DOKUZ EYLÜL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

DYNAMIC SCHEDULING OF FLEXIBLE JOB SHOPS UNDER CAPACITY AND SETUP CONSTRAINTS

by

Fatma Selen KARASLAN

January, 2017

İZMİR

DYNAMIC SCHEDULING OF FLEXIBLE JOB SHOPS UNDER CAPACITY AND SETUP CONSTRAINTS

A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of Dokuz Eylül University
In Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy in Industrial Engineering, Industrial Engineering Program

by
Fatma Selen KARASLAN

January, 2017
İZMİR

Ph.D. THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **"DYNAMIC SCHEDULING OF FLEXIBLE JOB SHOPS UNDER CAPACITY AND SETUP CONSTRAINTS"** completed by **FATMA SELEN KARASLAN** under supervision of **PROF. DR. ADİL BAYKASOĞLU** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Doctor of Philosophy.

Prof. Dr. Adil BAYKASOĞLU

Supervisor

Prof. Dr. Hilmi YÜKSEL

Thesis Committee Member

Assoc. Prof. Dr. Hasan SELİM

Thesis Committee Member

Prof. Dr. Rızvan EROL

Examining Committee Member

Prof. Dr. Hadi GÖKÇEN

Examining Committee Member

Prof.Dr. Emine İlknur CÖCEN
Director
Graduate School of Natural and Applied Sciences

ACKNOWLEDGMENTS

I would like to take this opportunity to acknowledge my deep sense of gratitude to my supervisor, Prof. Dr. Adil BAYKASOĞLU, for his guidance; unfailing advice, continuing support and encouragement that helped me overcome challenges throughout the progress of this thesis.

I would also like to thank to my committee members Prof. Dr. Hilmi YÜKSEL and Assoc. Prof. Dr. Hasan SELİM for their helpful comments and advice.

I would like to thank, everyone who have helped me throughout the progress of thesis, my colleagues and all staff member in department of Industrial Engineering.

Finally, I would like to express my deep thankfulness to my family, who have loved and supported me whole my life, Ali Suat MADENOĞLU, Ulviye MADENOĞLU and my brother İhsan MADENOĞLU. But the most, I am grateful to my husband Sermed Çağatay KARASLAN for his endless patience, encouragement, continuous support and for providing me comfortable condition to finish my thesis.

Fatma Selen KARASLAN

DYNAMIC SCHEDULING OF FLEXIBLE JOB SHOPS UNDER CAPACITY AND SETUP CONSTRAINTS

ABSTRACT

This thesis concerns the problem of dynamic flexible job shop scheduling. An event driven job shop scheduling policy under machine capacity constraints is presented. The proposed method makes use of the greedy randomized adaptive search procedure by also taking into account order's due dates and sequence-dependent setup times. Moreover, order acceptance/rejection decision and order review release mechanism are integrated with scheduling decision to meet customer due date requirement while attempting to execute capacity adjustments. A preemptive based multi-objective approach which accounts for four objectives: mean tardiness, schedule unstability, makespan and mean flow time is employed.

Within this context, there is no set of benchmark instances for testing of the proposed problem. First, the job shop scheduling problem benchmark data is solved by the proposed algorithm in order to test the performance of the proposed algorithm. Then the capability of the proposed algorithm is tested through a set of the dynamic job shop scheduling hypothetical cases. Second, the proposed algorithm sorts out the flexible job shop scheduling problem benchmark data in order to test the performance of the proposed algorithm. Then capability of the proposed algorithm is analyzed through a set of dynamic flexible job shop scheduling hypothetical cases. The performance of the proposed algorithm is tested through a set of computational experiments and computational results indicate that the proposed algorithm has satisfactory performances for solving the dynamic flexible job shop scheduling problem.

Keywords: Dynamic flexible job shop scheduling, sequence dependent setup times, capacity constraint, multi-objective optimization, greedy randomized adaptive search procedure, dispatching rules.

KAPASİTE VE AYAR KISITLARI ALTINDA ESNEK ATÖLYELERİN DİNAMİK ÇİZELGELENMESİ

ÖZ

Bu tez dinamik esnek atölye çizelgeleme problemini ele almaktadır. Makine kapasite kısıtı altında olaya dayalı atölye çizelgeleme politikası sunulmuştur. Önerilen metot, sipariş bazlı teslim tarihi ve sıra bağımlı hazırlık sürelerini dikkate alarak açgözlü rasgele seçimli uyarlanabilir arama prosedürünü kullanmaktadır. Ayrıca, kapasite ayarlamaları yapmaya çalışırken müşteri bazlı teslim tarihlerini gerçekleştirebilmek için çizelgeleme kararına sipariş kabul/ret kararı ve sipariş gözden geçirme/gönderme mekanizması entegre edilmiştir. Ortalama sipariş gecikmesi, çizelge değişkenliği, siparişlerin toplam tamamlanma zamanı ve ortalama siparişlerin akış süresi hedeflerini dikkate alan önleyici tabanlı, çok amaçlı bir yaklaşım kullanılmıştır.

Önerilen problemin test edilmesine yönelik kıyaslama örnek setleri bulunmamaktadır. İlk olarak, literatürdeki statik atölye çizelgeleme problemi kıyaslama örnekleri, önerilen algoritmanın performansını test etmek için çözülmüştür. Ardından, önerilen algoritmanın kabiliyeti, dinamik atölye çizelgeleme varsayımsal vaka seti aracılığıyla test edilmiştir. İkinci olarak, önerilen algoritmanın performansını test etmek için literatürdeki statik esnek atölye çizelgeleme problemi kıyaslama örnekleri çözülmüştür. Ardından, önerilen algoritmanın kabiliyeti, dinamik esnek atölye çizelgeleme varsayımsal vaka seti aracılığıyla analiz edilmiştir. Önerilen algoritmanın performansı, bir dizi hesaplama deneyiyle sınanmış ve hesaplanan sonuçlar, önerilen algoritmanın dinamik esnek atölye çizelgeleme problemini çözmek için tatmin edici performanslara sahip olduğunu göstermektedir.

Anahtar kelimeler: Dinamik esnek atölye çizelgeleme, sıra bağımlı hazırlık zamanı, kapasite kısıtı, çok amaçlı optimizasyon, açgözlü rasgele seçimli uyarlanabilir arama prosedürü, yönlendirme kuralları.

CONTENTS

Page

Ph.D. THESIS EXAMINATION RESULT FORM ii
ACKNOWLEDGMENTS iii
ABSTRACT iv
ÖZ v
LIST OF FIGURES ix
LIST OF TABLES xi

CHAPTER ONE - INTRODUCTION 1

1.1 Importance of the Study 1
1.2 Research Objectives 3
1.3 Outline of the Thesis 4

CHAPTER TWO - PROBLEM DEFINITION AND LITERATURE REVIEW 6

2.1 Introduction 6
2.2 Overview of Flexible Job Shop Scheduling Problem 9
2.2.1 Machine Assignment Rule 12
2.3 Dynamic Scheduling Approaches 13
2.4 Discussion of the Problem 18
2.4.1 Scheduling Constraints 18
2.4.1.1 Machine Non-Availability……...18
2.4.1.2 Sequence-Dependent Setup Times……...19
2.4.1.3 Order Acceptance/Rejection Decision...……..……...……21
2.4.1.4 Order Review Release(ORR)…..…….23
2.4.2 Rescheduling Under Different Disruptions 27
2.4.2.1 Machine Breakdown……27
2.4.2.2 Order Cancellation……...27
2.4.2.3 New Order Arrival……...28

2.4.2.4 Rush Order……..29
2.4.2.5 Change of the Due Dates……29
2.4.3 Fitness Evaluation by Preemptive Goal Programming..... 30
2.5 Literature Review 34

CHAPTER THREE - GREEDY RANDOMIZED ADAPTIVE SEARCH PROCEDURE 45

3.1 Chapter Introduction..... 45
3.2 Greedy Randomized Adaptive Search Procedure (GRASP)..... 45
3.3 GRASP for Dynamic Flexible Job Shop Scheduling Problem (DFJSSP) 47
3.3.1 Solution Representation..... 48
3.3.2. Generating New Schedule 48

CHAPTER FOUR - SOLVING THE DYNAMIC JOB SHOP SCHEDULING PROBLEM BY USING GREEDY RAMDOMIZED ADAPTIVE SEARCH PROCEDURE 64

4.1 Chapter Introduction..... 64
4.2 Experimental Design and Results..... 65
4.2.1 Experiment I 65
4.2.2 Experiment II 70
4.2.3 Experiment III..... 72
4.3 Chapter Conclusion 76

CHAPTER FIVE - SOLVING THE DYNAMIC FLEXIBLE JOB SHOP SCHEDULING PROBLEM BY USING GREEDY RANDOMIZED ADAPTIVE SEARCH PROCEDURE 78

5.1 Chapter Introduction..... 78
5.2 Experimental Design 78
5.2.1 Experiment I 79

5.2.2 Experiment II 83
5.2.3 Experiment III 97
5.3 Chapter Conclusion 108

CHAPTER SIX - CONCLUSION 109

6.1 Summary of the Thesis 109
6.2 Contributions of the Thesis 109
6.3 Future Research Directions 111

REFERENCES 112

APPENDICES 129
A1 Data set of DJSSP Introduced in Chapter 4 129
A2 Data set of DFJSSP Introduced in Chapter 5 131

LIST OF FIGURES

Page

Figure 2.1 Information flow diagram in a manufacturing system 7
Figure 2.2 Dynamic rescheduling policies (a) Periodic rescheduling policy (b) Event-driven rescheduling policy 17
Figure 2.3 Machine non-availability 20
Figure 2.4 Sequence-dependent setup times 21
Figure 2.5 Cooperation department in order acceptance/rejection decision 22
Figure 2.6 Order acceptance/rejection decision 26
Figure 2.7 Order review release 26
Figure 2.8 Machine breakdown. (a) Partial schedule at time 5 (b) A sample schedule 28
Figure 2.9 Order cancellation 28
Figure 2.10 New order arrival 31
Figure 2.11 Rush order 31
Figure 2.12 Due date change 31
Figure 2.13 Periodical Rescheduling. (a) Partial schedule at time 10 (b) A sample schedule 32
Figure 3.1 Illustration of candidate solution in JSSP 48
Figure 3.2 Relationship of schedule properties 49
Figure 3.3 Gannt-Chart for example problem 54
Figure 3.4 a) Swap local search b) Interchange local search c) Insert local search d)Change of the assigning machine 57
Figure 3.5 Local search process with swap move 58
Figure 3.6 Gannt Chart of job shop scheduling for construction phase 59
Figure 3.7 Gannt Chart of job shop scheduling after the local search phase 59
Figure 3.8 Flowchart of the proposed GRASP 60
Figure 3.9 Flowchart of the GRASP for DFJSSP under event-driven rescheduling policy 61
Figure 3.10 An event-driven rescheduling example for FJSSP with new order arrival 63

Figure 4.1 Gannt chart of the LA12 static job shop scheduling problem with GRASP 67
Figure 5.1 Comparison of the proposed GRASP and dispatching rules for 5x5 problems 91
Figure 5.2 Comparison of the proposed GRASP and dispatching rules for 10x10 problems 93
Figure 5.3 Comparison of the proposed GRASP and dispatching rules for 15x15 problems 95
Figure 5.4 Comparison of the event-driven and the periodical rescheduling policies for 5x5 problems..... 102
Figure 5.5 Comparison of the event-driven and the periodical rescheduling policies for 10x10 problems..... 104
Figure 5.6 Comparison of the event-driven and the periodical rescheduling policies for 15x15 problems..... 106

LIST OF TABLES

Page

Table 2.1 Processing time 11
Table 2.2 Example problem data 19
Table 2.3 An overview of the approaches in the literature on dynamic job shop scheduling problem 41
Table 3.1 Example 2x2 54
Table 3.2 Data of a job shop scheduling problem 56
Table 4.1 Comparison of GRASP from the literature for JSSP test instances 66
Table 4.2 Solution of 20x5 job shop scheduling problem (LA12) with GRASP 67
Table 4.3 Friedman ranks 70
Table 4.4 Comparison of the proposed GRASP and dispatching rules for 5x5 problem 73
Table 4.5 Comparison of the proposed GRASP and dispatching rules for 10x10 problem 74
Table 4.6 Comparison of the proposed GRASP and dispatching rules for 15x15 problem 75
Table 4.7 Comparison of event-driven and periodical rescheduling policies 76
Table 5.1 Makespan results for Kacem's instances 80
Table 5.2 Results for Brandimarte's data set 80
Table 5.3 Results for Dauzere-Peres and Paulli's data set 82
Table 5.4 Adjusted p-values for multiple comparisons among all methods 82
Table 5.5 Multiple comparisons for different significance level (α) 82
Table 5.6 Machine alternatives of operations for different problem sizes 83
Table 5.7 Comparison of the proposed GRASP algorithm and dispatching rules for 5x5 problems 85
Table 5.8 Comparison of the proposed GRASP algorithm and dispatching rules for 10x10 problems 87
Table 5.9 Comparison of the proposed GRASP algorithm and dispatching rules for 15x15 problems 89
Table 5.10 ANOVA results (p-value) for factor effects and interactions 97

Table 5.11 Comparison of event-driven and periodical rescheduling policies for 5x5 problems 99
Table 5.12 Comparison of event-driven and periodical rescheduling policies for 10x10 problems 100
Table 5.13 Comparison of event-driven and periodical rescheduling policies for 15x15 problems 101

CHAPTER ONE
INTRODUCTION

1.1 Importance of the Study

Production management is a managerial process in which a large number of decisions are taken over time in order to ensure the delivery of goods with the maximum quality, minimum cost, and minimum lead time. These decisions differ—among other issues—on their impact in the company, their scope, and on the time period to take them. They range from strategic, high-impact, long-range decisions such as deciding whether certain plant will manufacture or not a new product, to short-term, low-level, small-impact decisions such as which product is next to be manufactured in certain machine in the shop floor (Framinan et al., 2014).

In production management, scheduling plays an vital role that is important to achieve timely and cost effective production. Scheduling allocates the limited resources to jobs (tasks) over a time period to optimize the predefined objective or objectives. A schedule specifies which job should be operated for each resource and when. Scheduling provides additional information for different functions in the company as well as detailed production plan to be conducted. For example, it provides a capacity-check for materials requirement planning system, or decides orders acceptance/rejection. We can also see the close interaction between scheduling and other business functions.

The high level of interrelation among production management decisions (including scheduling) may be directed to give priority to these decisions in order to arrange the scarce financial and human resources effectively. This exposes which decision is more important than the other or others. However, this is the important point that better scheduling try to not help to improve the company objectives without an overall good operations management and it can be said that the same is applied to the different decisions.

Nowadays, industries confront with the intensified global competition, demanding customers, and rapidly changing world. More than in the past, companies push forward information and technology fields and conduct right business and production strategy to survive in a competitive marketplace. It is not enough only to keep up with the others. Today, success represents sustainability in introducing new standards and orientation towards the customer, to the quality and price of products, to flexibility, agility and promptness, to economising in resources and protection of the environment (Huang, 2010).

The globalization of markets has created entirely new dynamics of rapid environmental change. Flexibility is the capacity to deploy or redeploy production resources efficiently as required by changes in the environment. Flexibility is thus one of the basic characteristics of any production process. While it primarily corresponds to time, it is also linked to two other cross-functional objectives: quality and cost. Flexible production can be seen as an ideal state to be used in orienting efforts aimed at continuous improvement and elimination of waste (Duguay et al., 1997). An increacing number of companies prefer a flexible job shop system to adapt the changing world conditions.

The flexible job shop scheduling literature usually assumes that setup times are negligible or added to the processing times. Setups are processed on the machine before the main processing. Some examples of setups are inspecting materials, cleanings, tool changes. Setup times depend on the consecutive operations and it is named sequence dependent setup times. Sequence dependent setup times effects the schedule performance and it can be achieved by considering sequence dependent setup times between operations. Implication of sequence dependent setup times increases the hardness of the scheduling problem.

Production resource capacity is limited, but resource capacity isn’t considered in the flexible job shop scheduling literature generally and all orders are accepted as if machine capacities are limitless. Beside this, whenever the orders reach to the shop floor, all the orders are released immediately. This leads that some manufacturing

orders may delay and cause customer dissatisfaction in competitive marketplace. In order to achieve the company objectives, the companies should determine which orders to accept, how to schedule them, when orders are released to the shop floor by managing production capacity efficiently and maintaining customer satisfaction.

In classical scheduling theory, the scheduling problems are formulated as static, deterministic and are solved by exact, heuristic procedures. However, real systems are stochastic and dynamic in nature. After the unexpected events (order arrival, machine breakdown, order cancellation, due date change etc.) happen, these optimum schedules can become infeasible in real systems. Because of this reason, a feasible schedule generation can be the essential goal of the scheduling in dynamic environment.

The real flexible job shop manufacturing environment is dynamic and setup times, machine capacity constraint and dynamic events must be considered in manufacturing systems. At this point, the main endeavor of this thesis is to introduce the dynamic flexible job shop scheduling problem (DFJSSP) under event-driven rescheduling policy with sequence dependent setup times, machine capacity constraint, order acceptance/rejection decision, ORR mechanism by employing the proposed constructive, multi-start algorithm. This thesis fills a gap by focusing on implementations of real life practical constraints in dynamic manufacturing scheduling.

1.2 Research Objectives

The main goal of the thesis is to introduce the dynamic scheduling of flexible job shops under capacity and setup constraints and developing solution procedure. In the dynamic flexible job shop scheduling literature, dynamic events are considered partly. New order arrivals, machine breakdowns, due date changes, rush orders, order cancellations are handled simultaneously under event-driven rescheduling policy. Sequence dependent setup times and machine capacity constrainst are involved to the proposed problem. To manage production capacity orderly and satisfy the customer

demands, order acceptance/rejection decision which determines the accepted orders and ORR mechanism which determines the releasing time of the arriving orders to the shop floor is adapted to scheduling decision. A preemptive goal programming approach is utilized for evaluating the relative performance of the generated schedules according to the selected objective functions. In addition to event-driven rescheduling policy, a periodic rescheduling policy is also devised and both strategies are compared for different problems. This thesis presents one of the most comprehensive scheduling models with some practical criterias, constraints and mechanisms.

FJSSP is NP-hard and the proposed DFJSSP problem is also NP-hard. Because of this reason, a meta-heuristic algorithm is proposed to tackle the problem. The proposed constructive and multi-start algorithm is compared with dispatching rule-based scheduling approach as dispatching rules are usually utilized for generating schedules for such complex dynamic manufacturing scheduling problems.

1.3 Outline of the Thesis

Rest of the thesis involves five chapters. The following chapter contains the problem definition, dynamic scheduling approaches, the scope of our research problem and a literature survey about the dynamic job shop scheduling and dynamic flexible job shop scheduling.

Chapter 3 presents the Greedy Randomized Adaptive Search Procedure (GRASP) and the proposed GRASP for dynamic flexible job shop scheduling problem. Particularly, the Giffler & Thompson algorithm is utilized in construction phase of the proposed GRASP algorithm under dynamic environment is explained in detail.

The suitability of the proposed algorithm is conducted on the static job shop scheduling benchmark problems in Chapter 4. Then, the hypothetical cases of the proposed dynamic job shop scheduling problem are generated. The results of the

proposed algorithm are compared with six dispatching rules. The comparison of rescheduling policies (periodical vs. event-driven) is conducted in this Chapter.

Chapter 5 represent the results of the static flexible job shop scheduling to analyze the performance of the proposed algorithm. Then, the hypothetical cases of the proposed dynamic flexible job shop scheduling problem are generated. The results of the proposed algorithm are compared with six dispatching rules and the results of the proposed algorithm under periodical rescheduling policy.

Finally, the conclusions, the contributions of thesis, and suggestion for future research are discussed in Chapter 6.

CHAPTER TWO
PROBLEM DEFINITION AND LITERATURE REVIEW

2.1 Introduction

Scheduling is a very critical activity in production management due to its central role in systems performance. A production schedule specifies, for each resource required for production, the planned start time and end time of each job assigned to that resource. Scheduling is the process of creating a production schedule for a given set of jobs and resources. Many real scheduling problems in the manufacturing industries are quite complex and very difficult to be solved by conventional optimization techniques (Watanabe et al., 2005).

The scheduling process also interacts with the production planning process, which handles medium-term to long-term planning for the entire organization. This process intends to optimize the firm's overall product mix and long-term resource allocation based on inventory levels, demand forecasts and resource requirements. Decisions made at this higher planning level may impact the more detailed scheduling process directly (Pinedo, 2005). Figure 2.1 shows the information flow in a manufacturing system. In Figure 2.1, short-term capacity planning, for relatively short time horizon such as a month is categorized into the decision-making process to allocate resources and to specify worker shifts along with a shop floor schedule, according to the results of the upper-level capacity planning based on a master production schedule and customer orders. Short-term capacity planning is usually carried out by utilizing a Capacity Requirement Planning (CRP) system with in a Material Requirement Planning(MRP) system in modern manufacturing. Then the shop floor schedule, in other words, the detailed schedule for a certain period such as a week is derived from the monthly plan obtained by short-term capacity planning (Suwa & Sandoh, 2012). All the scheduling operations during the planning phase is called offline scheduling, because the uncertainties of the manufacturing environment are not considered.

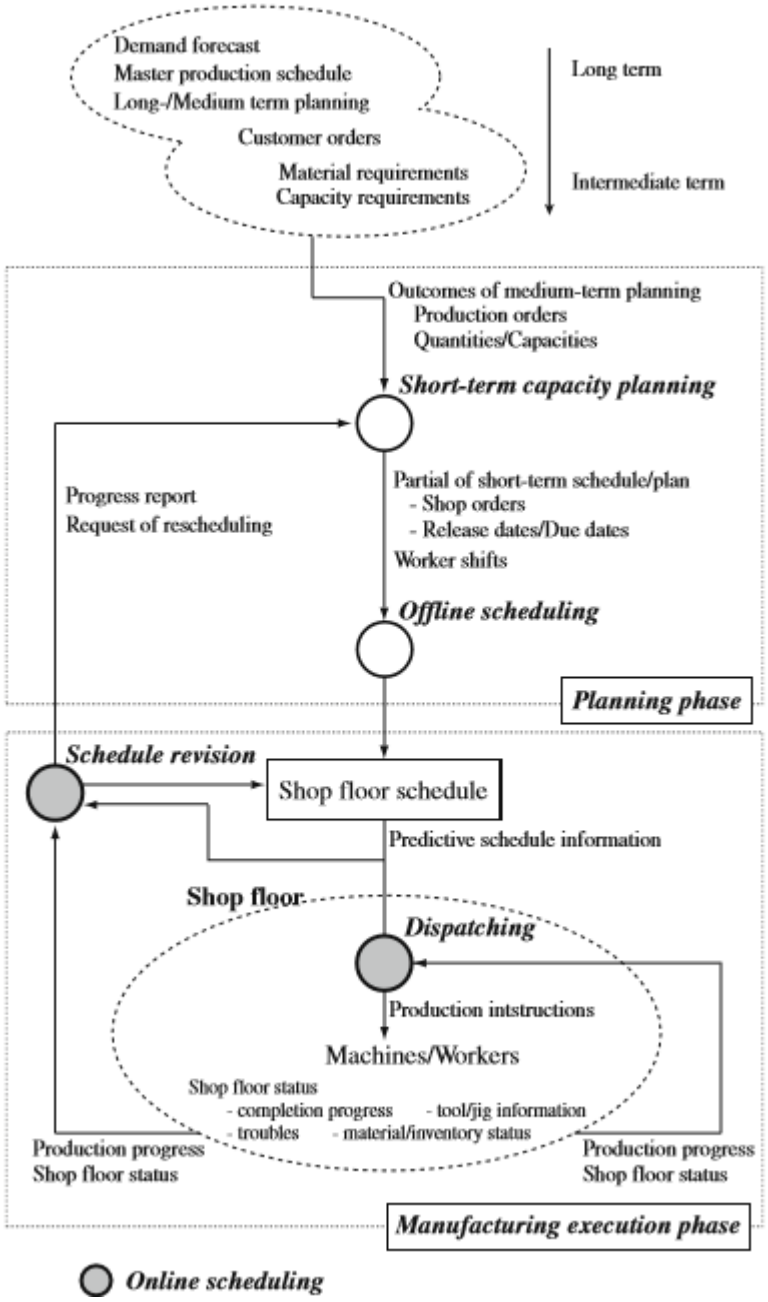


Figure 2.1 Information flow diagram in a manufacturing system (Suwa & Sandoh, 2012)

The job shop scheduling problem (JSSP) is a type of scheduling problem in manufacturing systems. In the classical JSSP, *n* jobs are processed for completion on *m* unrelated machines. For each job, technology constraints specify a complete, distinct routing which is fixed and known in advance. Each machine is continuously available from time zero, and operations are processed without preemption. The

appropriate sequencing of operations on the machines for optimizing the performance indicator is the aim of this problem. The JSSP is one of the hardest combinatorial optimization problems in the scheduling problem and NP-hard (Garey et al., 1976).

Manufacturing systems must be more flexible and efficient in order to achieve nowadays market requirements. The flexible scheduling systems are used in the systems to realize these objectives. The flexible job shop scheduling problem (FJSSP) is a much more complex version of the JSSP, so the FJSSP is strongly NP-hard and combinatorial. It incorporates all of the difficulties and complexities of its predecessor JSSP. It is more complex because of the additional need to determine the assignment of operations to the machines (Saidi-mehrabad & Fattahi, 2007; Yazdani et al., 2009). Two difficulties of the FJSSP are: to assign each operation to a machine from a set of capable machines and sequence the assigned operations on all machines.

Interestingly scheduling problems are usually considered and modeled as static problems in operations research literature. However, due to the nature of the production environment, the production environments encountered many dynamic events such as machine breakdowns, changes in due dates, job cancellations, arrivals of urgent jobs etc.. These random occurrences might cause some disturbances during later scheduling, delays on the shop floor schedule and make static scheduling approaches very complex. There is a new type of scheduling problem in most of the industries termed the dynamic (online) scheduling problem. In this condition, all orders are not available at the beginning of the scheduling and some dynamic events happen during the scheduling. After that, near optimal schedules generated by static approaches may become outdated when they are released to the shop floor. It is difficult to take into consideration these events while generating a schedule. Rescheduling is necessary to update a production schedule when they make the scheduling problem unfeasible and complex.

The main goal of this chapter is to provide a general understanding about the dynamic flexible job shop scheduling problem and the consideration of proposed problem in dynamic flexible job shop scheduling problem. The rest of this chapter is organized as follows. Section 2.2 gives information about flexible job shop scheduling problem. In Section 2.3, a brief introduction to the core concepts of the dynamic scheduling is given. The proposed problem is given in Section 2.4. A literature survey on DJSSP and DFJSSP is also given in Section 2.5.

2.2 Overview of Flexible Job Shop Scheduling Problem

FJSSP is an extension of the classical JSSP which allows an operation to be processed by any machine from a given set. In a flexible job shop there are a set of machining centers in which an operation can be performed by any machine in the work center.

The FJSSP can be defined as follows: There is a set of n independent jobs $J=\{J_1, J_2, J_3,\ldots, J_n\}$ and a set of m machines $M=\{M_1, M_2, M_3,\ldots, M_m\}$. Each job J_i consists of a sequence of operations $\{O_{i_1}, O_{i_2}, O_{i_3},\ldots\ldots\ldots, O_{i_{n_i}}\}$. Each operation O_{ij} can be processed on any among a subset $M_{ij} \subseteq M$ of compatible machines.

The FJSSP consists of a routing sub-problem, that is, assigning each operation to a machine out of a set of capable machines and the scheduling subproblem, which consists of sequencing the assigned operations on all machines in order to obtain a feasible schedule, minimizing a predefined objective function.

Assumptions of a FJSSP are as following:

- A set of n $J=\{J_1, J_2, J_3,\ldots, J_n\}$ and a set of m machines $M=\{M_1, M_2, M_3,\ldots, M_m\}$.
- Each job consists of a predetermined sequence of operations $O=\{O_{i_1}, O_{i_2}, O_{i_3},\ldots\ldots\ldots, O_{i_{n_i}}\}$.
- Each operation requires one machine that is selected from a set of available machines.

•All jobs and machines are available at time 0, and each machine can only execute one operation at a given time.

- Machines never breakdown and they are always available.
- The processing time of an operation (O_{ij}) on machine *k* is predetermined.
- The setup time of any operation is sequence independent and it is included in its processing time.
- Preemption is not allowed (A started operation cannot be interrupted).
- Each machine can perform at most one operation at any time.
- The precedence constraints of the operations in a job can be defined for any pair of operations.
- Machines are independent from each other.
- Jobs are independent from each other.

The notation is summarized in the following:

n: total number of jobs

m: total number of machines

n_i: total number of operation of job i

O_{ij}: the jth operation of job i

M_{ij}: the set of available machines for the operation O_{ij}

P_{ijk}: processing time of O_{ij} on machine k

t_{ijk}: start time of of operation O_{ij} on machine k

C_{ij}: completion time of the operation O_{ij}

i, h: index of jobs, i, h = 1, 2,...,n

k: index of machines, k = 1, 2,...,m

j, g: index of operation sequence, j, g = 1, 2,...,n_i

C_k is the completion time of M_k

$$x_{ijk} = \begin{cases} 1; & \text{if machine } k \text{ is selected for the operation } O_{ij} \\ 0; & \text{otherwise} \end{cases}$$

The standard FJJSP model is given as follows (Gen et al., 2009):

$$\min f_1 = \max_{1 \le k \le m} C_k \qquad (2.1)$$

$$\text{s.t. } C_{ij} - C_{i(j-1)} \geq P_{ijk} x_{ijk}\ ,\ j = 2, \ldots\ldots, n_i; \forall i, j \quad (2.2)$$

$$\left[\left(C_{hg} - C_{ij} - t_{hgk}\right) x_{ijk} x_{hgk} \geq 0 \ \vee \left(C_{ij} - C_{hg} - t_{hgk}\right) x_{ijk} x_{hgk} \geq 0\right],$$

$$\forall\ (i,j),(h,g),k \quad (2.3)$$

$$\sum_{k \in M_{ij}} x_{ijk}{}_{ijk} = 1, \forall\ i, j \quad (2.4)$$

Inequity (2.2) states the operation precedence constraint. Inequality (2.3) ensures the execution of operation O_{ik} cannot be overlapped in time with the execution of operation O_{hg}. Equation (2.4) states that one machine must be selected from the set of available machines for each operation.

Problem data can be arranged in a table, where rows are gave us the operations informationd and columns are gave us machines information. Table 2.1 shows the processing time of each operation for each machine. This table depicts total flexibility. In a partial flexibility, one operation cannot be operated on all machines.

Table 2.1 Processing time

	M_1	M_2	M_3
O_{11}	5	8	4
O_{12}	4	4	6
O_{21}	5	4	7
O_{22}	3	5	8
O_{23}	9	7	6

There is far more literature focusing on FJSSP. Due to the complexity of FJSSP, no exact methods have so far been proven to be efficient for solving the problem whereas meta-heuristic algorithms have become the practical solution techniques which can produce good schedules in a reasonable time. Some of the previous studies on the subject can be summarized as follows: Brucker and Schlie (1990) are among the first to address FJSSP. They developed a polynomial algorithm for solving the flexible job shop scheduling problem with two jobs. Brandimarte (1993) presented a hybrid tabu search algorithm with some dispatching rules for solving FJSSP. Baykasoğlu (2002) and Baykasoğlu et al. (2004) presented a linguistic-based meta-heuristic modeling and solution approach for FJSSP. Lee et al. (2002)

presented a mathematical model for this scheduling problem and then improved a genetic algorithm to minimize the makespan time. Saidi-mehrabad and Fattahi (2007) presented a mathematical model for solving the medium and large size problems of FJSP. They proposed a hierarchically approach algorithm based on a tabu search algorithm to solve this problem. Pezzella et al. (2008) presented a genetic algorithm for FJSSP in which a mix of different rules for generating the initial population, selection of individuals, and reproduction operators were used. Gao et al. (2008) used a hybrid of genetic algorithm and variable neighborhood descent for FJSSP. Bagheri et al. (2010) addressed an artificial immune algorithm for solving FJSSP to minimize makespan. Baykasoğlu and Özbakır (2010) analyzed the effects of dispatching rules on the scheduling performance within different machine flexibility levels. Rajkumar et al. (2010) presented a GRASP based algorithm with non-fixed availability constraints for FJSSP. Yazdani et al. (2010) presented a parallel variable neighborhood search algorithm which carries the responsibility of making changes in assignment and sequencing of operations for generating neighboring solutions for FJSSP. Rajkumar et al. (2011) presented a GRASP based algorithm with resource constraints for FJSSP. Zhang et al. (2011) suggested an effective genetic algorithm for solving FJSSP to minimize makespan. Xu et al. (2015) proposed an effective teaching–learning-based optimization algorithm for solving FJSSP with fuzzy processing times. Gao et al. (2016) presented an effective discrete harmony search algorithm with weighted combination of two minimization criteria namely, the maximum of the completion time and the mean of earliness and tardiness for FJSSP. Singh and Mahapatra (2016) used quantum-behaved particle swarm optimization for solving FJSSPs.

