>DP_5SI</i>	483,708	488,382	492,597	489,698	491,738	492,202	489,155	496,473
<i>SA_EG_1</i>	481,738	<b>485,167</b>	<b>487,886</b>	<b>481,628</b>	489,304	<b>482,321</b>	<b>485,384</b>	<b>489,072</b>
<i>SA_EG_2</i>	481,792	488,592	492,536	485,862	489,946	488,452	487,576	493,030
<i>IDP</i>	483,526	485,483	491,695	486,747	491,117	490,748	489,882	493,688
<i>IDP-TS</i>	<b>481,138</b>	485,215	491,205	485,594	<b>488,054</b>	489,390	488,601	492,850

Table 3.5. Computational results for instances with 15 facilities and 10 periods

Approach	Instance 25	Instance 26	Instance 27	Instance 28	Instance 29	Instance 30	Instance 31	Instance 32
<i>GADP (R)</i>	1,004,806	1,006,790	1,012,482	1,001,795	1,005,988	1,002,871	1,019,645	1,010,772
<i>GADP (U)</i>	987,887	<b>980,638</b>	985,886	976,025	982,778	973,912	982,872	987,789
<i>DP_10L</i>	986,811	985,154	989,081	979,139	986,029	976,917	985,535	990,844
<i>DP_10LI</i>	984,344	984,779	988,635	976,456	983,846	974,436	982,790	990,372
<i>DP_5L</i>	991,093	987,453	993,799	983,208	989,680	979,297	992,897	992,962
<i>DP_5LI</i>	988,322	985,147	993,318	982,632	985,966	978,683	989,272	988,959
<i>DP_10S</i>	986,592	984,601	990,218	978,726	984,975	976,610	987,019	990,247
<i>DP_10SI</i>	983,070	983,826	990,153	977,548	983,053	975,290	986,325	988,584
<i>DP_5S</i>	995,319	988,396	992,824	982,270	987,963	981,406	992,807	993,902
<i>DP_5SI</i>	991,801	985,360	990,794	982,112	982,893	979,731	988,870	990,376
<i>SA_EG_1</i>	982,298	982,714	<b>985,364</b>	974,994	<b>975,498</b>	<b>968,323</b>	<b>977,410</b>	988,304
<i>SA_EG_2</i>	984,013	983,550	988,465	980,045	982,191	973,199	985,270	989,520
<i>IDP</i>	981,771	986,862	987,729	974,597	985,972	987,317	987,484	987,129
<i>IDP-TS</i>	<b>980,431</b>	982,651	986,734	<b>973,650</b>	985,972	987,070	986,486	<b>985,917</b>



Table 3.6. Computational results for instances with 30 facilities and 5 periods

Approach	Instance 33	Instance 34	Instance 35	Instance 36	Instance 37	Instance 38	Instance 39	Instance 40
<i>GADP (R)</i>	603,339	589,834	592,475	586,064	580,624	587,797	588,347	590,451
<i>GADP (U)</i>	578,689	572,232	578,527	572,057	559,777	566,792	<b>567,873</b>	575,720
<i>DP_10L</i>	581,805	574,657	581,030	571,730	561,079	568,047	572,262	575,445
<i>DP_10LI</i>	579,741	<b>569,482</b>	578,506	569,874	561,079	568,047	568,196	575,445
<i>DP_5L</i>	583,082	576,592	581,691	575,024	561,424	570,435	573,878	576,091
<i>DP_5LI</i>	581,942	571,563	580,549	574,070	561,424	570,435	571,254	576,091
<i>DP_10S</i>	581,805	575,004	581,170	571,749	561,078	568,554	572,706	574,813
<i>DP_10SI</i>	579,741	570,906	577,402	569,596	558,792	568,554	568,721	574,813
<i>DP_5S</i>	582,858	576,106	581,262	574,110	562,857	570,356	572,797	576,149
<i>DP_5SI</i>	581,369	572,511	580,186	569,723	562,857	570,356	569,145	576,149
<i>SA_EG_1</i>	579,570	573,965	580,102	572,139	563,503	574,805	573,361	581,614
<i>SA_EG_2</i>	583,227	574,116	577,787	573,446	565,735	570,905	571,499	581,966
<i>IDP</i>	579,808	585,744	578,675	570,828	558,109	567,158	577,851	574,778
<i>IDP-TS</i>	<b>577,205</b>	577,728	<b>576,476</b>	<b>568,490</b>	<b>557,381</b>	<b>565,332</b>	574,741	<b>573,160</b>

Table 3.7. Computational results for instances with 30 facilities and 10 periods

Approach	Instance 41	Instance 42	Instance 43	Instance 44	Instance 45	Instance 46	Instance 47	Instance 48
<i>GADP (R)</i>	1,194,084	1,199,001	1,197,253	1,184,422	1,179,673	1,178,091	1,186,145	1,208,436
<i>GADP (U)</i>	1,169,747	1,168,878	1,166,366	1,154,192	1,133,561	1,145,000	1,145,927	1,168,657
<i>DP_10L</i>	1,174,773	1,175,323	1,174,023	1,155,879	1,128,136	1,145,858	1,143,814	1,168,142
<i>DP_10LI</i>	1,171,853	1,169,138	1,168,720	1,150,265	1,128,013	1,145,858	<b>1,143,144</b>	1,167,900
<i>DP_5L</i>	1,180,120	1,179,022	1,175,920	1,157,918	1,131,518	1,147,517	1,147,016	1,170,929
<i>DP_5LI</i>	1,171,413	1,174,421	1,170,019	1,156,016	1,131,518	1,147,517	1,145,934	1,170,929
<i>DP_10S</i>	1,172,434	1,175,551	1,175,240	1,155,998	1,129,143	1,144,539	1,143,788	<b>1,165,994</b>
<i>DP_10SI</i>	1,171,178	1,170,092	1,165,525	1,153,981	1,128,784	1,144,092	1,143,183	<b>1,165,994</b>
<i>DP_5S</i>	1,181,743	1,177,212	1,176,997	1,158,507	1,132,926	1,149,893	1,147,041	1,171,658
<i>DP_5SI</i>	1,180,087	1,170,810	1,173,529	1,156,517	1,132,926	1,149,893	1,146,987	1,171,428
<i>SA_EG_1</i>	1,173,483	1,173,015	1,166,295	1,154,196	1,141,738	1,158,322	1,157,505	1,179,888
<i>SA_EG_2</i>	1,174,815	1,177,743	1,171,932	1,154,945	1,140,116	1,158,227	1,163,761	1,177,565
<i>IDP</i>	1,169,304	1,170,561	1,164,542	1,149,681	1,128,783	1,144,088	1,163,224	1,186,306
<i>IDP-TS</i>	<b>1,167,445</b>	<b>1,166,556</b>	<b>1,162,825</b>	<b>1,148,820</b>	<b>1,126,964</b>	<b>1,142,510</b>	1,162,256	1,183,492

### **3.5. Conclusions**

In this chapter, DP and GA are hybridized to solve DFLP for RMS. The state-space of each stage of DP is determined using a heuristic procedure. In this procedure, the optimal or best layouts for each of the periods are to be found. For small-sized problems, SFLP can be solved exact using a mathematical programming approach. However, for larger instances of the problem, and because of the combinatorial NP-hard nature of the problem, it cannot be solved optimally in polynomial time. The GA is employed for larger instances to find near-optimal solutions. Then, the neighborhood solutions of the best layouts of all periods are generated, and the state-space of DP is shaped.

As the state-space is still large and enumerating all these layouts are highly time-consuming for DP, GA is used to select from this state-space in such a way that the best result for DFLP is obtained. Finally, the multi-period layout resulted from GA, and DP is improved using TS. Computational results show that the proposed approach outperforms other DP benchmarks in the literature in many instances.

### **Bibliography**

1. Azab, A., H. ElMaraghy, P. Nyhuis, J. Pachow-Frauenhofer, and M. Schmidt, *Mechanics of change: A framework to reconfigure manufacturing systems*. CIRP Journal of Manufacturing Science and Technology, 2013. 6(2): p. 110-119.
2. Al-Zubaidi, S.Q.D., G. Fantoni, and F. Failli, *Analysis of Drivers for Solving Facility Layout Problems: Literature Review*. Journal of Industrial Information Integration, 2020: p. 100187.
3. Anjos, M.F. and M.V. Vieira, *Mathematical optimization approaches for facility layout problems: The state-of-the-art and future research directions*. European Journal of Operational Research, 2017. 261(1): p. 1-16.
4. Hitchings, G., *Control, Redundancy, and Change in Layout Systems1*. AIIE Transactions, 1970. 2(3): p. 253-262.

5. Rosenblatt, M.J., *The dynamics of plant layout*. Management Science, 1986. 32(1): p. 76-86.
6. Sweeney, D.J. and R.L. Tatham, *An improved long-run model for multiple warehouse location*. Management Science, 1976. 22(7): p. 748-758.
7. Ballou, R.H., *Dynamic warehouse location analysis*. Journal of Marketing Research, 1968: p. 271-276.
8. Armour, G.C. and E.S. Buffa, *A heuristic algorithm and simulation approach to relative location of facilities*. Management Science, 1963. 9(2): p. 294-309.
9. Lacksonen, T. and E.E. Ensore Jr, *Quadratic assignment algorithms for the dynamic layout problem*. The International Journal of Production Research, 1993. 31(3): p. 503-517.
10. Urban, T.L., *A heuristic for the dynamic facility layout problem*. IIE transactions, 1993. 25(4): p. 57-63.
11. Balakrishnan, J., C.H. Cheng, and D.G. Conway, *An improved pair-wise exchange heuristic for the dynamic plant layout problem*. International Journal of Production Research, 2000. 38(13): p. 3067-3077.
12. Balakrishnan, J., C.H. Cheng, D.G. Conway, and C.M. Lau, *A hybrid genetic algorithm for the dynamic plant layout problem*. International Journal of Production Economics, 2003. 86(2): p. 107-120.
13. Erel, E., J. Ghosh, and J. Simon, *New heuristic for the dynamic layout problem*. Journal of the Operational Research Society, 2003. 54(12): p. 1275-1282.
14. Erel, E., J.B. Ghosh, and J.T. Simon, *Erratum: New heuristic for the dynamic layout problem*. Journal of the Operational Research Society, 2005. 56(8): p. 1001-1001.
15. Dunker, T., G. Radons, and E. Westkämper, *Combining evolutionary computation and dynamic programming for solving a dynamic facility layout problem*. European Journal of Operational Research, 2005. 165(1): p. 55-69.

16. Udomsakdigool, A. and S. Bangsaranthip, *Combining ant colony optimization and dynamic programming for solving a dynamic facility layout problem*. World Academy of Science, Engineering and Technology, 2010. 64: p. 523-527.
17. Baykasoglu, A., T. Dereli, and I. Sabuncu, *An ant colony algorithm for solving budget constrained and unconstrained dynamic facility layout problems*. Omega, 2006. 34(4): p. 385-396.
18. Pourvaziri, H. and B. Naderi, *A hybrid multi-population genetic algorithm for the dynamic facility layout problem*. Applied Soft Computing, 2014. 24: p. 457-469.
19. Azimi, P. and H.R. Charmchi. *A new heuristic algorithm for the dynamic facility layout problem with budget constraint*. in *Proceedings of International Conference on Modelling and Simulation (ICMS 2011)*. 2011.
20. Palekar, U. S., R. Batta, R.M. Bosch, and S. Elhence, *Modeling uncertainties in plant layout problems*. European Journal of Operational Research, 1992. 63(2): p. 347-359.
21. Balakrishnan, J., F.R. Jacobs, and M.A. Venkataramanan, *Solutions for the constrained dynamic facility layout problem*. European Journal of Operational Research, 1992. 57(2): p. 280-286.
22. Balakrishnan, J. and C.H. Cheng, *Genetic search and the dynamic layout problem*. Computers & Operations Research, 2000. 27(6): p. 587-593.
23. Rodríguez, J.M., et al. *Solving the Quadratic Assignment and Dynamic Plant Layout Problems Using a New Hybrid Meta-Heuristic Approach*. in *HPCS*. 2004.

## CHAPTER 4

### SEMI-ROBUST LAYOUT DESIGN FOR CELLULAR MANUFACTURING IN A DYNAMIC ENVIRONMENT

Chapter 3 deals with the changes in product mix and demand from a period to the next by rearranging the layouts. However, sometimes due to some restrictions the managerial decision is to avoid any rearrangements. To this aim, a semi-robust cellular approach is proposed in this chapter, which is able to cope with both the continuous change of product mix and the volatility in product demand. Cellular manufacturing system (CMS) which is one of the modern manufacturing paradigms is studied that has the advantages of high flexibility and throughput. In this chapter, the problems of cell formation (CF) and cellular layout (CL) encountered in the design of a CMS are investigated. At heart of the proposed robust approach, the facility layout is not being changed from one period to the other, but rather the positions of the pick-up/drop-off points of cells. The developed model and solution algorithms can concurrently make the decisions regarding the optimum number of cells and grouping facilities into different cells (CF), the layout of unequal-area facilities inside a cell (intra-cellular layout) and the layout of cells in the planar site (inter-cellular layout). The problem is formulated as a multi-objective mathematical programming model. A modified non-dominated sorting genetic algorithm (MNSGA-II) is then used to obtain Pareto-optimal solutions for the problems. In the proposed MNSGA-II, an improved non-dominated sorting strategy and a modified dynamic crowding distance procedure are implemented. The effectiveness of the MNSGA-II is evaluated against two well-known multi-objective optimization algorithms, namely multi-objective particle swarm optimization and non-dominated ranking genetic algorithm. To this aim first, the input parameters of all three algorithms are tuned using the response surface method. Then, several numerical examples and computational experiments are carried out; four metrics are employed to evaluate the quality of the developed algorithms. The results show that the proposed methodology is efficient in finding Pareto-optimal solutions.

## ***Nomenclature***

### **Indices:**

$i, j$  Index set of machines  $i, j \in \{1, 2, \dots, N\}$

$k, l$  Index set of cells  $k, l \in \{1, 2, \dots, K = N\}$

$t$  Index set of periods  $t \in \{1, 2, \dots, T\}$

### **Parameters:**

$L$  The length of the shop floor along the  $x$ -axis

$W$  The length of the shop floor along the  $y$ -axis

$l_i^M$  The length of the longer side of machine  $i$

$w_i^M$  The length of the shorter side of machine  $i$

$F_{ijt}^M$  The flow of materials between machine  $i$  and machine  $j$  in period  $t$

$c^M$  The intra-cellular MHC per unit distance

$c^{PDC}$  The inter-cellular MHC per unit distance

$\delta^M$  The clearance distance that is required around the machines

$\delta^C$  The clearance distance that is required around the cells

$H_{ij}$  The similarity between machines  $i$  and  $j$

$M$  A large enough number

### **Decision Variables:**

$(x_i^M, y_i^M)$  The coordinates of the centroid of machine  $i$

$(x_k^C, y_k^C)$  The coordinates of the centroid of cell  $k$

$(x_{kt}^{PDC}, y_{kt}^{PDC})$  The coordinates of the P/D point of cell  $k$  in period  $t$

$lx_i^M$  The length of machine  $i$  along the  $x$ -axis

$ly_i^M$  The length of machine  $i$  along the  $y$ -axis

$l_k^C$	The length of cell $k$ along the $x$ -axis
$w_k^C$	The length of cell $k$ along the $y$ -axis
$d_{ijt}^M$	The weighted distance between machine $i$ and machine $j$ in period $t$
$dx_{ij}^M$	The horizontal distance between the centroids of machines $i$ and $j$
$dy_{ij}^M$	The vertical distance between the centroids of machines $i$ and $j$
$d_{klt}^C$	The weighted distance between the P/D points of cells $k$ and $l$ in period $t$
$dx_{klt}^C$	The horizontal distance between the P/D points of cells $k$ and $l$ in period $t$
$dy_{klt}^C$	The vertical distance between the P/D points of cells $k$ and $l$ in period $t$
$dx_{ikt}^{MC}$	The horizontal distance between the centroid of machine $i$ and the P/D point of cell $k$ in period $t$
$dy_{ikt}^{MC}$	The vertical distance between the centroid of machine $i$ and the P/D point of cell $k$ in period $t$
$\alpha_{kt}$	The percentage of a side of cell $k$ for locating the P/D point in period $t$
$left_{ij}^M$	$= \begin{cases} 1 & \text{if machine } i \text{ is to the left of machine } j \\ 0 & \text{otherwise} \end{cases}$
$below_{ij}^M$	$= \begin{cases} 1 & \text{if machine } i \text{ is below machine } j \\ 0 & \text{otherwise} \end{cases}$
$left_{kl}^C$	$= \begin{cases} 1 & \text{if cell } k \text{ is to the left of cell } l \\ 0 & \text{otherwise} \end{cases}$
$below_{kl}^C$	$= \begin{cases} 1 & \text{if cell } k \text{ is below cell } l \\ 0 & \text{otherwise} \end{cases}$
$Sl_{kt}^C$	$= \begin{cases} 1 & \text{if the P/D point of cell } k \text{ is on its left side in period } t \\ 0 & \text{otherwise} \end{cases}$
$Sr_{kt}^C$	$= \begin{cases} 1 & \text{if the P/D point of cell } k \text{ is on its right side in period } t \\ 0 & \text{otherwise} \end{cases}$
$Su_{kt}^C$	$= \begin{cases} 1 & \text{if the P/D point of cell } k \text{ is on its upper side in period } t \\ 0 & \text{otherwise} \end{cases}$
$Sb_{kt}^C$	$= \begin{cases} 1 & \text{if the P/D point of cell } k \text{ is in its bottom side in period } t \\ 0 & \text{otherwise} \end{cases}$

$$O_i = \begin{cases} 1 & \text{if machine } i \text{ has horizontal orientation} \\ 0 & \text{otherwise} \end{cases}$$

$$V_{ik} = \begin{cases} 1 & \text{if machine } i \text{ is in cell } k \\ 0 & \text{otherwise} \end{cases}$$

$$U_{ikjl} = \begin{cases} 1 & \text{if machine } i \text{ is in cell } k \text{ and machine } j \text{ is in cell } l \\ 0 & \text{otherwise} \end{cases}$$

#### **4.1. Introduction**

With the rise in global competition, manufacturing firms try to reduce time-to-market and to become cost-effective while improving product quality. To achieve this goal, many manufacturing systems have turned out to cellular manufacturing, which is one of the modern manufacturing paradigms that captures the advantages of both job-shops (high flexibility allowing for the production of a wide variety of products) and flow-shops (high throughput) [1]. The implementation of cellular manufacturing systems (CMSs) results in improvement of the system's efficiency, system management and the quality of the products, as well as reduction of the setup time, work-in-process inventory and material flow [2, 3]. To reach all these advantages, the CMS should be designed efficiently. Cell formation (CF) and cellular layout (CL) problems are key elements in designing a CMS [4]. Since CF and CL problems are interrelated, it is important to integrate the two problems and obtain a global optimum solution for the combined problem [5].

CF is the problem of grouping machines and parts based on their similarities into machine cells and part families, respectively [6]. CL problem includes both the inter- and intra-cellular layout problems [7]. The inter-cellular layout problem includes that of locating the cells on the shop floor to minimize the total material handling cost (MHC) between cells and the intra-cellular layout problem involves laying out machines within each cell to minimize the total MHC between machines. An efficient layout design is a flexible one that can rapidly be adapted to the dynamicity, uncertainty, and in some cases the volatility even of market demand with variable products' mix and shorter product lifecycles. These variations change the material flow between machines and deem the optimal design of



CMS in a given period completely inefficient provided the following period's varying demand, and hence, necessitate rearrangements in the CMS.

There are two main approaches to handle the changes in the flow of materials between machines from a period to the next in a planning horizon. The first approach is called dynamic CMS (DCMS). In DCMS, it is assumed that the system can be reconfigured in each period. The reconfiguration of a manufacturing system involves some costly activities such as machine relocation, installation and uninstallation costs, as well as any lost production time [8]. In this case, finding a design that minimizes the total inter- and intra-cellular MHC and rearrangement costs over all the planning periods are of interest. The optimal DCMS layout is considered a trade-off where a compromise is being made between the cumulative increased MHC in the coming period because of a layout design that is deemed inefficient and obsolete versus the reconfiguration costs.

The second approach is to design a robust CMS (RCMS). The goal of RCMS is to find a single layout, called a robust layout, that is efficient over the planning horizon [9]. Hence, unlike DCMS, in RCMS the layout remains unchanged. The main reason for using the RCMS approach is that in some cases because of high layout rearrangement costs such as those due to interruption in production or moving of facilities, the managerial decision is to avoid any rearrangements. However, it is worthy to note that changing some particular limited aspects of the layout might not be necessarily costly. Therefore, it is effective in RCMS to change such elements in the layout in order to make the robust layout more adaptable to changes in product mix and demand. The locations of the pick-up/drop-off (P/D) points of the cells are one of these layout elements. In many CMSs, it is not costly to change the location of the P/D points of the cells. Doing such changes results in a semi-robust layout where on one hand the utility of the robust approach in eliminating layout rearrangement costs is preserved by keeping fixed positions for the facilities, while on the other hand, the advantage of the dynamic approach is to some extent preserved and kept. This chapter looks at the design of a semi-robust CMS (SRCMS) in a dynamic environment and presents a multi-objective mixed-integer nonlinear programming (MINLP) model for the integrated CF and CL (inter- and intra-cellular) problems.

As both the CF and CL problems are NP-hard combinatorial optimization problems [10, 11], the resulting integrated problem is clearly highly intractable and hence, finding the exact solution for large-sized problems are computationally prohibitive and heuristic or metaheuristic approaches are needed to find the near-optimum solutions. A modified non-dominated sorting genetic algorithm (MNSGA-II) metaheuristic is used. The structure of the rest of the chapter is as follows. Section 4.2 provides a critical review of the related work in the literature. The problem description and the mathematical model are explained in Sections 4.3 and 4.4, respectively. Section 4.5 presents the proposed multi-objective optimization algorithm to solve the problem. The computation results are reported in Section 4.6 and finally, the conclusions are given in Section 4.7.

## ***4.2. Literature Review***

Since this study develops a semi-robust layout for CMSs in a dynamic environment, only the studies that have addressed the layout (inter- and/or intra-cellular) problem in CMSs are included. The reviewed literature is classified according to the following: (1) dynamic cellular manufacturing system (2) robust cellular manufacturing system (3) conclusions.

### ***4.2.1. Dynamic Cellular Manufacturing System (DCMS)***

An integer nonlinear programming (INLP) model was proposed by Kia et al. [12] for simultaneously designing the CF and intra-cellular layout of equal-area facilities in a DCMS in a fuzzy environment, where they assumed demand for each part type in each period in the form of an asymmetrical trapezoidal fuzzy number. The model was linearized and solved using a fuzzy linear programming approach. Kia et al. [13] proposed an MINLP model for the integrated CF and inter- and intra-cellular layout problems in a DCMS. In their model, the equal area facilities were supposed to have a multi-row layout in each cell. The authors assumed that there was no physical partitioning between cells but the number of cells and the maximum and the minimum number of facilities that could be assigned to each cell were predetermined. They developed a simulated annealing (SA) metaheuristic procedure to solve the model. Another MINLP model was presented by Kia et al. [14] for integrated CF and intra-cellular layout problems in a CMS in a dynamic environment. In

their model, multi-row layouts for unequal-area facilities were obtained in a continual space with the objective of minimizing the total inter- and intra-cellular MHC and rearrangement costs. However, the number, shapes, locations, and dimensions of the cells, as well as the location of their P/D points and the maximum and the minimum number of machines that could be assigned to each cell were predetermined. It was also assumed that the orientations of the machines were fixed and known during the planning horizon. Moreover, the authors did not develop a solution approach for the proposed NP-hard model.

An integrated multi-objective INLP model was presented by Bagheri and Bashiri [15] to solve the CF, inter-cellular layout, and operator assignment problems of DCMS. The LP-metric approach was implemented to obtain the most preferred solution. Moreover, to validate the model, it was linearized and solved using a branch and bound (B&B) algorithm. The shortcoming of the proposed model was that the MHC and rearrangement costs only depended on the inter-cellular layout which was not realistic. This was because the model did not determine the locations of machines inside the cells. Besides, the number of cells and the maximum and the minimum number of machines that could be assigned to each cell were predefined and fixed for all the periods.

An MINLP model was proposed by Kia et al. [16] that was similar to the one presented by Kia et al. [14] yet overcame some of its shortcomings. First, this model optimized both the inter- and intra-cellular layouts. Second, the number, locations, and dimensions of the rectangular-shaped cells, as well as the locations of their P/D points were considered as decision variables that were optimally obtained by the model for each period on the continuous shop. An SA algorithm was developed to solve the model.

Sakhaii et al. [5] proposed an integrated mixed-integer linear programming (MILP) model for a CMS in a dynamic environment, where the demand for each part was changing from one period to the next. The model determined the CF, the inter-cellular layout in a discrete manner, the number of operators and their allocations to machines, and the production plan for each period. The authors solved the model using a branch and cut algorithm. The drawback of this model, the same as the one proposed by Bagheri and Bashiri [15], was that only the inter-cellular layout was determined and the locations of the machines inside

each cell were not considered. Hence, the real intra-cellular MHC was not calculated. Also, the number of cells to be constructed and the minimum and the maximum number of machines that could be assigned to each cell were predetermined and constant over all the planning periods.

A multi-objective MINLP model was proposed by Mehdizadeh and Rahimi [17] for solving the integrated CF, inter- and intra-cellular layout, and operator assignment problems in a DCMS. The authors assumed that the layout of machines inside each cell was linear. In their model, the first objective was to minimize the inter- and intra-cell material handling and rearrangement costs, the second was to minimize the costs related to machines and operators, and the third was to maximize the successive forward flow ratio. The model was linearized and solved by a B&B algorithm for small-sized instances and a multi-objective SA procedure and vibration-damping optimization algorithm were presented for solving large-sized instances. Although the model determined the locations of the cells and the machines inside the cells, for calculating the cost, the distance between two machines that were not located in the same cell was not calculated based on the real locations of machines inside the cells and was only calculated as the distance between two different cells that the machines were located in. Moreover, the number of cells and the maximum and the minimum number of machines that could be assigned to each cell were known and constant over all the planning periods.

The CF and discrete inter- and intra-cellular layout problems of a DCMS were addressed by Bayram and Şahin [18]. An integrated model was presented and two hybrid heuristic algorithms were proposed to solve the model. In their model, it was assumed that no physical partitioning existed between cells but the number of cells and the maximum and the minimum number of machines that could be assigned to each cell were predetermined. Kumar and Singh [19] considered the same assumptions and proposed a similarity score-based heuristic approach for solving the CF and discrete inter- and intra-cellular layout problems of a DCMS. This method had two phases. In the first phase, using the similarity scores between machines, the machine-cellular cluster formation was identified assuming that the number of cells and the number of machines to be assigned to cells were known in advance. In the second phase, based on the result of the first phase, the locations of

machines in each period were obtained in a way that the inter- and intra-cellular MHC and rearrangement costs were minimized over all the planning periods.

An integrated CF and CL problem was addressed by Golmohammadi et al. [20] for DCMS in a discrete manner. The demand for parts in each period was assumed to be uncertain and have a normal distribution. Their model objective function included but was not limited to the inter- and intra-cellular MHC, rearrangement costs and the cost of the difference between the estimated demand and its expected value. The authors formulated the problem as an MINLP model and solved it using a GA.

#### ***4.2.2. Robust Cellular Manufacturing System (RCMS)***

In this section, articles that have studied a robust layout for CMSs when there is a change or uncertainty in product mix and/or demand are covered. Wang et al. [21] presented a model for CL problem that minimized both the inter- and intra-cellular MHC under varying demand. Since the products' demands are not fixed over their lifecycles, the authors calculated the present value of the unit distance MHC of the products and determined a robust layout for the planning horizon. In their model, it was assumed that CF had been done in advance and machines were assigned to different cells. A bi-quadratic assignment problem (bi-QAP) was used to model the inter- and intra-cellular layout problems. An SA metaheuristic algorithm was adopted to solve the model.

Tavakkoli-Moghaddam et al. [22] developed an INLP model to minimize the total inter- and intra-cellular MHC when designing a robust layout for a CMS in a discrete manner. It was assumed that the demand had a normal distribution, the CF was done in the prior and the layout of machines in cells was U-shaped. The authors linearized the model using an approximate approach and solved it by B&B algorithm. A stochastic MINLP model was proposed by Ariaifar et al. [23] for the discrete layout problem in a CMS. The authors assumed that the demand for each product in the planning horizon was uniformly distributed and that CF was done beforehand. The result of their model was a robust layout based on the cumulative occurrence probability of demand scenarios. The model was solved by two solution approaches, a B&B algorithm and an enumeration method. An MILP was presented by Paydar et al. [24] for a CMS in a discrete manner that integrated

the CF problem, CL problem, supplier selection, and many other aspects. The demand of each part type was assumed to be uncertain (scenarios based) and hence the model was solved using a robust optimization model.

Kumar et al. [25] presented an embedded SA-based metaheuristic to solve an integrated sustainable robust stochastic CF and CL problem with discrete representation. The problem was modeled as a bi-QAP to minimize the total inter- and intra-cellular MHC. It was assumed that the demand had a normal distribution and the locations and the number of cells were predetermined. The authors generated layout alternatives using their proposed method and Big Data approach and obtained the final layout by applying multi-criteria decision-making techniques.

### ***4.2.3. Conclusions***

After reviewing the literature, it is revealed that with respect to the articles of CMS there are a few articles that have addressed finding the optimum layout in CMS in a dynamic environment. Gaps and overlaps in the literature could be identified by examining the synthesis matrix (Table 4.1). Some of these articles used a sequential approach wherein CF and CL problems are solved in two separate steps. However, CF and CL problems are interrelated problems and optimizing them sequentially may lead to a suboptimal solution.

Many other articles have paid less attention to important considerations such as optimization of the number of cells, P/D points, inequality in the area of facilities and multi-objective optimization, and have used restrictive assumptions. These restrictive assumptions make the model unrealistic and not applicable. For instance, despite the importance of the clearance distance, none of the studied articles have considered it. Clearance distance is the distance required around machines for loading and unloading materials or products, access of the workers to machines to operating them or carrying out maintenance [26]. However, the importance of considering clearance distance is high only when there is a continuous representation as the locations of facilities are predefined in discrete representation [27].

Furthermore, Table 4.1 shows that either a robust or dynamic approach was used to deal with the problem in a dynamic environment and none of them has used a semi-robust

approach to catch the benefits of both robust and dynamic approaches. As a consequence, to cope with these limitations, this research proposes a semi-robust optimization approach. The proposed approach can optimize the number of cells, the assignment of machines to the cells, the location of cells and the machines inside the cells, simultaneously.

Table 4.1. Summary of the related works

Article	Problem characteristics							Solution Characteristics		
	Cell Formation	Layout		Continues	Clearance Distance	P/D Point	Multi-Objective	Number of Cells	Integrated	Approach
Inter-cellular	Intra-cellular									
[21]	No	Yes	Yes	No	N/A	No	No	Predetermined	No	Robust
[22]	No	Yes	Yes	No	N/A	No	No	Predetermined	No	Robust
[23]	No	Yes	Yes	No	N/A	No	No	Predetermined	No	Robust
[12]	Yes	No	Yes	No	N/A	No	No	Predetermined	Yes	Dynamic
[13]	Yes	Yes	Yes	No	N/A	No	No	Predetermined	Yes	Dynamic
[14]	Yes	No	Yes	Yes	No	Yes	No	Predetermined	Yes	Dynamic
[15]	Yes	Yes	No	No	N/A	No	Yes	Predetermined	Yes	Dynamic
[24]	Yes	Yes	Yes	No	N/A	No	No	Predetermined	Yes	Robust
[16]	Yes	Yes	Yes	Yes	No	Yes	No	Optimized	Yes	Dynamic
[5]	Yes	Yes	No	No	N/A	No	No	Predetermined	Yes	Dynamic
[18]	Yes	Yes	Yes	No	N/A	No	No	Optimized	Yes	Dynamic
[17]	Yes	Yes	Yes	No	N/A	No	Yes	Predetermined	Yes	Dynamic
[19]	Yes	Yes	Yes	No	N/A	No	No	Predetermined	No	Dynamic
[20]	Yes	Yes	Yes	No	N/A	No	No	Predetermined	Yes	Dynamic
[25]	Yes	Yes	Yes	No	N/A	No	No	Predetermined	Yes	Robust
This work	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Optimized	Yes	Semi-Robust

### 4.3. Problem Description

The problem is to arrange unequal-area facilities within a shop floor and group them into different cells for a multi-period CMS. The CF and the layout of facilities are robust; *i.e.*, the assignment of facilities to the cells and the locations of the facilities and cells are fixed over all the planning periods. However, in the proposed model, the locations of the P/D points of the cells can be changed from one period to the next in order to minimize the cost and improve the robust layout for each period without doing any rearrangements. Therefore, an SRCMS can be defined as a multi-period CMS that has a single layout for all the periods in such a way that in each period, the assignment of facilities to the cells and

the locations of the facilities and the cells are fixed yet the locations of the P/D points of the cells are changed to adapt the layout with the changes in the flow. This is a good compromise between robust and dynamic layout design.