2.2.1 Machine Assignment Rule

Machine assignment rule is used to assign an operation to one of its capable machines for the FJSSP. In this proposed FJSSP, a machine which execute the operation with minimum finishing time is assigned to the operation and the assignment is fixed. This rule considers the global workload among all machines and can help find the better makespan.

2.3 Dynamic Scheduling Approaches

In most of the real manufacturing systems, unexpected events may occur after the scheduling process. The ongoing feasible schedule may turn infeasible when the unexpected event happens. MacCarthy and Liu (1993) presented the nature of the gap between the scheduling theory and scheduling practice, the failure of classical scheduling theory to respond to the needs of practical environments, and recent trends in scheduling research which attempt to make it more relevant and applicable. Shukla and Chen (1996), in their comprehensive survey on intelligent real-time control in flexible manufacturing systems, stated that comparison of theory and scheduling practice showed very little correspondence between the two. Cowling and Johansson (2002) addressed an important gap between scheduling theory and practice, and stated that scheduling models and algorithms are unable to make use of real-time information.

There is a new type of scheduling problem in most of industries named the *dynamic scheduling*. Single machine systems, parallel machined systems, flow shops, job shops, and flexible manufacturing systems include the real time events and these systems are examined in the literature on dynamic scheduling.

In dynamic scheduling problems, some real time events happen after the scheduling process. Real-time events have been classified into two categories (Stoop & Weirs, 1996; Suresh & Chaudhuri, 1993; Cowling & Johansson, 2002; Vieira et al., 2003):

- **Resource-related:** machine breakdown, operator illness, unavailability or tool failures, loading limits, delay in the arrival or shortage of materials, defective material (material with wrong specification), etc.
- **Job-related:** rush jobs, job cancellation, due date changes, early or late arrival of jobs, change in job priority, changes in job processing time, etc.

Near optimal schedules generated by static approaches may become outdated when dynamic event happens. After the dynamic events happen, the two decisions occur: *when-to-schedule* and *how-to-schedule*.

When-to-schedule policy determines the timing of scheduling points. Three policies have been proposed in the literature (Sabuncuoglu & Bayiz, 2000; Vieira et al., 2003): periodic, event-driven, and hybrid.

In the periodic policy, the timing of between consecutive schedule points is equally and the shop floor is monitored periodically. This means that the schedules are created at regular frequencies, which all avalaible information is gathered from the shop floor during the scheduling process. This policy separates the dynamic scheduling problem to the series of static problems. The schedule is generated and executed, and not revised until the next scheduling period. The new information are gathered from the shop floor status and then the schedule is updated.

In the event-driven policy, the shop floor is monitored continuously and the schedule is updated when an unexpeted event occurs. Most of the approaches for dynamic scheduling use this policy. The risk of this policy is unnecessary changes in the schedule because of the event which doesn't cause significant disruptions.

In the hybrid policy, rescheduling is executed periodically and also an unexpected event (such as machine breakdown, arrival of urgent orders, order cancellation, job priority changes) is triggerred the rescheduling process.

To clarify the difference with these rescheduling policies we presented in Figure 2.2, how two rescheduling strategies respond to the *'new order arrival'* dynamic event. In Figure 2.2-a, dynamic events do not trigger rescheduling process until the next scheduling period, but in Figure 2.2-b, rescheduling process is responsive to changing to the shop conditions.

How-to-schedule policy determines to tackle production schedule under uncertainty and concerns the definiton of rescheduling policies. The literature provided two main rescheduling policies (Sabuncuoglu & Bayiz, 2000; Cowling & Johansson, 2002; Vieira et al., 2003): (1) schedule repair, and (2) complete rescheduling.

Schedule repair regenerates the current schedule using some local adjustment and preferable features of schedule repair are saving in CPU times and providing the stability of the system.

Complete rescheduling generates a new schedule from scratch. Complete rescheduling obtains better optimal solutions. However, the current schedule is updated each rescheduling point. Besides, complete rescheduling doesn't provide stability because of the revision of the current schedule. As a result, complete rescheduling increases the schedule computation time, leads to additional production cost, results in instability.

Dynamic scheduling has been defined under three categories (Mehta & Uzsoy, 1998; Vieira et al., 2000; Vieira et al., 2003; Aytug et al., 2005; Herroelen & Leus, 2005): completely reactive scheduling, predictive–reactive scheduling, and robust pro-active scheduling.

In completely reactive scheduling, the schedule is revised when necessary in real time. However, only local information is used to construct a schedule which may not be globally optimal. Complete reactive scheduling is flexible and includes recent information from the shop floor. However, the overall perfomance of this approach is debatable in response to lack of coordination. Most of the machine learning based, knowledge based, neural network based, and fuzzy logic based approaches in which one schedule is made at a time when needed according to the changes in the system conditions are also classified into this category (Aytug et al., 2005).

In robust scheduling, a schedule is generated by satisfying performance requirements predictably in dynamic environment. The specification of the perdictability measures is the main difficulty. Mehta (1999) proposed a predictable scheduling model for a single machine exposed to breakdowns with the objective of minimizing the maximum lateness. The effect of disruption is measured by the deviation of the job completion time of the realised schedule from its planned completion time in the predictive schedule. The deviation is reduced through insertion of an additional time in the predictive schedule for achieving high predictability. Extensive computational experiments showed that predictable scheduling provides a significant improvement in the predictability at the expense of very little degradation in the maximum lateness.

The most common dynamic scheduling approach in manufacturing systems is called predictive-reactive scheduling. It combines static-deterministic scheduling as a first step with event-based reactive, possibly real-time updating of schedules in the meantime as second step until a new static-deterministic schedule is generated. Apart from its time coordination effect between the two steps, this approach is also intended to coordinate the broader view of the overall system to be scheduled in the first step with the myopic perspective of rescheduling in the second. However, as often confirmed empirically, large tolerances in the second step often significantly reduce the overall performance of the system since the finally implemented schedule may deviate significantly from the original schedule—and the deviations are a result of local, myopic considerations exclusively (Framinan et al., 2014).

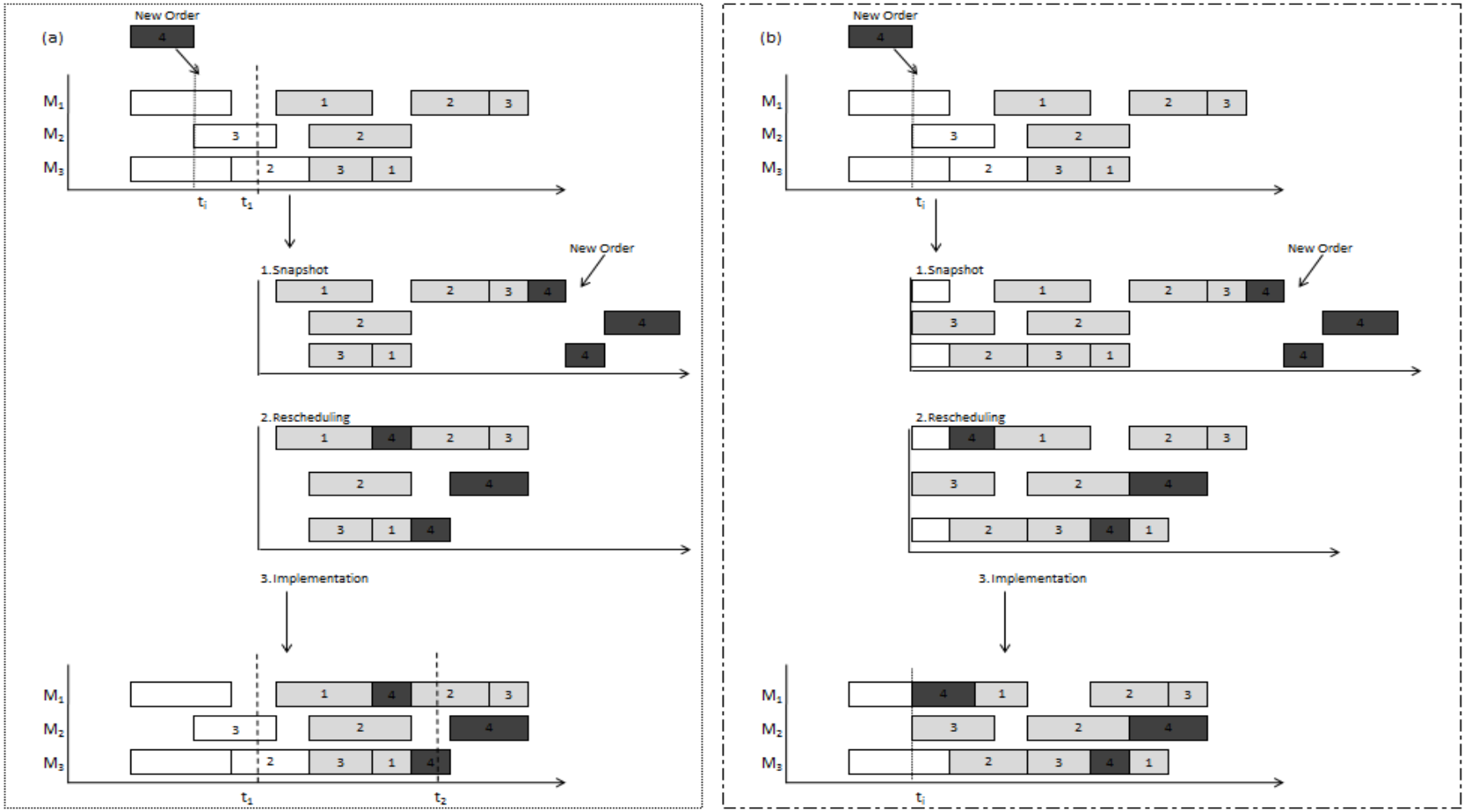


Figure 2.2 Dynamic rescheduling policies (a) Periodic rescheduling policy (b) Event-driven rescheduling policy

2.4 Discussion of the Problem

Scheduling constraints and differerent types of disruptions are discussed respectively in this section. Event-driven and periodic rescheduling policies for job shop scheduling problem under disruptions are also discussed. The proposed preemptive based multi-objective approach is explained.

2.4.1 Scheduling Constraints

2.4.1.1 Machine Non-Availability

It is generally assumed that machines are continuously available during the whole scheduling horizon in scheduling literature. This might be borne out in some cases, but in the practical manufacturing system, machines can not be ready for use all time. There are many reasons why machines are not available. Some of the examples are planned tasks and stochastic tasks. For example, preventive maintenance, cleaning, cooling are some planned tasks and these must be done a fixed time period for each machine. As a complex system with multiple components usually deteriorates with age, preventive maintenance is often performed to keep the system functioning in a good state to prolong its effective age (Lin & Huang, 2010). Preventive maintenance tasks are performed when the machines are idle and it represents one of the deterministic sources for machine non-availability (Duffuaa et al., 2000). These activities consume potential production time, but delaying these activities because of production demands may increase the chance of machine failure. In some manufacturing environment, there may be both programmed maintenance and unprogrammed maintenance activities must be considered simultaneously. If the machines reach to total aging levels, they require maintenance activity. In this thesis, the machines are maximum aging capacity. Whenever the machines reach maximum aging capacity, maintenance activity is carried out. This activity affect the machine availability and the machine utilization ratio. In this study unavailability periods are considered (which may correspond to some pre-determined preventive maintenance activity which are carried out after each shift or similar). Unavailability period can be different for each machine and they are appended at the end of each shift.

The data of 2x3 (job x machine) size for flexible job shop scheduling problem can be seen in Table 2.2. Each job arrives to the shop floor at time 0. Each machine has different unavailable time at the end of the each shift (each shift is 8 hours). In Figure 2.3 illustrates a schedule with unavailable time constraint for the data which is given in Table 2.2.

Table 2.2 Example problem data

Jobs	Operation	Machine no / Processing Time(hour)	Due date
Job 1	Operation 1 Operation 2	$p_{(1,1,1)}$ =2; $p_{(1,1,2)}$ =4; $p_{(1,1,3)}$ =3 $p_{(1,2,1)}$ =3; $p_{(1,2,2)}$ =3; $p_{(1,2,3)}$ =3	6
Job 2	Operation 1 Operation 2	$p_{(2,1,1)}$ =3; $p_{(2,1,2)}$ =4; $p_{(2,1,3)}$ =3 $p_{(2,2,1)}$ =4; $p_{(2,2,2)}$ =5; $p_{(2,2,3)}$ =5	14

Setup Times

Machine 1			Machine 2		
	Job 1	Job 2		Job 1	Job 2
Job 1	0.5	1	Job 1	0.5	1
Job 2	1	0.5	Job 2	1	0.5

2.4.1.2 Sequence-Dependent Setup Times

Setup time is a time which is required to prepare the necessary resources such as machines to perform a task. Setup activities may include, for example, (1) obtaining tools, positioning work in process material, returning tools, cleaning up, setting the required jigs and fixtures, adjusting tools, and inspecting material in a manufacturing system, (2) setting up a suitable environment to perform tasks in a service organization, (3) transferring programs and their dependent files as well as their run times in a computer system, and (4) holding the data signals steady before the clock event so that the signals sampled by the clock are reliable in synchronous circuits such as the flip-flop (Allahverdi & Soroush, 2008). Setup time is classified into two categories, sequence-dependent and sequence-independent setup time. If setup time depends on the processed operation and its preceding operation on the machine, it is called sequence-dependent setup time. Each operation of order has their own characteristic. The degree of similarity between consecutive operations; e.g. size, colors etc. influences on the setup duration. For example, sequence-dependent setup time encounters in painting industry, textile industry, chemical industry, printing

industry, paper industry, plastic industry, metal processing, and many other industries. In painting industry, a setup time is used to prepare the machine for operating the colour of the next job and setup time depends on the colour of the current job and the colour of following job. In textile industry, a setup time relies on the sequence of jobs for waving and dying operations. In plastic indusrty, different types, sizes and colours of the products effect the sequence-dependent setup duration. The other side, in the container/bottle industry, sizes and shapes of the container/bottle effect the sequence-dependent setup duration. On the other hand, setup time depends on only the processed task in the sequence-independent setup time. Furthermore, setup time might be anticipatory or non-anticipatory. Anticipatory setups are performed at the machine as soon as this machine finishes the previous job in the sequence. A machine cleaning is an example of anticipatory setup activity. Non-anticipatory setups are performed whenever the job is available at that machine for the setup. In standard FJSSP, setup time is included in the processing times or neglected to simplify the problem. But, this is not a realistic application. In our thesis, sequence-dependent and non-anticipatory setup times is considered. For example, the schedule in Figure 2.3 as the orijinal schedule and the setup times between jobs are given in Table 2.2, if the operation is the first assigned operation on the machine, setup time is taken half an hour. Figure 2.4 depicts a solution.

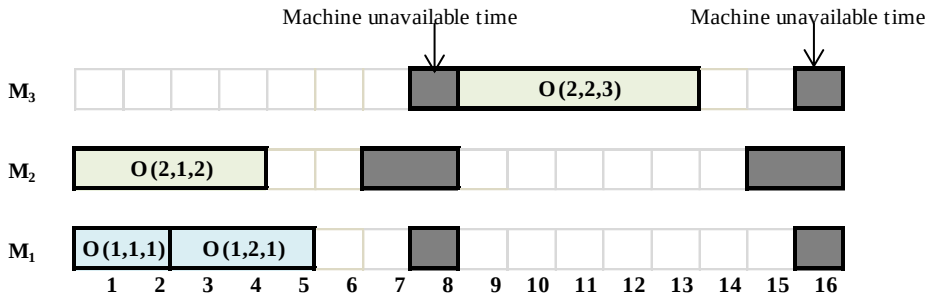


Figure 2.3 Machine non-availability

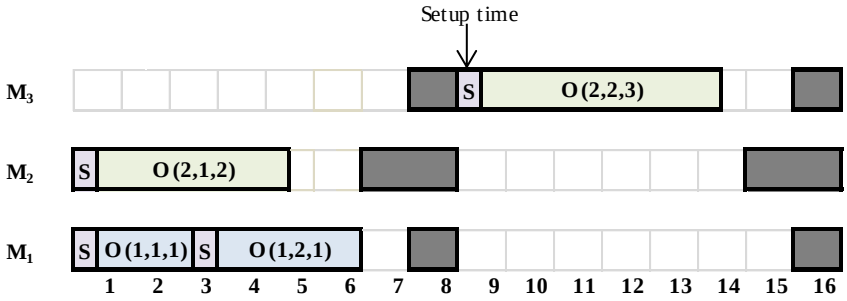


Figure 2.4 Sequence-dependent setup times

2.4.1.3 Order Acceptance/Rejection Decision

In today's competitive manufacturing environment, an organization must respect order deadlines agreed with customers, but order acceptance often takes place without consideration of the effect on the planning of the other jobs in the order portfolio (Nobibon et al., 2009). The order acceptance/rejection decision and scheduling are often functionally separated in the traditional manufacturing system and the order acceptance/rejection decision is made when an order arrives at the manufacturing system. The order acceptance/rejection decisions are made by sales department and production scheduling is prepared by the production department. Under these circumstances, the sales department is inclined to accept as many orders as possible with the due date of the customer orders to maintain customer satisfaction and boost sales and the production planning department attempts to live up to promised due dates under limited production capacity to maintain customer satisfaction and use the production capacity more efficiently. In the real manufacturing system, all of the arrrived orders can not be accepted because of the limitation of the machine capacity. In this confusion, a tactical planning activity that deals with acceptance and rejection customer orders which is named as order acceptance/rejection is applied. The order acceptance/rejection decision depends on the release time of each order, available shop capacity within the time window, the completion time of each order and the due date of each order. So a trial schedule is generated to make the acceptance/rejection decision. If the new order can be delivered on desired due date under available capacity, then the order can be accepted. Whenever processing requirements for unscheduled operations exceed available capacity, the new order can be rejected. If there is more than one such order

arriving in a scheduling period, the order is dispatched to the shop floor in accordance with the dispatching rules. Figure 2.5 shows the information flow between these departments. The order acceptance/rejection decision depends on the arrival time of each order, available shop capacity within the time window, the completion time of each order and the due date of each order. So a trial schedule is generated to make the acceptance/rejection decision. If the new order can be delivered on desired due date under available capacity with no lateness, the order can be accepted. Whenever processing requirements for unscheduled operations exceed available capacity, the new order can be rejected. The order acceptance/rejection decision process can be defined as follows. The total number of orders arriving in scheduling period is represented O, C_i is the completion time of the *i*th order, d_i is the due date of the *i*th order, T_i is the tardiness of the *i*th order.

Procedure Order acceptance/rejection decison

Initiliaze Scheduling period starts.

1. *Sort the new arriving orders based on dispatching rule EDD (earliest due date).*
   Set order number h=1
2. *Repeat until (h=O)*
   a.Calculate C_i for an order i,
   b. Accept the order, if $T_i \leq 0$ ($C_i \leq d_i$)
   c. Else Reject the order.
   d. Set h=h+1.

[End of the step 2 Loop]

[End Order acceptance/rejection decision]

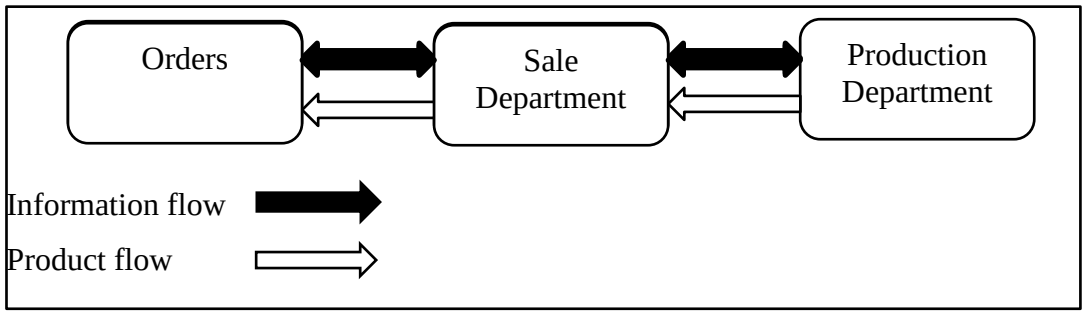


Figure 2.5 Cooperation department in order acceptance/rejection decision

Consider the schedule in Figure 2.4 as orijinal schedule with three machines and two order; at time 7, one additional order 3 identical with J_2 arrives and due date of the new order is 35. A sample result which is given in Figure 2.6 (b) is obtained and the new order can be delivered at time 21.5. The new order is accepted because this order is delivered on the desired due date (Figure 2.6 (b)).

2.4.1.4 Order Review Release(ORR)

The controlling of input releasing has a great effect on the system performance. ORR function which is mentioned as input sequencing (Hutchison, 1991), input/output control (Onur & Fabrycky, 1987), controlled release (Wight, 1970), input control (Baker, 1984), and input regulation (Uzsoy et al., 1994), performs the input regulation, in manufacturing systems.

The goal of ORR mechanism is to improve the system performance by controlling the flow of orders to the shop floor. These improvements can be achieved in terms of increased flexibility, decreaced work in process, improved delivery performance, reduced congestion and manufacturing lead times (Bobrowski & Park, 1989). Orders are not released to the shop floor immediately without considering any information about the system. The specification of the release time is a decision that effects the shop performance. ORR systems should be designed carefully and executed and implemented effectively. Early releases result in congestion in the shop floor, higher inventory costs, restricted factory space and late releases result in missed due dates, idle reources, increased lead times.

A complete ORR system has three major parts; (a) Order entry phase, (b) Pre-shop pool (PSP) management phase, (c) Order release phase. According to the guideline laid by Bergamaschi et al. (1997) the activities of ORR was summarized as follows:

(a) *Order Entry Phase*: The ‘order entry’ (OE) phase is defined to be the upstream interface of ORR with the planning system or directly with customers. Firstly, the acceptance/rejection decision of orders to the planning system is executed in this

phase. Then, production order preparation is managed. Order preparation is ensured that all accepted orders has all of the information, materials, tools, fixtures and all other resources required by the shop floor personnel (ie., determination of order's routing, assignment of due dates etc.). After the accepted order has been ensured, this order is inserted in the PSP.

(b) *PSP Management Phase*: In this phsse, all of the production orders are stored in the PSP storage area which can be a physical or electronic environment (i.e., database). The physical raw materials of orders are stored in the physical environment for PSP or the associated paperwork of orders are stored in the electronic environment for PSP. None of the production orders processed by the OE phase can reach the shop floor without having to pass through the PSP (Hendry & Wong 1994). Orders arriving from the OE phase are queued at this gateway according to the predetermined manner. The orders which are in the pre-shop pool, are sequenced according to their priority. In the literature, earliest due date (EDD), earliest release date (ERD), critical ratio (CR), capacity slack based rules are some of the rules which one used as priority rules.

(c) *The Order Release Phase*: The selection of orders from the PSP in order to release to the shop floor is the key issue to the successful use of an ORR strategy. The stored orders in the PSP are released to the shop floor according to a set of criteria. The order release mechanism determines when and which conditions the orders in PSP will be released to the shop floor .

In the literature, sevaral mechanisms have been studied. In this thesis, forward finite loading mechanism (FFLM) is applied. FFLM (Sabuncuoglu & Karapınar, 1999) situates all operations of the orders using available capacity and determines the releasing time of the orders. This mechanism uses information about the job (due date, number of operations, processing time) and the shop (setup time, the load profile for each machine). FFLM executes in the planning horizon which is considered in 8-hour shifts loading periods. The loading period uses a future time period with the unassigned workcentre capacity. This mechanism considers available

shop capacity over planning horizon and match machine requirement of the orders with available capacity. A trial schedule is constructed with ORR mechanism and it begins the loading process with the order's first operation and attempt to place each operation taking into account available capacity of the machines. ORR mechanism compares the due date of the orders with the completion time of the orders and the release time of the order is determined by the summation of the arrival of the order and the difference due date of the order and completion time of the order. If the job's release time is adjusted after the executing ORR mechanism, it keeps the total workload of orders until the operations are finished. However, whenever other dynamic events occur, the assigned operations and workloads can be changed.

Consider the schedule in Figure 2.6 (b) as orijinal schedule; the third order is completed before the due date at time 35. After the ORR mechanism is applied, a sample solution is constructed as shown in Figure 2.7 and third order is released to the shop floor at time 19.5.

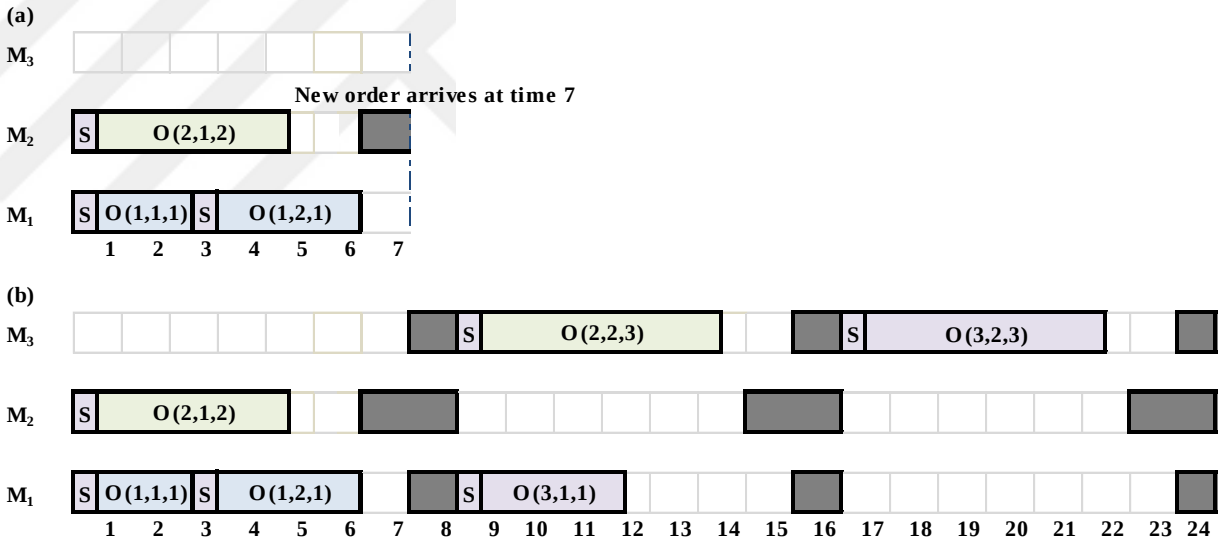


Figure 2.6 Order acceptance/rejection decision

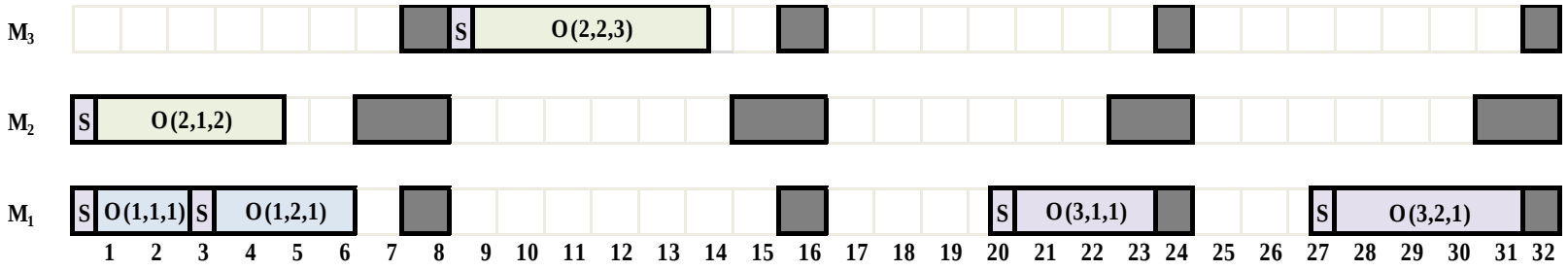


Figure 2.7. Order review release

2.4.2 Rescheduling Under Different Disruptions

In this thesis, we consider five most common types of disruptions: new order arrivals, machine breakdowns, rush order, changes of the due dates, and order cancellation.

2.4.2.1 Machine Breakdown

In real manufacturing system, when a machine breaks down, the non-started operations delay until the disruption is over or the non-started operations are reassigned to another machine after the rescheduling process. The beginning and completion time of non-started operations are arranged depending on repair time of the breakdown machine. During and after the machine repair process, the continued implementation of the schedule generated earlier would delay the completion of some jobs due to active precedence constraints. Such delays may create a contractual problem with customers. In order to minimise the delay, it is very important to re-optimise the remaining tasks at the time of machine breakdown and the schedule must be rescheduled again as soon as the repair is finished.

Consider the schedule in Figure 2.4 as original schedule with three machines and two order; at time 5 machine M3 breaks down and the repair time is 9. When the machine breaks down, some operations (O(1,1,1), O(2,1,2)) are completed and O(1,2,1) is ongoing. The initial partial schedule and a sample schedule is shown in Figure 2.8 (a) and 2.8 (b) respectively.

2.4.2.2 Order Cancellation

This disruption indicates that a job (order), already scheduled for processing in the shop, is no longer required (Subramaniam et al., 2005). After the order cancellation disruption, the operations of the cancelled order are not inserted, on the contrary, the operations of the cancelled order are removed. This dynamic events trigger the rescheduling process. Then the remaining operations of the canceled order are deleted from the schedulable task list and the rescheduling process is operated.

Consider Figure 2.8 as original schedule; at time 6 second order is cancelled. A sample schedule is shown in Figure 2.9.

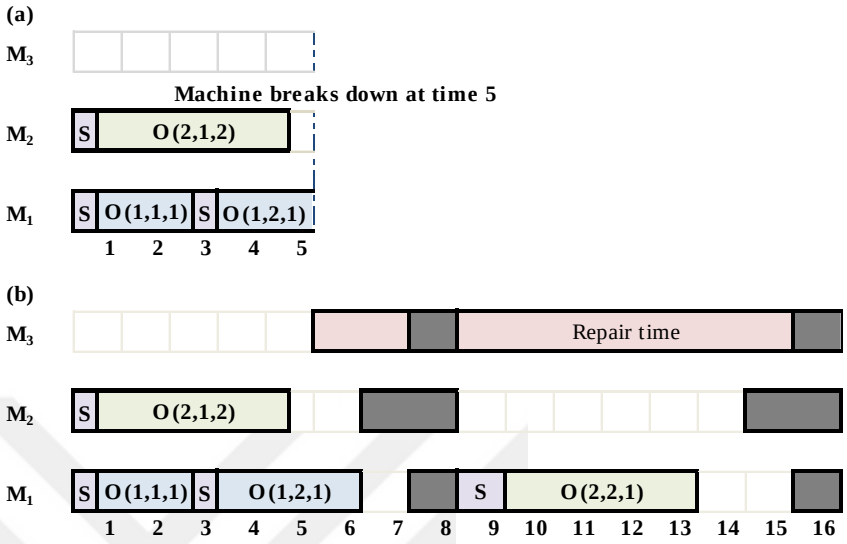


Figure 2.8 Machine breakdown. (a) Partial schedule at time 5. (b) A sample schedule

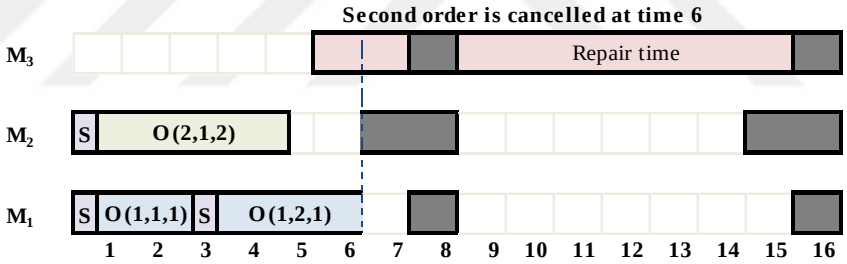


Figure 2.9 Order cancellation

2.4.2.3 New Order Arrival

In the real manufacturing systems, new order arrivals may occur during the production process. Rescheduling is triggered with the new order arrival. But, all of the orders mustn't be accepted, some orders may be rejected because of the machine capacity constraint. In the proposed system, after an order arrives, the order acceptance/rejection decision is executed and then the decision is determined. After this process, accepted orders are not released to the shop floor immediately. ORR mechanism identifies the release time of orders.