Two objectives are considered, namely minimization of the total MHC and maximization of the similarity (minimization of the dissimilarity) between the facilities in each cell. The MHC includes both the total inter- and intra-cellular transportation costs. Because cells do not usually have tangible physical boundaries, changing the locations of P/D points does not incur a considerable cost to the system but at the same time reduces the total MHC.

Since the most important objective in designing the layout of facilities is to minimize the total MHC, an optimal layout tries to position the facilities as close as possible to each other. In real-world problems, though, a minimum space is required around each facility which is called the clearance distance. To elaborate, for instance, in manufacturing systems a clearance distance is necessary around each machine for safety issues and to allow for maintenance and repair of the machines, temporarily storing work-in-process (WIP), proper ventilation and minimizing the vibration effects of neighboring machines [26, 27]. However, some authors integrate the clearance distance into the dimensions of the facilities by adjusting their length and width [28] which simplifies the model yet does not provide the possibility of sharing the clearance distances between facilities when applicable.

#### **Assumptions:**

- The flow of materials changes from one period to the other; the periods can have any length (a month, a season, or even a year).
- The facilities and the shop floor have rectangular shapes with known dimensions.
- Facilities can rotate and have either horizontal or vertical orientations.
- Facilities can be located at any place within the shop floor yet they should not overlap with each other or the boundaries of the cells/shop floor.
- A clearance distance is required around each facility.
- Cells should not overlap with each other.
- The intra-cellular MHC per unit distance is less than the inter-cellular MHC per unit distance.
- The material flow of each period is known and deterministic.



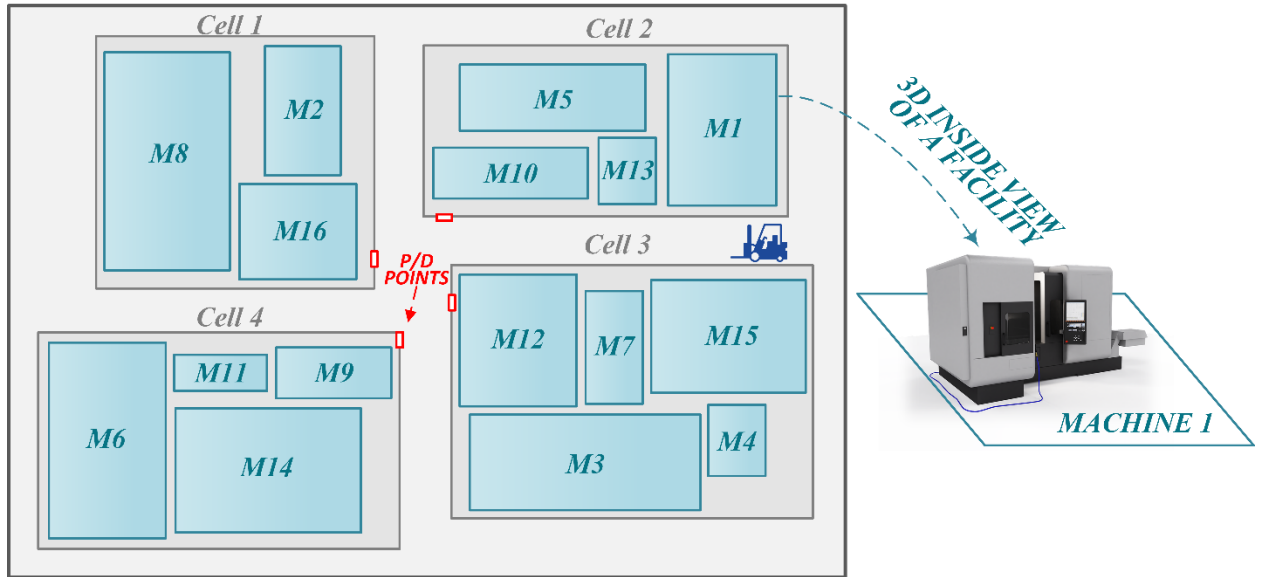


Figure 4.1. An example of the SRCMS with 16 facilities and 4 cells

#### 4.4. Mathematical Model

In this section, an MINLP formulation is presented to solve the integrated CF and CL problem of an SRCMS with unequal-area facilities in a dynamic environment. Since each facility contains a machine as shown in Figure 4.1, from this point on and to the end of the manuscript a facility is simply referred to as a machine without loss of generality.

##### 4.4.1. Objective Functions

The first objective function (4.1) minimizes the total MHC over all the planning periods. This cost objective function is calculated by summing the product of the flow of materials between each pair of machines in each period and the weighted distance between them. The term weighted distance is used since the cost of material handling per unit distance is not constant. To be precise, it is assumed that the cost of moving the materials between cells per unit distance is more than that of moving the materials between machines within a cell.

The second objective function (4.2) minimizes the dissimilarity of machines in each cell by avoiding assigning machines that are less similar to each other in the same cell. The

similarity here is the likeness of the machines based on their processes. A number of researchers have used different types of similarity and dissimilarity coefficients for solving the CF problem [29]. Among these coefficients, the Jaccard similarity coefficient  $H_{ij} = \frac{a}{a+b+c}$  is used to calculate the similarity between machines where  $a$  is the number of parts that need to be processed in both machines,  $b$  the number of parts that need to be processed in machine  $i$  but not  $j$ , and  $c$  the number of parts that need to be processed in machine  $j$  but not  $i$  [30, 31]. According to Yin and Yasuda [31], the Jaccard similarity coefficient is the most stable similarity coefficient among all other coefficients.

$$\text{Min } Z_1 = \sum_{t=1}^T \sum_{i=1}^{N-1} \sum_{j=i+1}^N F_{ijt}^M d_{ijt}^M \quad (4.1)$$

$$\text{Min } Z_2 = \sum_{k=1}^K \sum_{i=1}^{N-1} \sum_{j=i+1}^N U_{ikjk} (1 - H_{ij}) \quad (4.2)$$

If machines  $i$  and  $j$  are located in the same cell, the distance between them is the rectilinear distance between their centroids. This distance is calculated using constraints (4.3)-(4.7). The first two terms in constraint (4.3) calculate the weighted rectilinear distance between machines  $i$  and  $j$  while the third term ensures that machines  $i$  and  $j$  are in the same cell. It should be noted that as the locations of the machines are fixed over all the planning periods, this distance does not depend on  $t$ .

$$d_{ijt}^M \geq c^M dx_{ij}^M + c^M dy_{ij}^M - M(1 - U_{ikjl}) \quad \forall i, j, t, k = l \quad (4.3)$$

$$dx_{ij}^M \geq x_i^M - x_j^M \quad \forall i, j \quad (4.4)$$

$$dx_{ij}^M \geq x_j^M - x_i^M \quad \forall i, j \quad (4.5)$$

$$dy_{ij}^M \geq y_i^M - y_j^M \quad \forall i, j \quad (4.6)$$

$$dy_{ij}^M \geq y_j^M - y_i^M \quad \forall i, j \quad (4.7)$$

If machines  $i$  and  $j$  are located in different cells, then the distance separating them is the total of the weighted rectilinear distance between the centroid of each machine to the P/D point of its cell plus the weighted distance between the P/D points of their cells as shown in Figure 4.2. Constraints (4.8)-(4.18) are used to calculate the distance between machines that are not in the same cell. The first term in constraint (4.8) is the distance between the P/D points of the cells which is calculated using constraints (4.9)-(4.13). The second and

third terms in constraint (4.8) are the distances between the centroid of each machine and the P/D point of its cell which are calculated using constraints (4.14)-(4.18). And finally, the last term of constraint (4.8) is to ensure that two machines are not located in the same cell.

As the locations of the P/D points of the cells are changing from one period to the next, these distances are all dependent on the  $t$ .

$$d_{ijt}^M \geq c^{PDC} d_{klt}^C + c^M d_{ikt}^{MC} + c^M d_{jlt}^{MC} - M(1 - U_{ikjl}) \quad \forall i, j, t, k \neq l \quad (4.8)$$

$$d_{klt}^C \geq dx_{klt}^C + dy_{klt}^C \quad \forall k, l, t \quad (4.9)$$

$$dx_{klt}^C \geq x_{kt}^{PDC} - x_{lt}^{PDC} \quad \forall k, l, t \quad (4.10)$$

$$dx_{ijt}^C \geq x_{lt}^{PDC} - x_{kt}^{PDC} \quad \forall k, l, t \quad (4.11)$$

$$dy_{klt}^C \geq y_{kt}^{PDC} - y_{lt}^{PDC} \quad \forall k, l, t \quad (4.12)$$

$$dy_{klt}^C \geq y_{lt}^{PDC} - y_{kt}^{PDC} \quad \forall k, l, t \quad (4.13)$$

$$d_{ikt}^{MC} \geq dx_{ikt}^{MC} + dy_{ikt}^{MC} \quad \forall i, k, t \quad (4.14)$$

$$dx_{ikt}^{MC} \geq x_{kt}^{PDC} - x_i^M \quad \forall i, k, t \quad (4.15)$$

$$dx_{ikt}^{MC} \geq x_i^M - x_{kt}^{PDC} \quad \forall i, k, t \quad (4.16)$$

$$dy_{ikt}^{MC} \geq y_{kt}^{PDC} - y_i^M \quad \forall i, k, t \quad (4.17)$$

$$dy_{ikt}^{MC} \geq y_i^M - y_{kt}^{PDC} \quad \forall i, k, t \quad (4.18)$$

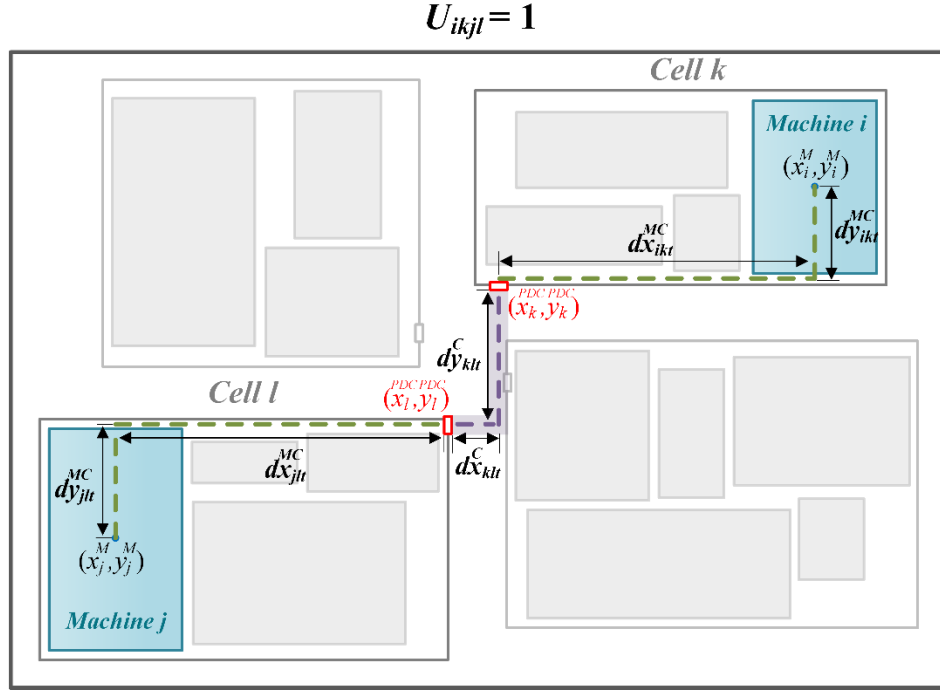


Figure 4.2. The distance between machine  $i$  in cell  $k$  and machine  $j$  in cell  $l$

#### 4.4.2. Cell Formation Constraints

The decision variable  $V_{ik}$  is equal to one if machine  $i$  is assigned to cell  $k$ . For any cell  $k$ , if the value of  $V_{ik}$  is equal to zero for all  $i$ , it means that cell  $k$  is not generated. Constraint (4.19) ensures that each machine is assigned to one and only one cell and constraint (4.20) controls the value of the dependent variable  $U_{ijkl}$ , which should be equal to one only if both the variables  $V_{ik}$  and  $V_{jl}$  are equal to one.

$$\sum_{k=1}^K V_{ik} = 1 \quad \forall i \quad (4.19)$$

$$U_{ijkl} \geq V_{ik} + V_{jl} - 1 \quad \forall i, j, k, l \quad (4.20)$$

#### 4.4.3. Non-Overlapping Constraints

Constraints (4.21) and (4.22) are to determine the length of a machine along the  $x$ - and  $y$ -axis based on the orientation of the machine. If machine  $i$  has a horizontal orientation,  $O_i = 1$ , the longer side of the machine is located along the  $x$ -axis and its shorter side is along the

y-axis. If machine  $i$  has a vertical orientation,  $O_i = 0$ , the shorter side of it is along the  $x$ -axis and its longer side is along the  $y$ -axis.

$$lx_i^M = l_i^M O_i + w_i^M (1 - O_i) \quad \forall i \quad (4.21)$$

$$ly_i^M = l_i^M (1 - O_i) + w_i^M O_i \quad \forall i \quad (4.22)$$

Constraints (4.23)-(4.25) are to prevent any overlapping between machines. A machine can be below, above, to the left, or the right of the other machine. Two decision variables  $left_{ij}^M$  and  $below_{ij}^M$  determine the position of machine  $i$  relative to machine  $j$ . If machine  $i$  is to the left of machine  $j$ , constraint (4.23) holds and hence, the  $x$ -coordinate of the centroid of machine  $i$  plus half of its length along the  $x$ -axis plus the clearance distance of machines should be less than or equal to the  $x$ -coordinate of the machine  $j$  minus the half of its length along the  $x$ -axis.

In a similar fashion, if machine  $i$  is below machine  $j$ , constraint (4.24) is imposed and hence, the  $y$ -coordinate of the centroid of machine  $i$  plus the half of its length along the  $y$ -axis plus the clearance distance of machines should be less than or equal to the  $y$ -coordinate of the machine  $j$  minus the half of its length along the  $y$ -axis.

Constraint (4.25) ensure that only one of its four terms is equal to one, which in turn ensures that only one of the two constraints (4.23) or (4.24) holds.

$$(x_i^M + 0.5lx_i^M) + \delta^M \leq (x_j^M - 0.5lx_j^M) + M(1 - left_{ij}^M) \quad \forall i \neq j \quad (4.23)$$

$$(y_i^M + 0.5ly_i^M) + \delta^M \leq (y_j^M - 0.5ly_j^M) + M(1 - below_{ij}^M) \quad \forall i \neq j \quad (4.24)$$

$$left_{ij}^M + left_{ji}^M + below_{ij}^M + below_{ji}^M = 1 \quad \forall i > j \quad (4.25)$$

Constraints (4.26)-(4.28) ensure that there is no overlapping between cells. A cell can be below, above, to the left, or the right of the other cell. Two decision variables  $left_{kl}^C$  and  $below_{kl}^C$  determine the position of cell  $k$  relative to cell  $l$ . If cell  $k$  is to the left of cell  $l$ , constraint (4.26) is imposed and hence, the  $x$ -coordinate of the centroid of cell  $k$  plus half of its length along the  $x$ -axis plus the clearance distance of cells should be less than or equal to the  $x$ -coordinate of the cell  $l$  minus the half of its length along the  $x$ -axis.