Consider the schedule in Figure 2.9 as the original schedule; at time 7, one additional order 3 identical with J_2 arrives and due date of the new order is 35. When the new order arrive at time 7, some operations (O(1,1,1), O(1,2,1), O(2,1,2)) are completed. Order acceptance/rejection decision and ORR mechanism is applied. A sample result which is given in Figure 2.10 is obtained. The new order is accepted because this order is delivered on the desired due date (Figure 2.10 and the new order is released to shop floor at time 19.5.

2.4.2.4 Rush Order

When the customer wants to receive the order as soon as earlier, the order is called rush order. The rush order must be executed immediately, because due date's of the rush order is near. After the rush order arrives, it has maximal priority and the rescheduling process is triggered. The schedule is reconstructed with this additional rush order constraint.

Consider Figure 2.10 as original schedule; at time 8, a rush order (order 4) arrives and the order is identical with J_1. A sample schedule is shown in Figure 2.11.

2.4.2.5 Change of the Due Dates

In real manufacturing systems, changes of the due dates are adapted to the scheduling process because of the contractual problem with customer. Although the customers declare the due dates, some customers may postpone due date to get a later date or get an earlier date, the due date changes trigger the rescheduling process. Some customers may request late delivery of order, and then these orders will be postponed or early delivery of order. When due date is changed, the non-started operations are suspended and ORR determines the starting time of each non-started operations according to the due date.

Consider Figure 2.11 as original schedule; at time 8 due date of the third order is getting earlier as soon as possible. A sample schedule is shown in Figure 2.12.

Besides the event-driven rescheduling policy, the periodical rescheduling policy is also illustrated. Consider the schedule in Figure 2.4 as the original schedule with three machines and two orders; suppose that the rescheduling interval is 10, the original schedule can not be changed until the next scheduling point, 20. After the schedule is executed in the shop floor, some dynamic events happen: Machine 3 breaks down at time 5; Order 2 is cancelled at time 6; Order 3 (which is identical with Job 2) arrives at time 7 with due date 35; rush order (Order 4, which is identical with Job 1) arrives at time 8 and the due date of the third order is changed at time 8. Dynamic events are depicted in Figure 2.13 (a). A sample schedule which is generated by the periodical rescheduling policy is shown in Figure 2.13 (b).

2.4.3 Fitness Evaluation by Preemptive Goal Programming

An objective function is a measure of how good a solution is (Baykasoğlu, 2005). Minimizing mean tardiness, minimizing makespan, minimizing mean flow time, minimizing schedule unstability are considered as objective function in this thesis. Minimizing mean tardiness, minimizing makespan, minimizating mean flow time have been chosen to evaluate the solutions efficiency. Makespan can be defined as the time difference between the start (zero) and completion time of the last job. Tardiness is calculated by taking the difference between the completion time and due date for each job which the completion time happens after the due date. Mean tardiness is the average tardiness of all jobs in the schedule. Flow time is the time a job spends in the manufacturing system. Mean flow time is the average flow time of all jobs in the schedule. When dynamic events occur, the non-started operations are scheduled with the proposed GRASP algorithm and this inevitably will result in deviation from the previous schedule. A stable schedule deviates minimally from the original schedule. The deviation from the previous schedule is a significant criteria for the performance of the dynamic scheduling. For this reason, an unstability criterion is added to the list of objectives to be optimized. Schedule unstability is added to the list of

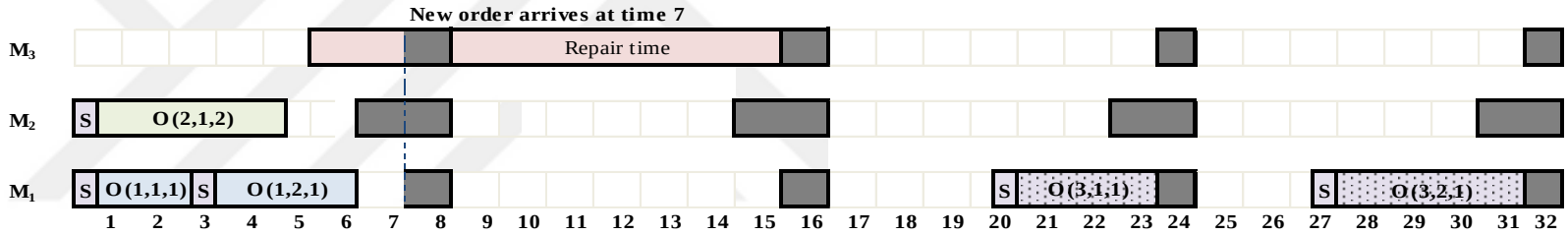


Figure 2.10 New order arrival

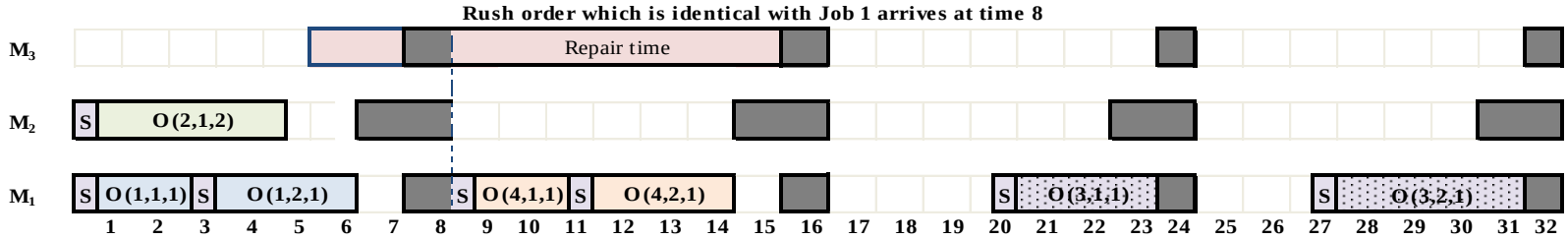


Figure 2.11 Rush order

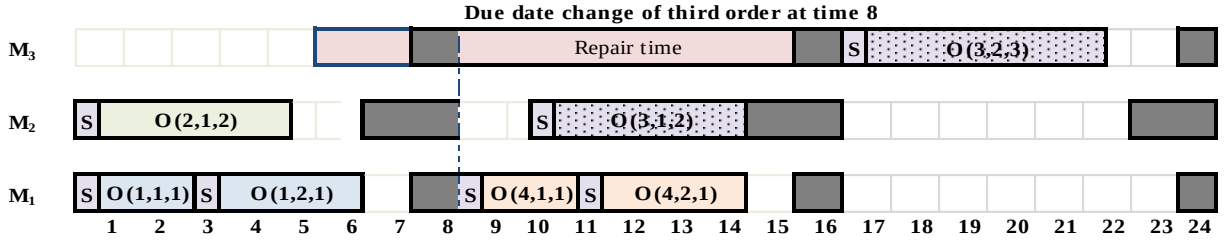


Figure 2.12 Due date change

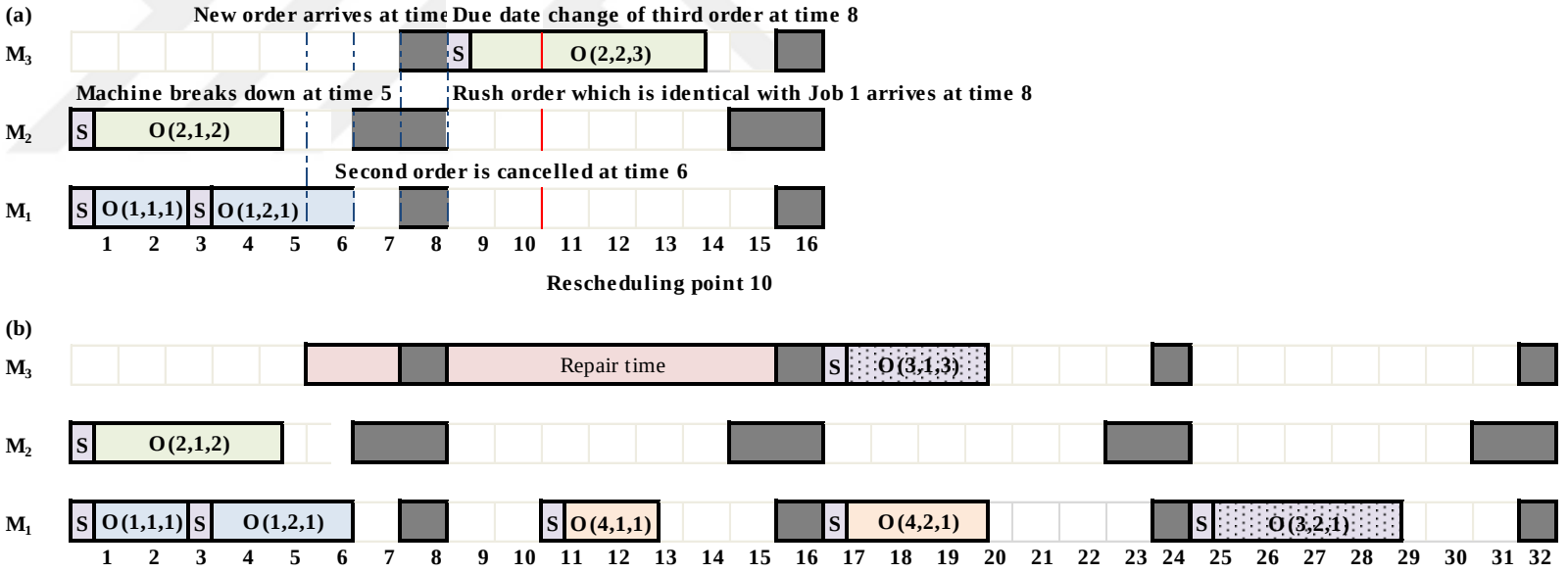


Figure 2.13 Periodical rescheduling. (a) Partial schedule at time 10 (b) A sample schedule

objectives to be optimized. Schedule unstability is modeled as shown in Equation 2.5. The measure is the total deviation for all of the operations between the starting times in a new schedule and old schedule.

$$Schedule\ Unstability = Starting\ time\ deviation = \sum_{all\ eligible\ n} \sum_k \sum_m \left| t^l_{n,k,m} - t_{n,k,m} \right| \quad (2.5)$$

Where, $t_{n,k,m}$ is starting time of operation *k* of job *n* on the machine *m* per a schedule built at one rescheduling point. $t^l_{n,k,m}$ is starting time of operation *k* of job *n* on the machine *m* per a schedule built at the next rescheduling point.

Generated solutions are evaluated by making use of preemptive goal programming logic. We make preferences among the above mentioned objectives and put them in a priority order. The current best solution within the neighbourhood solutions is determined by preserving this priority. This evaluation procedure is adopted from Baykasoğlu (2005). For example; assume that the primary objective is minimizing mean tardiness. If the primary objective of the neighborhood solution is better than the primary objective of the current best solution, the best solution is updated. If the primary objective of the neighborhood solution and the primary objective of the current best solution are equal, then the secondary objective is evaluated. The secondary objective is minimizing schedule unstability. If the secondary objective of the neighbourhood solution is better than the secondary objective of the current best solution, the current best solution is updated. The third objective is minimizing makespan and fourth objective is minimizing mean flow time. The explained procedure is applied to the other objectives, respectively. The preemptive goal programming based evaluation logic can be formalley stated as follows:

$$\begin{aligned} \{solution(j)\ \ is\ better\ than\ solution(i)\} \rightarrow & \left\{ if\ \left(MeanTardiness(j) < MeanTardiness(i) \right) \right\} \vee \\ & \left\{ if \begin{pmatrix} MeanTardiness(j) = MeanTardiness(i)\ \wedge \\ ScheduleUnstability(j) < ScheduleUnstability(i) \end{pmatrix} \right\}_{\vee,\ldots,\vee} \\ & \left\{ if \begin{pmatrix} MeanTardiness(j) = MeanTardiness(i)\ \wedge \\ ScheduleUnstability(j) = ScheduleUnstability(i)\ \wedge,\ldots,\wedge \\ Makespan(j) = Makespan(i)\ \wedge \\ MeanFlowTime(j) < MeanFlowTime(i) \end{pmatrix} \right\} \end{aligned} \quad (2.6)$$

Some examples are given in the following to clarify this process.

Best feasible solution so far :	Current feasible solution:	Decision:
{50, 1200, 120, 600}	{45, 1500, 100, 650}	improved Solution
{40, 1000, 110, 500}	{40, 950, 100, 450}	improved Solution
{38, 1500, 140, 700}	{38, 1500, 140, 600}	improved Solution
{42, 1450, 150, 800}	{42, 1490, 150, 800}	non-improved solution
{42, 1450, 150, 800}	{42, 1450, 150, 950}	non-improved solution

At the first example, mean tardiness value is decreased in the current feasible solutions. So, there is no need to look into other objectives. We accept the current solution as the best solution. Example two illustrates that unstability value is decreased and then the solution is accepted. At example three, the last objective mean flow time value is smaller than current solution value and the current solution is accepted. Unstability value is greater than unstability value of the current solution and the current solution is rejected at example four. At the last example, the last objective value is greater in current solution. So, the best solution found so far is not updated in the last two examples.

2.5 Literature Review

We review the research papers that are related to dynamic job shop scheduling problem and dynamic flexible job shop scheduling problem. A classification related to these problems is constructed to demonstrate how complex the proposed problem is. In Table 2.3, a taxonomy to a better understanding of previous works has been proposed. This taxonomy contains seven main sections: the objective function, method, production, problem type, data, modelling, frequency of scheduling, dynamic events. The performance of the system is evaluated by using various measures based on the objective function. Objective function section identifies the specific goals of each paper. The optimization techniques of the problem solution are classified in the utilized method section. Production feature section specifies the performing an operation on alternative machines and the possibility of producing the same manufacturing feature with alternative operations or sequences of operations. Problem feature section defines the specific features of the manufacturing systems such as daily production capacity, setup times. Data type section states which types

of data (deterministic, stochastic, fuzzy or hybrid) is used. Modelling approach section defines implementation techniques of the problem environment. The rescheduling policies are described with the frequency of scheduling section. Three policies have been proposed in the literature: periodic, event-driven, and hybrid. Dynamic events section specifies uncertainty and disturbance sources in the manufacturing systems.

The first study in dynamic job shop scheduling was published by Nelson et al. (1977) who proposed a scheduling system for a job shop with intermittent job arrivals. They implemented a multi-pass procedure by generating schedules periodically. Fang and Xi (1997) used a hybrid of genetic algorithms and dispatching rules to solve job shop scheduling problem with sequence-dependent setup time and due date constraints under job arrivals, machines breakdown, due date change dynamic events. Holthaus (1999) presented the simulation-based analysis of dispatching rules for dynamic job shop scheduling under machine breakdowns with performance measures: mean and maximum flow time, variance of flowtime, mean and maximum tardiness, percentage of tardy jobs and variance of tardiness. Kutanoglu (1999) tested new scheduling techniques using simulation under new order arrivals. Aydin and Oztemel (2000) presented an intelligent agent based dynamic scheduling system under new job arrivals. Sabuncuoglu and Bayiz (2000) tested different scheduling policies under machine breakdowns in dynamic job shop system and measured the effect of system size, type of work allocation on the system performance, mean tardiness and makespan criteria. Subramaniam et al. (2000) suggested three machine selection rules for a job shop scheduling problem. Qi et al. (2000) proposed parallel multipopulation genetic algorithms for the dynamic job-shop scheduling problem under new job arrival. Chryssolouris and Subramaniam (2001) presented a genetic algorithm for dynamic flexible job shop scheduling under machine breakdowns. Kutanoglu and Sabuncuoglu (2001) proposed iterative simulation-based scheduling mechanisms for manufacturing systems that operated in dynamic and stochastic environments. Dewan and Joshi (2002) developed an auction mechanism that can be used to make a schedule within a distributed decision-making environment for DJSSP. Rangsaritratsamee et al. (2004) suggested a genetic

local search prosedure to construct the schedules at each periodic rescheduling point for improving efficiency and schedule unstability measures with new job arrivals. Dominic et al. (2004) proposed an efficient dispatching rules for dynamic job shop scheduling by combining different dispatching rules using simulation under various rules and performance measures. Shnits et al. (2004) suggested a two-level control hierarchy for the DFJSSP. A dominant decision ciriterion and relevant scheduling rules are determined in the first step. And then simulation is used to obtain the best scheduling policy to be selected. New orders, changes in order priorities, machine breakdowns and machine flexibility, alternative route of production are considered. Liu et al. (2005) applied a metaheuristic to solve the job dynamic job shop scheduling problem. Branke and Mattfeld (2005) presented an evoluationary algorithm for the DFJSSP under order arrivals non-deterministically over time. Yu and Ram (2006) proposed response threshold model for dynamic manufacturing scheduling with flexible routing and sequence dependent setups. They only modeled machines as agents and compared their approach with several dispatching rules. Kang et al. (2007) considered the dynamic events such as new order arrival, machine brakdown, order cancellation in the job shop scheduling problem with the new structure of multi-agent system which is combined with ant colony optimization algorithm. Suwa and Sandoh (2007) proposed a new when-to-schedule policy in reactive scheduling under job shop environment with machine breakdowns. Xiang and Lee (2008) present a multi-agent system equipped with ant colony intelligence to provide an efficient scheduling with sequence-dependent setup constraints, and various dynamic disturbances. Vinod and Sridharan (2008) developed a discrete event simulation model of the job shop system in which setup times are sequence dependent. The new order arrival is considered as dynamic events. The results of this paper indicate that sequence-dependent setup time has a significant impact on the shop performance. Zhou et al. (2009) used ant colony optimization in dynamic job shop with event driven scheduling strategy and new job arrival dinamic events and compared the results with the results of dispatching rules. Vinod and Sridharan (2009) developed regression-based models for scheduling in the dynamic job shop environment with sequence-dependent setup times. Gholami and Zandieh (2009) integrated simulation into genetic algorithm for the flexible job shop scheduling

problem with machine breakdowns. Zandieh and Adibi (2010) presented a variable neighborhood search method for a multi-objective DJSS problem. They considered a JSP with random job arrivals and machine breakdowns under event-driven scheduling strategy. Their multiobjective performance measure consisted of two efficiency criteria (makespan and tardiness). The proposed method was compared to some common dispatching rules by using a simulated job shop under varied conditions. The results indicated that the performance of the proposed method is significantly better than those of the common dispatching rules. Fattahi and Fallahi (2010) developed a multi-objective mathematical model and a genetic algorithm for dynamic flexible job shop scheduling problem. Vinod and Sridharan (2011) analyzed the effects of due-date assignment methods and scheduling rules on the performance of the dynamic job shop production system by using a discrete-event simulation model. The performance of the system is evaluated by using various measures based on flow time and tardiness of jobs. Rajabinasab and Mansour (2011) developed a multi-agent scheduling system which used a pheromonebased approach for coordination among agents to obtain a schedule under job shop problem considering dynamic events such as stochastic job arrivals, uncertain processing time, and unexpected machine breakdowns. The considered job shop problem had routing and process flexibility. The proposed approach is compared with five dispatching rules by using simulation tools. The mean flow time, maximum flow time, variance of flow time, percentage of tardy jobs, mean tardiness, maximum tardiness, and variance of tardiness are the performance measures. Al-Hinai and ElMekkawy (2011) proposed a two-stage hybrid genetic algorithm for the flexible job shop scheduling problem with random machine breakdowns under bi-objective robustness and stability measure. Lou et al. (2012) proposed multi-agent-based proactive-reactive scheduling for dynamic job shop scheduling problem. Wang et al. (2013) proposed effective novel genetic algorithm which uses special chromosomes encoding to handle flexible job shop scheduling problems under machine disruption. Geyik and Dosdoğru (2013) used genetic algorithm to solve the optimization phase of solution methodology and discrete-event system simulation model is utilized to these machine-operation pairings for estimating their performances. Zhang et al. (2013) used hibrid scheduling strategy with a hybrid genetic algorithm and tabu

search for multi-objective dynamic job shop scheduling problem under random job arrivals and machine breakdowns to achieve two performance measures, which contain the schedule efficiency and the schedule stability and compared their results with dispatching rules. Scholz-Reiter et al. (2013) presented a new scheduling method that combined shifting bottleneck prosedure and the variable neighborhood search for job shop scheduling under new job arrivals to minimize job tardiness and constructed the schedule with rolling time horizon procedure. They compare the results with the the results of dispatching rules. Lu and Romanowski (2013) described multicontextual functions the unique characteristics of a dynamic job shop at a specific time. 11 basic dispatching rules and 33 composite rules made with multicontextual functions that describe machine idle time and job waiting time are analyzed. Nie et al. (2013) presented an approach based on gene expression programming which automatically constructs reactive scheduling policies for the dynamic flexible job shop scheduling problem with job release dates. Qiu and Lau (2013) presented an idiotypic network model whose antibody's action part is the dispatching rule for solving the multiobjective dynamic online job shop scheduling problem and they demonstrated the capability and efficiency of the proposed model. The used performance measures are minimizing the weighted mean flow time, minimizing the maximal flow time, minimizing the weighted variance of flow time, minimizing the weighted mean tardiness, minimizing the maximal tardiness, minimizing the weighted variance of tardiness, minimizing the percentage of tardy jobs, minimizing the weighted mean lateness, minimizing the maximal lateness, minimizing the weighted variance of lateness, maximizing the utilization level of machines. He and Sun (2013) combined new idle time insertion strategy and routing changing combined with right-shift strategy to obtain robust and stable performance under dynamic job shop scheduling environment. Xiong et al. (2013) presented a multi-objective evolutionary algorithm for a flexible job-shop scheduling problem with random machine breakdowns under robust scheduling objective. Sarker et al. (2013) proposed a hybrid evolutionary algorithm for solving job shop scheduling problem under machine breakdowns. Adibi and Shahrabi (2014) proposed a clustering-based modified variable neighborhood search (VNS) algorithm for dynamic job shop scheduling problem and used an event driven scheduling strategy.

The results are compared with the results of VNS, dispatching rules and the performence of the poposed method is better than VNS, dispatching rules. Zambrano et al. (2014) presented two hybridisations of genetic algorithm and distributed arrival-time control with the minimisation of due date mean-squared deviation for dynamically controlling a flexible job shop scheduling problem. Gao et al. (2015) presented a two-stage artificial bee colony algorithm for flexible job shop scheduling problem with new job arrivals. Sharma and Jain (2015) presented a stochastic dynamic job shop scheduling problem (SDJS) while considering sequence-dependent setup times. A discrete-event simulation model of the dynamic job shop manufacturing system was developed. The performance of nine dispatching rules which were taken from literature was assessed. The performance meaasures are makespan, mean flow time, maximum flow time, mean tardiness, maximum tardiness, number of tardy jobs, total setups and mean setup time. Ali et al. (2015) proposed a genetic algorithms to solve multiple objective job shop scheduling problems with alternative routes and dynamic job arrival in order to simultaneously minimize the maximum lateness and makespan. Shen and Yao (2015) suggested a multi-objective evolutionary algorithm-based proactive–reactive method and a new mathematical model for the multi-objective dynamic flexible job shop scheduling problem. Liu et al. (2015) proposed a fast estimation of distribution algorithm for the dynamic flexible job-shop scheduling problem with fuzzy processing time. Kundakcı and Kulak (2016) proposed efficient hybrid genetic algorithm methodologies for minimizing makespan in dynamic job shop scheduling problem. Various benchmark problems were generated and detailed numerical experiments were carried out to evaluate the performance of the proposed methodologies. Jin et al. (2016) presented a novel mixed integer linear programming and non-dominated sorting genetic algorithm-II with three criteria for an integrated process planning and schedule to analyze the impact of peridic and event-driven rescheduling on manufacturing environment. Gao et al. (2016) investigated two stages artificial bee colony algorithm to solve the flexible job shop scheduling problem with fuzzy processing time and new job insertion. Ning et al. (2016) presented a new multi-objective mathematical model for the dynamic flexible job shop scheduling problem with periodic and event-driven rescheduling policy. Xiong et al. (2017) considered a simulation based

approach for dynamic job shop scheduling problem to analyze dispatching rules with batch release and extended technical precedence constraints.

Table 2.3 summarize job shop scheduling papers in the literature under dynamic events and depicted how complex the proposed problem is. There are a few papers in the literature which handle real life like cases such as new order arrivals, machine breakdowns, rush orders, order cancellations, due date changes, setup times, machine capacity as a whole. In this thesis, we proposed a scheduling system which reflects the real life like situations such as machine capacities, setup times, dynamic events, order acceptance rejection and order review release mechanism all together.

Table 2.3 An overview of the approaches in the literature on dynamic job shop scheduling problem

Publications	Objective Function	Utilized method				Production feature		Problem feature			Data type				Modelling approach			Frequency of scheduling			Dynamic events					
		Exact	Heuristic	Metaheuristic	Hibrid	Altemative Routes	Altemative Process Plan	Capacitated	Setup Time	All	Deterministic	Stochastic	Fuzzy	Hibrdi	Math. Prog.	Agent Based	Simulation	Periodic	Event driven	Hibrid	New order arrival	Machine breakdown	Due date changing	Job cancellation	Rush order	Other
Fang & Xi (1997)	Multiobjective		X	X		X			X		X							X	X		X	X	X			
Holthaus (1999)	Mean- max variance of flowtime, mean-max tardiness, percentage of tardy jobs, variance of tardiness		X									X					X		X			X				
Kutanoglu (1999)	Weighted tardiness		X									X					X		X		X					
Aydin & Öztemel (2000)	Mean tardiness		X									X				X	X		X		X					
Sabuncuoglu & Bayiz (2000)	Mean tardiness, Makespan		X									X					X			X	X	X				X
Subramaniam et al. (2000)	Mean job cost, mean job tardiness		X			X						X					X		X			X				
Qi et al. (2000)	Multiobjective		X	X							X								X		X					
Chryssolouris & Subramaniam (2001)	Mean tardiness, mean cost			X		X						X							X			X				
Kutanoglu & Sabuncuoglu (2001)	Wegihted tardiness		X									X					X	X				X				X
Dewan & Joshi (2002)	Multiobjective		X								X				X	X			X		X					

Table 2.3 An overview of the approaches in the literature on dynamic job shop scheduling problem (continue)

Publications	Objective Function	Utilized method				Production feature		Problem feature			Data type				Modelling approach			Frequency of scheduling			Dynamic events					
		Exact	Heuristic	Metaheuristic	Hibrid	Alternative Routes	Alternative Process Plan	Capacitated	Setup Time	All	Deterministic	Stochastic	Fuzzy	Hibrdi	Math. Prog.	Agent Based	Simulation	Periodic	Event driven	Hibrid	New order arrival	Machine breakdown	Due date changing	Job cancellation	Rush order	Other
Rangsaritratsamee et al. (2004)	Makespan, Tardiness, Stability			X								X						X			X					
Dominic et al.(2004)	Mean –max flow time, mean tardiness, number of tardy jobs, tardiness variance		X									X					X		X		X					
Shnits et al. (2004)	Multiobjective		X			X	X					X					X		X		X	X				X
Branke & Mattfeld (2005)	Mean tardiness, flexibility		X	X		X					X							X			X					
Liu et al. (2005)	Makespan			X							X								X		X	X				
Yu & Ram (2006)	Several		X			X			X		X					X			X		X					
Kang et al. (2007)	Makespan, tardiness			X							X					X		X			X	X		X		
Suwa & Sandoh (2007)	Maximum Tardiness			X								X						X	X			X				
Xiang & Lee (2008)	Several			X		X			X			X				X			X		X	X		X		
Vinod & Sridharan (2008)	Mean flow time, mean tardiness		X						X			X					X		X		X					
Zhou et al.(2009)	Makespan, mean flow time, mean tardiness			X							X								X		X					X

Table 2.3 An overview of the approaches in the literature on dynamic job shop scheduling problem (continue)

Publications	Objective Function	Utilized method				Production feature		Problem feature			Data type				Modelling approach			Frequency of scheduling			Dynamic events					
		Exact	Heuristic	Metaheuristic	Hibrid	Alternative Routes	Alternative Process Plan	Capacitated	Setup Time	All	Deterministic	Stochastic	Fuzzy	Hibrdi	Math. Prog.	Agent Based	Simulation	Periodic	Event driven	Hibrid	New order arrival	Machine breakdown	Due date changing	Job cancellation	Rush order	Other
Gholami & Zandieh (2009)	Makespan, mean tardiness			X		X						X						X	X			X					
Vinod & Sridharan (2009)	Mean flow time, mean tardiness, mean setup time, mean number of setups		X						X			X					X		X		X					
Zandieh & Adibi (2010)	Mean flow time			X							X								X		X	X				
Fattahi & Fallahi (2010)	Makespan, Tardiness, Stability	X		X		X					X				X			X			X					X
Rajabinasab & Mansour (2011)	Several		X			X	X					X				X			X		X	X				X
Al-Hinai & ElMekkawy (2011)	Robustness, stability			X		X						X							X			X				
Vinod & Sridharan (2011)	Mean flow time, mean tardiness		X									X					X				X					
Lou et al. (2012)	Makespan		X									X				X			X			X			X	
Geyik & Dosdoğru (2013)	Mean flow time			X		X	X					X					X		X		X					X
Zhang et al.(2013)	Efficiency, stability			X								X								X	X	X				
Scholz-Reiter et al. (2013)	Weighted tardiness			X								X						X			X					
Lu & Romanowski (2013)	Makespan, mean flow time		X									X							X		X					X
Nie et al. (2013)	Makespan, mean flowtime, mean tardiness		X				X					X							X		X					
Qiu & Lau (2013)	Multi-objecitve	X	X									X					X		X		X	X				

Table 2.3 An overview of the approaches in the literature on dynamic job shop scheduling problem (continue)

Publications	Objective Function	Utilized method				Production feature		Problem feature			Data type				Modelling approach			Frequency of scheduling			Dynamic events					
		Exact	Heuristic	Metaheuristic	Hibrid	Altemative Routes	Altematice Process Plan	Capacitated	Setup Time	All	Deterministic	Stochastic	Fuzzy	Hibrdi	Math. Prog.	Agent Based	Simulation	Periodic	Event driven	Hibrid	New order arrival	Machine breakdown	Due date changing	Job cancellation	Rush order	Other
He & Sun (2013)	Average relative robust, average stability		X			X	X					X							X			X				
Xiong et al. (2013)	Makespan, Robustness			X		X						X							X			X				
Sarker et al. (2013)	Makespan			X								X							X			X				
Wang et al. (2013)	Makespan			X		X						X							X			X				
Adibi et al.(2014)	Mean flow time			X								X							X		X	X				
Zambrano et al. (2014)	Mean-squared deviarion for due dates			X		X						X						X	X		X	X				
Sharma & Jain(2015)	Several		X							X		X							X			X				
Ali et al. (2015)	Max lateness, makespan			X			X					X							X			X				
Shen & Yao(2015)	Multi-objective		X	X		X						X			X		X			X			X			X
Liu et al. (2015)	Makespan			X		X								X					X			X				
Gao et al. (2015)	Makespan			X		X					X									X		X				
Kundakcı & Kulak (2016)	Makespan			X							X								X		X	X				X
Jin et al. (2016)	Makespan, Total tardiness, stability	X		X			X				X				X			X	X		X					
Gao et al. (2016)	Fuzzy completion time			X		X							X						X		X					
Ning et al. (2016)	Makespan, stability	X				X						X			X	X		X	X		X	X				
Xiong et al. (2017)	Total tardiness		X									X					X		X		X					

CHAPTER THREE
GREEDY RANDOMIZED ADAPTIVE SEARCH PROCEDURE

3.1 Chapter Introduction

Several solution techniques including heuristic methods and metaheuristics are applied to DFJSSP, as it is stated previously. However, GRASP is not employed to solve DFJSSP so far. In this chapter, the proposed GRASP algorithm is presented for solving DFJSSP.

The rest of this chapter is organized as follows. In section 3.2, the general consept of GRASP algorithm is presented. The proposed GRASP algorithm is explained for solving DFJSSP.

3.2 Greedy Randomized Adaptive Search Procedure (GRASP)

GRASP is a multi-start, iterative metaheuristic (Feo & Resende, 1989; Feo & Resende, 1995) for difficult combinatorial optimisation problems. GRASP has been applied to many NP-hard combinatorial optimization problems with much success (Robertson, 2001). GRASP has been effectively applied to solve combinatorial problems such as single machine problems (Bard et al., 1993; Gupta & Smith, 2006), chinese postman routing problem (Corberán et al., 2002), aircraft routing problem (Argüello et al., 1997), flow shop scheduling (Prabhaharan et al., 2006), job shop scheduling problems (Aiex et al., 2003), flexible job shop scheduling problems (Rajkumar et al., 2010; Rajkumar et al., 2011), the maximum independent set problem (Feo et al., 1994), flight scheduling and maintenance base planning (Feo & Bard, 1989) and the quadratic assignment problem (Pardalos & Resende, 1994).

An implementation of a GRASP consists of four components: a greedy function, random selection, an adaptive method and a local search algorithm. These components are combined in two phases: construction phase and local search phase. The pseudo-code illustrates the main blocks of a GRASP procedure.

```
procedure GRASP( )
1       For GRASP stopping criteria not satisfied
2           Constructsolution(solution)
3           Localsearch(solution)
4           Updatesolution(solution, bestsolutionfound)
5       end for
6       return (bestsolutionfound)
end GRASP
```

At each iteration of the construction phase, the set of candidate elements is constructed to be added the partial solution without destroying feasibility. The element is designated exclusively for the problem. The choise of the next element for the construction of partial solution is determined by the evaluation of all candidate elements in a candidate list *C* with respect to *a greedy function g:* $C \rightarrow R$. This function measures the benefit of selecting each element to be included in the solution. The evaluation of the elements by this function leads to the creation of a restricted candidate list (RCL) formed by the best elements. The RCL is associated with a parameter $\alpha \in [0,1]$ and a threshold value $\mu = g_{min} + \alpha(g_{max} - g_{min})$. In fact, all candidate elements i whose incremental cost g(i) is no greater than the threshold value are inserted into the RCL, i.e. $g(i) \in [g_{min}, \mu]$. The parameter α is one and only parameter of the GRASP algorithm to determine the length of RCL. The parameter α controls the amounts of the greediness and randomness in the algorithm. A value $\alpha=0$ corresponds to a pure greedy algorithm and a value $\alpha=1$ is equivalent to a random construction. The amount of the greediness and randomness in the algorithm is determined by α. The probabilistic component of a GRASP is presented by *randomly choosing* one of the best candidates in the RCL, but not necessarily the top candidate. And the candidates in the RCL are assigned equal probabilities of being chosen. This choice technique for different solutions to be obtained at each GRASP iteration, but does not necessarily compromise the power of the adaptive greedy component of the method. After the selected element is incorporated into the partial solution, the candidate list *C* is updated at each iteration of the construction phase in order to include the new information that is gathered after the selection of the previous element *(adaptive method).* The above steps are repeated at each iteration of the construction phase until at least one candidate solution is constructed. The following pseudo code describes a basic GRASP construction phase.

procedure Greedy_randomized_construction(g(.),α,solution)
1 *Solution* ←Ø;
2 Initialize the set of candidate elements *C*;
3 Evaluate the candidate elements;
4 **while** at least one candidate element **do;**
5 g_{min}=min{g(t) | t ϵ *C*};
6 g_{max}=max{g(t) | t ϵ *C*};
7 RCL={s ϵ *C* | $g(s) \leq g_{min} + \alpha (g_{max} - g_{min})$};
8 Select an element s from the RCL at random;
9 *Solution*←*Solution* ∪{s};
10 Update the set of candidate elements *C*;
11 Reevaluate the incremental cost for all element in C
12 **end**
13 **return** *Solution*
end Greedy_randomized_construction

The construction phase of a GRASP doesn't guarantee optimality, and it is almost always beneficial to apply a local search to attempt to improve each constructed solution. A local search algorithm executes iterativelly by replacing the current solution by a better solution in the neighborhood of the current solution. It terminates when no better solution is obtained in the neighborhood. The pseudo-code of a basic local search algorithm starting from the solution *Solution* contructed in the construction phase and using a neighborhood *N* is given below.

procedure local_search(*Solution*)
1 **while** *Solution* is not locally optimal **do**
2 Find s^1 ϵ N (*Solution*) with $f(s^1)$<f(*Solution*);
3 *Solution*← s^1
4 **end**
5 return *Solution*
end local_search

3.3 GRASP for Dynamic Flexible Job Shop Scheduling Problem (DFJSSP)

GRASP is a constructive search method, it starts with an empty solution and adds solution elements to a partial solution iteratively and returns a complete solution. At each generation a new solution is re-constructed from scratch. Because of this reason any additional efforts like repairing, modifying a solution are not necessarily to handle dynamic events (Baykasoğlu & Ozsoydan, 2015). It is enable us to tailor the

length of the operation sequence whenever the dynamic event (new order arrival, order cancellation, rush order) which change the length of the operation sequence, happen. The main advantages of GRASP are its few parameters which need to be tuned, it is easy to parallelize (every process executes a complete GRASP) (Resende & Ribeiro, 2003) and it may be easily integrated into other search techniques (Blum & Rali, 2003). Therefore GRASP algorithm is convenient for the present DFJSSP (Baykasoğlu & Ozsoydan, 2015).

3.3.1 Solution Representation

In the proposed GRASP algorithm, we contemplate a single operation to be the building block of the candidate solution. Each element in the candidate solution is formed by triples *(i,j,k)*, where

- *i* depicts the order number;
- *j* depicts the progressive number of operation for order *i*;
- *k* depicts the assigned machine to operation *j*.

For example, a solution of 2 orders, 3 machines and 6 operations problem is shown in Figure 3.1. The first element O(2,1,3), means that the first operation of order 2 is processed on machine 3.

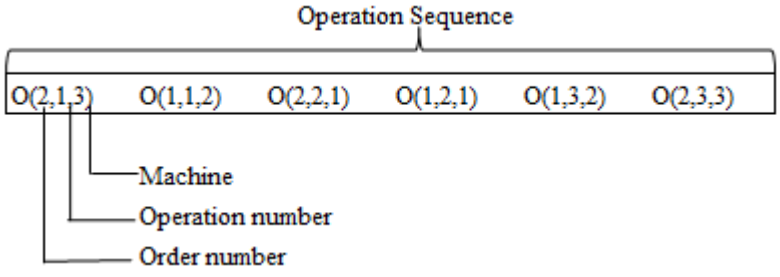


Figure 3.1 Illustration of candidate solution in JSSP

3.3.2 Generating New Schedule

The construction phase plays a very important role with respect to building high-quality starting solutions for the local search (Glover & Kochenberger, 2006). It is

difficult to find optimal solution form within all possible schedules (both feasible and infeasible), because $(n!)^m$ is the total number of all possible schedules for n jobs and m machines in the JSSP. For example, for one of the Fisher–Thompson benchmark problem (ten jobs to ten machines), it has a search space with a size of about 3.96 x 10^{65}. Therefore, the search space may be restricted and a search process may be applied. There are four types of feasible schedules in the JSSP, namely inadmissible, semi-active, active, and non-delay. Inadmissible schedules contain an excessive idle time, and they can be improved by forward-shifting operations until no excess idle time exists. Semi-active schedules contain no excess idle time, but they can be improved by shifting some operations to the front without delaying others. Active schedules contain no idle time, and no operation can be finished earlier without delaying other operations. Non-delay schedules are active schedules, in which the operations are placed into the schedule such that the machine's idle time is minimized and that no machine is kept idle if some operations can be processed (Bierwirth & Mattfeld, 1999). Figure 3.2, taken from Ge et al. (2008) depicts their relationships. The optimal schedules compose of the active schedules class (Morton & Pentico, 1993).

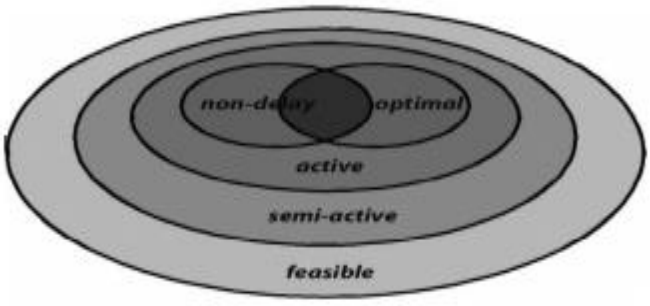


Figure 3.2 Relationship of schedule properties

Giffler and Thompson's (G&T) is adapted into GRASP construction phase to produce active schedules similar to Baykasoğlu (2002), Baykasoğlu and Özbakır (2010), Baykasoğlu et al. (2014). G&T algorithm is used to adjust job orders on each machine to resolve the infeasibility and to convert spring into active schedules. G&T algorithm may produce the easiest diverse schedules in JSSP. G&T algorithm is described as follows:

Notation

İ	job
j	operation
k	machine
O(i, j, k)	the j'th operation of job i that needs to be processed on machine k
(i,j)	the j'th operation of job i
S	the partial schedule that contains scheduled operations
Ω	the set of schedulable operations
F	the finished time of the set of schedulable operations
$s_{(i,j,k)}$	the earliest time at which operation (i,j,k) belonging to Ω could be started
$p_{(i,j,k)}$	the processing time of operation (i,j,k)
$f_{(i,j,k)}$	the earliest time at which operation (i,j,k) belonging to Ω could be finished, f(i,j,k) = s(i,j,k) + p(i,j,k)

G&T algorithm:

Step 1: Initialize S = Ø; Ω is initialized to contain all operations without predecessors.

Step 2: Determine f* = minO(i,j,k) ϵ Ω {f(i,j,k)} and the machine m* on which f* could be realized. If more than one such machine exists, tie is broken by a random choice.

Step 3: (1) Identify the operation set O(i,j,k) ϵ Ωsuch that(i,j,k)requires machine k*, and s(i,j,k)< f*.

(2) Choose O(i,j,k) from the operation set identified in 3(1) with the largest priority.

(3) Add O(i,j,k) to S.

(4) Assign s(i,j,k) as the starting time of O (i,j,k).

Step 4: If a complete schedule has been generated, stop. Else, delete O(ij,,k) from Ω and includeits immediate successor in Ω , then go to Step 2.

In the proposed GRASP algorithm, we contemplate a single operation to be the building block of the construction phase. Until all operations are scheduled, a

feasible schedule is built by assigning individual operations step by step. We can decode a partial schedule into an active schedule by using G&T heuristic. G&T algorithm is integrated into the GRASP construction phase to handle feasible solution. RCL is incorporated into the G&T algorithm and the assigned operation is chosen from the RCL randomly. G&T algorithm ensures the adaptive function of the GRASP algorithm by updating the schedulable operations and meanwhile updating completion time of the schedulable operations at each iteration to reflect the changes brought on by the selection of the previous operation.

While constructing a feasible schedule in the first step, the partial schedule that contains scheduled operations is equal to empty and the set of the schedulable operations is initialized to contain all operations without predecessors. The set of schedulable operations form the candidate operations list (C). The finishing time of the schedulable operations are calculated as a greedy function (g: C→R) which represents the incremental increase in the cost function, when one candidate operation joins the current partial solution under construction. The operation is assigned to the machine which executes the operation with minimum finishing time. RCL is embedded in G&T algorithm to adjust the greediness and randomness of the proposed algorithm. RCL is composed by best operations which have the smallest finishing time incorporated to the current partial solution (greedy aspect of the algorithm). The restricted candidate list (RCL) is defined as:

$g_{min} = min \{ f_{(i,j,k)} \mid O(i,j,k) \in \Omega \};$

$g_{max} = max \{ f_{(i,j,k)} \mid O(i,j,k) \in \Omega \};$

$RCL = \{ g(s) \in F \mid g(s) \leq g_{min} + \alpha(g_{max} - g_{min}) \};$

where α is parameter such that $0 \leq \alpha \leq 1$.

The next operation to be scheduled is chosen at random from the RCL (probabilistic aspect of the algorithm) and the selected operation is added to the partial schedule that contains scheduled operations, resulting in a new partial schedule. If a complete schedule has been generated, the construction phase ends. Otherwise the chosen operation is added to the partial schedule that contains

scheduled operations and the chosen operation is deleted from the set of the schedulable operations and its immediate successor is included in the set of schedulable operations. The iteration continues until all operations have been scheduled. The construction phase of the proposed GRASP algorithm is given as follows:

Step 1: Initialize S = Ø; Ω is initialized to contain all operations without predecessors.

Step 2: Machine, which execute the operation with minimum finishing time, are assigned to the operation.

Step 3: Compute the finish time of each operation in the set of the schedulable operations O(i,,j,k) ∈ Ω.

Step 4: Find out the minimum and maximum value of the finishing times of work load of the set of the schedulable operations and determine RCL.

g_{min} =min $\{f_{(i,j,k)} \mid O(i,j,k) \in \Omega\}$;

g_{max} =max $\{f_{(i,j,k)} \mid O(i,j,k) \in \Omega\}$;

RCL=$\{ g(s) \in F \mid g(s) \leq g_{min} + \alpha(g_{max} - g_{min})\}$;

Step 5: Randomly choose one operation from RCL and add the selected operation to the partial solution.

Step 6: If a complete schedule has been generated, stop. Else, delete the selected operation from the set of the schedulable operations and include its immediate successor in the set of the schedulable operations, then go to Step 2.

Table 3.1 shows the data of 2x2 example and the mechanism of the construction phase of the proposed GRASP algorithm is explained below and the Gannt-Chart is depicted in Figure 3.3.

Initialization

Step 1: S = Ø; Ω = {(1,1), (2,1)}

Iteration 1

Step 2: s(1,1,1) = 0, s(1,1,2) = 0; f(1,1,1) = 2; f(1,1,2) = 3,

f* = min{ f(1,1,1) ,f(1,1,2) } = 2, m* =1.

s(2,1,1) = 0, s(2,1,2) = 0; f(2,1,1) = 1; f(2,1,2) = 4,

f* = min{ f(2,1,1) ,f(2,1,2) } = 1, m* =1.

Step 3: Ω = {O(1,1,1), O(2,1,1)}, F={2, 1}.

Step 4: ṣ=1, ṡ=2,α=0.4, g≤1.4, RCL ={1}.

Step 5: S = {O(2,1,1)}.

Step 6: Update Ω = {(1,1), (2,2)}; go to Step 2.

Iteration 2

Step 2: s(1,1,1) = 1, s(1,1,2) = 0; f(1,1,1) = 3; f(1,1,2) = 3,

f* =min{ f(1,1,1), f(1,1,2) }= 3, select one machine randomly m* =2.

s(2,2,1) = 1; s(2,2,2) = 1; f(2,2,1) = 3; f(2,2,2) = 3,

f* = min{ f(2,2,1) ,f(2,2,2) }= 3,select one machine randomly m* =1.

Step 3: Ω = {O(1,1,2), O(2,2,2)}, F={3, 3}.

Step 4: ṣ=3, ṡ=3,α=0.4, g≤3, RCL ={3,3}.

Step 5: S = {O(2,1,1), O(1,1,2)}.

Step 6: Update Ω = {(1,2), (2,2)}; go to Step 2.

Iteration 3

Step 2: s(1,2,1) = 3, s(1,2,2) = 3; f(1,2,1) = 6; f(1,2,2) = 4,

f* = min{ f(1,2,1), f(1,2,2) } = 4, m* =2.

s(2,2,1) = 1; s(2,2,2) = 3; f(2,2,1) = 3; f(2,2,2) = 5,

f* = min{ f(2,2,1) ,f(2,2,2) } = 3, m* =1.

Step 3: Ω = {O(1,2,2), O(2,2,1)}, F={4, 3}.

Step 4: ṣ=3, ṡ=4, α=0.4, g≤3.4, RCL ={3}.

Step 5: S = {O(2,1,1), O(1,1,2), O(2,2,1)}.

Step 6: Update Ω = {(1,2)}; go to Step 2.

Iteration 4

Step 2: s(1,2,1) = 3, s(1,2,2) = 3; f(1,2,1) = 6; f(1,2,2) = 4,

f* = min{ f(1,2,1), f(1,2,2) } = 4, m* =2.

Step 3: Ω = {O(1,2,2)}, F={4}.

Step 4: ṣ=4, ṡ=4,α=0.4, g≤4, RCL ={4}.

Step 5: S = {O(2,1,1), O(1,1,2), O(2,2,1), O(1,2,2)}.

Step 6: A complete schedule has beeg generated, so stop the process.

Table 3.1 Example 2x2

Jobs	Operation	Machine no/Processing Time(hour)
1	Operation 1	$p_{(1,1,1)}$ =2; $p_{(1,1,2)}$ =3
	Operation 2	$p_{(1,2,2)}$ =1; $p_{(1,2,1)}$ =3
2	Operation 1	$p_{(2,1,2)}$ =4; $p_{(2,1,1)}$ =1
	Operation 2	$p_{(2,2,2)}$ =2; $p_{(2,2,1)}$ =2

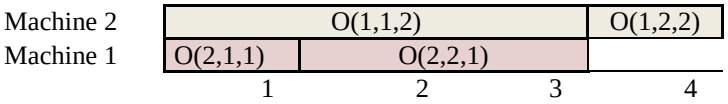


Figure 3.3 Gannt-Chart for example problem

A complete schedule is constructed by the construction phase of the proposed GRASP algorithm but generated complete schedule is not guaranteed to be a local optimum. Local search phase tries to improve the constructed solution. The effectiveness of a local search procedure depends on several aspects, such as the neighborhood structure, the neighborhood search technique, the fast evaluation of the cost function of the neighbors, and the starting solution itself. Simple neighborhoods are usually used. The neighborhood search may be implemented using either a best-improving or a first-improving strategy. In the case of the best-improving strategy, all neighbors are investigated and the current solution is replaced by the best neighbor. In the case of a first-improving strategy, the current solution moves to the first neighbor whose cost function value is smaller than that of the current solution. In practice, we observed on many applications that quite often both strategies lead to the same final solution, but in smaller computation times when the first-improving strategy is used (Glover & Kochenberger, 2006).

In this thesis, the set of possible local search operations is defined by a neighborhood of the current solution with first-improving strategy. The local search operation list is explained as follows: In swap operation, pairwise tasks are interchanged. On multiple machine problems, a pairwise interchange of tasks does not protect feasibility since the technological constraints on the task orders bring about infeasibility. Because of this reason, the swap is carried out when the obtained result is feasible. In the insertion operation, the operation at an *i*th position is removed and inserted into a random position among the feasible ones. In the

interchange operation, the operation at an *i*th and *j*th are interchanged, provided that the interchange is feasible. The assigning machine is changed with one machine which is in the machine set. In each local search iteration, the move is selected in the set of possible moves randomly. Figure 3.4 illustrates the local search operations. For every possible move, the deviation of the objective function value is computed. If the deviation is improvement, it is accepted. The accepted solution become the current solution. The local search phase of the proposed GRASP algorithm is given as follows:

Step 1: Set the number of iterations (i).
Step 2: Select a move from the set of moves (swap, interchange, insertion, changing of the assigning machine).
Step 3: Construct the neighborhood solution.
Step 4: Calculate deviation of the objective function value.
Step 5: If deviation is an improvement, neighborhood sequence is accepted and update the best schedule.
Step 6: Repeat all the steps until i reaches the local search iteration number.

Figure 3.5 shows an illustrative example of the revision process for swap move. Let us consider, at time *t* machine M_3 breaks down and rescheduling process is executed. After the construction phase, the solution, which is given in Figure 3.5(a), is obtained. In the local search phase, swap move is selected. Swap operations between jobs 1 and 4 on machine M3 which is selected randomly (Figure 3.5 (b)). However, there is a conflict for job 1 on machine M_2 and M_3 (Figure 3.5 (b)). In order to maintain feasibility, moreover, job 3 and job 1 on machine M_2 are interchanged (Figure 3.5 (c)).

The data of 4x6 (job x machine) job shop scheduling problem can be seen in Table 3.2. After the construction phase, this solution which is given in Figure 3.6, is obtained. The local search phase can be illustrated with an example which is given 3.7. Insertion move is selected from the neighborhood list randomly. In Figure 3.6,

the second operation of job 1 was inserted into 4'th point, resulting in an improvement of the makespan.

(i) Before the neighborhood move
Solution:
{O(1,1,6),O(2,1,2),O(2,2,1),O(4,1,4),O(4,2,5),O(3,1,6),O(2,3,1),***O(1,2,1),***O(3,2,3), O(1,3,4)}Makespan: 16

(ii) After the neighborhood move
Solution:
{O(1,1,6),O(2,1,2),O(2,2,1***),O(1,2,1)***,O(4,1,4),O(4,2,5),O(3,1,6),O(2,3,1),O(3,2,3), O(1,3,4)}Makespan:13

Table 3.2 Data of a job shop scheduling problem

Job	Machine Sequence	Processing Time
1	O(1,1,6)O(1,2,1) O(1,3,4)	$p_{(1,1,6)}$=3, $p_{(1,2,1)}$=2, $p_{(1,3,4)}$=4
2	O(2,1,2), O(2,2,1), O(2,3,1)	$p_{(2,1,2)}$=5, $p_{(2,2,1)}$=1, $p_{(2,3,1)}$=4
3	O(3,1,6) O(3,2,3)	$p_{(3,1,6)}$=6, $p_{(3,2,3)}$4
4	O(4,1,4) O(4,2,5)	$p_{(4,1,4)}$=2, $p_{(4,2,5)}$=1

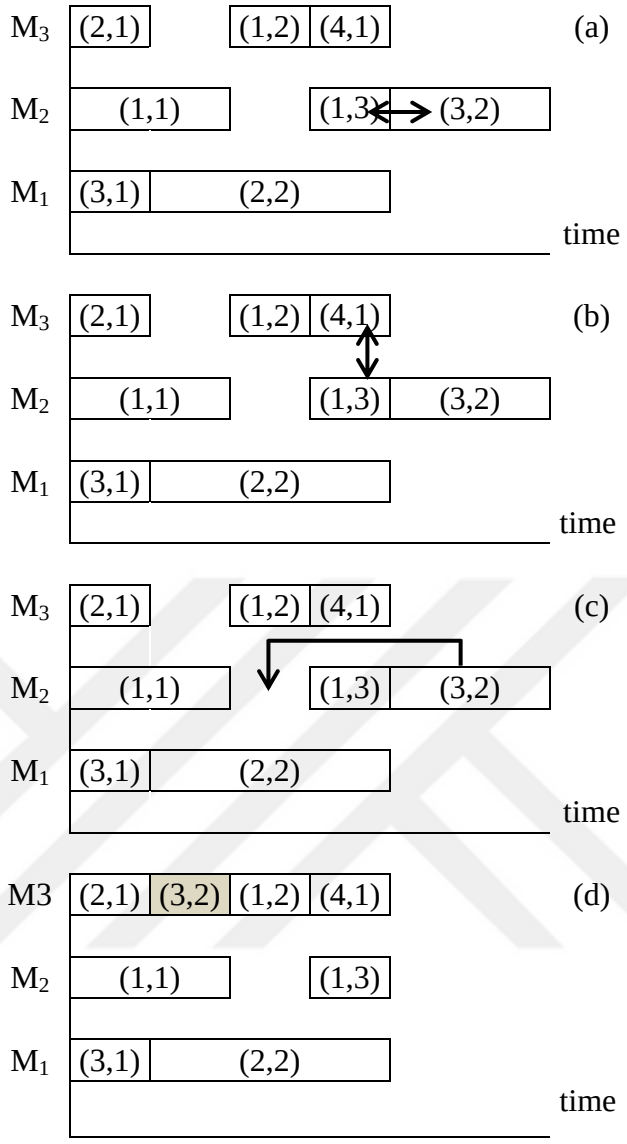


Figure 3.4 a) Swap local search b) Interchange local search c) Insert local search d)Change of the assigned machine

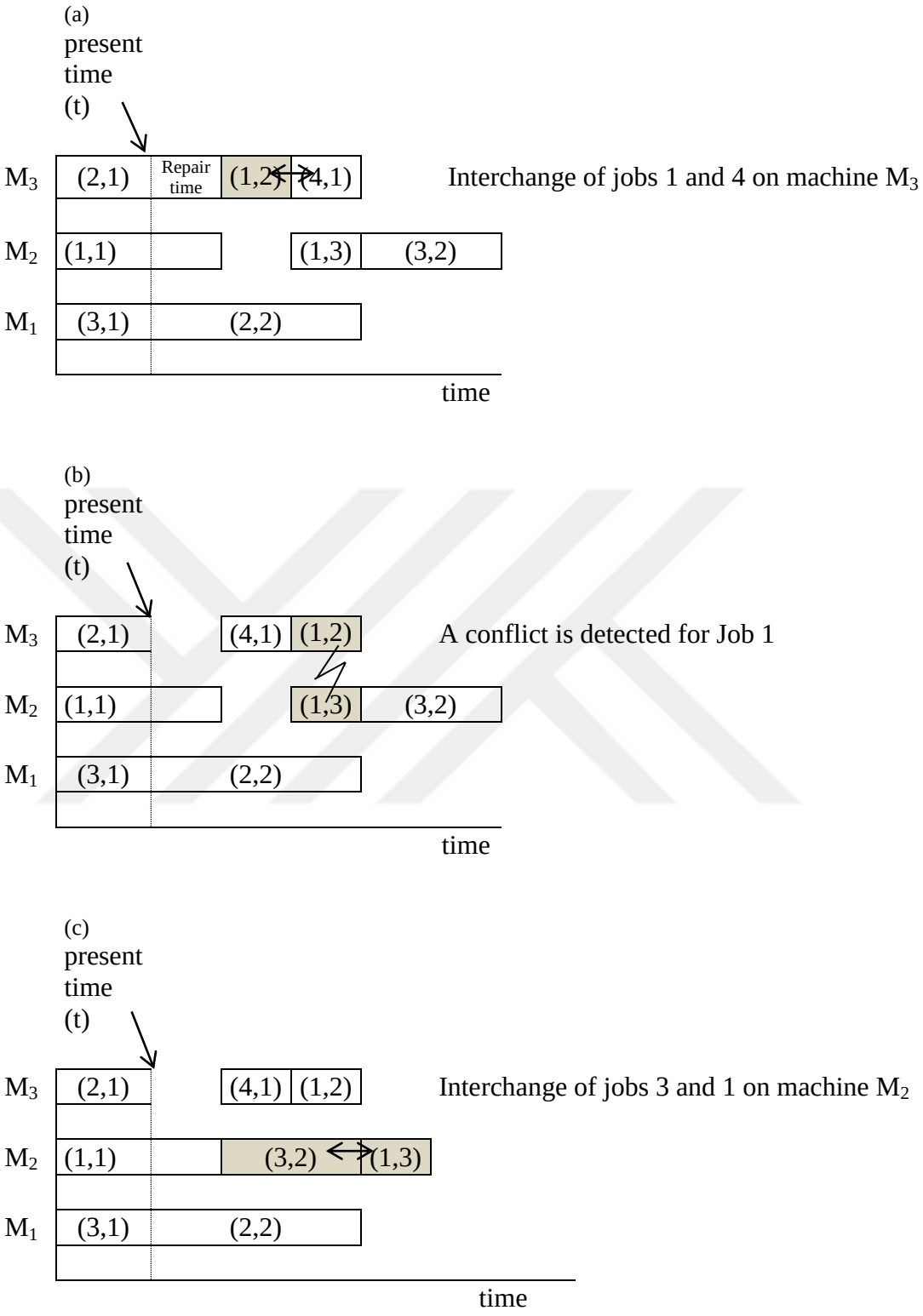


Figure 3.5 Local search process with swap move

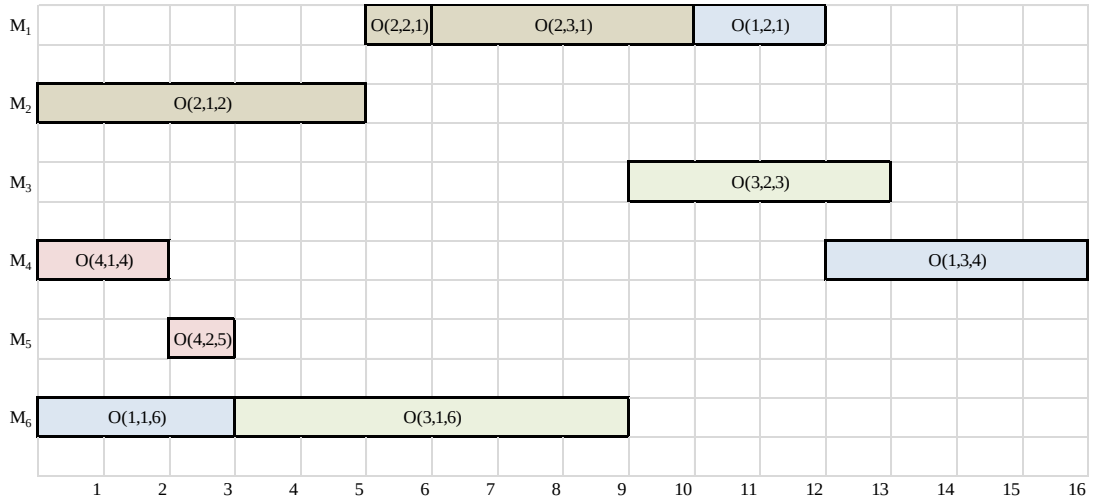


Figure 3.6 Gannt Chart of job shop scheduling for the construction phase

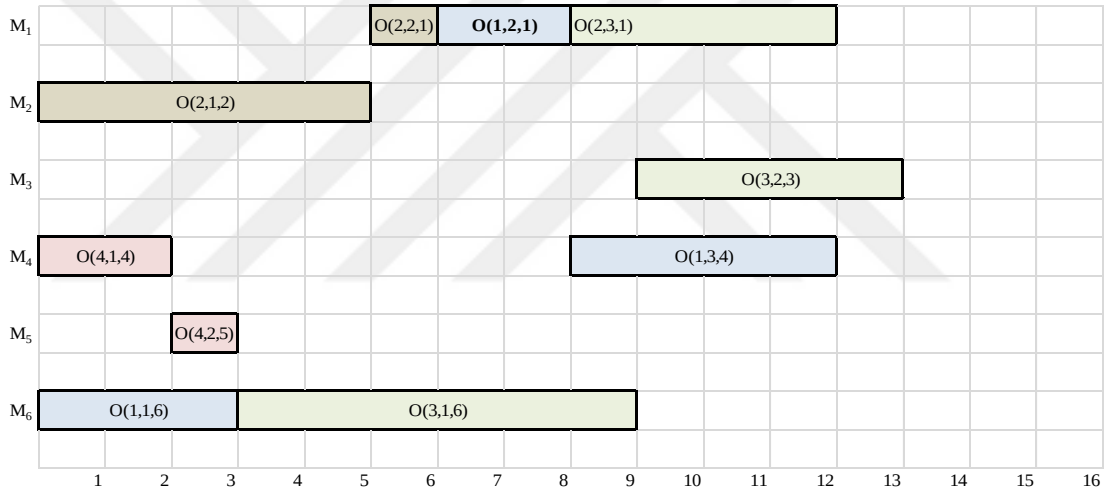


Figure 3.7 Gannt Chart of job shop scheduling after the local search phase

At first, the proposed GRASP algorithm (see Figure 3.8) which is explained above is used to construct a schedule. If there is no dynamic events, the schedule will be executed until all the operations are executed. When a dynamic event occurs, it affects the ongoing schedule. Rescheduling process is triggered and it is executed on the existing operations which are not ongoing at the dynamic event time. After rescheduling, the new schedule is realized on the shop floor, until the next dynamic event occurs. The flow chart of the proposed GRASP algorithm for DFJSSP is showed in Figure 3.9.