Similarly, if cell  $k$  is below cell  $l$ , constraint (4.27) is imposed and hence, the  $y$ -coordinate of the centroid of cell  $k$  plus the half of its length along the  $y$ -axis plus the clearance distance of cells should be less than or equal to the  $y$ -coordinate of the cell  $l$  minus the half of its length along the  $y$ -axis. Constraint (4.28) ensure that only one of its four terms is equal to one, which in turn ensures that only one of the two constraints (4.26) or (4.27) holds.

$$(x_k^C + 0.5l_k^C) + \delta^C \leq (x_l^C - 0.5l_l^C) + M(1 - left_{kl}^C) \quad \forall k \neq l \quad (4.26)$$

$$(y_k^C + 0.5w_k^C) + \delta^C \leq (y_l^C - 0.5w_l^C) + M(1 - below_{kl}^C) \quad \forall k \neq l \quad (4.27)$$

$$left_{kl}^C + left_{lk}^C + below_{kl}^C + below_{lk}^C = 1 \quad \forall k > l \quad (4.28)$$

#### 4.4.4. Within-Site Boundaries Constraints

Constraints (4.29)-(4.32) guarantee that if a machine is assigned to a cell, it is located within the boundaries of that cell considering its clearance distance.

$$(x_i^M + 0.5lx_i^M) + \delta^M \leq (x_k^C + 0.5l_k^C) + M(1 - V_{ik}) \quad \forall i, k \quad (4.29)$$

$$(x_i^M - 0.5lx_i^M) - \delta^M \geq (x_k^C - 0.5l_k^C) - M(1 - V_{ik}) \quad \forall i, k \quad (4.30)$$

$$(y_i^M + 0.5ly_i^M) + \delta^M \leq (y_k^C + 0.5w_k^C) + M(1 - V_{ik}) \quad \forall i, k \quad (4.31)$$

$$(y_i^M - 0.5ly_i^M) - \delta^M \geq (y_k^C - 0.5w_k^C) - M(1 - V_{ik}) \quad \forall i, k \quad (4.32)$$

Constraints (4.33)-(4.36) ensure that the cells are located within the boundaries of the shop floor considering their clearance distances.

$$x_k^C + 0.5l_k^C + \delta^C \leq L \quad \forall k \quad (4.33)$$

$$x_k^C - 0.5l_k^C - \delta^C \geq 0 \quad \forall k \quad (4.34)$$

$$y_k^C + 0.5w_k^C + \delta^C \leq W \quad \forall k \quad (4.35)$$

$$y_k^C - 0.5w_k^C - \delta^C \geq 0 \quad \forall k \quad (4.36)$$

#### 4.4.5. Constraints for Determining the Locations of P/D Points

The P/D point of a cell can be located on any side of the cell. As the locations of P/D points change in each period, the coordinates of the P/D points and their corresponding decision variables are a function of  $t$ . Constraint (4.37) ensures that there is one and only one P/D point for each cell. Constraints (4.38)-(4.43) determine the  $x$ -coordinate of the P/D point of a cell, based on the side where the P/D point is located. For instance, if the P/D point of cell  $k$  is located on the left side of the cell as shown in Figure 4.3, constraints (4.40) and (4.41) are imposed and constraints (4.38), (4.39), (4.42), and (4.43) are relaxed.

$$Sl_{kt}^C + Sr_{kt}^C + Su_{kt}^C + Sb_{kt}^C = 1 \quad \forall k, t \quad (4.37)$$

$$x_{kt}^{PDC} \leq (x_k^C + (\alpha_{kt} - 0.5)l_k^C) + M(1 - Su_{kt}^C - Sb_{kt}^C) \quad \forall k, t \quad (4.38)$$

$$x_{kt}^{PDC} \geq (x_k^C + (\alpha_{kt} - 0.5)l_k^C) - M(1 - Su_{kt}^C - Sb_{kt}^C) \quad \forall k, t \quad (4.39)$$

$$x_{kt}^{PDC} \leq (x_k^C - 0.5l_k^C) + M(1 - Sl_{kt}^C) \quad \forall k, t \quad (4.40)$$

$$x_{kt}^{PDC} \geq (x_k^C - 0.5l_k^C) - M(1 - Sl_{kt}^C) \quad \forall k, t \quad (4.41)$$

$$x_{kt}^{PDC} \leq (x_k^C + 0.5l_k^C) + M(1 - Sr_{kt}^C) \quad \forall k, t \quad (4.42)$$

$$x_{kt}^{PDC} \geq (x_k^C + 0.5l_k^C) - M(1 - Sr_{kt}^C) \quad \forall k, t \quad (4.43)$$

Likewise, constraints (4.44)-(4.49) determine the  $y$ -coordinate of the P/D point of a cell, based on the side that P/D point is located on. For instance, if the P/D point of cell  $k$  is located on the left side of the cell as shown in Figure 4.3, constraints (4.44) and (4.45) are imposed and constraints (4.46)-(4.49) are relaxed.

$$y_{kt}^{PDC} \leq (y_k^C + (\alpha_{kt} - 0.5)w_k^C) + M(1 - Sl_{kt}^C - Sr_{kt}^C) \quad \forall k, t \quad (4.44)$$

$$y_{kt}^{PDC} \geq (y_k^C + (\alpha_{kt} - 0.5)w_k^C) - M(1 - Sl_{kt}^C - Sr_{kt}^C) \quad \forall k, t \quad (4.45)$$

$$y_{kt}^{PDC} \leq (y_k^C + 0.5w_k^C) + M(1 - Su_{kt}^C) \quad \forall k, t \quad (4.46)$$

$$y_{kt}^{PDC} \geq (y_k^C + 0.5w_k^C) - M(1 - Su_{kt}^C) \quad \forall k, t \quad (4.47)$$

$$y_{kt}^{PDC} \leq (y_k^C - 0.5w_k^C) + M(1 - Sb_{kt}^C) \quad \forall k, t \quad (4.48)$$

$$y_{kt}^{PDC} \geq (y_k^C - 0.5w_k^C) - M(1 - Sb_{kt}^C) \quad \forall k, t \quad (4.49)$$

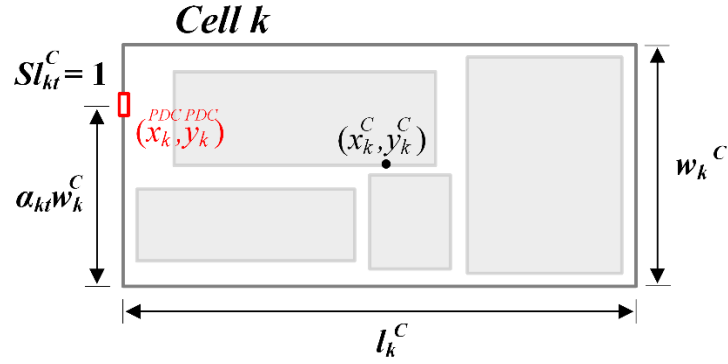


Figure 4.3. An example for the location of P/D point of cell  $k$

#### 4.4.6. Domain Constraints

The decision variables of the model are defined by constraints (4.50)-(4.52).

$$0 \leq \alpha_{kt} \leq 1 \quad \forall k, t \quad (4.50)$$

$$O_i, V_{ik}, U_{ikjl}, left_{ij}^M, below_{ij}^M, left_{kl}^C, below_{kl}^C, Sl_{kt}^C, Sr_{kt}^C, Su_{kt}^C, Sb_{kt}^C \in \{0,1\} \quad \forall i, j, k, l, t \quad (4.51)$$

$$x_i^M, y_i^M, x_k^C, y_k^C \geq 0 \quad \forall i, j, k, \quad (4.52)$$

#### 4.5. The Proposed Multi-Objective Optimization Algorithm

Some real-world optimization problems, such as the one that is studied in this chapter, have more than one possibly contradictory objectives that need to be optimized simultaneously.

The general form of a multi-objective optimization problem can be formulated as:

$$Max / Min f_m(\bar{x}) \quad m = 1, 2, \dots, N_{obj} \quad (4.53)$$

Subject to:

$$g_p(\bar{x}) \geq 0 \quad p = 1, 2, \dots, N_{con1} \quad (4.54)$$

$$h_q(\bar{x}) = 0 \quad q = 1, 2, \dots, N_{con2} \quad (4.55)$$



$$x_v^l \leq x_v \leq x_v^u \quad x_v \in \bar{x}, v = 1, 2, \dots, N_v \quad (4.56)$$

where the objective functions are represented by  $f_m$ , and inequality and equality constraints are represented by  $g_p$  and  $h_q$ , respectively. Decision variables are represented by  $x_v$  that are the components of a solution vector  $\bar{x}$  with the size of  $N_v$  [32].

Despite the single-objective optimization problems, the multi-objective ones do not have a single optimum solution and instead have multiple equivalent alternative solutions that are not dominated by any solution [33]. For a minimization problem, a solution  $\bar{x}_1$  dominates the other solution  $\bar{x}_2$  if  $\bar{x}_1$  is strictly better than the  $\bar{x}_2$  in at least one of the objectives and no worse than  $\bar{x}_2$  in any of the objectives [34]. Hence,  $\bar{x}_1$  dominates  $\bar{x}_2$ , expressed using  $\bar{x}_1 < \bar{x}_2$ , if:

$$1) \quad f_m(\bar{x}_1) \leq f_m(\bar{x}_2) \quad \forall m \in \{1, 2, \dots, N_{obj}\} \quad (4.57)$$

and

$$2) \quad f_m(\bar{x}_1) < f_m(\bar{x}_2) \quad \exists m \in \{1, 2, \dots, N_{obj}\} \quad (4.58)$$

Solution  $\bar{x}_1$  is said to be a Pareto-optimal solution if it is not dominated by any other solution in the solution space.

Evolutionary algorithms such as genetic algorithms (GAs) are known to be suitable for solving multi-objective optimization problems [35]. In this chapter, an MNSGA-II is proposed to solve the SRCMS problem. To this aim, in Section 4.5.1, the modified non-dominated sorting procedure is explained and in Section 4.5.2, the steps of the proposed MNSGA-II are explained.

#### ***4.5.1. The Modified Non-Dominated Sorting Procedure***

This sorting approach sorts the population based on the non-dominated fronts that they belong to. The set of solutions that are not dominated by any other solution forms the Pareto-optimal front, which is also referred to simply as the front one,  $F_1$ . To find the solutions that belong to other fronts, it should be noted that some solutions are only dominated by Pareto-optimal solutions. Hence, if the  $F_1$  solutions are assumed to be removed from the population, another set of solutions can be found that are not dominated

by any solution, and hence termed front two,  $F_2$ . Likewise, further removing the solutions that belong to  $F_2$ , another set of non-dominated solutions can be found that are in front three,  $F_3$ , and so forth.

The concept of crowding distance, first introduced by Deb et al. [36] is to ensure that not only the algorithm converges to the Pareto-optimal front but also it obtains a diverse set of solutions [36]. This concept demonstrates and quantifies the density of solutions around a particular one in the same non-dominated front. It calculates the distance of two neighbors of a solution. This distance is computed separately for each objective. Then, the average of the distances for all objectives is calculated to evaluate the crowding distance.

To calculate the crowding distance, first, all the solutions are sorted in ascending order according to the values of each respective objective function. Second, for each solution, the absolute distances of two solutions on either side of that solution are calculated and are divided by the distance between the minimum and the maximum solutions of that objective function. The crowding distance of the individuals in a non-dominated front that have the minimum/maximum value along any of the objective functions is set to be infinite.

$$CD_i^m = \frac{|f_{i+1}^m - f_{i-1}^m|}{f_{max}^m - f_{min}^m} \quad (4.59)$$

$$CD_i = 1/m \sum_m CD_i^m \quad (4.60)$$

where  $CD_i^m$  is the crowding distance of  $i^{\text{th}}$  individual along with the  $m^{\text{th}}$  objective,  $CD_i$  is the crowding distance of individual  $i$ ,  $f_i^m$  is the  $m^{\text{th}}$  objective value of the  $i^{\text{th}}$  individual, and  $f_{max}^m$  and  $f_{min}^m$  are the maximum and minimum values of the  $m^{\text{th}}$  objective, respectively.

After sorting the population using the crowding distance, the excess population of a front is truncated all at once. To clarify, in Fig 4, let us say there are three individuals that should be removed. Solutions D, E, and F have the lowest values of crowding distance and will be deleted simultaneously. Hence, after this truncation, there are no solutions between points C and G which is not the desired result. To overcome this issue, Luo et al. [37] proposed a dynamic crowding distance strategy. This way, the excess individuals are removed one at a time and after each elimination, the crowding distances of the remaining population are recalculated.

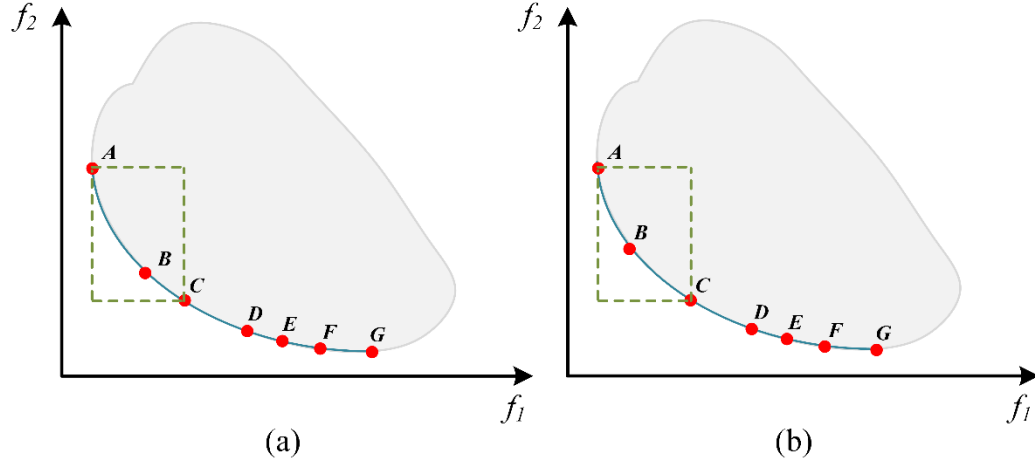


Figure 4.4. Illustration of the drawbacks of the original crowding distance method

Despite how useful the concept of crowding distance is it provides limited information on the uniformity of solutions on a front. To further explain, in Figure 4.4 the crowding distance of solution B for both cases (a) and (b) are the same. However, in Figure 4.4 (a), the solution B is very close to solution C and in terms of uniformity, the position of B in Figure 4.4 (b) is preferred. To cope with this limitation, a new method for calculating the crowding distance is presented that takes into account the distance of a solution to each of the neighboring solutions on either side of it. The average of these two distances, which is the crowding distance of that solution, and their variance are calculated. The lower the variance, the better the solution is in terms of uniformity. For the presented two-objective problem, Equations (4.61)-(4.64) are proposed to calculate the crowding distance.

$$D_{i,i+1} = \sqrt{\left(\frac{|f_{i+1}^1 - f_i^1|}{f_{max}^1 - f_{min}^1}\right)^2 + \left(\frac{|f_{i+1}^2 - f_i^2|}{f_{max}^2 - f_{min}^2}\right)^2} \quad (4.61)$$

$$D_{i-1,i} = \sqrt{\left(\frac{|f_i^1 - f_{i-1}^1|}{f_{max}^1 - f_{min}^1}\right)^2 + \left(\frac{|f_i^2 - f_{i-1}^2|}{f_{max}^2 - f_{min}^2}\right)^2} \quad (4.62)$$

$$DCD_i = 1/2 (D_{i-1,i} + D_{i,i+1}) \quad (4.63)$$

$$V_i^{CD} = 1/2 [(D_{i,i+1} - DCD_i)^2 + (D_{i-1,i} - DCD_i)^2] \quad (4.64)$$

where,  $D_{i-1,i}$  and  $D_{i,i+1}$  are the Euclidean distances between each individual and its neighboring solutions when the solutions are sorted based on one of the objectives, the

$DCD_i$  is the dynamic crowding distance (DCD) of individual  $i$ , and the  $V_i^{CD}$  is the variance of crowding distances of individual  $i$ .

In the proposed sorting procedure, the chromosomes are initially sorted according to their front. Then the solutions in each front are sorted ascendingly based on their crowding distance value. If the crowding distances (rounded to two decimal places) of two chromosomes in the same front are equal, the one that has a lower variance of crowding distance is preferred. The procedure of the non-dominated sorting is described in Algorithm 1.