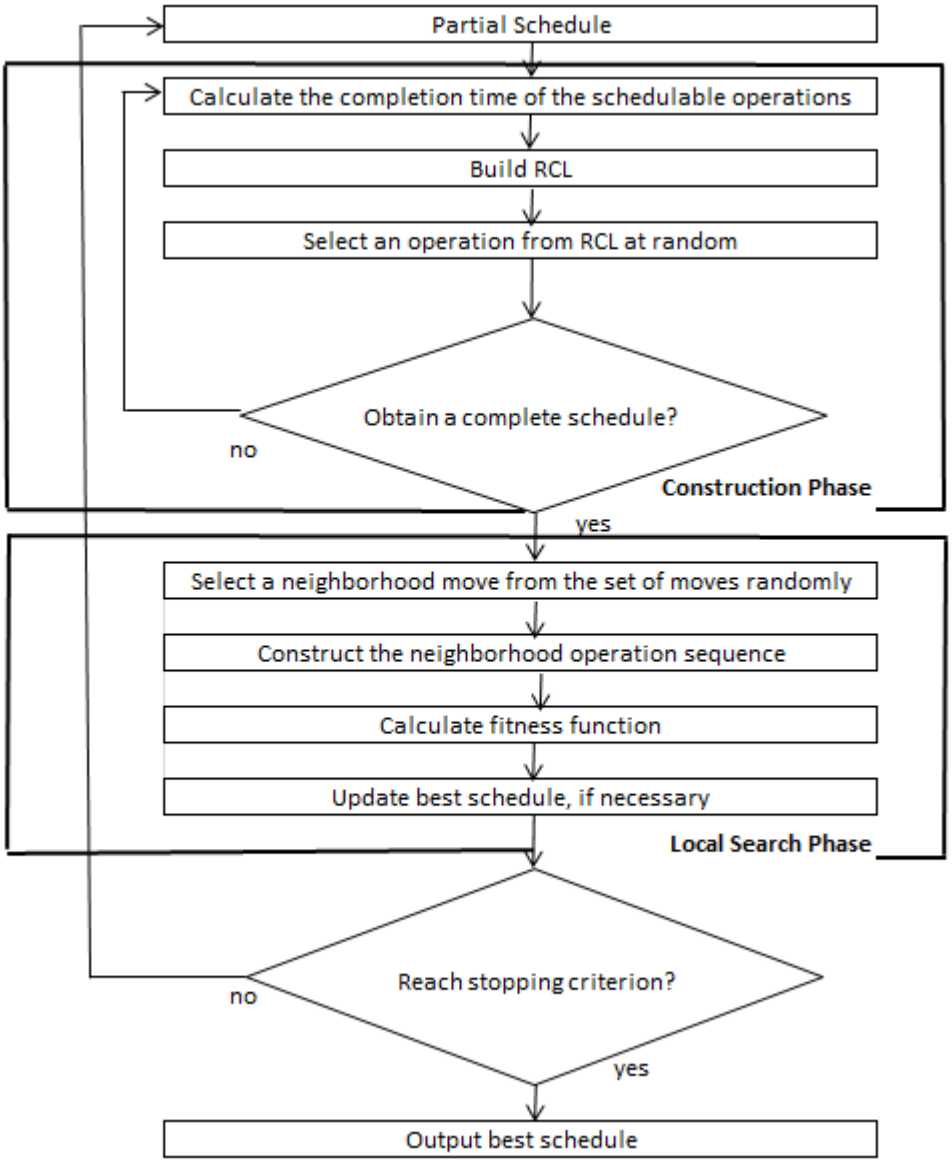


Figure 3.8 Flowchart of the proposed GRASP

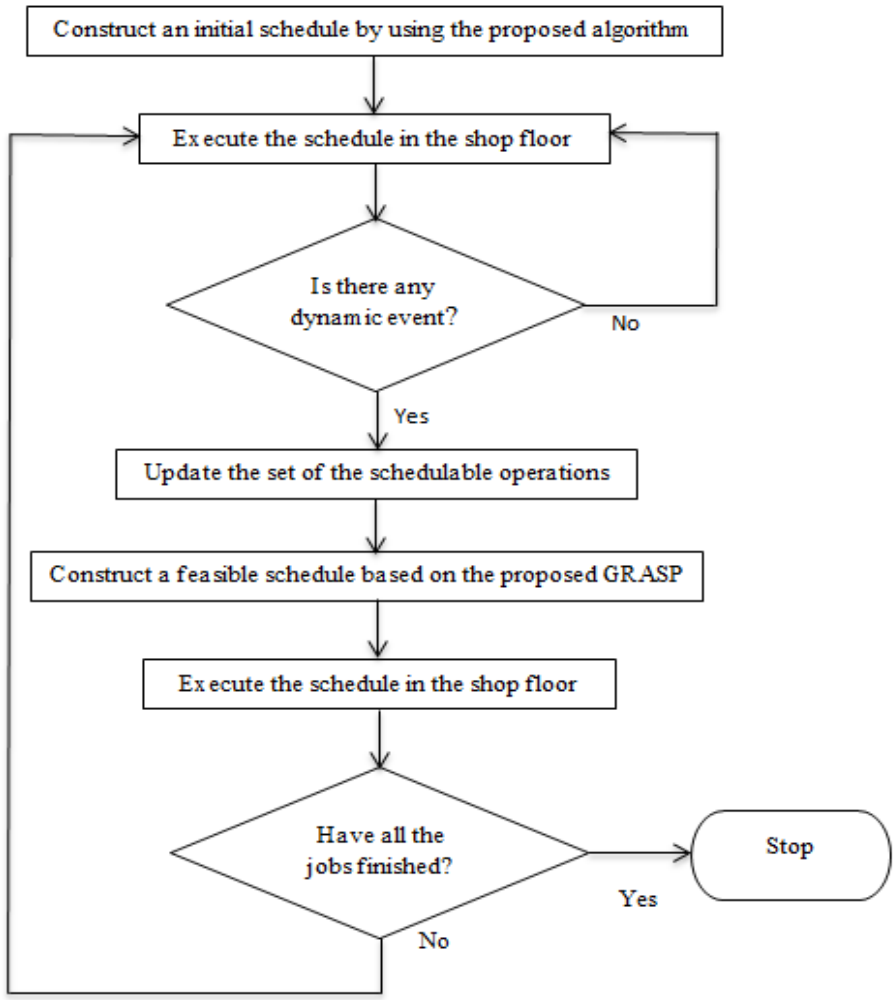


Figure 3.9 Flowchart of the GRASP for DFJSSP under event-driven rescheduling policy

The construction phase of GRASP starts with an empty solution and the selected new elements are added to the partial solution at each iteration, until a complete solution is obtained. After then, local search pahse is applied to improve the solution quality. The obtained solution is executed in the shop floor until a dynamic event occurs. When a dynamic event occurs, it affects the ongoing schedule. Rescheduling process is triggered and it is executed on the existing operations which are not ongoing at the dynamic event time. After rescheduling, the new schedule is executed on the shop floor, until the next dynamic event occurs. An example is given to explain how the proposed mechanism is executed: The new job (Job 3) arrives to the 2x2 example (given above) job floor at time 1 ($p_{(3,1,1)}$ =1; $p_{(3,1,2)}$ =3; $p_{(3,2,1)}$=2 ; $p_{(3,2,2)}$=1). The solution procedure of the proposed construction phase is expressed in detail and Figure 3.10 shows Gannt-Chart of the example.

Initialization

Step 1: S = Ø; Ω = { (1,2), (2,2), (3,1)}.

Iteration 1

Step 2: $s_{(1,2,1)} = 3$, $s_{(1,2,2)} = 3$; $f_{(1,2,1)} = 6$; $f_{(1,2,2)} = 4$, $f^* = \min\{ f_{(1,2,1)}, f_{(1,2,2)} \} = 4$, $m^* = 2$.

$s_{(2,2,1)} = 1$, $s_{(2,2,2)} = 3$; $f_{(2,2,1)} = 3$; $f_{(2,2,2)} = 5$, $f^* = \min\{ f_{(2,2,1)}, f_{(2,2,2)} \} = 3$, $m^* = 1$.

$s_{(3,1,1)} = 1$, $s_{(3,1,2)} = 3$; $f_{(3,1,1)} = 2$; $f_{(3,1,2)} = 6$, $f^* = \min\{ f_{(3,1,1)}, f_{(3,1,2)} \} = 2$, $m^* = 1$.

Step 3: Ω = {O(3,1,1)}, F={2}.

Step 4: ṣ=2, ṡ=2,α=0.4, g≤2, RCL ={2}.

Step 5: S = {O(2,1,1), O(1,1,2), O(3,1,1)}.

Step 6: Update Ω = {(1,2),(2,2), (3,2)}; go to Step 2.

Iteration 2

Step 2: $s_{(1,2,1)} = 3$, $s_{(1,2,2)} = 3$; $f_{(1,2,1)} = 6$; $f_{(1,2,2)} = 4$, $f^* = \min\{ f_{(1,2,1)}, f_{(1,2,2)} \} = 4$, $m^* = 2$.

$s_{(2,2,1)} = 2$, $s_{(2,2,2)} = 3$; $f_{(2,2,1)} = 4$; $f_{(2,2,2)} = 5$, $f^* = \min\{ f_{(2,2,1)}, f_{(2,2,2)} \} = 4$, $m^* = 1$.

$s_{(3,2,1)} = 2$, $s_{(3,2,2)} = 3$; $f_{(3,2,1)} = 4$; $f_{(3,2,2)} = 4$, $f^* = \min\{ f_{(3,2,1)}, f_{(3,2,2)} \} = 4$, $m^* = 2$.

Step 3: Ω = {O(1,2,2), O(2,2,1), O(3,2,2)}, F={4}.

Step 4: ṣ=4, ṡ=4,α=0.4, g≤4, RCL ={4}.

Step 5: S = {O(2,1,1), O(1,1,2), O(3,1,1), O(2,2,1)}.

Step 6: Update Ω = {(1,2), (3,2)}; go to Step 2.

Iteration 3

Step 2: $s_{(1,2,1)} = 4$, $s_{(1,2,2)} = 3$; $f_{(1,2,1)} = 7$; $f_{(1,2,2)} = 4$, $f^* = \min\{ f_{(1,2,1)}, f_{(1,2,2)} \} = 4$, $m^* = 2$.

$s_{(3,2,1)} = 4$, $s_{(3,2,2)} = 3$; $f_{(3,2,1)} = 6$; $f_{(3,2,2)} = 4$, $f^* = \min\{ f_{(3,2,1)}, f_{(3,2,2)} \} = 4$, $m^* = 2$.

Step 3: Ω = {O(1,2,2), O(3,2,2)}, F={4}.

Step 4: ṣ=4, ṡ=4,α=0.4, g≤4, RCL ={4}.

Step 5: S = {O(2,1,1), O(1,1,2), O(3,1,1), O(2,2,1), O(3,2,2)}.

Step 6: Update Ω = {(1,2)}; go to Step 2.

Iteration 4

Step 2: $s_{(1,2,1)} = 4$, $s_{(1,2,2)} = 4$; $f_{(1,2,1)} = 7$; $f_{(1,2,2)} = 5$, $f^* = \min\{ f_{(1,2,1)}, f_{(1,2,2)} \} = 5$, $m^* = 2$.

Step 3: Ω = {O(1,2,2)}, F={5}.

Step 4: ṣ=5, ṡ=5,α=0.4, g≤5, RCL ={5}.

Step 5: S = { O(2,1,1), O(1,1,2), O(3,1,1), O(2,2,1), O(3,2,2), O(1,2,2) }.

Step 6: A complete schedule has beeg generated, so stop the process.

The constructed schedule (Figure 3.10 (a)) is executed on the shop floor until time 1. The ongoing operations are not interrupted, but the rescheduling process is conducted. In the real manufacturing system, after the constructed schedule is executed on the shop floor, some dynamic events occur inevitable and rescheduling method is triggered to construct a new schedule. Whenever a rescheduling process begins, all non-started operations are rescheduled according to the proposed algorithm, which is explained above (see Figure 3.8). After rescheduling process, the newly generated schedule is implimented in the shop floor until a new dynamic event occurs (Figure 3.10 (b)). A feasible solution is obtained by the proposed GRASP algorithm at each scheduling point and implemented in the shop floor.

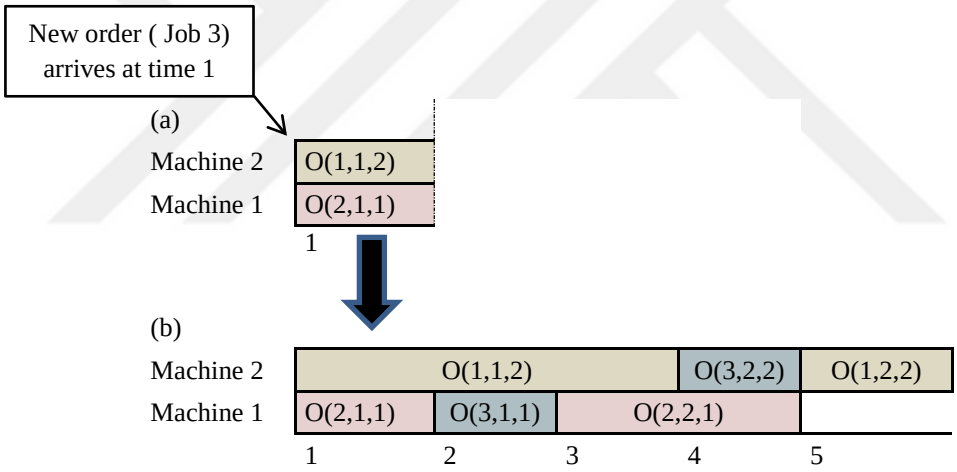


Figure 3.10 An event-driven rescheduling example for FJSSP with new order arrival

CHAPTER FOUR
SOLVING THE DYNAMIC JOB SHOP SCHEDULING PROBLEM BY USING GREEDY RAMDOMIZED ADAPTIVE SEARCH PROCEDURE

4.1 Chapter Introduction

Manufacturing systems are complex, dynamic, and stochastic systems with a wide variety of products, processes, production levels, and unexpected disturbances (Xiang & Lee, 2008). Mostly, it may be possible to formulate these problems, but solving these to optimality may require an enormous amount of computer time (Pinedo & Chao, 1999). Therefore, instead of the optimal solution, maintaining a feasible solution alone can sometimes be the only goal of the scheduling practice (Sabuncuoglu & Kizilisik, 2003). When a dynamic and stochastic manufacturing environment is encountered in which static scheduling may be impractical, the use of real time scheduling approaches is required (Xiang & Lee, 2008).

This chapter proposes a GRASP based algorithm for DJSSP to address several real-life situations. Rescheduling process is performed whenever a dynamic event is encountered in the production environment such as new order arrivals, machine breakdowns, arrival of urgent orders, order cancellations, and due date changes. GRASP which is explained in Chapter 3, is proposed for DJSSP in which the setup times are sequence dependent under machine capacity constraints. The dynamic event; order cancellation decreases the burden of the production system, but the others increase the burden of the production system. Because of this reason order acceptance/rejection decision and an ORR mechanism are integrated into scheduling procedure in order to meet the customer due date requirements while attempting to execute capacity adjustments. For evaluating the relative performance of the generated schedules, a preemptive goal programming approach is utilised with mean tardiness, schedule unstability, makespan, and mean flow time objectives. The explained dynamic events and mechanisms are discussed in Chapter 2.

The rest of this chapter is organized as follows: Experimental study is presented in Section 4.2 and conclusion is given in Section 4.3.

4.2 Experimental Design and Results

The proposed algorithm is coded by using MATLAB programming language and tested on a PC with Intel Core i5 2.40 GHz CPU and 16 GB of RAM. This section consists of three groups of experiments: (I) The first set of experiments are carried out to verify the suitability of the proposed algorithm; (II) The second set of experiments are carried out to verify the performance of the proposed algorithm for DJSSP; (III) The third set of experiments are conducted for comparison of the rescheduling policies (periodical vs. event-driven).

4.2.1 Experiment I

This group of the experiments is conducted to verify the suitability of the GRASP for solving JSSP. The performance of the proposed GRASP is evaluated by using the benchmark problem set ft06, ft10, ft20 from Fisher and Thompson (1963) and LA01-LA20 from Lawrence (1984). This JSSP set include 23 problem instances in which machine x job numbers vary between 36 and 100. A scheduling problem set with medium complexity is chosen in order to determine the best GRASP parameters for this study. The proposed algorithm can be defined by control parameters, the main parameter alfa which is defined in Chapter 3, local iteration number, and global iteration number. A test instance; 15 jobs-5 machines (LA06) is conducted 10 times and 360 experiments are carried out. The best value of parameters are 0.4, 2000, and 150 respetively. The proposed GRASP is compared with some novel studies, which also employ G&T algorithm within their schedule generation mechanism in order to have a more direct comparison with the proposed GRASP based algorithm. These algorithms are ‘‘particle swarm optimization algorithm (PSO)’’ of Yen and Ivers (2009), ‘‘memetic algorithm’’ of Yang et al. (2008), “GRASP” of Binato et al. (2002). Each problem is tested 10 times and every experiment is terminated by obtaining the best possible solution or by reaching the maximum iteration number. The results of the experiments are given in Table 4.1. As it can be seen from Table

4.1 the proposed algorithm is very effective. For example, the solution and Gannt chart for LA12 benchmark problem is given in Table 4.2 and Figure 4.1, respectively.

Table 4.1 Comparison of GRASP from the literature for JSSP test instances

Problem	JxM	The best solution	Yen and Ivers (2009) PSO	Yang et al. (2008) MA	Binato et al. (2002) GRASP	The proposed GRASP based algorithm
FT06	6x6	55	55	55	55	55
FT10	10x10	930	930	930	938	930
FT20	20x5	1165	1165	1165	1169	1165
LA01	10x5	666	666	666	666	666
LA02	10x5	655	668	655	655	655
LA03	10x5	597	606	597	604	597
LA04	10x5	590	611	590	590	590
LA05	10x5	593	593	593	593	593
LA06	10x5	926	926	926	926	926
LA07	10x5	890	890	890	890	890
LA08	15x5	863	863	863	863	863
LA09	15x5	951	951	951	951	951
LA10	15x5	958	958	958	958	958
LA11	20x5	1222	1222	1222	1222	1222
LA12	20x5	1039	1039	1039	1039	1039
LA13	20x5	1150	1150	1150	1150	1150
LA14	20x5	1292	1292	1292	1292	1292
LA15	20x5	1207	1207	1207	1207	1207
LA16	10x10	945	945	945	946	946
LA17	10x10	784	784	784	784	784
LA18	10x10	848	848	848	848	848
LA19	10x10	842	842	844	842	842
LA20	10x10	902	902	907	907	902

A multiple problem can be analyzed with non-parametric statistical tests preferably. Table 4.1 summarizes the results obtained from the competing algorithms. We check the independence, normality and heteroscedasticity properties whether we can apply or not a parametric statistical test. Independence condition can be given that one occurring does not affect the probability distribution of the other one occurring. Independence condition is satisfied in the experiments. All the tests which are used in this experiment represent abnormality shape. The p-value of the experiments are lower than the level of significance ($\alpha = 0.05$) and this indicates a

Table 4.2 Solution of 20x5 job shop scheduling problem (LA12) with GRASP

Machines																				
M1	Job Sequence	J9	J10	J11	J18	J20	J14	J1	J2	J8	J16	J3	J17	J15	J19	J13	J5	J4	J7	J6	J12
	Start Time	0	7	26	91	113	121	196	278	296	328	408	460	523	535	624	650	731	785	876	955
	Finish Time	7	26	89	113	121	196	278	296	328	408	460	523	535	624	650	731	785	876	955	1029
M2	Job Sequence	J18	J15	J1	J20	J12	J4	J2	J3	J7	J17	J5	J13	J6	J19	J9	J16	J14	J10	J8	J11
	Start Time	0	33	48	71	109	151	213	242	294	360	396	457	537	626	722	761	822	866	949	1033
	Finish Time	33	48	71	109	151	213	242	294	360	396	457	537	626	722	761	822	866	949	1033	1039
M3	Job Sequence	J20	J14	J11	J8	J16	J17	J18	J2	J3	J10	J19	J4	J9	J15	J5	J1	J6	J12	J13	J7
	Start Time	0	18	89	153	208	251	313	323	344	382	390	454	528	592	612	642	687	776	874	961
	Finish Time	18	57	153	208	251	313	323	344	382	390	454	528	592	612	642	687	776	874	961	981
M4	Job Sequence	J2	J20	J8	J18	J16	J10	J3	J12	J4	J5	J15	J11	J17	J14	J13	J1	J6	J19	J7	J9
	Start Time	0	50	65	73	91	117	124	178	239	296	364	443	534	630	654	729	776	787	882	915
	Finish Time	50	65	73	91	117	124	178	239	296	364	443	534	630	654	729	767	787	882	915	969
M5	Job Sequence	J3	J20	J10	J14	J8	J2	J18	J2	J4	J12	J16	J15	J19	J17	J5	J13	J9	J11	J7	J6
	Start Time	0	18	41	81	103	127	168	278	362	399	414	443	454	630	731	788	794	856	915	955
	Finish Time	16	41	81	103	127	168	173	362	399	414	436	451	518	670	788	794	850	890	935	1036

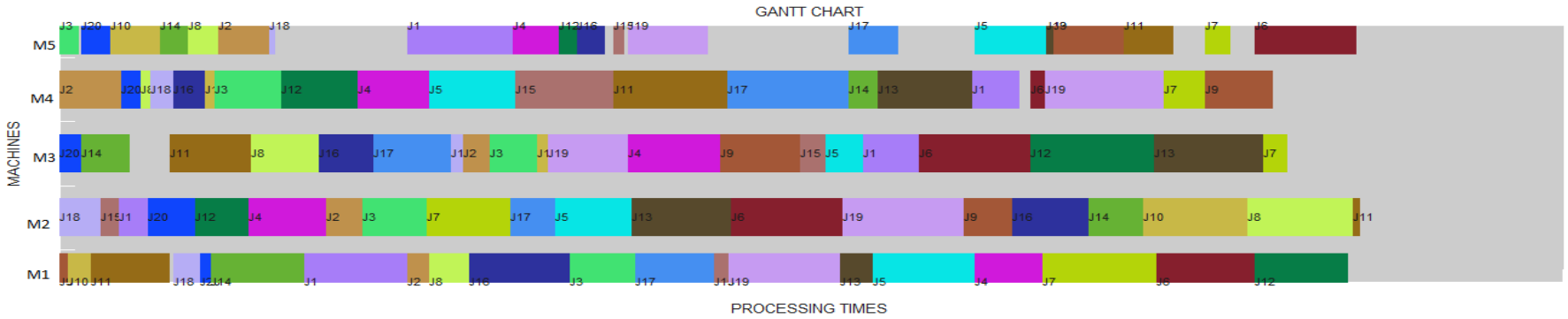


Figure 4.1. Gannt chart of the LA12 static job shop scheduling problem with GRASP

non-normal distribution and the normality condition is not satisfied. Statistical software package MINITAB is used to perform all computations. Heteroscedasticity condition can be controlled by applying the Levene's test, which checks homogeneity of variances of the different algorithms. Homogeneties condition is not satisfied, because we reject the null hypothesis at a level of significance, $\alpha = 0.05$. The parametric test conditions are not fulfilled perfectly. These results induce us to using nonparametric statistics in order to analyze the results.

Derrac et al. (2011) suggested usage of Friedman test for multiple comparisons (N x N). The Friedman test (Friedman two-way analysis of variances by ranks) is a nonparametric analog of the parametric two-way analysis of variance. It can be used for answering the following question: in a set of k samples (where $k \geq 2$), do at least two of the samples represent populations with different median values? The Friedman test is the analog of the repeated measures ANOVA in nonparametric statistical procedures; therefore, it is a multiple comparisons test that aims to detect significant differences between the behaviors of two or more algorithms (Derrac et al., 2011).

The null hypothesis for Friedman's test states equality of medians between the populations. The alternative hypothesis is defined as the negation of the null hypothesis, so it is non-directional (Derrac et al., 2011).

The ranks of the Friedman test can be computed for each problem, rank values from 1 (best result) to k (worst result) and then average ranks of all problems for each algorithm give us the final rank of each algorithm. In case of ties, average ranks are computed. Under the null hypothesis, formed from supposing that the results of the algorithms are equivalent and, therefore, their rankings are also similar, the Friedman's statistic

$$\chi_F^2 = \frac{12\ N}{k\ (k+1)} \left[\sum_j R_j^2 - \frac{k\ (k+1)^2}{4} \right]$$

is distributed according to χ^2_F with k − 1 degrees of freedom, being Rj =1/N $\sum_i r_i^j$, and N the number of functions. The critical values for the Friedman's statistic coincide with the established in the χ^2 distribution when N>10 and k>5.

Iman and Davenport (1980) presented a derivation from the Friedman's statistic given that this last metric produces a conservative undesirably effect. The proposed statistic is

$$F_F = \frac{(N-1)\chi_F^2}{N(k-1)-\chi_F^2}$$

and it is distributed according to a F distribution with k −1 and (k −1)(N −1) degrees of freedom.

The rejection of the null hypothesis in both tests described above does not involve the detection of the existing differences among the algorithms compared. They only inform us about the presence of differences among all samples of results compared. In order to conducting pairwise comparisons within the framework of multiple comparisons, we can proceed with a post-hoc procedure. In this case, a control algorithm (maybe a proposal to be compared) is usually chosen. Then, the post-hoc procedures proceed to compare the control algorithm with the remain k−1 algorithms (Derrac et al., 2011).

Table 4.3 shows the ranks computed through the Friedman test. The experiments depict the proposed GRASP as the best performing algorithm with a rank of 2.35 and GRASP (Binato et. al., 2002) as the worst performing algorithm with a rank of 2.67. The Friedman test controls whether the computed average ranks are significantly different from the mean rank R_J=2.5 expected under the null-hypothesis:

$$\chi_F^2 = \frac{12 \cdot 23}{4 \cdot 5}\left[(2.54^2 + 2.43^2 + 2.67^2 + 2.35^2) - \frac{4 \cdot 5^2}{4}\right] = 0.8217$$

$$F_F = \frac{22 \cdot 0.8217}{23 \cdot 3 - 0.8217} = 0.2651$$

F_F is distributed according to the F distribution with 4-1 = 3 and (4-1) x (23-1) = 66 degrees of freedom, with four algorithms and 23 data sets. The critical value of F(0.2651) for $\alpha = 0.05$ is 0.8503, so we reject the null-hypothesis. The post-hoc rest is not applied because of the insignificant differences between the algorithms.

Table 4.3 Friedman ranks

Problem	Yen and Ivers (2009) PSO	Yang et al. (2008) MA	Binato et al. (2002) GRASP	The proposed GRASP based algorithm
FT06	2.5	2.5	2.5	2.5
FT10	2	2	4	2
FT20	2	2	4	2
LA01	2.5	2.5	2.5	2.5
LA02	4	2	2	2
LA03	4	1.5	3	1.5
LA04	4	2	2	2
LA05	2.5	2.5	2.5	2.5
LA06	2.5	2.5	2.5	2.5
LA07	2.5	2.5	2.5	2.5
LA08	2.5	2.5	2.5	2.5
LA09	2.5	2.5	2.5	2.5
LA10	2.5	2.5	2.5	2.5
LA11	2.5	2.5	2.5	2.5
LA12	2.5	2.5	2.5	2.5
LA13	2.5	2.5	2.5	2.5
LA14	2.5	2.5	2.5	2.5
LA15	2.5	2.5	2.5	2.5
LA16	1.5	1.5	3.5	3.5
LA17	2.5	2.5	2.5	2.5
LA18	2.5	2.5	2.5	2.5
LA19	2	4	2	2
LA20	1.5	3.5	3.5	1.5
Average	2.54	2.43	2.67	2.35

4.2.2 Experiment II

In the related literature, there is a lack of detailed benchmark problems for the DJSSP. In order to evaluate the performance of the proposed algorithm, new benchmark instances are generated in this study. The generated problem set is classified as small, medium and large sized problems. Each problem takes into

account dynamic events such as new order arrival, machine breakdowns, arrival of urgent order, change of the due dates, order cancellation, machine capacity and sequence dependent setup times. Problems which have less than 25 operations are defined small size problems (5x5 size problem). Problems which have between 25 and 100 operations are defined medium size problems (10x10 size problem). Problems which have 100 or more then 100 operations are defined large size problems (15x15 size problem). The benchmark instances are given in Table A1.1, Table A1.2, and Table A1.3.

In Appendix-A1, event type "1" indicates new order arrival. For instance, a new order J1 arrives at 50th minutes of the schedule. Event type "2" indicates machine breakdown. For instance, machine M1 breakdowns at 120th minutes of the schedule and the repair time of this machine takes 30 minutes. Event type "3" indicates order cancellation. For instance, fourth order is cancelled at 350th minutes of the schedule. Event type "4" indicates rush order. For instance, the seventh order arrives at 100th minutes of the schedule as rush order. Event type "5" indicates change of due date. For instance, the due date of the sixth order is changed at 120th minutes of the schedule and the new due date is assigned 800 minutes. Setup times are uniformly distributed (8,15). Each shift is considered 480 minutes.

Each benchmark instance is designated by a set of parameters: U (resource utilization range), due date type (tight or loose due date) with two levels. Parameter U has two levels: low (-) = [0.85 0.95] (ranges are specified as a (min, max) pair) and high (+) = [0.50 0.80]. Parameter due date type has two levels: tight (T) (delivered on desired time) and loose (L) (late delivery of %15). The proposed methods are run 10 times.

The results are compared with the following dispatching rules: FIFO: First arrival at shop, first served (r_i); EDD: Earliest due date (d_i); SPT: Shortest processing time (p_{ij}); LPT: Longest processing time ($-p_{ij}$); SL: Smallest slack ($d_i - t - \sum_{q=j}^{m_i} p_{iq}$); CR: Smallest critical ratio ($(d_i - t)/\sum_{q=j}^{m_i} p_{iq}$). Where, r_i is release time of job i, p_{ij} is processing time of operation j of job i, q is index of remaining operations (q = j,...,

m_i), d_i is due date of job i, t is time. The last two dispatching rules (SL, CR) are multi dimensional as they make use of due date and processing time information; others are single dimensional. Computational results are summarized in Table 4.4, 4.5 and 4.6. In these tables, the best fitness values and the number of accepted orders for each method are presented. The results obtained from the proposed GRASP algorithm are usually better than dispatching rules. Better schedules can be generated by the proposed approach in comparison to classical dispatching rule based approach which is commonly used in dynamic scheduling environments. Especially the first objective, mean tardiness is considerably better, as this is the top priority objective in our computational settings we can say that GRASP is able to provide better results in most of the settings. Multi-dimensional dispatching rules are also able to generate competitive results in some problem sets. But we should mention here that GRASP is able to provide competitive results by accepting more jobs than dispatching rules. In such a case we can say that GRASP is able to provide more acceptable solutions in practice.

4.2.3 Experiment III

In this group of experiments, the periodical rescheduling policy with different time intervals is compared with event-driven rescheduling policy, by using the problem set which is given in section 5.2. Each problem is run 10 times with the previously designated parameters. A periodic that is too short may lead the rescheduling to occur frequently, which affects the scheduling stability, but a periodic that is too long may decrease the utilization of the machines. We set the scheduling periodic to 120, 180, 240 min. Table 4.7 presents the comparisons of periodical and event-driven rescheduling policies. It is not possible to make a ranking between the scheduling intervals in terms of the solution quality as the best scheduling interval varies according to the problem. The frequency of the occurrence of the dynamic events affects solution quality for each scheduling intervals of the periodical rescheduling policy. But, event-driven rescheduling policy usually provides better results than the periodical rescheduling policy for the studied problems.

Table 4.4 Comparison of the proposed GRASP and dispatching rules for 5x5 problem

Jobs	Machines	Utilization	Due date tightness	Methods	Mean Tardiness	Unstability	Makespan	Mean Flow Time	Number of Orders	Number of accepted orders
5	5	-	T	GRASP	81	5103	814	557	7	6
				FIFO	581	2391	1025	604	7	4
				EDD	124	5687	1051	600	7	4
				SPT	183	3036	874	641	7	5
				LPT	262	3986	888	607	7	4
				SL	102	5230	893	593	7	4
				CR	131	694	702	530	7	4
5	5	-	L	GRASP	47	3098	814	651	7	6
				FIFO	360	1861	871	600	7	3
				EDD	111	2724	1125	694	7	5
				SPT	126	4175	1051	672	7	5
				LPT	394	2499	888	661	7	4
				SL	99	1872	1054	714	7	5
				CR	86	3847	768	562	7	5
5	5	+	T	GRASP	135	7557	802	695	7	5
				FIFO	681	2635	1125	740	7	3
				EDD	201	6274	1153	651	7	4
				SPT	167	4065	1051	847	7	3
				LPT	270	12513	832	603	7	4
				SL	146	2745	1054	516	7	3
				CR	167	1405	755	539	7	5
5	5	+	L	GRASP	106	8619	802	695	7	5
				FIFO	225	2	736	546	7	3
				EDD	190	2846	1127	599	7	4
				SPT	244	1821	1151	980	7	3
				LPT	415	5048	1174	829	7	4
				SL	253	5658	1231	875	7	5
				CR	234	4394	1248	814	7	5

Table 4.5 Comparison of the proposed GRASP and dispatching rules for 10x10 problem

Jobs	Machines	Utilization	Due date tightness	Methods	Mean Tardiness	Unstability	Makespan	Mean Flow Time	Number of Orders	Number of accepted orders
10	10	-	T	GRASP	399	65973	2981	2006	13	12
				FIFO	856	1	2645	1790	13	3
				EDD	1140	51497	4214	2670	13	7
				SPT	1014	66023	3755	2288	13	8
				LPT	1142	38106	3033	2574	13	5
				SL	574	33080	3466	2310	13	7
				CR	603	33906	2773	2247	13	11
10	10	-	L	GRASP	309	65347	3195	2126	13	13
				FIFO	553	2	2484	1718	13	3
				EDD	1650	122340	6330	3460	13	8
				SPT	2019	78847	5186	3776	13	9
				LPT	1261	54493	3450	2954	13	6
				SL	482	34341	3466	2401	13	9
				CR	891	58638	3567	3752	13	11
10	10	+	T	GRASP	693	47168	3065	2493	13	11
				FIFO	1828	9	6397	3610	13	6
				EDD	742	19295	3208	2283	13	4
				SPT	1245	47336	3922	2910	13	5
				LPT	1990	84743	5381	3862	13	6
				SL	673	28424	2646	2308	13	5
				CR	373	22287	2504	2150	13	6
10	10	+	L	GRASP	641	67223	3442	2662	13	10
				FIFO	1566	3	3605	2464	13	3
				EDD	1352	47752	5584	3199	13	5
				SPT	1503	59144	4882	2887	13	8
				LPT	1632	44822	3678	3208	13	5
				SL	898	36812	3545	2853	13	5
				CR	1604	71501	4421	3664	13	11

Table 4.6 Comparison of the proposed GRASP and dispatching rules for 15x15 problem

Jobs	Machines	Utilization	Due date tightness	Methods	Mean Tardiness	Unstability	Makespan	Mean Flow Time	Number of Orders	Number of accepted orders
15	15	-	T	GRASP	810	384590	4730	3520	20	18
				FIFO	1101	8	3926	2748	20	3
				EDD	2160	153930	7280	4780	20	6
				SPT	3450	593410	9310	6330	20	13
				LPT	1894	52871	5931	4211	20	7
				SL	980	139540	5370	3690	20	14
				CR	940	110340	3990	3690	20	16
15	15	-	L	GRASP	650	412050	4760	3710	20	20
				FIFO	2019	18742	7071	3801	20	6
				EDD	2890	336040	9750	5850	20	8
				SPT	5640	918840	12560	8500	20	17
				LPT	2540	204990	7090	5390	20	7
				SL	1240	158510	5620	4390	20	15
				CR	820	211000	4390	4030	20	18
15	15	+	T	GRASP	1180	291040	5070	3860	20	14
				FIFO	2983	1744	8333	5263	20	5
				EDD	2780	161380	8390	5470	20	5
				SPT	2250	184110	8240	5190	20	8
				LPT	2380	105430	6570	5320	20	6
				SL	192	33755	3132	2839	20	8
				CR	145	7363	2754	2685	20	6
15	15	+	L	GRASP	990	255240	5010	4070	20	15
				FIFO	2673	1735	8333	5263	20	5
				EDD	2390	162790	8390	5100	20	6
				SPT	2690	443400	9290	6000	20	9
				LPT	2540	108930	7040	5340	20	5
				SL	876	88836	5138	4037	20	12
				CR	1040	129190	4480	4210	20	12

Table 4.7 Comparison of event-driven and periodical rescheduling policies

#Jobs	#Machines	Utilization	Duedate tightness	GRASP (Evet driven, periodic: 120,180,240)	Mean Tardiness	Schedule unstability	Makespan	Mean Flow Time	Number of Orders	Number of accepted orders
5	5	-	T	Event driven	81	5103	814	557	7	6
				120	115	7850	866	677	7	6
				180	132	10781	848	658	7	6
				240	40	17412	906	588	7	6
5	5	-	L	Event driven	47	3098	814	651	7	6
				120	145	1753	926	790	7	6
				180	126	3589	1052	726	7	6
				240	93	8223	880	633	7	6
5	5	+	T	Event driven	135	7705	802	695	7	5
				120	129	9225	802	635	7	5
				180	177	20751	1051	722	7	5
				240	97	10943	1051	650	7	5
5	5	+	L	Event driven	106	8619	802	695	7	5
				120	251	10064	1051	883	7	5
				180	154	25803	816	641	7	5
				240	133	11051	1153	754	7	5
10	10	-	T	Event driven	399	65973	2981	2006	13	12
				120	374	57607	2723	1948	13	10
				180	475	60764	6065	2109	13	13
				240	580	155590	3090	2220	13	13
10	10	-	L	Event driven	309	65347	3195	2126	13	13
				120	429	86660	2985	2199	13	11
				180	526	82639	3253	2288	13	13
				240	587	49166	3090	2402	13	13
10	10	+	T	Event driven	693	47168	3065	2493	13	11
				120	747	63990	3465	2312	13	10
				180	850	125841	3850	2541	13	11
				240	990	148530	3680	2630	13	12
10	10	+	L	Event driven	641	67223	3442	2662	13	10
				120	628	56291	3574	2419	13	11
				180	910	136280	3920	2760	13	12
				240	977	74929	3671	2824	13	12
15	15	-	T	Event driven	810	384590	4730	3520	20	18
				120	1020	453650	4870	3630	20	19
				180	960	155770	4660	3600	20	19
				240	1220	187780	4880	3890	20	19
15	15	-	L	Event driven	650	412050	4760	3710	20	20
				120	950	100720	4870	4000	20	18
				180	640	100992	4560	3650	20	19
				240	690	217660	4490	3720	20	19
15	15	+	T	Event driven	1180	291040	5070	3860	20	14
				120	1590	249030	5550	4340	20	17
				180	1540	481210	5360	4350	20	18
				240	1810	198790	5940	4570	20	18
15	15	+	L	Event driven	990	255240	5010	4070	20	15
				120	1480	212200	5920	4500	20	18
				180	1310	163670	5610	4470	20	18
				240	1400	198140	5910	4550	20	18

4.3 Chapter Conclusion

This chapter presents a GRASP based metaheuristic algorithm for modeling and solving multi-objective DJSSP in which new order arrivals, machine breakdowns, arrival of urgent orders, order cancellations, change of due dates occur under machine capacity constraints with sequence dependent setup times, order acceptance

rejection and an ORR mechanism. The proposed approach handles many practical scheduling factors which are faced in many manufacturing environments. A preemptive goal programming logic is used to assess the relative performance of the generated schedules. A Matlab based computer program is also developed for possible implementations.

In experimental study, static benchmark job shop scheduling problems are conducted. Then, extensive experimental studies for dynamic shop environments under event-driven and periodical rescheduling policies are presented to provide numerical evidences.

CHAPTER FIVE
SOLVING THE DYNAMIC FLEXIBLE JOB SHOP SCHEDULING PROBLEM BY USING GREEDY RANDOMIZED ADAPTIVE SEARCH PROCEDURE

5.1 Chapter Introduction

GRASP has not yet been adapted to DFJSSP in the literature (see Chapter2). The primary aim of this chapter is to solve the proposed DFJSSP based on the proposed GRASP algorithm (see Chapter 3). The following assumptions are made in model development:

- Jobs arrive at the shop floor continuously during the production process.
- Each operation can be executed on a set of machines.
- Each job has its own due date.
- The arrival time and the due date of jobs are not known until the job arrives.
- Each job has a predetermined sequence of operations.
- All machines are available at time 0.
- Transportation time is not considered.
- Each machine performs only one job at a time.

This chapter presents one of the most comprehensive scheduling models with many practical criteria, constraints and mechanisms (see Chapter 2).

The rest of this chapter is organized as follows: Experimental study is presented in Section 5.2 and conclusion is given in Section 5.3.

5.2 Experimental Design

In this chapter, experiments are conducted in order to evaluate the performance of the proposed approach. This section consists of three groups of experiments. First of all, performance of the proposed GRASP algorithm is tested for static FJSSP by

using benchmark problem instances from the literature. Then the performance of the proposed algorithm is investigated for DFJSSP by using test problems which are developed in this research as there are no standard benchmarks for comparison. Lastly, performances of the event-driven rescheduling policy and the periodical rescheduling policy are compared. The proposed approach is implemented by using MATLAB programming language on a PC with Intel Core i5 2.40 GHz CPU.

5.2.1 Experiment I

In order to have an idea about the general performance of the proposed GRASP based dynamic scheduling algorithm, we have run experiments on static FJSSP benchmarks from the literature. Three FJSSP benchmark data sets which are widely used in the literature, namely, Kacem data (Kacem et al., 2002), BRdata (Brandimarte, 1993) and DPdata (Dauzere-Peres & Paulli, 1997) are utilized.

Our computational results for the proposed GRASP algorithm are reported in Tables 5.1, 5.2, and 5.3. The best values among the referred approaches depicted with bold values. The first data set contains three problems. Table 5.1 shows the results of the proposed GRASP algorithm along with PSO+SA of (Xia & Wu, 2005), PVNS of (Yazdani et al., 2010), hGA of (Gao et al., 2008) and PSO of (Nouiri et al., 2015). The first column includes the size of the instance and the following columns include the result of the given algorithms. The proposed GRASP algorithm outperforms PSO+SA in 1 out of 3 instances, outperforms PVNS in 1 out of 3 instances, and outperforms PSO in 2 out of 3 instances.

The second data set contains ten problems. Table 5.2 demonstrates the results of the proposed GRASP algorithm along with PVNS of (Yazdani et al., 2010), CDDS of (Yuan & Xu, 2013), PSO of (Nouiri et al., 2015), SSPR of (González et al., 2015) for Brandimarte data set. The name and size of the instances are given the first and second columns in Table 5.2-5.3. The third column shows the ‘flexibility’ which is defined in terms of the average number of alternative machines for each operation. The best lower bound of the instance is given in the fourth column of Tables 5.2-5.3.

The proposed GRASP algorithm is able to provide the best results for 7 instances out of 10 instances among the five algorithms.

The third data set contains eighteen problems. Table 5.3 shows the results of the proposed GRASP algorithm with hGA of (Gao et al., 2008), eGA of (Zhang et al., 2011), CDDS of (Yuan & Xu, 2013), SSPR of (González et al., 2015) and QPSO of (Singh & Mahapatra, 2016). The proposed GRASP algorithm is able to provide the best results for 4 instances out of 18 instances among the five algorithms.

Table 5.1 Makespan results for Kacem's instances

Problem size (n x m)	PSO+SA (Xia et al., 2005)	PVNS (Yazdani et al., 2010)	hGA (Gao et al, 2008)	PSO (Nouiri et al., 2015)	**Proposed GRASP**
8 x 8	15	**14**	**14**	17	**14**
10 x 10	**7**	**7**	**7**	8	**7**
15 x 10	12	12	**11**	**11**	**11**

Table 5.2 Results for Brandimarte's data set

Problem	n x m	flex.	LB	PVNS (Yazdani et al., 2010)	CDDS (Yuan & Xu, 2013)	PSO (Nouiri et al., 2015)	SSPR (González et al., 2015)	**Proposed GRASP**
Mk 01	10 x 6	2.09	36	**40**	**40**	41	**40**	**40**
Mk 02	10 x 6	4.10	24	**26**	**26**	**26**	**26**	**26**
Mk 03	15 x 8	3.01	204	**204**	**204**	207	**204**	**204**
Mk 04	15 x 8	1.91	48)	**60**	**60**	65	**60**	**60**
Mk 05	15 x 4	1.71	168	173	173	**171**	172	172
Mk 06	10 x 15	3.27	33	60	58	61	**57**	64
Mk 07	20 x 5	2.83	133	141	**139**	173	**139**	**139**
Mk 08	20 x 10	1.43	523	**523**	**523**	**523**	**523**	**523**
Mk 09	20 x 10	2.53	299	309	**307**	**307**	**307**	**307**
Mk 10	20 x 15	2.98	165	206	197	312	**196**	205

The proposed GRASP algorithm is run 20 times for each test problem in order to obtain the given results (makespan). The performance of proposed GRASP algorithm is superior than or equal to the cited algorithms for Kacem's and Brandimarte's data sets. In order to better compare the performance of algorithms with Dauzere-Peres and Paulli's data sets, we decided to perform statistical tests. A multiple problem can be analyzed with non-parametric statistical tests preferably. Table 5.4 summarizes the results obtained from the competing algorithms. We check the independence, normality and heteroscedasticity properties whether we can apply or not a parametric statistical test. The parametric test conditions are not fulfilled perfectly. These results induce us to using non parametric statistics in order to analyze the results.

The ranks of the Friedman test can be computed for each problem, rank values from 1 (best result) to k (worst result) and then average ranks of all problems for each algorithm give us the final rank of each algorithm. The experiments depict SSPR as the best performing algorithm with a rank of 2.11 and hGA as the worst performing algorithm with a rank of 5.14. Iman-Davenport extension (F_F =10.24036, p-value = 0.0001) strongly suggest the existence of differences among the algorithms. After accepting differences of the algorithms, the application of a post-hoc test can conduct pair-wise comparisons with the multiple comparisons framework. In NxN comparisons, k(k-1)/2 comparisons among algorithms are conducted by its p-value from lowest to highest.

Friedman test:

$$z = (R_i - R_j) / \sqrt{\frac{k(k+1)}{6N}}$$

where R_i and R_j are the average ranking by the Friedman test of the algorithms compared (Derrac et al., 2011).

The test statistic value of z for comparing the *i*th and *j*th algorithms is taken from the table of normal distribution (p-value). Nemenyi and Holm's post-hoc procedures are selected for the implementation of the Friedman test applying multiple comparisons process. Table 5.4 lists all z-values, unadjusted –values, and all of the post-hoc hypotheses. Using a level α=0.1, only six hypotheses are rejected by the Nemelyi method. These hypotheses show the improvement of SSPR and QPSO over hGA and CDDS, and GRASP and eGA over hGA. The Holm rejects additional hypotheses, the improvement of GRASP over CCDS. The remaining eight hypotheses cannot be rejected by using these methods. However, using a level α=0.05, six hypotheses are rejected by the Holm method. Table 5.5 summarizes the rejected hypotheses for different α level. Based on the computational results obtained by solving static FJSSPs we can say that the proposed GRASP algorithm is able to generate competitive schedules for static problems. This can be an indication for its applicability to dynamic scheduling problems.

Table 5.3 Results for Dauzere-Peres and Paulli's data set

Problem	*n* x *m*	flexibility	LB	hGA (Gao et al., 2008)	eGA (Zhang et al., 2011)	CDDS (Yuan & Xu, 2013)	SSPR (González et al, 2015)	QPSO (Singh& Mahapatra, 2016)	**Proposed GRASP**
1a	10 x 5	1.13	2505	2518	2516	2518	**2505**	**2505**	**2505**
2a	10 x 5	1.69	2228	2231	2231	2231	**2229**	2230	2230
3a	10 x 5	2.56	2228	2229	2232	2229	**2228**	2229	2229
4a	10 x 5	1.13	2503	2515	2515	2503	2503	**2498**	2503
5a	10 x 5	1.69	2189	2217	2208	2216	2211	**2207**	**2207**
6a	10 x 5	2.56	2162	2196	2174	2196	2183	**2170**	**2170**
7a	15 x 8	1.24	2187	2307	**2217**	2283	2274	2264	2264
8a	15 x 8	2.42	2061	2073	2073	2069	**2064**	2073	2073
9a	15 x 8	4.03	2061	2066	2066	2066	**2062**	2066	2066
10a	15 x 8	1.24	2178	2315	**2189**	2291	2269	2205	2205
11a	15 x 8	2.42	2017	2071	2063	2063	2051	**2050**	**2050**
12a	15 x 8	4.03	1969	2030	2019	2031	**2018**	2019	2019
13a	20 x 10	1.34	2161	2257	**2194**	2257	2248	2253	2253
14a	20 x 10	2.99	2161	2167	2167	2167	**2163**	2167	2167
15a	20 x 10	5.02	2161	2165	2165	2165	**2162**	2165	2165
16a	20 x 10	1.34	2148	2256	**2211**	2256	2244	2252	2252
17a	20 x 10	2.99	2088	2140	**2109**	2140	2130	2134	2134
18a	20 x 10	5.02	2057	2127	**2089**	2127	2119	2123	2123

Table 5.4 Adjusted p-values for multiple comparisons among all methods

i	Hypothesis	z	Unadjusted *p*	Nemenyi	Holm
1	hGA versus SSPR	4.7661588	0.0001000	0.0015000	0.0015000
2	CDDS versus SSPR	4.008919	0.0001000	0.0015000	0.0014000
3	hGA versus QPSO	3.6080268	0.0003000	0.0045000	0.0039000
4	hGA versus GRASP	3.4298526	0.0006000	0.0090000	0.0072000
5	hGA versus eGA	3.2071349	0.0013000	0.0195000	0.0143000
6	CDDS versus QPSO	2.8507866	0.0044000	0.0660000	0.0440000
7	CCDS versus GRASP	2.6726124	0.0075000	0.1125000	0.0675000
8	CDDS versus eGA	2.4498947	0.0143000	0.2145000	0.1144000
9	eGA versus SSPR	1.5590239	0.1190000	1	0.8330000
10	GRASP versus SSPR	1.3363062	0.1814000	1	1
11	QPSO versus SSPR	1.1581320	0.2468000	1	1
12	hGA versus CDDS	0.7572402	0.4489000	1	1
13	eGA versus QPSO	0.4008919	0.6885000	1	1
14	eGA versus GRASP	0.2227177	0.8238000	1	1
15	GRASP versus QPSO	0.1781742	0.8586000	1	1

Table 5.5 Multiple comparisons for different significance level (α)

Alg. Versus	hGA		eGA		CDDS		SSPR		QPSO		GRASP	
alg.	0.1	0.05	0.1	0.05	0.1	0.05	0.1	0.05	0.1	0.05	0.1	0.05
hGA	-	-	+	+	-	-	+	+	+	+	+	+
eGA	-	-	-	-	-	-	-	-	-	-	-	-
CDDS	-	-	-	-	-	-	+	+	+	+	+	-
SSPR	-	-	-	-	-	-	-	-	-	-	-	-
QPSO	-	-	-	-	-	-	-	-	-	-	-	-
GRASP	-	-	-	-	-	-	-	-	-	-	-	-

5.2.2 Experiment II

New benchmark instances are generated in this chapter in order to demonstrate the performance of the proposed algorithm for solving DFJSSP. Extensive computational experiments are conducted for different levels of problem flexibility, shop utilization and due date tightness.

The generated problem set is classified as small, medium and large size. Each problem size has three flexibility levels which are called as small, medium and large flexibility level. Machine sets are formed randomly. Each problem takes into account dynamic events such as new order arrival, machine breakdowns, arrival of urgent order, change of due dates, order cancellation, machine capacity and sequence dependent setup times. Problems that have less than 25 operations are defined as small size problems (5x5 size problems). Problems that have 25 to 100 operations are defined as medium size problems (10x10 size problems). Problems that have 100 or more than 100 operations are defined as large size problems (15x15 size problem). Machine alternatives of operations for different problem sizes are given in Table 5.6. Flexibility levels are represented as follow: Small flexibility level: 1, Medium flexibility level: 2, Large flexibility level: 3. Non-flexible DJSSP is demonstrated with 0. The benchmark instances are given in Table A2.1, Table A2.2, and Table A2.3. The effects of resource utilization and due date tightness on the test problems are investigated at two levels. Low (-) resource utilization level, ranges [0.85 0.95] (ranges are defined as (min, max) pair) and High (+) resource utilization level ranges [0.50 0.80]. Tight (T) due date (delivered on time) and loose (L) due date (late delivery of %15) are examined.

Table 5.6 Machine alternatives of operations for different problem sizes

Problem	Machine flexibility		
	Small	Medium	Large
5 x 5	2	3	4
10x10	2	5	8
15x15	3	5	12

In order to evaluate the performance of the proposed algorithm, comparisons are made with dispatching rules similar to other studies in the DFJSSP literature. The

following dispatching rules are utilized: FIFO (first arrival at shop), EDD (earliest due date), SPT (shortest processing time), LPT (longest processing time), SL (smallest slack), CR (smallest critical ratio).

Results of the computational experiments are shown in Table 5.7, Table 5.8, and Table 5.9). The proposed GRASP algorithm employs the preemptive goal programming logic to evaluate generated schedules. Therefore, the first objective function (mean tardiness) is utilized to compare the proposed GRASP algorithm with heuristics. This is mainly due to the fact that in a preemptive programming logic secondary objectives can only be improved if primary objectives are not worsened. Therefore,it is not easy to compare other criteria directly. Moreover, the number of accepted orders must also be taken into account to make fair comparisons as number of accepted orders can be different for different rules/algorithms. Summary of the obtained results (mean tardiness and number of accepted orders) for each problem are shown in Figure 5.1, 5.2, and 5.3. For each problem, the proposed GRASP algorithm provided better mean tardiness values than dispatching rules (Figure 5.1, 5.2, 5.3) by accepting more orders. We can say that the proposed GRASP algorithm is able to provide much better solutions than the compared dispatching rules. In addition, results of all combinations are also analyzed by ANOVA for statistical significance. The results of analysis of variance (ANOVA) tests are shown in Table 5.10. The *F*-test *p*-values show that *'shop utilization'* and *'degree of problem flexibility'* are observed as the most effective factors on the scheduling environment for all different problem sizes (*p*-value < 0.05). The effect of '*due date tightness'* factor cannot be proved statistically (*p*-value > 0.05). This means that the effect of this factor on the flexible job shop scheduling performance is negligible. Similarly, Sabuncuoglu and Karapınar (1999) represent that due date tightness factor is not significant for FFLM method. Operations of jobs finish around their due dates for this reason, some jobs delay without consideration of the due date tightness levels. Furthermore, the *p*-values for the factor interactions indicate that there are some interactions among the factors. For example, the interaction between *'due date tightness'* and *'flexibility'* is shown in Table 5.10 (*p*-value <0.05) for all problem sizes.

Table 5.7 Comparison of the proposed GRASP algorithm and dispatching rules for 5x5 problems

Jobs	Machines	Utilization	Due date tightness	Methods	Flexibility level[a]	Mean Tardiness	Unstability	Makespan	Mean Flow Time	Number of Orders	Number of accepted orders
5	5	-	T	GRASP	0	81	5103	814	557	7	6
					1	34	5302	844	622	7	6
					2	0	4018	719	513	7	5
					3	0	3583	747	545	7	5
5	5	-	T	FIFO	0	581	2391	1025	604	7	4
					1	274	2287	1264	835	7	5
					2	256	821	990	513	7	5
					3	225	6223	1025	618	7	5
5	5	-	T	EDD	0	124	5687	1051	600	7	4
					1	437	5632	1661	928	7	6
					2	152	4561	1052	605	7	5
					3	22	4471	922	579	7	5
5	5	-	T	SPT	0	183	3036	874	641	7	5
					1	242	1156	1052	684	7	6
					2	185	1261	1095	651	7	6
					3	133	13288	867	596	7	6
5	5	-	T	LPT	0	262	3986	888	607	7	4
					1	221	13397	1026	690	7	6
					2	428	11181	1155	860	7	6
					3	155	8344	847	645	7	6
5	5	-	T	SL	0	102	5230	893	593	7	4
					1	271	7197	1362	818	7	6
					2	269	3278	926	608	7	6
					3	193	4490	990	593	7	5
5	5	-	T	CR	0	131	694	702	530	7	4
					1	88	3241	921	722	7	5
					2	200	4859	837	594	7	6
					3	1	2081	806	571	7	5
5	5	-	L	GRASP	0	47	3098	814	651	7	6
					1	19	2758	825	650	7	6
					2	0	8300	845	516	7	6
					3	0	4672	845	563	7	6
5	5	-	L	FIFO	0	360	1861	871	600	7	3
					1	640	8684	1692	960	7	6
					2	31	2059	831	516	7	5
					3	0	2486	749	491	7	5
5	5	-	L	EDD	0	111	2724	1125	694	7	5
					1	161	9239	1152	679	7	6
					2	0	1651	856	554	7	5
					3	0	1207	755	549	7	5
5	5	-	L	SPT	0	126	4175	1051	672	7	5
					1	415	4635	1467	948	7	6
					2	174	10066	1097	629	7	6
					3	100	9467	871	692	7	6
5	5	-	L	LPT	0	394	2499	888	661	7	4
					1	250	8376	920	740	7	6
					2	186	4826	1053	712	7	6
					3	30	5039	843	589	7	6
5	5	-	L	SL	0	99	1872	1054	714	7	5
					1	185	12143	1151	765	7	6
					2	211	9521	1121	674	7	6
					3	101	4277	887	652	7	5
5	5	-	L	CR	0	86	3847	768	562	7	5
					1	35	3351	853	676	7	5
					2	133	8536	884	605	7	6
					3	71	1991	862	690	7	6

[a] 0: Static problem, 1: Small flexibility level, 2: Medium flexibility level, 3: Large flexibility level.

Table 5.7 Comparison of the proposed GRASP algorithm and dispatching rules for 5x5 problems (continue)

Jobs	Machines	Utilization	Due date tightness	Methods	Flexibility level	Mean Tardiness	Unstability	Makespan	Mean Flow Time	Number of Orders	Number of accepted orders
5	5	+	T	GRASP	0	135	7557	802	695	7	5
					1	71	4012	777	642	7	6
					2	6	5858	787	553	7	5
					3	2	5884	802	537	7	5
5	5	+	T	FIFO	0	681	2635	1125	740	7	3
					1	151	41	1051	664	7	3
					2	151	0	1051	559	7	4
					3	0	1383	834	519	7	4
5	5	+	T	EDD	0	201	6274	1153	651	7	4
					1	151	41	1051	664	7	3
					2	112	2814	1053	607	7	3
					3	358	3185	1258	633	7	4
5	5	+	T	SPT	0	167	4065	1051	847	7	3
					1	252	8977	1219	780	7	5
					2	677	2735	1116	647	7	4
					3	672	2437	1121	648	7	4
5	5	+	T	LPT	0	270	12513	832	603	7	4
					1	510	36205	1506	980	7	5
					2	135	1466	1051	755	7	5
					3	472	4558	1247	872	7	5
5	5	+	T	SL	0	146	2745	1054	516	7	3
					1	151	811	1051	664	7	3
					2	111	2814	1052	607	7	4
					3	116	1032	799	624	7	4
5	5	+	T	CR	0	167	1405	755	539	7	5
					1	134	7920	850	725	7	4
					2	308	8886	775	582	7	5
					3	177	7151	1053	564	7	5
5	5	+	L	GRASP	0	106	8619	802	695	7	5
					1	66	4761	829	719	7	6
					2	33	517	571	462	7	3
					3	11	1341	817	505	7	4
5	5	+	L	FIFO	0	225	2	736	546	7	3
					1	439	2399	1504	960	7	5
					2	247	87	1092	598	7	4
					3	0	1383	834	519	7	4
5	5	+	L	EDD	0	190	2846	1127	599	7	4
					1	499	9492	1532	882	7	5
					2	377	58	114	779	7	4
					3	253	1419	1153	618	7	4
5	5	+	L	SPT	0	244	1821	1151	980	7	3
					1	224	9046	1095	708	7	5
					2	173	1300	811	585	7	4
					3	479	5968	1212	799	7	6
5	5	+	L	LPT	0	415	5048	1174	829	7	4
					1	451	36205	1506	990	7	6
					2	129	1845	833	647	7	5
					3	359	13954	1252	829	7	6
5	5	+	L	SL	0	253	5658	1231	875	7	5
					1	268	11643	1157	850	7	5
					2	0	918	832	552	7	4
					3	192	3264	834	753	7	5
5	5	+	L	CR	0	234	4394	1248	814	7	5
					1	230	10705	1151	945	7	5
					2	131	1481	762	581	7	5
					3	272	6010	1247	946	7	6

Table 5.8 Comparison of the proposed GRASP algorithm and dispatching rules for 10x10 problems

Jobs	Machines	Utilization	Due date tightness	Methods	Flexibility level	Mean Tardiness	Unstability	Makespan	Mean Flow Time	Number of Orders	Number of accepted orders
10	10	-	T	GRASP	0	399	65973	2981	2006	13	12
					1	256	43663	2719	1867	13	11
					2	111	54287	2501	1799	13	12
					3	81	78965	2356	1640	13	12
10	10	-	T	FIFO	0	856	1	2645	1790	13	3
					1	2785	14040	7514	4566	13	10
					2	1810	26815	5590	3430	13	10
					3	1540	163550	4560	2690	13	12
10	10	-	T	EDD	0	1140	51497	4214	2670	13	7
					1	2120	44108	6924	3901	13	11
					2	2840	285190	7990	4130	13	12
					3	1920	18814	6820	3330	13	12
10	10	-	T	SPT	0	1014	66023	3755	2288	13	8
					1	2510	156570	6800	4030	13	12
					2	1460	251950	4740	2980	13	12
					3	1410	251500	4490	2850	13	12
10	10	-	T	LPT	0	1142	38106	3033	2574	13	5
					1	1590	201040	4660	2940	13	12
					2	1220	194790	4530	2610	13	11
					3	1500	202430	4010	2740	13	12
10	10	-	T	SL	0	574	33080	3466	2310	13	7
					1	1760	130650	5600	3500	13	11
					2	970	15592	4630	2670	13	12
					3	270	123720	3290	1990	13	12
10	10	-	T	CR	0	603	33906	2773	2247	13	11
					1	2400	189830	5160	3900	13	11
					2	1100	164420	3440	2880	13	12
					3	606	91817	3160	2385	13	12
10	10	-	L	GRASP	0	309	65347	3195	2126	13	12
					1	188	35071	2701	2109	13	11
					2	177	38290	2612	2008	13	12
					3	128	57343	2480	1917	13	12
10	10	-	L	FIFO	0	553	2	2484	1718	13	3
					1	2703	27927	8090	4688	13	10
					2	2550	118290	8370	4320	13	11
					3	1970	213220	6180	3050	13	12
10	10	-	L	EDD	0	1650	122340	6330	3460	13	8
					1	1916	47935	6924	3901	13	11
					2	1950	150540	6340	3050	13	12
					3	1900	23046	7610	3720	13	12
10	10	-	L	SPT	0	2019	78847	5186	3776	13	9
					1	1490	202830	5110	3430	13	11
					2	1610	170110	5350	3310	13	12
					3	1090	184490	4730	2980	13	12
10	10	-	L	LPT	0	1261	54493	3450	2954	13	6
					1	2200	265140	5700	4000	13	12
					2	1940	174960	4680	3410	13	12
					3	1580	202430	4010	2740	13	12
10	10	-	L	SL	0	482	34341	3466	2401	13	9
					1	2168	58321	6656	3910	13	11
					2	1130	117390	4520	3120	13	12
					3	310	57374	3096	2193	13	12
10	10	-	L	CR	0	891	58638	3567	3752	13	11
					1	1430	127120	4720	3420	13	12
					2	837	59020	3451	2825	13	11
					3	685	76858	3450	2610	13	12

Table 5.8 Comparison of the proposed GRASP algorithm and dispatching rules for 10x10 problems (continue)