---

**Algorithm 1.** Modified Non-Dominated Sorting Procedure

---

**Input:**  $Pop, N_{pop}$

**Output:** Sorted population,  $Pop^{sorted}$ , the list of solutions that belong to each front,  $F$ .

```

1: [ $Pop, F$ ]  $\leftarrow$  DetermineFront( $Pop$ )
2: [ $Pop.DCD, Pop.V^{CD}$ ]  $\leftarrow$  CalculateDCD( $Pop, F$ )
3:  $Pop^{sorted} \leftarrow$  EmptyPopulationSize $N_{pop}$ 
4: for  $i = 1$  to NumberOfFronts do
5:      $Pop(F\{i\}) \leftarrow$  AscendingSort_BasedOnDCD( $Pop(F\{i\})$ )
6:     for  $j = 1$  to  $|F\{i\}|-1$  do
7:         for  $k = j + 1$  to  $|F\{i\}|$  do
8:             if  $Pop(F\{i\})(j).DCD = Pop(F\{i\})(k).DCD$  then
9:                 if  $Pop(F\{i\})(k).V^{CD} < Pop(F\{i\})(j).V^{CD}$  then
10:                     $temp = Pop^{sorted}(k)$ 
11:                     $Pop^{sorted}(k) = Pop^{sorted}(j)$ 
12:                     $Pop^{sorted}(j) = temp$ 
13:                end if
14:            end if
15:        end for
16:    end for
17:     $Pop^{sorted} \leftarrow Pop^{sorted} \cup Pop(F\{i\})$ 
18: end for

```

---

#### 4.5.2. *Implementation of Modified Non-Dominated Sorting Genetic Algorithm (MNSGA-II)*

The proposed MNSGA-II is the improved version of the original fast non-dominated sorting genetic algorithm (NSGA-II) introduced by Deb et al. [36]. The steps of the MNSGA-II are as follows:

The parameters of the proposed MNSGA-II algorithm are set; the initial parent population,  $Pop$ , of size  $N_{pop}$  is generated randomly; the fitness of each individual in the population (chromosome) is evaluated according to each objective function; and all the chromosomes are sorted based on the proposed non-dominated sorting procedure (lines 1-3). Then, the main loop of the proposed MNSGA-II is repeated until the stopping criteria is met. The termination criterion is to reach the maximum number of iterations,  $MaxIt$ .

In each iteration, some individuals in the population are selected as parents to generate the offspring population using crossover and mutation operators. For crossover,  $pc$  percentage of the population is selected as parent chromosomes using the binary tournament selection strategy and is pair-wisely combined to generate offspring chromosomes,  $Pop_{cross}$  (line 5). To generate offspring chromosomes using the mutation operation,  $Pop_{mute}$ , the selection of the parent chromosomes which are the  $pm$  percentage of the population is done randomly (line 6). When the offspring are generated, they are merged with the rest of the population and their fitness are evaluated (lines 7 and 8) and the duplicate solutions are removed from the population (line 9).

Since the size of the new population could expand more than  $N_{pop}$ , excess individuals in the new population should be removed. Hence, the new population is ranked using the modified non-dominated sorting procedure and the truncation process is done one at a time until the size of the population decreases to  $N_{pop}$  in a way that each time the last individual of the population is removed from the population, the solutions are sorted again using the non-dominated sorting procedure (lines 10-13). Next, the Pareto-optimal solutions of the population in that iteration are improved using an improvement algorithm (line 14) and then the population is sorted again (line 15).

---

**Algorithm 2.** Procedure of MNSGA-II

---

**Input:**  $N_{pop}, MaxIt, pc, pm$ **Output:** The Pareto-optimal solutions  $Pop(F\{1\})$ 

```
1:  $Pop.Sol \leftarrow InitializePop(N_{pop})$ 
2:  $Pop.Cost \leftarrow CalculateCost(Pop)$ 
3:  $[Pop, F] \leftarrow NonDominatedSorting(Pop)(based\ on\ Algorithm\ 1)$ 
4: for  $i = 1$  to  $MaxIt$  do
5:      $Pop_{cross} \leftarrow Crossover(Pop, pc)$ 
6:      $Pop_{mute} \leftarrow Mutate(Pop, pm)$ 
7:      $Pop \leftarrow Pop \cup Pop_{cross} \cup Pop_{mute}$ 
8:      $Pop.Cost \leftarrow CalculateCost(Pop)$ 
9:      $Pop \leftarrow RemoveDuplicateSolutions(Pop)$ 
10:    while  $|Pop| > N_{pop}$  do
11:         $[Pop, F] \leftarrow NonDominatedSorting(Pop)(based\ on\ Algorithm\ 1)$ 
12:         $Pop \leftarrow Remove(Pop(end))$ 
13:    end while
14:     $Pop(F\{1\}) \leftarrow ImproveSolutions(Pop(F\{1\}))$ 
15:     $[Pop, F] \leftarrow NonDominatedSorting(Pop)(based\ on\ Algorithm\ 1)$ 
16: end for
```

---

#### 4.5.2.1. Chromosome Representation

Solution representation is one of the important issues that affect the performance of metaheuristics. In GAs a solution is represented by a chromosome. A chromosome of the proposed MNSGA-II contains genes that all take continuous random numbers between  $[0, 1]$ . Figure 4.5 depicts the proposed chromosome and its corresponding first-period layout for a problem with ten machines ( $N = 10$ ) and three periods ( $T = 3$ ). As shown in Fig 5, the chromosome consists of seven segments. The segment “A” determines the number of cells to be generated,  $N_{cell}$ . If this number is between  $[0, 1/K]$ , where  $K$  is the maximum number of cells, one cell is created,  $N_{cell} = 1$ ; if it is between  $(1/K, 2/K]$ , two

cells are created,  $N_{cell} = 2$ , and so forth. Taking into account that the maximum number of cells is equal to the number of machines,  $K = N$ , in the example chromosome of Figure 4.5, as the value 0.26 *lays in the interval*  $(\frac{2}{10}, \frac{3}{10}]$ , three cells are generated and hence  $N_{cell} = 3$ . Segments “B” and “C” each consist of  $i = 1, 2, \dots, N$  genes, which show the  $\hat{x}$  and  $\hat{y}$  of the centroid of machines, respectively. According to the values of  $\hat{x}$  and  $\hat{y}$  and Equations (4.65)-(4.71) the  $x$ - and  $y$ -coordinates of the centroid of machines are obtained.

$$x_i^M = x_i^{M,min} + (x_i^{M,max} - x_i^{M,min})\hat{x}_i \quad \forall i \quad (4.65)$$

$$y_i^M = y_i^{M,min} + (y_i^{M,max} - y_i^{M,min})\hat{y}_i \quad \forall i \quad (4.66)$$

$$x_i^{M,min} = \frac{lx_i^M}{2} + \delta^M \quad \forall i \quad (4.67)$$

$$x_i^{M,max} = L - \frac{lx_i^M}{2} - \delta^M \quad \forall i \quad (4.68)$$

$$y_i^{M,min} = \frac{ly_i^M}{2} + \delta^M \quad \forall i \quad (4.69)$$

$$y_i^{M,max} = W - \frac{ly_i^M}{2} - \delta^M \quad \forall i \quad (4.70)$$

$$0 \leq \hat{x}_i, \hat{y}_i \leq 1 \quad \forall i \quad (4.71)$$

where, the  $x_i^{M,min}$  and  $x_i^{M,max}$  are the minimum and maximum values that the  $x$ -coordinate of machine  $i$  can take, and the  $y_i^{M,min}$  and  $y_i^{M,max}$  are the minimum and maximum values that the  $y$ -coordinate of machine  $i$  can accept, respectively.

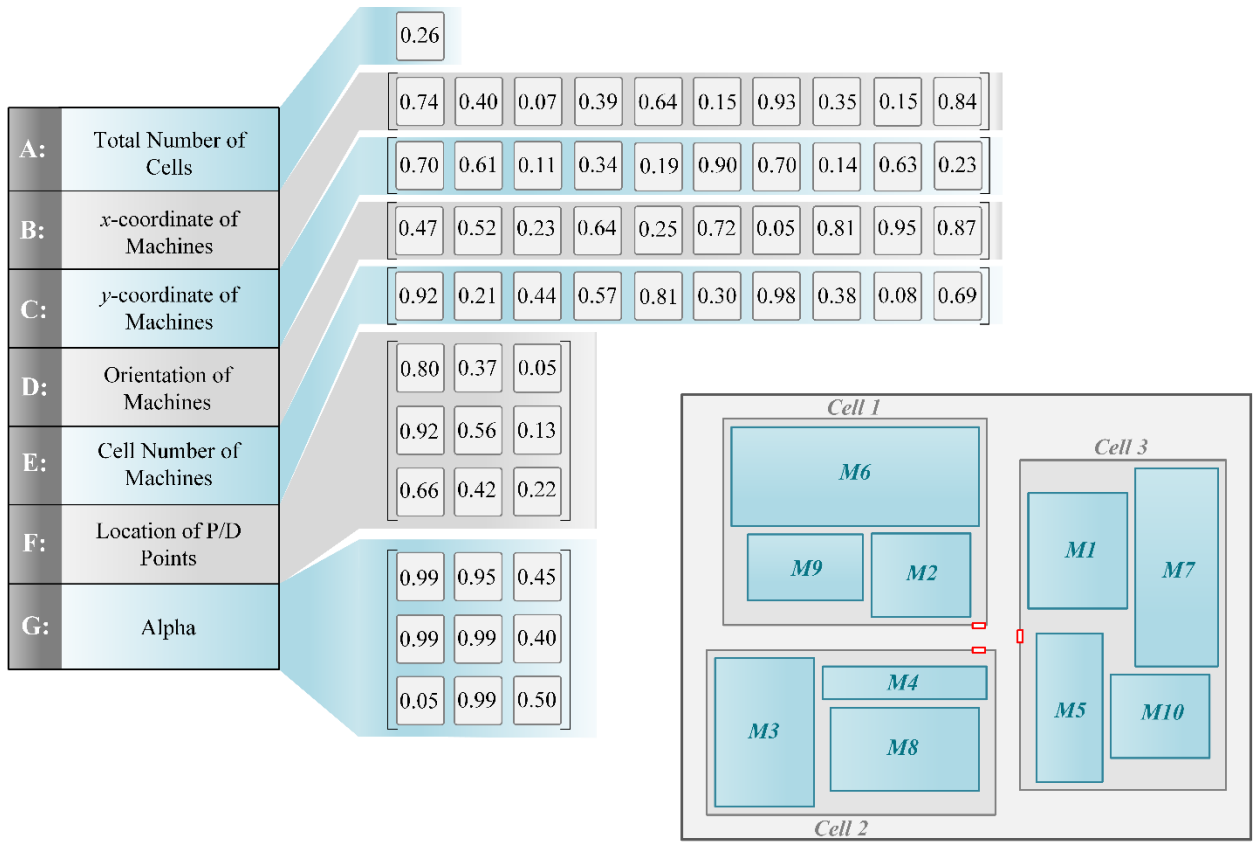


Figure 4.5. An example of a chromosome and its corresponding layout for first period

The segment “D” is used to determine the orientation of machines. It consists of  $i = 1, 2, \dots, N$  genes each of which corresponds to an orientation of a machine. If the value of a gene  $i$  is between  $[0, 0.5]$ , the machine  $i$  takes a vertical orientation; otherwise, the machine  $i$  takes a horizontal orientation.

The segment “E” determines how machines are assigned to cells. It consists of  $i = 1, 2, \dots, N$  genes where gene  $i$  determines the cell number of machine  $i$ . Machine  $i$  is assigned to cell 1 if gene  $i$  takes a value between  $\left[0, \frac{1}{N_{cell}}\right]$ , where  $N_{cell}$  is the number of cells that has already been obtained in segment “A”. In a similar manner, machine  $i$  is assigned to cell 2 if its corresponding gene  $i$  take a value between  $\left(\frac{1}{N_{cell}}, \frac{2}{N_{cell}}\right]$ , and so forth. For example, in Figure 4.5, the value of gene  $i = 1$  is 0.97 and given segment “A”



it is known that three cells are generated ( $N_{cell} = 3$ ). As 0.97 is between  $(\frac{2}{3}, \frac{3}{3}]$ , machine 1 is assigned to cell 3.

Segments “F” and “G” are responsible to determine the positions of the P/D points. Since the positions of the P/D points can change from one period to the other, these segments make  $T \times N_{cell}$  matrices where  $T$  is the number of periods and  $N_{cell}$  is the number of created cells.

Segment “F” specify which side of the cell is selected for locating the P/D point in each period. The values in the range  $[0, 0.25]$ ,  $(0.25, 0.5]$ ,  $(0.5, 0.75]$  or  $(0.75, 1]$  indicate that the P/D point is located on the left, upper, right, or bottom side of the cell, respectively. For instance, in Figure 4.5, the corresponding gene for cell 1 in period 1 is 0.80. As this value is in the range  $(0.75, 1]$ , the P/D point of cell 1 in period 1 is located on its bottom side. Through segment “G” the exact position of P/D point on the selected side of the cell is determined. Each side is graded from zero to one from the left corner to the right or from the bottom corner to the top. In this way, for instance, the value 0.5 indicates that the P/D point is located in the middle of the side.

#### *4.5.2.2. Binary Tournament Selection*

In the binary tournament selection, a pair of chromosomes are chosen at random from the population and are compared in terms of their rank, the value of the crowding distance (rounded to two decimal places), and variance of crowding distance. If they belong to different fronts, the one which is in a lower rank is preferred. If both belong to the same non-dominated front, the solution that has a higher value of DCD is selected. If the value of DCD is the same for both, then the one that has the lower value of  $V^{CD}$  is deemed better.

#### *4.5.2.3. Crossover Operator*

The crossover operator generates two new chromosomes (offspring) with new characteristics being inherited from the two chromosomes of the initial population (their parents). There are different types of crossover (single-point, double-point, arithmetic, and uniform) that can be used in the MNSGA-II algorithm. In this study, two different crossover operators are used with equal probability.

The first crossover is an arithmetic crossover for all the fields of the chromosome. Using this crossover operator, two parent chromosomes are selected by the binary tournament selection strategy and are linearly combined to generate two new offspring. For instance, using Equations (4.72) and (4.73) the arithmetic crossover is applied on the  $x$ -coordinate ( $\hat{x}$ ) of the machines of two parents to obtain the  $x$ -coordinate ( $\hat{x}$ ) of the machines of offspring, where  $r$  is a random number between zero and one.

$$\hat{x}_{new1} = r\hat{x}_{parent1} + (1 - r)\hat{x}_{parent2} \quad (4.72)$$

$$\hat{x}_{new2} = (1 - r)\hat{x}_{parent1} + r\hat{x}_{parent2} \quad (4.73)$$

A single-point crossover is applied to  $x$  and  $y$ -coordinates ( $\hat{x}, \hat{y}$ ), orientation, and the cell number of machines and an arithmetic crossover for  $\alpha$ , and the location of the P/D points.

#### 4.5.2.4. Mutation Operator

The mutation is used to maintain diversity in individuals of the population and to prevent premature convergence of the algorithm. To generate a new chromosome (offspring) from a parent chromosome in the proposed MNSGA-II, two numbers are selected randomly between 1 to  $N$ . Then the elements of the parent vector that are between these two numbers are reversed. The mutation is applied only to the  $x$  and  $y$ -coordinates of machines ( $\hat{x}, \hat{y}$ ), the orientation of machines, and the cell number of machines that correspond to the segments B, C, D and E of the chromosome, respectively.

#### 4.5.2.5. Improvement Algorithm

The improvement algorithm is applied to the Pareto front chromosomes. It implements a set of minor changes in the position of machines and makes them closer to each other to create a more coherent layout. These minor changes do not affect the CF and hence, the second objective remains unchanged. However, these changes lead to a decrease in MHC. In the first step of the improvement algorithm, the machines are randomly selected one at a time. In the second step, the position of the selected machine is slightly changed by moving it in four main directions. After each of the four displacements is applied, the MHC is calculated and the one that leads the best MHC is executed. Next, the orientation of the machine is changed and in a similar manner, the best orientation for each machine is selected. After improving the position and orientation of machines, the cells are selected

randomly one at a time and the position of the P/D point of the cell is slightly changed in both directions alongside the side it is located; then, the change that leads to the best MHC is applied.

#### ***4.6. Computational Results***

In this section, a set of computational experiments is conducted to evaluate the effectiveness of the proposed method. To this aim, in Section 4.6.1, a set of data from literature is taken into account. Section 4.6.2. describes how the parameters of the algorithms are tuned. Metrics to evaluate the algorithms are enumerated in Section 4.6.3. In Section 4.6.4, the efficiency of the proposed MNSGA-II is evaluated. As there is no appropriate benchmark available to validate the performance of the proposed MNSGA-II, two other popular metaheuristic algorithms, namely non-dominated ranking genetic algorithm (NRGA) and multi-objective particle swarm optimization algorithm (MOPSO), are used. These algorithms are used because they are known as efficient population-based algorithms for multi-objective optimization problems. The solution structure and function evaluation procedure in both algorithms are akin to the ones in the proposed MNSGA-II. The details of the implemented NRGA are available in Mousavi et al. [38], and details of MOPSO are in Nemati-Lafmejani et al. [39]. It is noteworthy to note that the parameter tuning was done using Design Expert 7 and the algorithms were coded using MATLAB software (release 2018a) and executed on a computer with core (TM) i5-CPU 2.40 GHz, RAM 4.00 GB.