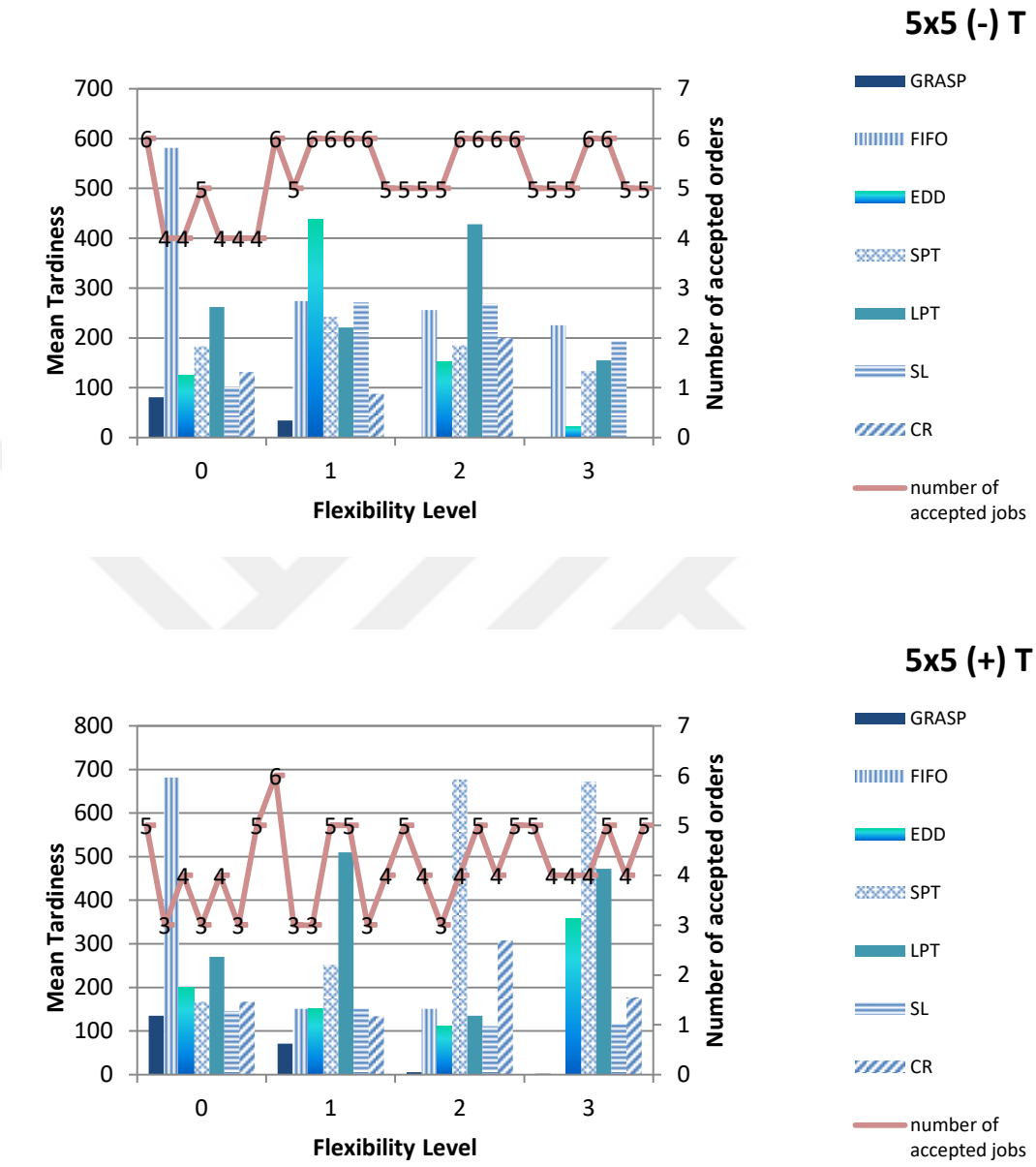
Jobs	Machines	Utilization	Due date tightness	Methods	Flexibility level	Mean Tardiness	Unstability	Makespan	Mean Flow Time	Number of Orders	Number of accepted orders
10	10	+	T	GRASP	0	693	47168	3065	2493	13	8
					1	455	67567	2981	2288	13	11
					2	301	84278	2720	2092	13	11
					3	202	85685	2981	1924	13	11
10	10	+	T	FIFO	0	1828	9	6397	3610	13	6
					1	2753	29885	6900	4258	13	9
					2	3160	237490	8260	4680	13	11
					3	3210	241230	8850	4390	13	11
10	10	+	T	EDD	0	742	19295	3208	2283	13	4
					1	2330	125170	7030	4140	13	9
					2	3040	288920	9430	4870	13	11
					3	2420	29760	7030	3750	13	11
10	10	+	T	SPT	0	1245	47336	3922	2910	13	5
					1	2590	27930	6320	3880	13	11
					2	2140	243550	5830	3720	13	11
					3	2120	278930	6480	3970	13	11
10	10	+	T	LPT	0	1990	84743	5381	3862	13	6
					1	3370	189080	7040	4840	13	10
					2	2210	387480	5540	4040	13	11
					3	1920	318700	4630	3580	13	10
10	10	+	T	SL	0	673	28424	2646	2308	13	5
					1	1983	94133	6343	3368	13	9
					2	1850	101880	6340	3680	13	11
					3	1200	202550	4520	3030	13	11
10	10	+	T	CR	0	373	22287	2504	2150	13	6
					1	2502	95103	5129	4326	13	9
					2	1250	106820	3920	3070	13	11
					3	1238	97060	3683	3070	13	11
10	10	+	L	GRASP	0	641	67223	3442	2662	13	10
					1	435	62862	3189	2495	13	11
					2	337	60368	3192	2302	13	11
					3	216	68223	2714	2195	13	11
10	10	+	L	FIFO	0	1566	3	3605	2464	13	3
					1	3303	3771	8887	5026	13	11
					2	2480	191700	8260	4540	13	11
					3	2210	215400	6542	3520	13	12
10	10	+	L	EDD	0	1352	47752	5584	3199	13	5
					1	3180	150350	10180	5240	13	11
					2	3370	299050	10660	5430	13	11
					3	2890	306020	9700	4950	13	11
10	10	+	L	SPT	0	1503	59144	4882	2887	13	8
					1	2750	184780	7410	4810	13	11
					2	3020	302190	7990	4800	13	11
					3	2230	284340	6450	4290	13	11
10	10	+	L	LPT	0	1632	44822	3678	3208	13	5
					1	3610	228390	8250	5670	13	11
					2	2240	259690	5520	4030	13	11
					3	1970	259690	5520	4030	13	11
10	10	+	L	SL	0	898	36812	3545	2853	13	5
					1	2460	12700	7510	4520	13	11
					2	1540	136880	4900	3600	13	11
					3	1210	150770	4630	3270	13	11
10	10	+	L	CR	0	1604	71501	4421	3664	13	11
					1	2480	188760	6400	4540	13	11
					2	1530	165480	4660	3580	13	11
					3	1070	128420	3920	3130	13	11

Table 5.9 Comparison of the proposed GRASP algorithm and dispatching rules for 15x15 problems

Jobs	Machines	Utilization	Due date tightness	Methods	Flexibility level	Mean Tardiness	Unstability	Makespan	Mean Flow Time	Number of Orders	Number of accepted orders
15	15	-	T	GRASP	0	810	384590	4730	3520	20	18
					1	340	211260	4010	3060	20	19
					2	300	272860	4170	3040	20	19
					3	260	236910	4250	3060	20	19
15	15	-	T	FIFO	0	1101	8	3926	2748	20	3
					1	6700	1075800	16400	8700	20	19
					2	5500	1346200	14600	7400	20	19
					3	5500	1863100	15500	7100	20	19
15	15	-	T	EDD	0	2160	153930	7280	4780	20	6
					1	5800	1331100	16900	8600	20	19
					2	5100	1600400	14500	7100	20	19
					3	5000	1786800	15000	7700	20	19
15	15	-	T	SPT	0	3450	593410	9310	6330	20	13
					1	4900	2671700	12300	7500	20	19
					2	5600	2596200	14100	8400	20	19
					3	3400	1672900	10300	5900	20	19
15	15	-	T	LPT	0	1894	52871	5931	4211	20	7
					1	3900	1604800	9700	6800	20	19
					2	2800	1440300	7300	5400	20	19
					3	2300	1526900	7000	5200	20	19
15	15	-	T	SL	0	980	139540	5370	3690	20	14
					1	3030	955430	10480	5910	20	19
					2	730	503320	5060	3610	20	19
					3	690	338650	4650	3570	20	19
15	15	-	T	CR	0	940	110340	3990	3690	20	16
					1	3900	1021300	8600	6700	20	19
					2	1480	361940	5110	4760	20	19
					3	1080	515090	4750	3960	20	19
15	15	-	L	GRASP	0	650	412050	4760	3710	20	20
					1	370	260710	4590	3500	20	19
					2	350	226870	4200	3270	20	19
					3	320	245340	4250	3410	20	19
15	15	-	L	FIFO	0	2019	18742	7071	3801	20	6
					1	8000	1137600	18200	9400	20	19
					2	5200	1723800	13800	6900	20	19
					3	4700	1874800	13700	6900	20	19
15	15	-	L	EDD	0	2890	336040	9750	5850	20	8
					1	5300	117300	16700	8600	20	19
					2	5000	1413700	15000	7600	20	19
					3	4900	1732600	14600	7200	20	19
15	15	-	L	SPT	0	5640	918840	12560	8500	20	17
					1	3700	1810200	10800	6500	20	19
					2	5400	2198400	15100	8000	20	19
					3	4100	2553100	12800	6900	20	19
15	15	-	L	LPT	0	2540	204990	7090	5390	20	7
					1	3300	1395100	8600	6400	20	19
					2	3100	1785200	8100	6400	20	19
					3	3000	1796700	9200	6300	20	19
15	15	-	L	SL	0	1240	158510	5620	4390	20	15
					1	2850	652160	10970	6130	20	19
					2	1270	368890	5930	4560	20	19
					3	1040	338830	5590	4320	20	19
15	15	-	L	CR	0	820	211000	4390	4030	20	18
					1	3880	786810	8880	6760	20	19
					2	1340	505560	4740	4220	20	19
					3	1010	44509	4860	4290	20	19

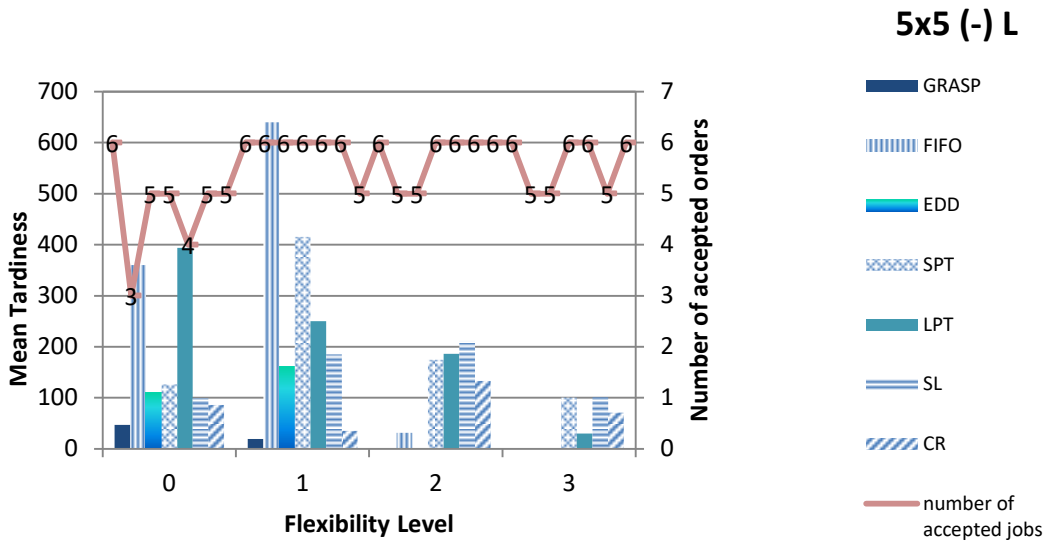
Table 5.9 Comparison of the proposed GRASP algorithm and dispatching rules for 15x15 problems (continue)

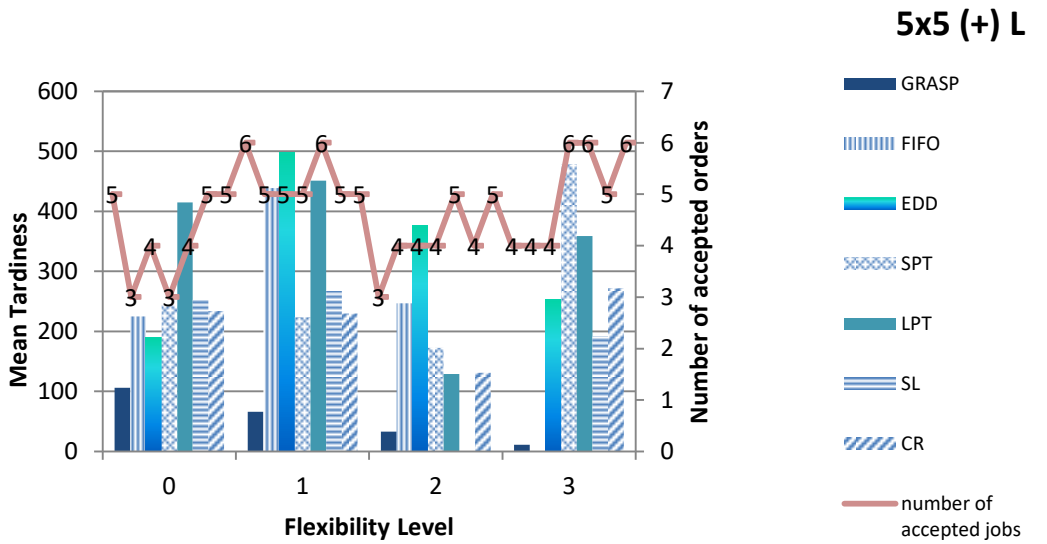
Jobs	Machines	Utilization	Due date tightness	Methods	Flexibility level	Mean Tardiness	Unstability	Makespan	Mean Flow Time	Number of Orders	Number of accepted orders
15	15	+	T	GRASP	0	1180	291040	5070	3860	20	14
					1	730	676920	4570	3540	20	18
					2	610	620390	4170	3420	20	19
					3	550	547830	4150	3420	20	19
15	15	+	T	FIFO	0	2983	1744	8333	5263	20	5
					1	8480	967010	19460	10810	20	16
					2	8400	1766900	20400	98000	20	19
					3	7500	2980700	1900	8600	20	19
15	15	+	T	EDD	0	2780	161380	8390	5470	20	5
					1	9700	1993900	22300	12000	20	19
					2	8300	3097600	21900	11200	20	19
					3	6600	2697300	18900	8700	20	19
15	15	+	T	SPT	0	2250	184110	8240	5190	20	8
					1	8600	4137100	18800	11000	20	19
					2	5400	3526100	15600	8300	20	19
					3	3100	329500	9900	5600	20	19
15	15	+	T	LPT	0	2380	105430	6570	5320	20	6
					1	5600	2416300	12700	8500	20	19
					2	5000	2683200	11300	7600	20	19
					3	2500	2856400	8300	5400	20	19
15	15	+	T	SL	0	192	33755	3132	2839	20	8
					1	3700	1165000	13100	6600	20	18
					2	1820	731660	6570	4700	20	19
					3	1120	645570	5540	4000	20	19
15	15	+	T	CR	0	145	7363	2754	2685	20	6
					1	5700	1643900	11900	8600	20	18
					2	2000	645870	5610	5290	20	19
					3	1820	904240	5830	5100	20	19
15	15	+	L	GRASP	0	990	255240	5010	4070	20	15
					1	700	430340	4660	3880	20	19
					2	630	438730	4900	3760	20	19
					3	540	533430	4900	3700	20	19
15	15	+	L	FIFO	0	2673	1735	8333	5263	20	5
					1	9100	1712200	21400	11400	20	19
					2	10400	2424700	24600	11900	20	19
					3	7000	3414500	18500	9500	20	19
15	15	+	L	EDD	0	2390	162790	8390	5100	20	6
					1	8700	1857400	20500	11400	20	19
					2	6800	2352000	17900	9300	20	19
					3	6600	2697300	18900	8700	20	19
15	15	+	L	SPT	0	2690	443400	9290	6000	20	9
					1	7100	3443200	16400	10400	20	19
					2	5000	3827500	15700	8000	20	19
					3	4300	3409500	11900	6900	20	19
15	15	+	L	LPT	0	2540	108930	7040	5340	20	5
					1	5200	1796600	11900	8200	20	19
					2	4800	2462200	10700	8100	20	19
					3	4400	2349100	10400	7700	20	19
15	15	+	L	SL	0	876	88836	5138	4037	20	12
					1	5000	1170100	14100	8200	20	19
					2	1840	570560	6820	5120	20	19
					3	1400	805480	6480	4680	20	19
15	15	+	L	CR	0	1040	129190	4480	4210	20	12
					1	5100	152300	10700	8000	20	18
					2	1980	804960	5440	4860	20	19
					3	1580	636320	5100	4460	20	19



****(-) low resource utilization, (+) high resource utilization, T-Tight due date, L-loose due date.*

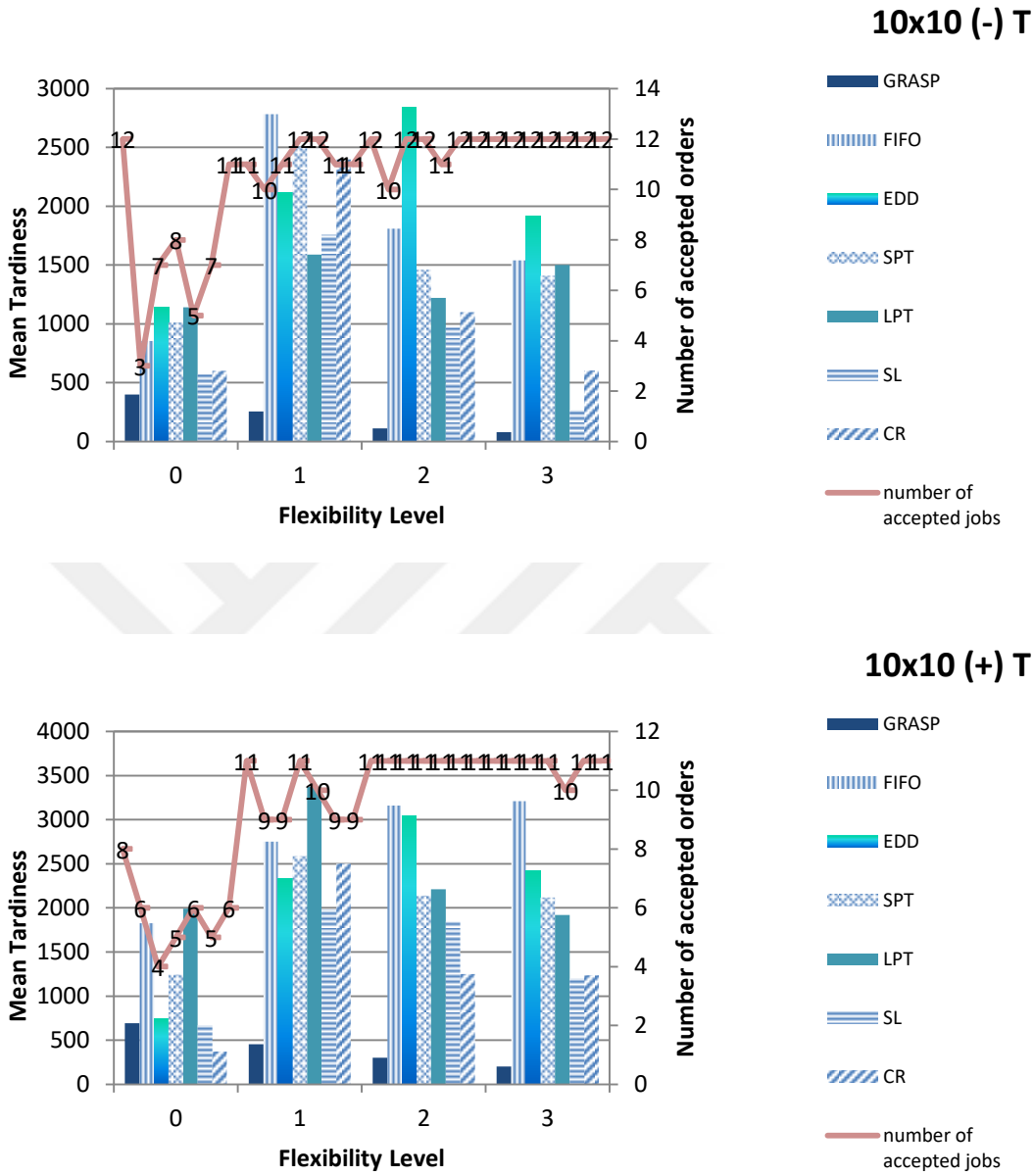
Figure 5.1 Comparison of the proposed GRASP and dispatching rules for 5x5 problems





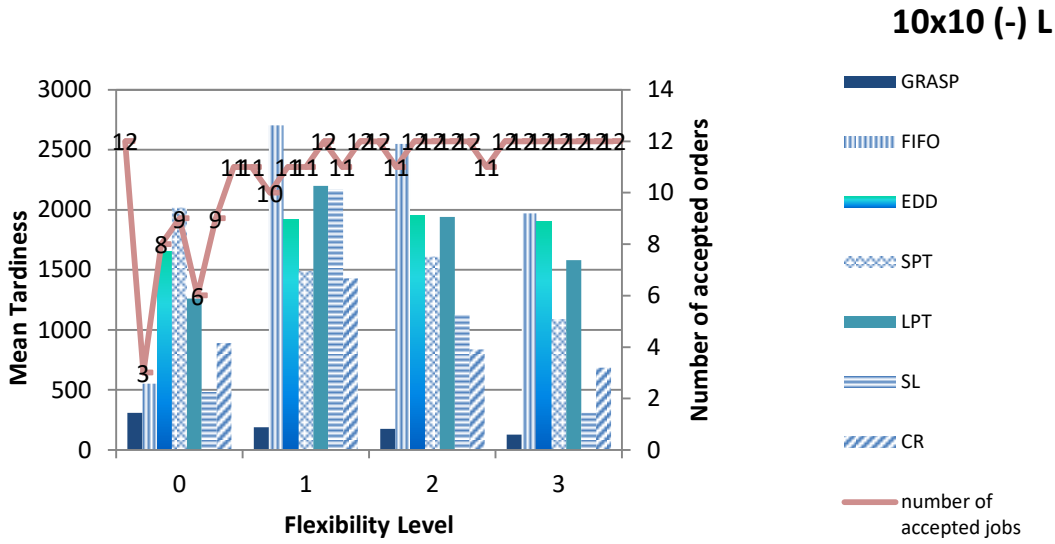
****(-) low resource utilization, (+) high resource utilization, T-Tight due date, L-loose due date.*

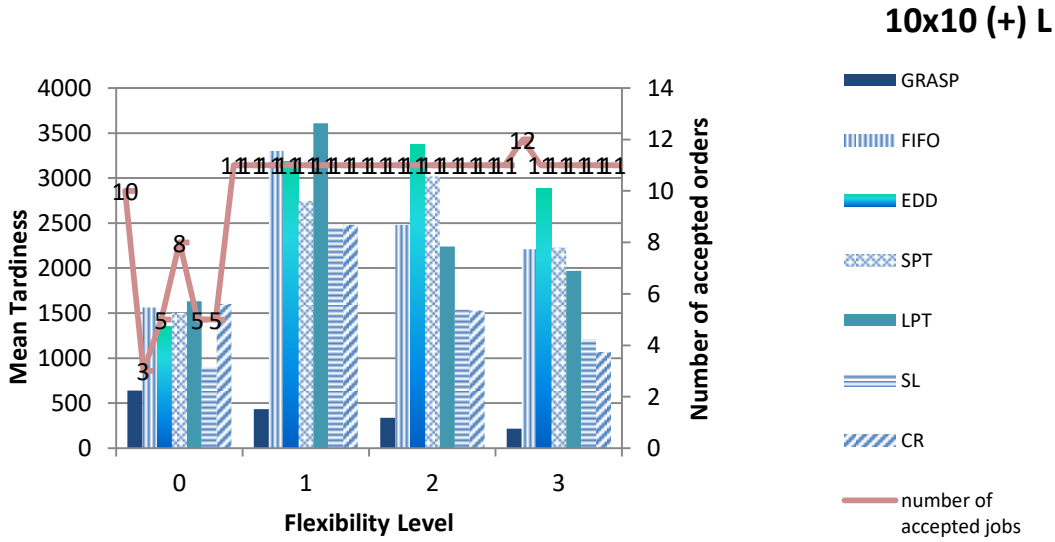
Figure 5.1 Comparison of the proposed GRASP and dispatching rules for 5x5 problems (continue)



****(-) low resource utilization, (+) high resource utilization, T-Tight due date, L-loose due date.*

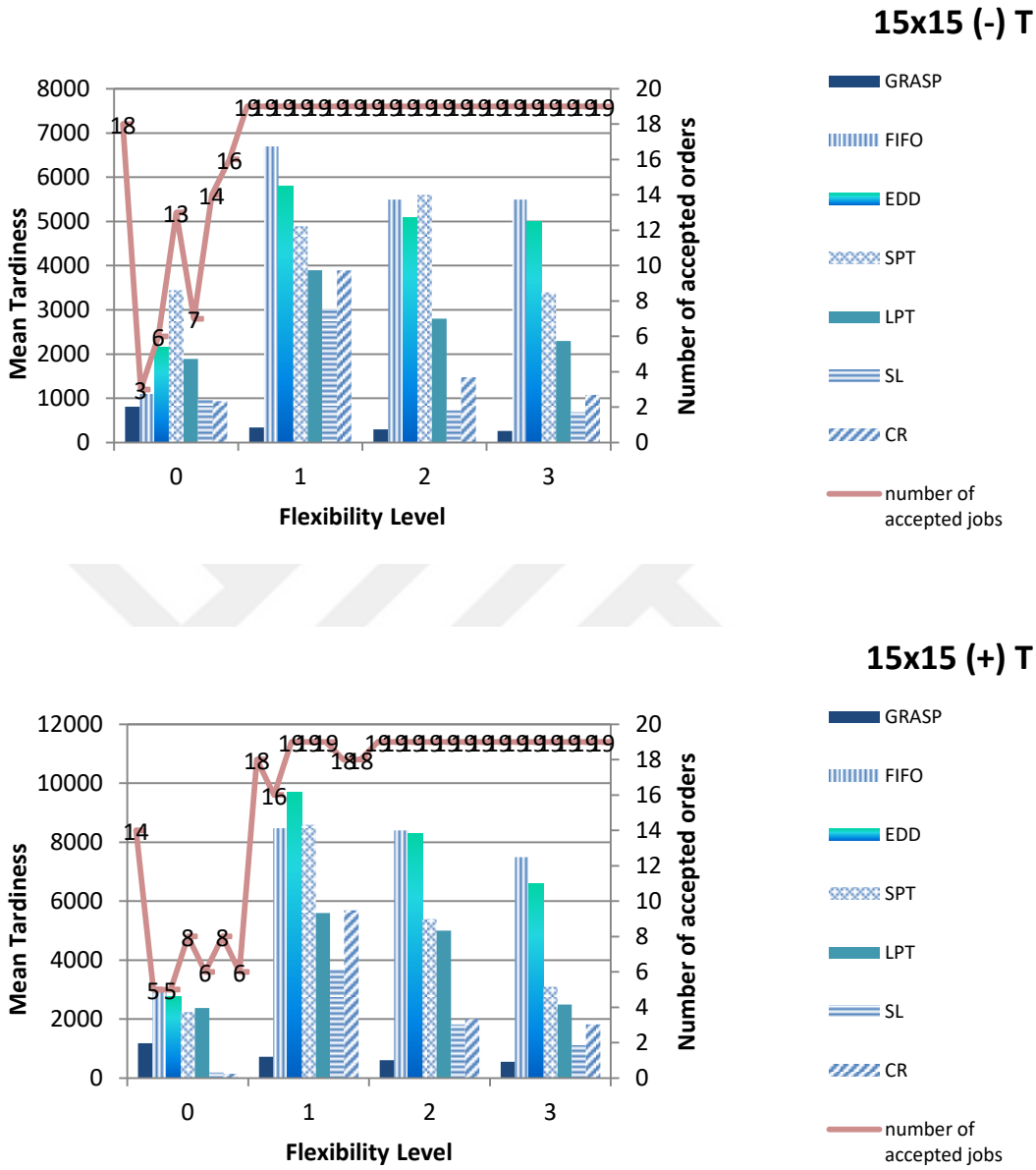
Figure 5.2 Comparison of the proposed GRASP and dispatching rules for 10x10 problems





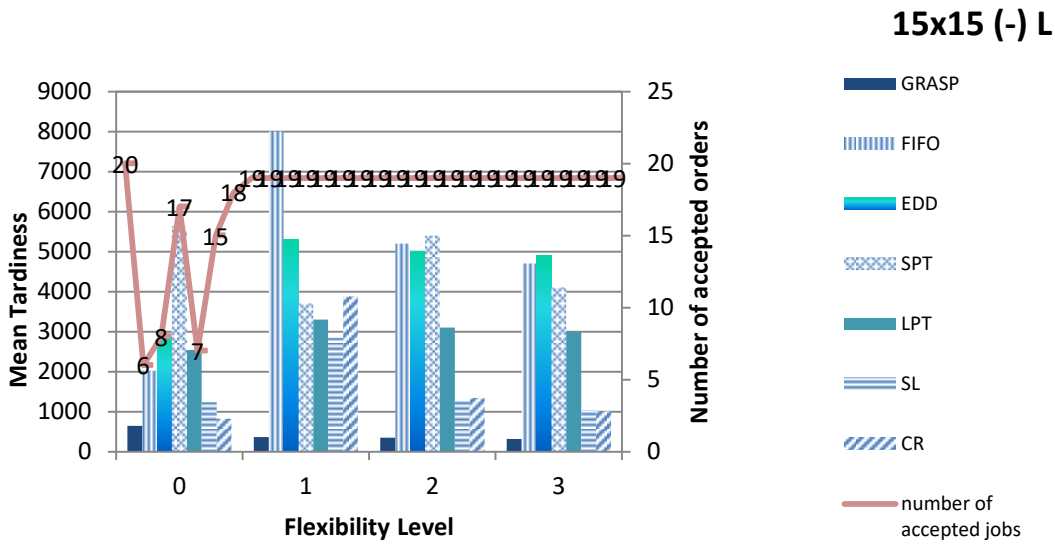
****(-) low resource utilization, (+) high resource utilization, T-Tight due date, L-loose due date.*

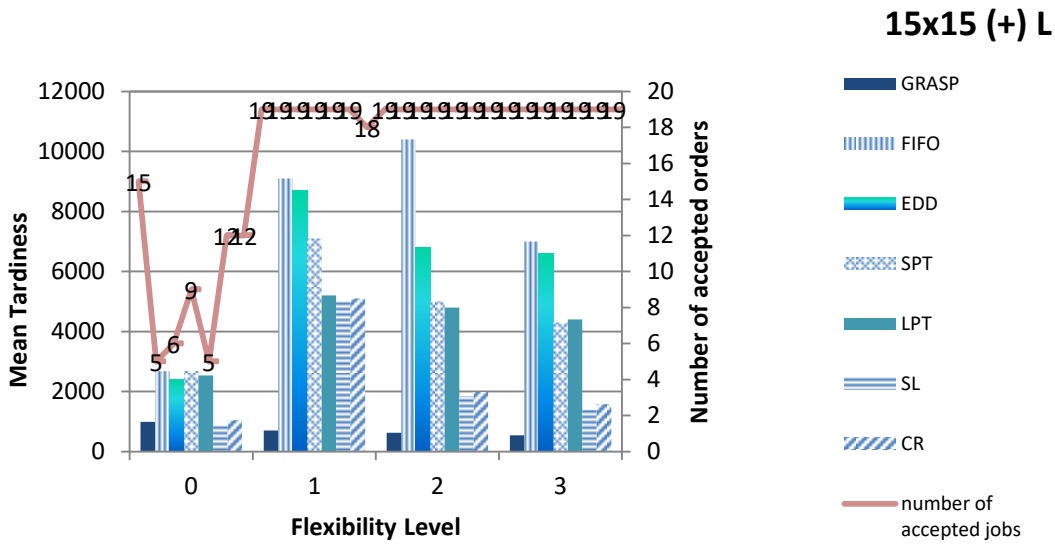
Figure 5.2 Comparison of the proposed GRASP and dispatching rules for 10x10 problems (continue)



****(-) low resource utilization, (+) high resource utilization, T-Tight due date, L-loose due date.*

Figure 5.3 Comparison of the proposed GRASP and dispatching rules for 15x15 problems





****(-) low resource utilization, (+) high resource utilization, T-Tight due date, L-loose due date.*

Figure 5.3 Comparison of the proposed GRASP and dispatching rules for 15x15 problems (continue)

Table 5.10 ANOVA results (*p*-value) for factor effects and interactions

Source/ Problem sizes	P (5 x 5)	P (10 x 10)	P (15 x 15)
Shop utilization	0.000	0.000	0.000
Due date tightness	0.098	0.487	0.284
Flexibility	0.000	0.000	0.000
Shop utilization * Due date tightness	0.017	0.303	0.189
Shop utilization * Flexibility	0.000	0.000	0.240
Due date tightness * Flexibility	0.000	0.019	0.011

5.2.3 Experiment III

In this group of experiments, the results of the proposed GRASP algorithm under event-driven rescheduling policy is compared with the results of the proposed GRASP algorithm under periodical rescheduling policy in order to compare the performance of different rescheduling policies. The periodical rescheduling policy generates the schedules at regular interval and the scheduling periods are considered as 120 minutes, 180 minutes, and 240 minutes. The comparisons of periodical and event-driven rescheduling policies are given in Tables 5.11, 5.12, and 5.13. Mean tardiness and the number of accepted orders for each problem are shown in Figure 5.4, 5.5, and 5.6. It can be seen that flexibility helps to improve scheduling performance with decredation of the mean tardiness or the increment of the number of the accepted orders for periodic and event-driven rescheduling policies.