##### ***4.6.1. Data Set***

This section provides the details of the data sets used to test the proposed approach for solving the SRCMS. The data sets which are taken from the literature include the dimensions of the machines, the demand, and the sequence of the operations of the products. Since these data sets were produced for single period problems and demands for other periods are not available, the demand is randomly generated using the uniform distribution (see Table 4.3-4.7). Other missing data such as the length and width of the shop floor and the inter- and intra-cellular MHC are made available in Table 4.2. As shown

in Table 4.2, there are ten different problem sizes according to the number of machines ( $M$ ), periods ( $T$ ) and products ( $P$ ). For each size, three test problems are generated by changing the length and width of the shop floor and inter- and intra-cellular MHC. The shop floor (*i.e.*, its length and width) is constructed in a way such that the total area of the machines constitutes 85% of the area of the shop floor. For all test problems, a clearance distance of one meter is considered between machines and three meters between cells.

Table 4.2. The input data for test problems

Problem		Parameters added to each problem					
No	Source	Size ( $M \times T \times P$ )	Demand	Shop floor (m)		Cost(unit per meter)	
				Length	Width	Inter-cellular	Intra-cellular
1	Souilah [40]	8×3×17	[Table 4.3]	101	60	20	32
2		8×3×17	[Table 4.3]	100	77	11	16
3		8×3×17	[Table 4.3]	111	77	20	29
4		8×6×17	[Table 4.3]	112	84	10	20
5		8×6×17	[Table 4.3]	87	96	19	30
6		8×6×17	[Table 4.3]	71	77	9	15
7		Mohammadi and Forghani [4]	8×3×20	[Table 4.4]	28	22	5
8	8×3×20		[Table 4.4]	22	24	10	15
9	8×3×20		[Table 4.4]	22	20	12	20
10		8×6×20	[Table 4.4]	24	16	12	15
11		8×6×20	[Table 4.4]	20	24	17	25
12		8×6×20	[Table 4.4]	23	20	19	28
13		Feng et al. [41]	12×3×15	[Table 4.5]	28	29	5
14	12×3×15		[Table 4.5]	24	27	8	16
15	12×3×15		[Table 4.5]	28	32	5	10
16		12×6×15	[Table 4.5]	25	32	14	25
17		12×6×15	[Table 4.5]	25	24	11	20
18		12×6×15	[Table 4.5]	39	27	10	15
19		Allahyari and Azab [28]	16×3×7	[Table 4.6]	97	131	15
20	16×3×7		[Table 4.6]	81	159	17	25
21	16×3×7		[Table 4.6]	68	71	10	18
22		16×6×7	[Table 4.6]	75	95	20	28
23		16×6×7	[Table 4.6]	61	63	21	35
24		16×6×7	[Table 4.6]	77	108	18	30
25		Mohammadi and Forghani [4]	25×3×40	[Table 4.7]	96	207	25
26	25×3×40		[Table 4.7]	82	58	18	35
27	25×3×40		[Table 4.7]	45	72	8	15
28		25×6×40	[Table 4.7]	56	53	10	15
29		25×6×40	[Table 4.7]	95	412	19	25
30		25×6×40	[Table 4.7]	104	98	20	30

Table 4.3. Product demand for problems with 17 products [40]

Product	Period						Product	Period					
	1	2	3	4	5	6		1	2	3	4	5	6
1	5	11	12	7	10	6	10	18	24	23	16	20	17
2	11	15	13	5	13	7	11	18	12	10	14	15	14
3	17	17	16	21	22	19	12	12	11	11	11	6	7
4	45	36	35	43	52	34	13	1	2	2	1	2	1
5	6	12	5	12	11	5	14	10	7	3	5	11	12
6	10	4	10	9	10	2	15	5	2	5	7	9	8
7	24	26	40	28	38	40	16	57	50	43	51	54	56
8	93	101	93	93	85	105	17	7	7	2	7	7	2
9	8	8	12	13	10	6							

Table 4.4. Product demand for problems with 20 products [4]

Product	Period						Product	Period					
	1	2	3	4	5	6		1	2	3	4	5	6
1	83	82	95	85	89	125	11	92	93	96	92	76	122
2	112	96	105	82	72	127	12	90	122	93	121	126	85
3	94	91	93	84	92	100	13	103	72	83	105	88	118
4	105	89	129	122	124	106	14	106	90	83	88	121	115
5	73	85	122	78	91	112	15	118	110	105	76	103	73
6	106	110	123	84	122	118	16	77	129	130	90	78	125
7	117	71	111	73	76	98	17	74	71	97	112	91	118
8	92	117	89	91	128	118	18	121	121	105	110	128	108
9	79	117	87	103	124	78	19	73	99	89	110	97	106
10	87	76	88	114	85	111	20	101	103	121	79	71	115

Table 4.5. Product demand for problems with 15 products [41]

Product	Period						Product	Period					
	1	2	3	4	5	6		1	2	3	4	5	6
1	47	33	31	37	34	40	9	40	32	43	38	26	29
2	37	43	35	54	51	43	10	48	44	34	40	33	39
3	49	33	47	47	35	45	11	45	54	50	58	55	54
4	27	25	27	31	30	35	12	26	35	27	21	28	30
5	29	28	40	28	33	27	13	43	39	42	41	42	40
6	29	26	45	31	27	43	14	27	43	27	35	28	34
7	29	32	28	42	37	39	15	30	39	32	28	33	20
8	40	30	26	25	27	35							

Table 4.6. Product demand for problems with 7 products [28]

Product	Period					
	1	2	3	4	5	6
1	189	199	236	189	199	236
2	377	386	406	377	386	406
3	466	518	544	466	518	544
4	256	252	221	256	252	221
5	230	240	151	230	240	151
6	176	153	194	176	153	194
7	225	221	202	225	221	202

Table 4.7. Product demand for problems with 40 products [4]

Product	Period						Product	Period					
	1	2	3	4	5	6		1	2	3	4	5	6
1	155	147	176	156	161	133	21	75	88	90	94	99	114
2	160	176	206	205	183	152	22	100	86	79	89	75	60
3	135	117	135	111	117	116	23	140	151	145	139	126	132
4	150	127	150	179	204	177	24	62	57	58	67	79	93
5	210	236	212	246	282	300	25	85	102	96	77	82	96
6	230	229	190	215	255	263	26	185	202	183	163	152	150
7	85	72	75	63	65	66	27	55	52	56	58	69	65
8	90	100	97	115	98	111	28	130	115	125	147	138	153
9	95	107	110	108	115	117	29	125	113	131	106	114	97
10	86	99	109	102	102	114	30	135	149	130	134	113	106
11	55	56	66	70	60	71	31	65	55	49	49	42	41
12	120	102	118	107	86	72	32	90	104	94	102	100	108
13	142	138	139	156	169	164	33	100	109	88	101	116	117
14	140	158	132	143	148	172	34	90	95	92	91	80	79
15	100	110	115	116	99	96	35	120	138	165	191	178	174
16	65	62	58	65	69	77	36	130	133	153	143	126	106
17	85	86	78	69	82	87	37	145	138	163	140	131	131
18	125	148	161	137	129	118	38	250	276	287	282	282	243
19	102	93	81	65	61	69	39	60	70	56	55	63	71
20	105	126	127	140	134	142	40	90	96	113	106	90	104

#### 4.6.2. Evaluation Metrics of the Metaheuristic Algorithms

In general, the performance criteria for a multi-objective optimization algorithm determine the Intensification (*i.e.*, how close Pareto solutions are to the ideal Pareto solution) and Diversification (*i.e.*, how wide and uniform the solutions are distributed along the Pareto front) of the search. In order to measure the above performance criteria, a variety of metrics have been proposed. The pros and cons of the various performance metrics have been

discussed by Zitzler et al. [42]. Four metrics are used in this study: i) spacing, ii) mean ideal distance, iii) max spread, and iv) computational time.

#### 4.6.2.1. Spacing (SP)

The performance metric spacing (SP) that was first introduced by Schott [43] calculates the standard deviation of the rectilinear distances between each individual in the Pareto-optimal set and its closest neighbor (Equations (4.70)-(4.72)).

$$SP(S) = \sqrt{\frac{1}{|S|-1} \sum_{i=1}^{|S|} (d_i - \bar{d})^2} \quad (4.70)$$

$$d_i = \min_{\substack{s_j \in S \\ s_j \neq s_i}} \sum_{m=1}^{N_{obj}} |f_i^m - f_j^m| \quad (4.71)$$

$$\bar{d} = \frac{\sum_{i=1}^{|S|} d_i}{|S|} \quad (4.72)$$

where  $S$  is the set of Pareto-optimal solutions.

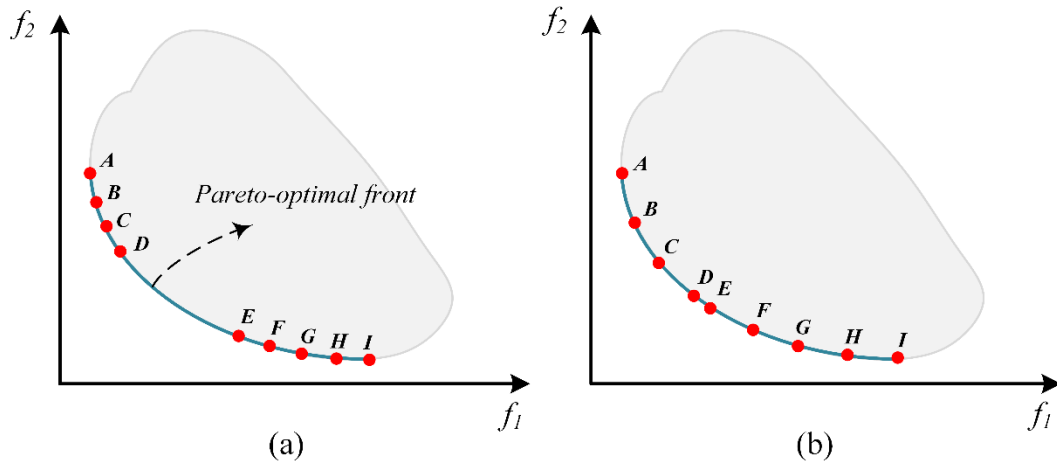


Figure 4.6. The drawback of the original SP metric

As can be seen in Figure 4.6 (a), the rectilinear distances between each individual and its closest neighbor are the same for all individuals. Hence, the SP metric is zero although the solutions are not spread on the Pareto-optimal front uniformly. On the other hand, Figure 4.6 (b) depicts better solutions in terms of uniformity but as the distance of the closest neighbors for D and E are less than the other individuals, the SP metric is greater than zero.



To overcome this issue, a modified SP metric (MSP) is presented in this section to check how evenly the solutions are spread over the Pareto front. In the modified formulation, instead of calculating the shortest distance, the Euclidean distance of each individual with its successive neighbor is calculated. This method works well when there are only two objectives; it is not possible to define neighboring solutions if there are more than two objectives.

To calculate the MSP, first, the solutions should be sorted with respect to one of the objectives and then the Euclidian distance between each individual and the one that is next to it in the sorted order is calculated. Second, the standard deviation of such distances, the MSP, is calculated as outlined in Equations (4.73)-(4.75).

$$MSP(S) = \sqrt{\frac{1}{|S|-1} \sum_{i=1}^{|S|} (D_{i,i+1} - \bar{D})^2} \quad (4.73)$$

$$D_{i,i+1} = \sqrt{\left(\frac{|f_{i+1}^1 - f_i^1|}{f_{max}^1 - f_{min}^1}\right)^2 + \left(\frac{|f_{i+1}^2 - f_i^2|}{f_{max}^2 - f_{min}^2}\right)^2} \quad (4.74)$$

$$\bar{D} = \frac{\sum_{i=1}^{|S|-1} D_{i,i+1}}{|S|-1} \quad (4.75)$$

#### 4.6.2.2. Mean Ideal Distance (MID)

The MID determines the average of distances of Pareto solutions from an ideal point (the maximum objective function value in a maximizing function or the minimum objective function value in a minimizing function). The MID index is calculated by Equation (4.76):

$$MID = \frac{\sum_{i=1}^{|S|} \sqrt{(f_1^{ideal} - f_1^i)^2 + (f_2^{ideal} - f_2^i)^2}}{|S|} \quad (4.76)$$

where  $f_1^{ideal}$  and  $f_2^{ideal}$  are the values of the first and the second objective functions of the ideal point and  $f_1^i$  and  $f_2^i$  are the values of the first and second objectives functions of solution  $i$ , respectively. According to the definition of MID, the smaller the MID the better the quality of obtained solutions. Since the objectives of the proposed SRCMS problem are to be minimized, the values of  $f_1^{ideal}$  and  $f_2^{ideal}$  are zero.

#### 4.6.2.3. Max Spread (MS)

The MS reports the spread of the Pareto solutions set and is measured by Equation (4.77). The algorithm with the higher value of MS has a better performance.

$$MS = \sqrt{(f_1^{max} - f_1^{min})^2 + (f_2^{max} - f_2^{min})^2} \quad (4.77)$$

#### 4.6.2.4. Computational Time (CPU time)

The computational time (CPU time) represents the required computational time to run an algorithm.

### 4.6.3. Calibration of Parameters

It is common knowledge that in order for a metaheuristic to achieve its potential, parameters need to be finely tuned using experimental design. To ensure a fair comparison, all of the parameters of the developed algorithms are tuned via the response surface method (RSM). To this aim, the parameters of the algorithms and their levels are determined based on preliminary experiments. These parameter levels are shown in Table 4.8. Then,  $2^k$  experiments are designed and implemented. Finally, regression analysis is applied to provide the functional relationship between the algorithms parameters and the response variable. For more information on our implemented experiments, see the article by Najafi et al. [44]. We consider the multi-objective coefficient of variation (MOCV) metric as a response variable of each experiment. MOCV, presented by [45], can support algorithm intensification through the inclusion of an MID metric and algorithm diversification through MS metric (Equation 4.78). The lower value of MOCV is desirable.

$$MOCV = \frac{MID}{MS} \quad (4.78)$$

Table 4.8. The parameters of the algorithms and their optimum values

Algorithm	Parameter	Description	Low level (-1)	High level (+1)	Optimum
MNSGA-II	$N_{pop}$	Initial pop size	100	300	<b>291</b>
	$pc$	Percent of crossover	0.75	0.85	<b>0.77</b>
	$pm$	Percent of mutation	0.03	0.08	<b>0.03</b>
	$MaxIt$	Number of iteration	50	150	<b>124</b>
NRGA	$N_{pop}$	Initial pop size	100	300	<b>300</b>
	$pc$	Percent of crossover	0.75	0.85	<b>0.84</b>
	$pm$	Percent of mutation	0.03	0.08	<b>0.07</b>
	$MaxIt$	Number of iteration	50	150	<b>123</b>
MOPSO	$N_{pop}$	Initial pop size	200	400	<b>262</b>
	$C_1$	Cognitive acceleration coefficient	1	2	<b>1</b>
	$C_2$	Social acceleration coefficient	1	2	<b>2</b>
	$W$	Inertia weight	0.5	0.99	<b>0.77</b>
	$N_{rep}$	Number of repositories	100	200	<b>102</b>
	$N_{grid}$	Number of adaptive grid	4	10	<b>6</b>
	$MaxIt$	Number of iteration	300	500	<b>498</b>

After implementing the experiments, the quadratic equations are developed using a stepwise regression method ( $\alpha = 0.1$ ) to show the relationship between the parameters of the algorithm and MOCV.

$$\begin{aligned}
 MOCV_{NSGA-II} = & 23.38782 - 4.89478E - 003 \times Iteration - 0.21881 & (4.79) \\
 & \times Popsiz e - 17.50400 \times Pc + 33.75406 \times Pm + 0.13577 \\
 & \times Popsiz e \times Pc + 0.12243 \times Popsiz e \times Pm + 3.98433E \\
 & - 004 \times Popsiz e^2
 \end{aligned}$$

$$\begin{aligned}
 MOCV_{NRGA} = & 11.36890 + 1.89622E - 003 \times Popsiz e - 0.055968 & (4.80) \\
 & \times Iteration - 1.28201 \times Pc - 30.52260 \times Pm - 4.04264E \\
 & - 003 \times Popsiz e \times Pc + 2.25214E - 004 \times Iteration^2 \\
 & + 112.05599 \times Pm^2
 \end{aligned}$$

$$\begin{aligned}
MOCV_{MOPSO} = & 40.38886 - 0.076039 \times Popsiz e - 1.42919 \times C1 & (4.81) \\
& - 20.56748 \times C2 + 4.34297 \times W - 0.017938 \times Nrep \\
& - 0.73296 \times Ngrid - 5.42501E - 003 \times Iteration \\
& + 1.01410 \times C1 \times C2 + 0.023116 \times C2 \times Nrep + 0.47581 \\
& \times C2 \times Ngrid - 0.020118 \times W \times Nrep + 1.23867E - 004 \\
& \times Popsiz e^2 + 3.02861 \times C2^2
\end{aligned}$$

According to the analysis of variance, shown in Tables 4.9-4.11, it can be said that the regression functions can properly estimate the response variable. By optimizing the Equations (4.79)-(4.81), the optimum values for each algorithm are obtained. The optimum value for each factor of the metaheuristics is highlighted in Table 4.8.

Table 4.9. ANOVA results of the regression model

	Source	Sum of squares	Degrees of freedom	Mean square	F-Test	P-value
MNSGA-II	Regression	63.608	7	9.086	22.021	< 0.0001
	Residual error	9.077	22	0.412		
	Total	72.685	29			
	$R^2 = 87.51\%$ , $R^2(adj) = 83.54\%$ , $R^2(pred) = 70.93\%$					
NRGA	Regression	12.349	7	1.764	21.498	< 0.0001
	Residual error	1.805	22	0.082		
	Total	14.154	29			
	$R^2 = 0.8725\%$ , $R^2(adj) = 83.19\%$ , $R^2(pred) = 70.54\%$					
MOPSO	Regression	157.371	13	12.105	20.636	< 0.0001
	Residual error	21.117	36	0.5865		
	Total	178.488	49			
	$R^2 = 88.17\%$ , $R^2(adj) = 83.90\%$ , $R^2(pred) = 76.84\%$					

#### ***4.6.4. Evaluating the Proposed Metaheuristic***

In this section, the MNSGA-II is compared with two other popular algorithms, namely, MOPSO and NPGA. Table 4.10 shows the values of the four performance measures; the best values are boldfaced. From the results exhibited in Table 4.10, it is obvious that MNSGA-II surpasses the other algorithms in regard to the SP metric almost in all experimental problems, especially for large-sized instances. On the other hand, NPGA generally performed better than MOPSO in SP. As for the MS metric, MNSGA-II and MOPSO have better performance than NPGA. In terms of MID, MNSGA-II and NPGA carry out better than MOPSO. Finally, MOPSO performs far better than the two other algorithms in terms of computational time.

Table 4.10. Results of the algorithms for all the test problems

	MNSGA-II				NRGA				MOPSO			
	MS	SM	MID	Time (s)	MS	SM	MID	Time (s)	MS	SM	MID	Time (s)
1	1024147	<b>0.165</b>	<b>1544768</b>	688	<b>1052484</b>	0.179	1756184	683	665382	0.335	1668132	<b>484</b>
2	<b>821959</b>	<b>0.137</b>	980928	714	321561	0.142	<b>956398</b>	598	172958	0.240	1094890	<b>460</b>
3	<b>2038273</b>	0.191	<b>1670227</b>	772	712526	<b>0.179</b>	1755753	739	427367	0.375	2013282	<b>469</b>
4	1655830	<b>0.131</b>	<b>2104390</b>	922	1369679	0.193	2219112	1048	<b>3484697</b>	0.225	2725490	<b>588</b>
5	2467344	0.153	3599694	850	1984163	<b>0.132</b>	<b>3537936</b>	1034	<b>4152602</b>	0.344	3978479	<b>612</b>
6	<b>2100129</b>	<b>0.093</b>	1743029	1007	1234466	0.206	<b>1678296</b>	1151	412526	0.166	2001310	<b>685</b>
7	424302	<b>0.112</b>	921410	772	327864	0.141	<b>879007</b>	728	<b>662238</b>	0.322	901350	<b>483</b>
8	<b>1141433</b>	<b>0.157</b>	1583483	727	549421	0.162	<b>1557200</b>	727	256801	0.265	1989792	<b>482</b>
9	<b>727231</b>	<b>0.169</b>	2077145	771	698730	0.235	<b>1987927</b>	751	242233	0.248	2249046	<b>555</b>
10	<b>3022966</b>	<b>0.085</b>	<b>3797670</b>	974	2280413	0.116	4035371	1095	1360985	0.140	4664961	<b>763</b>
11	1343311	0.171	<b>4954910</b>	854	4208639	<b>0.151</b>	5381018	979	<b>5848025</b>	0.346	6165932	<b>790</b>
12	<b>2499547</b>	0.206	<b>6090287</b>	816	2100684	<b>0.200</b>	6137290	942	1901971	0.235	7565420	<b>717</b>
13	285543	<b>0.168</b>	<b>457761</b>	1501	166259	0.222	473655	1482	<b>353928</b>	0.280	692734	<b>1077</b>
14	<b>452282</b>	<b>0.083</b>	<b>718725</b>	3029	443045	0.094	740415	3378	386735	0.239	905171	<b>1054</b>
15	<b>262085</b>	<b>0.102</b>	<b>462506</b>	4472	245548	0.125	485505	5467	158308	0.177	575936	<b>1239</b>
16	877735	<b>0.076</b>	2668660	6209	1322725	0.102	<b>2524441</b>	8220	<b>1423896</b>	0.181	3668959	<b>1506</b>
17	1258626	<b>0.092</b>	2104559	8561	796235	0.121	<b>1926970</b>	11338	<b>3044395</b>	0.230	2399439	<b>2343</b>
18	<b>1069712</b>	<b>0.063</b>	<b>1685588</b>	10897	559692	0.090	1713831	14153	240679	0.091	1887924	<b>1419</b>
19	<b>26540934</b>	<b>0.115</b>	<b>15737103</b>	2290	7570759	0.155	17005403	2912	6478592	0.224	29941893	<b>1614</b>
20	10979148	<b>0.132</b>	19014829	2209	7475829	0.168	<b>18564290</b>	4326	<b>18674301</b>	0.237	28814720	<b>1303</b>
21	<b>9081027</b>	<b>0.146</b>	11499361	2515	3415467	0.180	<b>10930829</b>	5992	4535023	0.192	14316496	<b>1712</b>
22	13404009	<b>0.148</b>	51103053	3168	<b>18874596</b>	0.214	<b>50576704</b>	7045	17858019	0.176	60146703	<b>2725</b>
23	20227118	<b>0.104</b>	<b>50117513</b>	3730	<b>49249536</b>	0.132	51550925	9850	16231462	0.120	65588122	<b>3135</b>
24	<b>75479777</b>	<b>0.139</b>	<b>45314096</b>	4552	21925751	0.177	46463213	10289	19388846	0.282	60013970	<b>2434</b>
25	25354580	<b>0.124</b>	<b>26455610</b>	6764	25057011	0.198	32673518	8304	<b>32508482</b>	0.293	35958566	<b>4767</b>
26	14838519	<b>0.158</b>	<b>17264486</b>	6557	8229776	0.209	18002576	7840	<b>46532095</b>	0.314	35242996	<b>4128</b>
27	4936999	<b>0.117</b>	7956421	6456	4852961	0.155	<b>7706014</b>	7644	<b>15249143</b>	0.204	13758519	<b>4326</b>
28	<b>11019840</b>	<b>0.094</b>	<b>17702025</b>	11494	10658531	0.129	18779445	16152	6803162	0.228	22348198	<b>7855</b>
29	25465204	<b>0.095</b>	<b>33654478</b>	12181	<b>42115457</b>	0.119	37156843	17070	9515728	0.173	45175378	<b>7951</b>
30	22736354	<b>0.106</b>	<b>44606554</b>	13905	113252841	0.139	54967862	18729	<b>203869049</b>	0.321	80821170	<b>9718</b>

In order to determine if the various algorithms perform significantly different, a single factor ANOVA is conducted. In order to perform the ANOVA test, the results of Table 10 are initially normalized by the relative percentage deviation (RPD) as formulated in Equation (4.82).

$$RPD_{ij} = \frac{|Alg_{sol}(ij) - Best_{sol}(i)|}{Best_{sol}(i)} \times 100 \quad i = 1, \dots, n \quad (4.82)$$

where,  $RPD_{ij}$  is the RPD of algorithm  $j$  in problem  $i$ ,  $Alg_{sol}$  is the metric's value algorithm  $j$  in problem  $i$ ,  $Best_{sol}(i)$  is the best value of the metric among all algorithms, and  $n$  is the number of problems. The obtained ANOVA results are shown in Table 11. A P-value less than 0.05 indicates the rejection of the Null hypothesis (equality of the algorithms). This means that there is a significant difference between algorithms.

Table 4.11. ANOVA results for all performance metrics

	Source	Sum of Squares	Degrees of freedom	Mean Square	F-Test	P-Value
Max Spread	Algorithm	2692.2	2	1346.11	1.47	0.2355
	Error	79644.7	87	915.46		
	Total	82336.9	89			
Spacing	Algorithm	139097.6	2	69548.8	65.85	3.85E-18
	Error	91885.3	87	1056.2		
	Total	230982.9	89			
Mean Ideal Distance	Algorithm	19422.7	2	9711.36	43.92	6.51E-14
	Error	19236.2	87	221.11		
	Total	38658.9	89			
CPU Time	Algorithm	375437.4	2	187718.7	11.12	5.03E-05
	Error	1469313	87	16888.6		
	Total	1844750	89			

According to Table 11, it is clear that the algorithms have significant differences in terms of all performance metrics except for the MS metric. For a deeper investigation of the algorithms, Tukey test is utilized. The 95% Tukey simultaneous confidence intervals are calculated and are shown in Figure 4.7. According to Figure 4.7, it can be concluded that in terms of the MS metric all of the algorithms are performing similarly. However, MNSGA-II is relatively better than two other algorithms. Regarding the SP metric, it can be statistically concluded that MNSGA-II has the highest quality among all the algorithms.

The main reason for this occurrence is the modified crowding distance applied in the MNSGA-II, which made it capable of obtaining a well-distributed Pareto front. Based on the MID metric, the efficiency of the MNSGA-II and NRGAs is at the same level. MOPSO performs too weak with regard to this metric. Finally, in terms of CPU time, the MOPSO algorithm is dramatically better than the two other algorithms. Meanwhile, MNSGA-II and NRGAs are statistically at the same level.

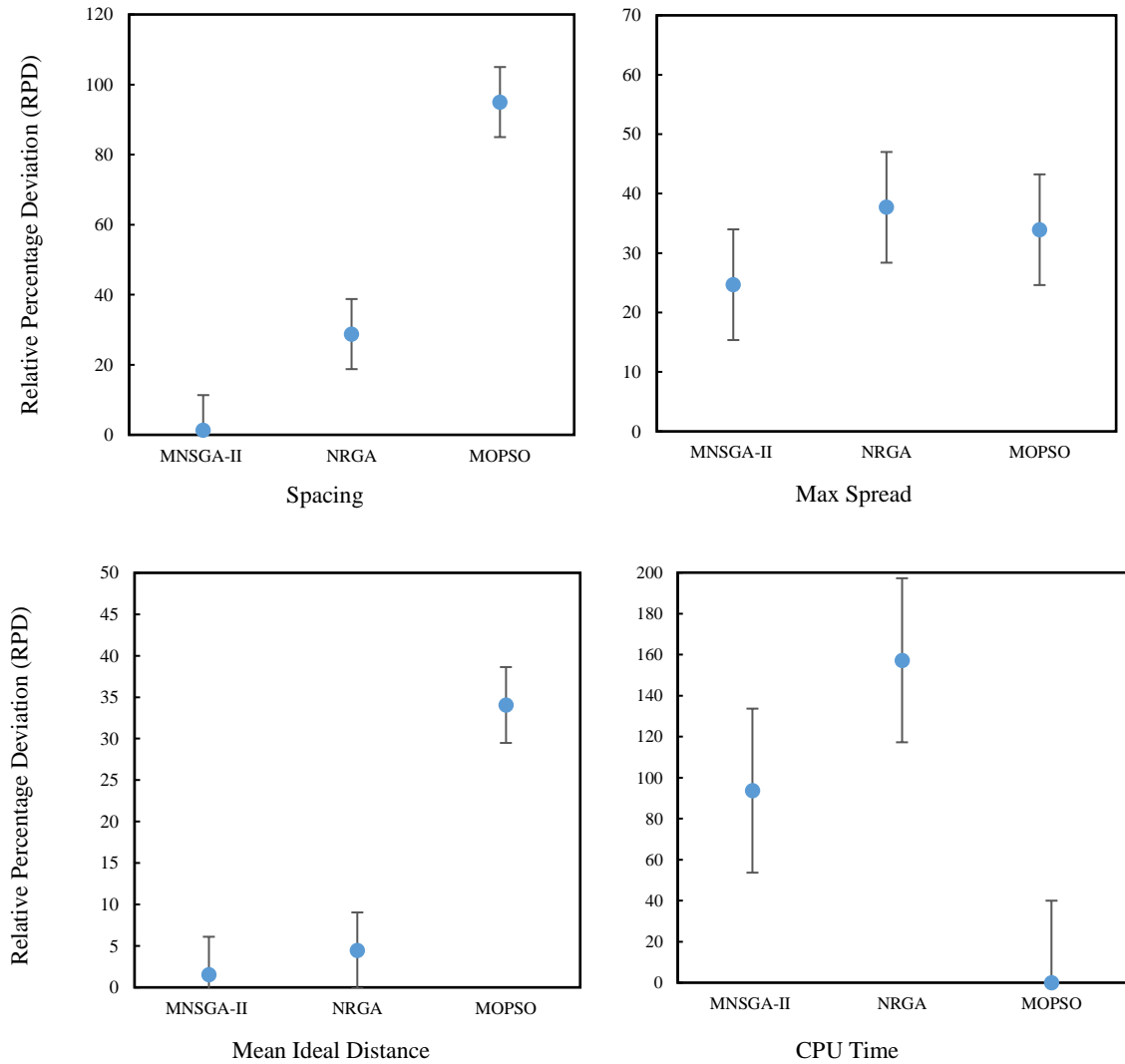


Figure 4.7. Tukey simultaneous confidence intervals of the algorithms.

Besides, problems 19 and 25 are selected as two large-sized problems that the convergence graphs of the algorithms for these problems are shown in Figs. 8 and 9, respectively. These



graphs are plotted by calculating the average objective function values of the Pareto-front solutions in each iteration. Examining Figure 4.8, it is seen MNSGA-II and NRGGA converge during the first 300 iterations to a near-to-optimal solution. By contrast, MOPSO requires around 500 iterations to converge. Since the convergence curve for MNSGA-II is lower than those of the other algorithms, it has a stronger ability to search than the other methods.

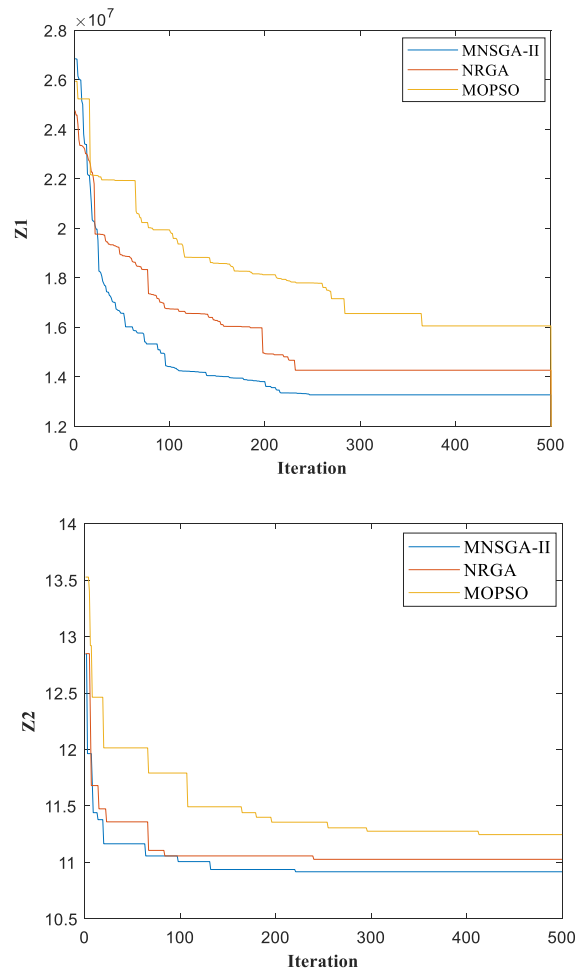


Figure 4.8. Convergence graph of the algorithms for problem 19

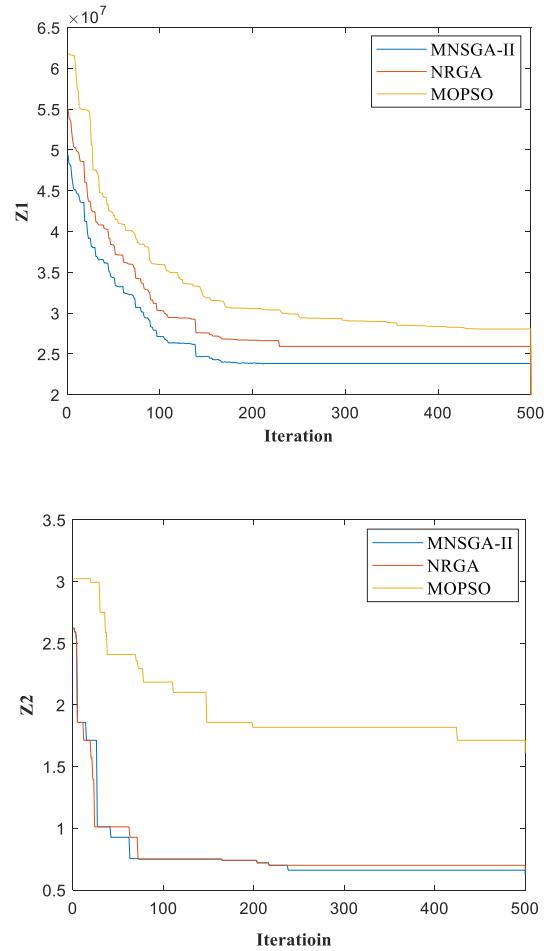


Figure 4.9. Convergence graph of the algorithms for problem 25

#### 4.7. Conclusions

In this chapter, a semi-robust approach is presented for the integrated CF and CL (inter- and intra-cellular) problems in a CMS in a dynamic environment. Higher management in the industry usually looks for robust layouts in order to avoid any interruption in production or rearrangement of facilities. The advantage of the proposed SRCMS is that although it is robust and the location and orientation of machines are fixed for all the periods, it still adapts to changes in the demand from a period to the next by accommodating the locations of the P/D points of the cells.

The problem is formulated as a multi-objective MINLP model and is solved using a modified NSGA-II. An improved modified non-dominated sorting strategy is employed in

the proposed MNSGA-II that results in a better uniformity in the obtained solutions of the Pareto–optimal front. The performance of the proposed MNSGA-II is compared, in terms of the multi-objective optimization criteria, against the two other well-known multi-objective optimization algorithms, MOPSO and NPGA. Given the obtained results, it can be concluded that the proposed MNSGA-II has better performance as far as the SP metric is concerned, which is attributed to the employed modified crowding distance used. Also, the improved sorting algorithm results in a better performance in terms of the MS and MID metrics.

For future work, the demand in each period can be considered to be stochastic. In this chapter, it is assumed that the pick-up and drop-off points of a cell both are at the same place the optimum location of which is determined by the model. Distinct pick-up and drop-off points for each cell can be considered for future work. Moreover, the material handling system can also be addressed in an integrated manner with the CF and CL problems when designing an SRCMS. Future research might also investigate a robust layout for flexible manufacturing systems in a dynamic environment that can be adapted to the changes of product mix and demand from one period to the next by changing the process routing of the products.

### ***Bibliography***

1. Rahimi, V., J. Arkat, and H. Farughi, *A vibration damping optimization algorithm for the integrated problem of cell formation, cellular scheduling, and intercellular layout*. Computers & Industrial Engineering, 2020: p. 106439.
2. Baykasoğlu, A., *A meta-heuristic algorithm to solve quadratic assignment formulations of cell formation problems without presetting number of cells*. Journal of Intelligent Manufacturing, 2004. 15(6): p. 753-759.
3. Urban, T.L., W.C. Chiang, and R.A. Russell, *The integrated machine allocation and layout problem*. International journal of production research, 2000. 38(13): p. 2911-2930.

4. Mohammadi, M. and K. Forghani, *Designing cellular manufacturing systems considering S-shaped layout*. Computers & Industrial Engineering, 2016. 98: p. 221-236.
5. Sakhaii, M., R. Tavakkoli-Moghaddam, M. Bagheri, and B. Vatani, *A robust optimization approach for an integrated dynamic cellular manufacturing system and production planning with unreliable machines*. Applied Mathematical Modelling, 2016. 40(1): p. 169-191.
6. Ahi, A., M.B. Aryanezhad, B. Ashtiani, and A. Makui, *A novel approach to determine cell formation, intracellular machine layout and cell layout in the CMS problem based on TOPSIS method*. Computers & Operations Research, 2009. 36(5): p. 1478-1496.
7. Forghani, K. and S.F. Ghomi, *Joint cell formation, cell scheduling, and group layout problem in virtual and classical cellular manufacturing systems*. Applied Soft Computing, 2020: p. 106719.
8. Balakrishnan, J. and C.H. Cheng, *Multi-period planning and uncertainty issues in cellular manufacturing: A review and future directions*. European Journal of Operational Research, 2007. 177(1): p. 281-309.
9. Pillai, V.M., I.B. Hunagund, and K.K. Krishnan, *Design of robust layout for dynamic plant layout problems*. Computers & Industrial Engineering, 2011. 61(3): p. 813-823.
10. Kusiak, A., *Intelligent Manufacturing Systems*. Rentice Hall Press, 200 Old Tappan Road, Old Tappan, NJ 07675, USA, 1990, 448, 1990.
11. Chung, S.H., T.H. Wu, and C.C. Chang, *An efficient tabu search algorithm to the cell formation problem with alternative routings and machine reliability considerations*. Computers & Industrial Engineering, 2011. 60(1): p. 7-15.
12. Kia, R., M.M. Paydar, M.A. Jondabeh, N. Javadian, and Y. Nejatbakhsh, *A fuzzy linear programming approach to layout design of dynamic cellular manufacturing systems with route selection and cell reconfiguration*. International Journal of Management Science and Engineering Management, 2011. 6(3): p. 219-230.

13. Kia, R., A. Baboli, N. Javadian, R. Tavakkoli-Moghaddam, M. Kazemi, and J. Khorrami, *Solving a group layout design model of a dynamic cellular manufacturing system with alternative process routings, lot splitting and flexible reconfiguration by simulated annealing*. *Computers & Operations Research*, 2012. 39(11): p. 2642-2658.
14. Kia, R., J. Khorrami, I. Mahdavi, N. Javadian, and M. Kazemi, *Designing an intra-cell layout model in dynamic cellular manufacturing systems with unequal-area facilities*. *International Journal of Management Science and Engineering Management*, 2012. 7(1): p. 10-19.
15. Bagheri, M. and M. Bashiri, *A new mathematical model towards the integration of cell formation with operator assignment and inter-cell layout problems in a dynamic environment*. *Applied Mathematical Modelling*, 2014. 38(4): p. 1237-1254.
16. Kia, R., H. Shirazi, N. Javadian, and R. Tavakkoli-Moghaddam, *Designing group layout of unequal-area facilities in a dynamic cellular manufacturing system with variability in number and shape of cells*. *International Journal of Production Research*, 2015. 53(11): p. 3390-3418.
17. Mehdizadeh, E. and V. Rahimi, *An integrated mathematical model for solving dynamic cell formation problem considering operator assignment and inter/intra cell layouts*. *Applied Soft Computing*, 2016. 42: p. 325-341.
18. Bayram, H. and R. Şahin, *A comprehensive mathematical model for dynamic cellular manufacturing system design and Linear Programming embedded hybrid solution techniques*. *Computers & Industrial Engineering*, 2016. 91: p. 10-29.
19. Kumar, R. and S.P. Singh, *A similarity score-based two-phase heuristic approach to solve the dynamic cellular facility layout for manufacturing systems*. *Engineering Optimization*, 2017. 49(11): p. 1848-1867.
20. Golmohammadi, A., A. Asadi, Z. Amiri, and M. Behzad, *Design of a facility layout problem in cellular manufacturing systems with stochastic demands*. *Management Science Letters*, 2018. 8(11): p. 1133-1148.

21. Wang, T.Y., K.B. Wu, and Y. Liu, *A simulated annealing algorithm for facility layout problems under variable demand in cellular manufacturing systems*. Computers in industry, 2001. 46(2): p. 181-188.
22. Tavakkoli-Moghaddam, R., N. Javadian, B. Javadi, and N. Safaei, *Design of a facility layout problem in cellular manufacturing systems with stochastic demands*. Applied Mathematics and Computation, 2007. 184(2): p. 721-728.
23. Ariafar, S.H., N. Ismail, S.H. Tang, M.K.A.M Ariffin, and Z. Firoozi, *A stochastic facility layout model in cellular manufacturing systems*. International Journal of Physical Sciences, 2011. 6(15): p. 3754-3758.
24. Paydar, M.M., M. Saidi-Mehrabad, and E. Teimoury, *A robust optimisation model for generalised cell formation problem considering machine layout and supplier selection*. International Journal of Computer Integrated Manufacturing, 2014. 27(8): p. 772-786.
25. Kumar, R., S.P. Singh, and K. Lamba, *Sustainable robust layout using Big Data approach: A key towards industry 4.0*. Journal of Cleaner Production, 2018. 204: p. 643-659.
26. Zuo, X., C. Murray, and A. Smith, *Sharing clearances to improve machine layout*. International Journal of Production Research, 2016. 54(14): p. 4272-4285.
27. Keller, B. and U. Buscher, *Single row layout models*. European Journal of Operational Research, 2015. 245(3): p. 629-644.
28. Allahyari, M.Z. and A. Azab, *A novel bi-level continuous formulation for the cellular manufacturing system facility layout problem*. Procedia CIRP, 2015. 33: p. 87-92.
29. Jeon, G. and H.R. Leep, *Forming part families by using genetic algorithm and designing machine cells under demand changes*. Computers & operations research, 2006. 33(1): p. 263-283.
30. Jaccard, P., *Nouvelles recherches sur la distribution florale*. Bull. Soc. Vaud. Sci. Nat., 1908. 44: p. 223-270.

31. Yin, Y. and K. Yasuda, *Similarity coefficient methods applied to the cell formation problem: a comparative investigation*. Computers & industrial engineering, 2005. 48(3): p. 471-489.
32. Suo, X.S., X.Q. Yu, and H.S. Li, *Subset simulation for multi-objective optimization*. Applied Mathematical Modelling, 2017. 44: p. 425-445.
33. Bajestani, M.A., M. Rabbani, A.R. Rahimi-Vahed, and G.B. Khoshkhou, *A multi-objective scatter search for a dynamic cell formation problem*. Computers & operations research, 2009. 36(3): p. 777-794.
34. Sengupta, R.N., A. Gupta, and J. Dutta, *Decision sciences: theory and practice*. 2016: Crc Press.
35. Fonseca, C.M. and P.J. Fleming, *An overview of evolutionary algorithms in multiobjective optimization*. Evolutionary computation, 1995. 3(1): p. 1-16.
36. Deb, K., A. Pratap, S. Agarwal, and T.A.M.T. Meyarivan, *A fast and elitist multiobjective genetic algorithm: NSGA-II*. IEEE transactions on evolutionary computation, 2002. 6(2): p. 182-197.
37. Luo, B., J. Zheng, J. Xie, and J. Wu, *Dynamic crowding distance? A new diversity maintenance strategy for MOEAs*. in *2008 Fourth International Conference on Natural Computation*. 2008. IEEE.
38. Mousavi, S.M., J. Sadeghi, S.T.A. Niaki, and M. Tavana, *A bi-objective inventory optimization model under inflation and discount using tuned Pareto-based algorithms: NSGA-II, NRGGA, and MOPSO*. Applied Soft Computing, 2016. 43: p. 57-72.
39. Nemati-Lafmejani, R., H. Davari-Ardakani, and H. Najafzad, *Multi-mode resource constrained project scheduling and contractor selection: Mathematical formulation and metaheuristic algorithms*. Applied Soft Computing, 2019. 81: p. 105533.
40. Souilah, A., *Simulated annealing for manufacturing systems layout design*. 1993.

41. Feng, H., L. Xi, T. Xia, and E. Pan, *Concurrent cell formation and layout design based on hybrid approaches*. Applied Soft Computing, 2018. 66: p. 346-359.
42. Zitzler, E., L. Thiele, M. Laumanns, C.M. Fonseca, and V.G. Da Fonseca, *Performance assessment of multiobjective optimizers: An analysis and review*. IEEE Transactions on evolutionary computation, 2003. 7(2): p. 117-132.
43. Schott, J.R., *Fault tolerant design using single and multicriteria genetic algorithm optimization*. 1995, Air force inst of tech Wright-Patterson afb OH.
44. Najafi, A.A., S.T.A. Niaki, and M. Shamsavari, *A parameter-tuned genetic algorithm for the resource investment problem with discounted cash flows and generalized precedence relations*. Computers & Operations Research, 2009. 36(11): p. 2994-3001.
45. Rahmati, S.H.A., V. Hajipour, and S.T.A. Niaki, *A soft-computing Pareto-based meta-heuristic algorithm for a multi-objective multi-server facility location problem*. Applied Soft Computing, 2013. 13(4): p. 1728-1740.



## CHAPTER 5

### CONCLUSIONS

#### *5.1. Concluding Remarks*

In this dissertation, the FLP in manufacturing systems is studied. A constructive metaheuristic algorithm is developed to solve a UA-FLP. The layout generated using the proposed algorithm can be used as an initial layout for metaheuristic algorithms which results in feasible solutions even when the required vs available areas/aspect-ratio is tight.

As in today's competitive world, the product's life cycles are shorter, and their demand is changing more frequently, this research mainly focuses on considering this dynamicity and its impact on FLP. One of the most common approaches in solving FLP in a dynamic environment is the DFLP, which has been the subject of much research. In this dissertation, the DFLP for RMS is studied and solved using the DP approach which is one of the well-known solution approaches used in the literature. Even though enumerating all the possible layouts of each period of DFLP will result in an optimal solution, only small-sized problems can be solved in polynomial time. Hence, a solution algorithm that can obtain a subset of layouts as the state-space of DP that results in a near-optimal solution for larger problems is very challenging. To this aim, a heuristic algorithm is proposed in this dissertation that obtains the most promising layouts of each period and a hybrid metaheuristic algorithm to select the best subset amongst them as the state-space of DP. Computational results demonstrate the outperformance of the proposed approach against many DP benchmark instances in the literature.

The proposed DP is for solving discrete DFLP with equal-area machines. To expand the problem under study and to be more realistic, the unequal-area cellular FLP in a dynamic environment is investigated. This approach suits a cellular FMS or a conventional machine tool job shop. The presented approach is able to obtain the optimum number of cells, assignment of machines to different cells, the locations of machines inside each cell, and the locations of cells on the shop floor. A nonlinear mixed-integer mathematical model is developed that is solved using a multi-objective metaheuristic algorithm. The obtained solution is semi-robust against the changes of product mix and demands such that only a

single robust layout is used for all the periods and the locations of the cells and machines are not changing from a period to the next, yet the locations of the P/D points of the cells are changing. Because of this change, even though the machines are not physically relocated, the distance of each machine to its cell's P/D point and the distance between cells will vary from a period to the next. This is in line with the minimization of the material handling cost of the selected robust layout with respect to the changes of demand. The developed model and solution algorithm are validated using cellular manufacturing test problems that are the benchmark in the literature.

## ***5.2. Future Work***

Simulation techniques can be used as an integrated part of the solution approach. By using simulation, the dynamic and stochastic phenomena of the manufacturing systems are handled. Moreover, simulation can address the behavior of the material handling system in an integrated manner with the layout problems.

The hybrid metaheuristic-dynamic programming approach presented in Chapter 3 is tested for dynamic facility layout problems. However, this framework can also be applied to other combinatorial optimization problems. Therefore, testing the applicability of the proposed method for other optimization problems is suggested as future research.

Regarding the presented mathematical model in Chapter 4, although the assumptions are reasonable, relaxing them can be a good direction for future research. For instance, it is assumed that parameters related to the material flow between machines are deterministic. Extensions of the model can consider these parameters to be stochastic to make the model more realistic. Another possible extension can be developing the models that can simultaneously optimize the layout decisions and other important manufacturing systems decisions such as scheduling, reliability, inventory control, etc.

The proposed multi-objective model in Chapter 4 is developed to minimize the total material handling cost and dissimilarity of machines. A more realistic representation of the manufacturing system can be provided by considering performance criteria such as work-in-process, machine utilization, and transportation time.

## VITA AUCTORIS

NAME: Saeideh Salimpour

PLACE OF BIRTH: Tehran, Iran

YEAR OF BIRTH: 1988

EDUCATION: Azad University (Central Tehran Branch), B.Sc. in Electrical Engineering-Electronics, Tehran, Iran, 2011

Iran University of Science and Technology, M.Sc. in Industrial Engineering, Tehran, Iran, 2014