In order to compare the event-driven rescheduling policy with the periodical rescheduling policy statistically, we check the independence, normality and heteroscedasticity properties in order to see whether we can apply or not parametric statistical tests. The parametric test conditions are not fulfilled perfectly. These results induce us to using non parametric statistics to analyze the results. Friedman test is applied. The null hypothesis for Friedman's test remarks the equality of medians between the algorithms. Computation of the p-value is given a F_F statistic which is done according to Friedman test procedure. Holm's post-hoc procedures are selected for the implementation of the Friedman test for applying multiple comparisons process. This hypothesis for 5x5 problems show the improvement of the event-driven rescheduling policy over the periodical rescheduling policy with 120, 180, and 240 scheduling frequency and the improvement of periodical rescheduling

policy with 240 scheduling interval over the periodical rescheduling policy with 120, and 180 scheduling frequency, and the improvement of the periodical rescheduling policy with 180 scheduling frequency over the periodical rescheduling policy with 120 scheduling frequency. This hypothesis for 10x10 problems shows the improvement of the periodical rescheduling policy with 120, 180 and 240 scheduling frequency over the event-driven rescheduling policy and the improvement of the periodical rescheduling policy with 120 and 180 scheduling frequency over the periodical rescheduling policy with 240 scheduling frequency and the improvement of periodical rescheduling policy with 120 over 180 scheduling frequency. This hypothesis for 15x15 problems shows the improvement of the periodical rescheduling policy with 120, 180 and 240 scheduling frequency over the event-driven rescheduling policy and the improvement of periodical rescheduling policy with 120 and 180 scheduling frequency over the 240 scheduling frequency and the improvement of the periodical rescheduling policy with 120 over the 180 scheduling frequency. The periodical rescheduling policy gathers all of the information from the shop floor at the specific time period and evaluates this information during the rescheduling process. This inevitably enables it to provide better results and the production decision may be more accurate because of the accumulated information. However, event-driven rescheduling policy takes into account dynamic events instantly and this may lead ineffective schedules in some occasions. Results clearly indicate that the selection of the rescheduling policy is an important and critical decision for dynamic scheduling performance.

Table 5.11 Comparison of event-driven and periodical rescheduling policies for 5x5 problems

Jobs	Machines	Utilization	Due date tightness	GRASP (event driven, periodic: 120, 180,240)	Flexibility level[a]	Mean Tardiness	Unstability	Makespan	Mean Flow Time	Number of Orders	Number of accepted orders
5	5	-	T	Event driven	0	81	5103	814	557	7	6
					1	34	5302	844	622	7	6
					2	0	4018	719	513	7	5
					3	0	3583	747	545	7	5
5	5	-	T	120	0	115	7850	866	677	7	6
					1	110	4259	774	576	7	5
					2	90	1708	773	592	7	5
					3	90	474	835	536	7	6
5	5	-	T	180	0	132	10781	848	658	7	6
					1	73	60	774	517	7	5
					2	71	47	772	445	7	5
					3	103	1032	899	728	7	6
5	5	-	T	240	0	40	17412	906	588	7	6
					1	65	14525	827	609	7	6
					2	50	6164	911	607	7	5
					3	42	9053	908	600	7	5
5	5	-	L	Event driven	0	47	3098	814	651	7	6
					1	19	2758	825	650	7	6
					2	0	8300	845	516	7	6
					3	0	4672	845	563	7	6
5	5	-	L	120	0	145	1753	926	790	7	6
					1	86	1868	864	669	7	5
					2	58	600	838	669	7	6
					3	42	1163	884	606	7	6
5	5	-	L	180	0	126	3589	1052	726	7	6
					1	64	6519	895	606	7	6
					2	63	55	895	558	7	6
					3	60	5210	899	600	7	6
5	5	-	L	240	0	93	8223	880	633	7	6
					1	72	47	878	547	7	5
					2	100	548	885	674	7	6
					3	18	999	873	600	7	6
5	5	+	T	Event driven	0	135	7557	802	695	7	5
					1	71	4012	777	642	7	6
					2	6	5858	787	553	7	5
					3	2	5884	802	537	7	5
5	5	+	T	120	0	129	9225	802	635	7	5
					1	86	8430	914	586	7	6
					2	97	1324	672	540	7	6
					3	79	208	780	550	7	6
5	5	+	T	180	0	177	20751	1051	722	7	5
					1	100	3961	676	577	7	6
					2	78	9863	837	656	7	3
					3	70	37837	813	593	7	5
5	5	+	T	240	0	97	10943	1051	650	7	5
					1	94	3272	788	667	7	5
					2	28	9375	828	572	7	5
					3	22	4096	781	565	7	6
5	5	+	L	Event driven	0	106	8619	802	695	7	5
					1	66	4761	829	719	7	6
					2	33	517	571	462	7	3
					3	11	1341	817	505	7	4
5	5	+	L	120	0	251	10064	1051	883	7	5
					1	120	3125	831	703	7	5
					2	119	32769	1024	763	7	6
					3	105	6756	809	653	7	6
5	5	+	L	180	0	154	25803	816	641	7	5
					1	114	648	1052	738	7	6
					2	127	0	987	666	7	6
					3	65	2079	988	665	7	6
5	5	+	L	240	0	133	11051	1153	754	7	5
					1	89	47	988	654	7	6
					2	81	25	990	600	7	6
					3	0	4785	800	592	7	6

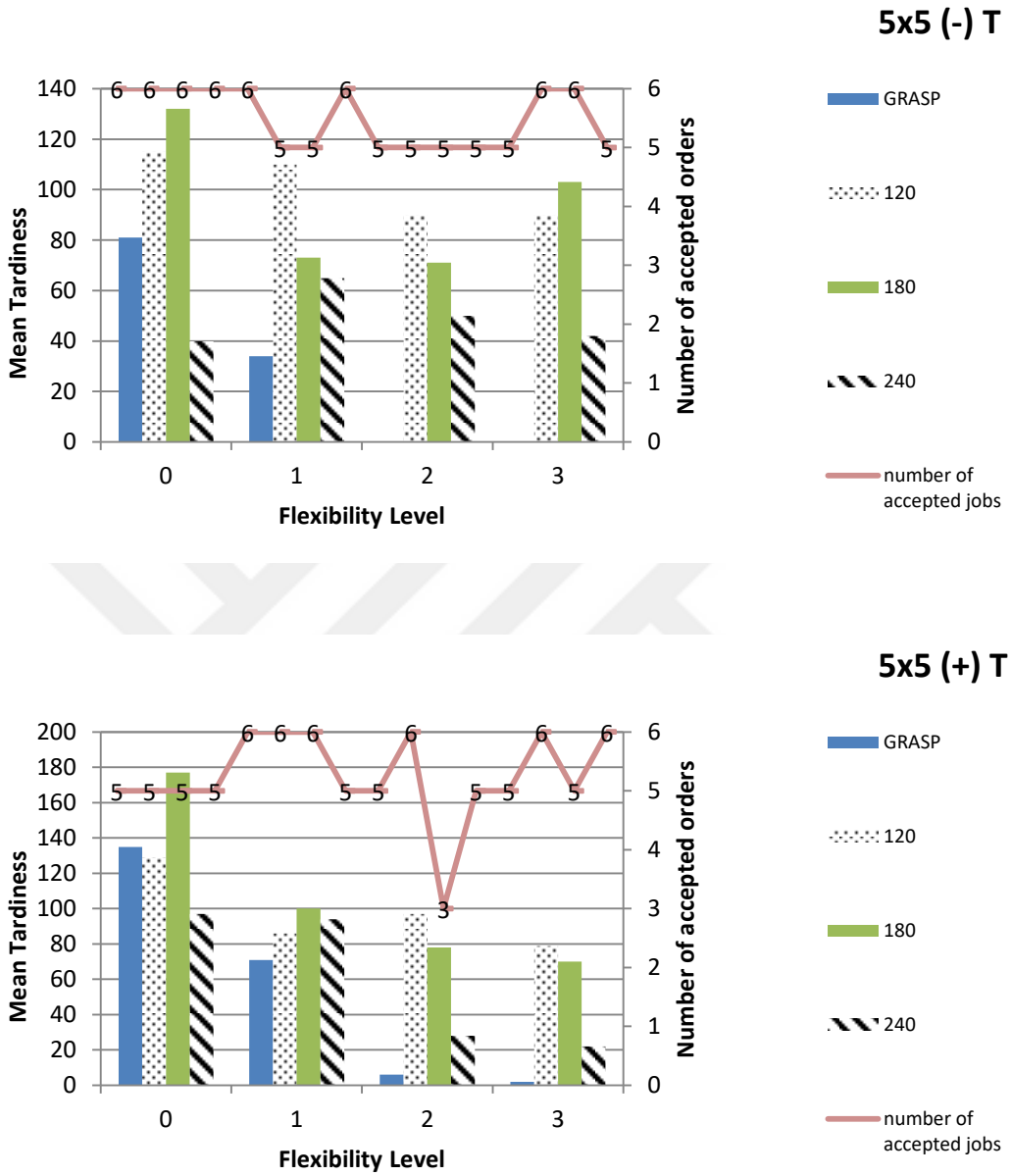
[a] 0: Static problem, 1: Small flexibility level, 2: Medium flexibility level, 3: Large flexibility level.

Table 5.12 Comparison of event-driven and periodical rescheduling policies for 10x10 problems

Jobs	Machines	Utilization	Due date tightness	GRASP (event driven , periodic: 120, 180,240)	Flexibility level	Mean Tardiness	Unstability	Makespan	Mean Flow Time	Number of Orders	Number of accepted orders
10	10	-	T	Event driven	0	399	65973	2981	2006	13	12
					1	256	43663	2719	1867	13	11
					2	111	54287	2501	1799	13	12
					3	81	78965	2356	1640	13	12
10	10	-	T	120	0	374	57607	2723	1948	13	10
					1	149	382	1784	1511	13	12
					2	44	2939	1884	1529	13	12
					3	0	131	1756	1443	13	12
10	10	-	T	180	0	475	60764	6065	2109	13	12
					1	51	718	1784	1414	13	12
					2	23	34	1779	1395	13	12
					3	4	241	1786	1390	13	12
10	10	-	T	240	0	580	155590	3090	2220	13	12
					1	98	554	1786	1400	13	12
					2	79	1006	2009	1529	13	11
					3	61	699	1880	1451	13	11
10	10	-	L	Event driven	0	309	65347	3195	2126	13	12
					1	188	35071	2701	2109	13	11
					2	177	38290	2612	2008	13	12
					3	128	57343	2480	1917	13	12
10	10	-	L	120	0	429	86660	2985	2199	13	11
					1	156	39360	2162	1841	13	12
					2	17	5387	2159	1709	13	11
					3	16	1416	2108	1488	13	12
10	10	-	L	180	0	526	82639	3253	2288	13	12
					1	65	804	2164	1512	13	12
					2	20	337	2159	1482	13	12
					3	17	452	2159	1571	13	12
10	10	-	L	240	0	587	49166	3090	2402	13	12
					1	67	165	2269	1502	13	12
					2	66	207	2260	1513	13	12
					3	48	1237	2107	1550	13	12
10	10	+	T	Event driven	0	693	47168	3065	2493	13	11
					1	455	67567	2981	2288	13	11
					2	301	84278	2720	2092	13	11
					3	202	85685	2981	1924	13	11
10	10	+	T	120	0	747	63990	3465	2312	13	10
					1	117	16152	2026	1786	13	11
					2	118	12598	2021	1608	13	11
					3	94	57	2024	1806	13	11
10	10	+	T	180	0	850	125841	3850	2541	13	11
					1	128	6025	2087	1606	13	11
					2	110	6617	2021	1621	13	11
					3	96	216	2021	1661	13	11
10	10	+	T	240	0	990	148530	3680	2630	13	12
					1	281	3696	2130	1838	13	11
					2	204	3734	2024	1764	13	11
					3	156	2799	2088	1765	13	11
10	10	+	L	Event driven	0	641	67223	3442	2662	13	10
					1	435	62862	3189	2495	13	11
					2	337	60368	3192	2302	13	11
					3	216	68223	2714	2195	13	11
10	10	+	L	120	0	628	56291	3574	2419	13	11
					1	120	176290	2280	1930	13	11
					2	89	2699	2141	1808	13	11
					3	63	3059	2267	1782	13	11
10	10	+	L	180	0	910	136280	3920	2760	13	12
					1	107	389	2163	1782	13	11
					2	74	3351	2108	1753	13	11
					3	64	411	2087	1730	13	11
10	10	+	L	240	0	977	74929	3671	2824	13	12
					1	216	394	2244	1785	13	11
					2	116	6462	2271	1690	13	11
					3	85	6983	2276	1769	13	11

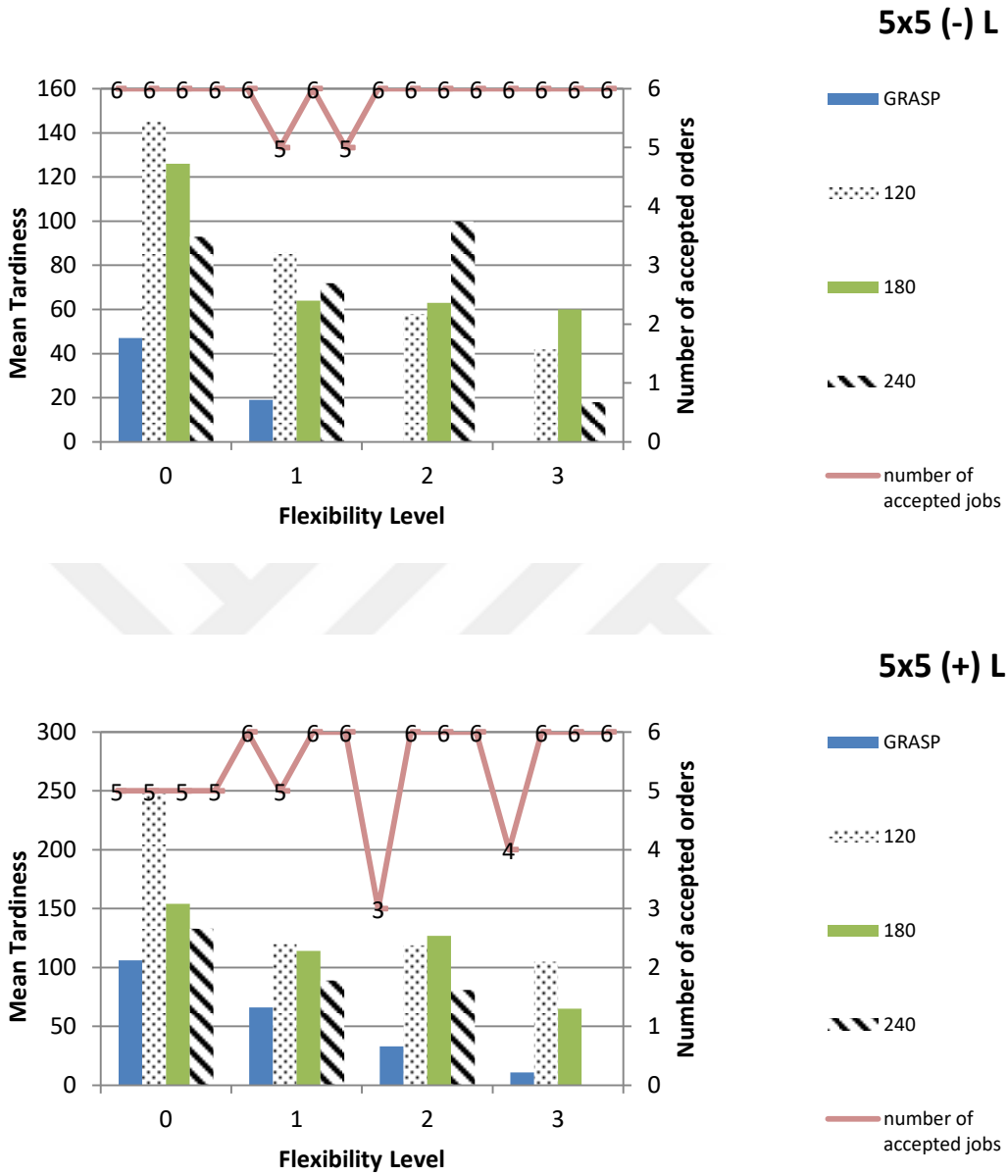
Table 5.13 Comparison of event-driven and periodical rescheduling policies for 15x15 problems

Jobs	Machines	Utilization	Due date tightness	GRASP (event driven , periodic: 120, 180,240)	Flexibility level	Mean Tardiness	Unstability	Makespan	Mean Flow Time	Number of Orders	Number of accepted orders
15	15	-	T	Event driven	0	810	384590	4730	3520	20	18
					1	340	211260	4010	3060	20	19
					2	290	295930	4010	3050	20	19
					3	260	236910	4250	3060	20	19
15	15	-	T	120	0	1020	453650	4870	3630	20	19
					1	79	6572	2770	2289	20	19
					2	3	1571	2661	2191	20	18
					3	32	6241	2844	2259	20	19
15	15	-	T	180	0	960	155770	4660	3600	20	19
					1	73	14104	2825	2280	20	19
					2	72	20755	2746	2214	20	19
					3	71	29655	2843	2253	20	19
15	15	-	T	240	0	1220	187780	4880	3890	20	19
					1	57	4111	2775	2315	20	19
					2	28	3917	2771	2318	20	19
					3	27	29100	2760	2345	20	19
15	15	-	L	Event driven	0	650	412050	4760	3710	20	20
					1	370	260710	4590	3500	20	19
					2	350	226870	4200	3270	20	19
					3	320	245340	4250	3410	20	19
15	15	-	L	120	0	950	100720	4870	4000	20	18
					1	58	2944	3250	2490	20	19
					2	31	751	3056	2399	20	19
					3	31	782	3241	2445	20	19
15	15	-	L	180	0	640	100992	4560	3650	20	19
					1	104	33311	3251	2472	20	19
					2	89	46372	3336	2400	20	19
					3	62	45345	3332	2408	20	19
15	15	-	L	240	0	690	217660	4490	3720	20	19
					1	106	36613	3249	2542	20	19
					2	39	42734	3251	2505	20	19
					3	37	46881	3244	2501	20	19
15	15	+	T	Event driven	0	1180	291040	5070	3860	20	14
					1	730	676920	4570	3540	20	18
					2	610	620390	4170	3420	20	19
					3	550	547830	4150	3420	20	19
15	15	+	T	120	0	1590	249030	5550	4340	20	17
					1	113	2094	2765	2513	20	19
					2	92	5887	2762	2491	20	19
					3	76	1426	2770	2512	20	19
15	15	+	T	180	0	1540	481210	5360	4350	20	18
					1	87	8075	2765	2414	20	19
					2	81	8120	2946	2424	20	19
					3	78	8410	2945	2480	20	19
15	15	+	T	240	0	1810	198790	5940	4570	20	18
					1	111	3903	3029	2519	20	19
					2	105	2654	2986	2634	20	19
					3	86	4366	2752	2581	20	19
15	15	+	L	Event driven	0	990	255240	5010	4070	20	15
					1	700	430340	4660	3880	20	19
					2	630	438730	4900	3760	20	19
					3	540	533430	4900	3700	20	19
15	15	+	L	120	0	1480	212200	5920	4500	20	18
					1	134	4372	3429	2702	20	19
					2	98	2184	3244	2668	20	19
					3	97	1471	3245	2664	20	19
15	15	+	L	180	0	1310	163670	5610	4470	20	18
					1	149	30676	3429	2666	20	19
					2	121	41833	3251	2619	20	19
					3	118	14627	3240	2596	20	19
15	15	+	L	240	0	1400	198140	5910	4550	20	18
					1	194	37757	3435	2833	20	19
					2	191	22673	3255	2789	20	19
					3	140	22705	3246	2792	20	19



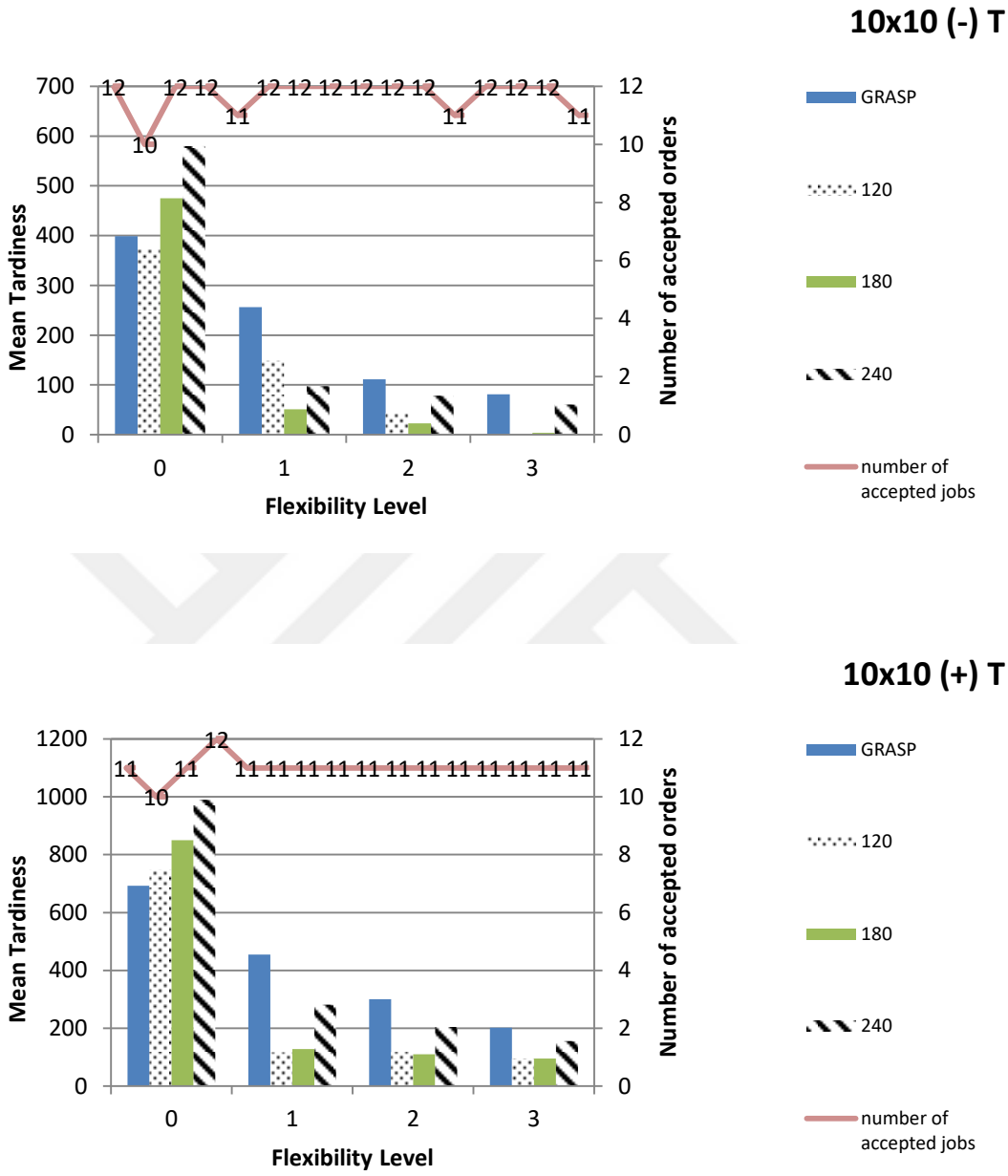
****(-) low resource utilization, (+) high resource utilization, T-Tight due date, L-loose due date.*

Figure 5.4 Comparison of the event-driven and the periodical rescheduling policies for 5x5 problems



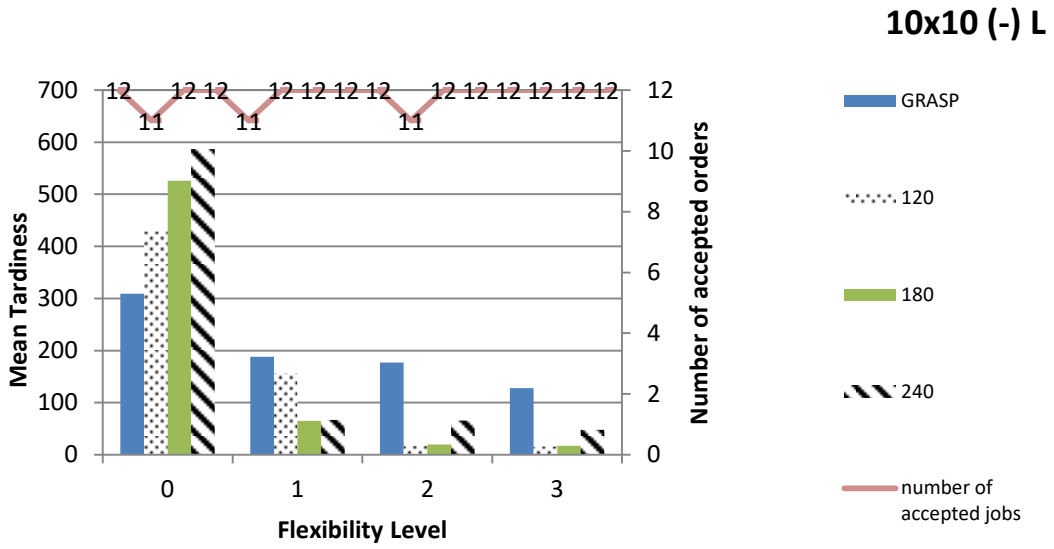
****(-) low resource utilization, (+) high resource utilization, T-Tight due date, L-loose due date.*

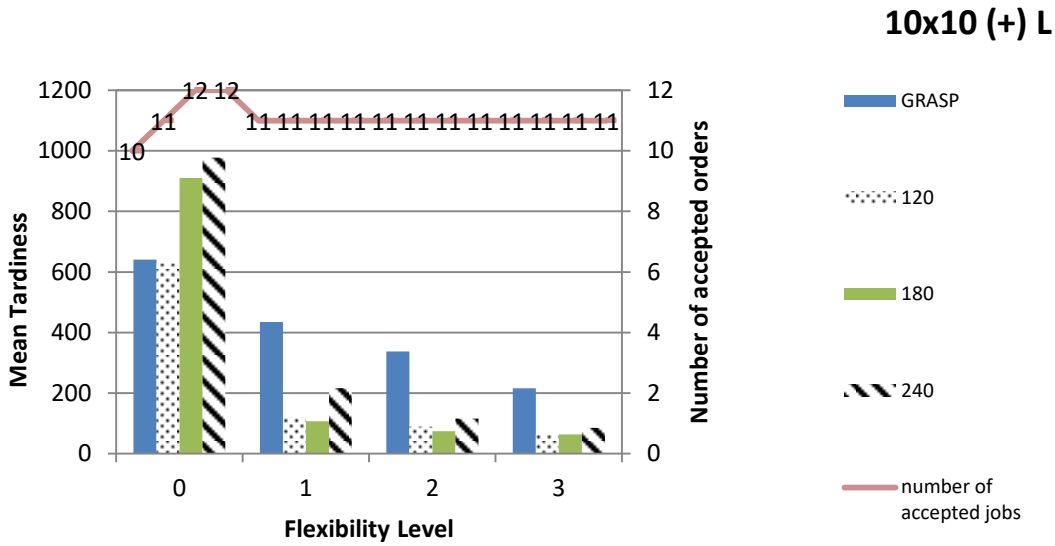
Figure 5.4 Comparison of the event-driven and the periodical rescheduling policies for 5x5 problems (continue)



****(-) low resource utilization, (+) high resource utilization, T-Tight due date, L-loose due date.*

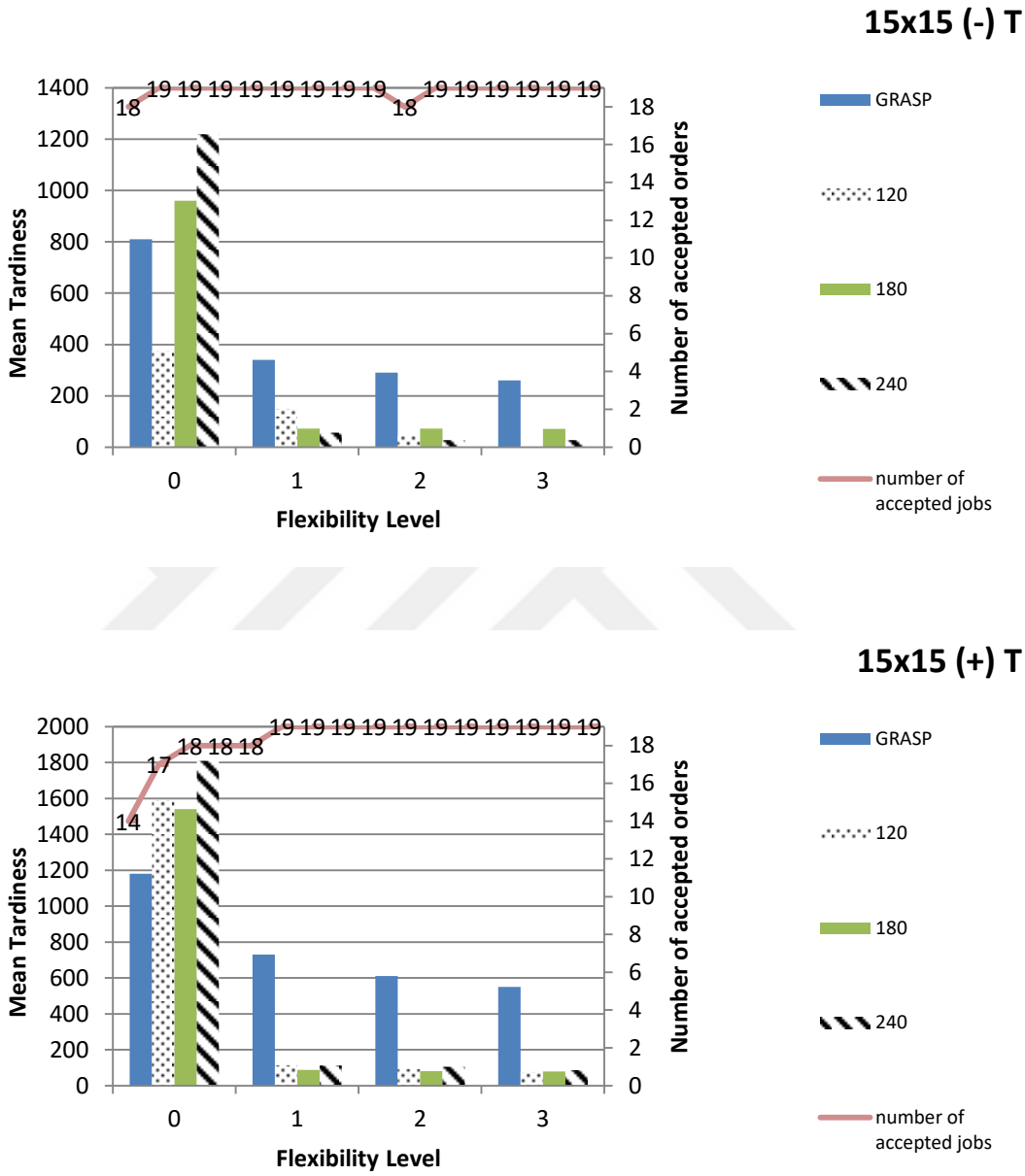
Figure 5.5 Comparison of the event-driven and the periodical rescheduling policies for 10x10 problems (continue)





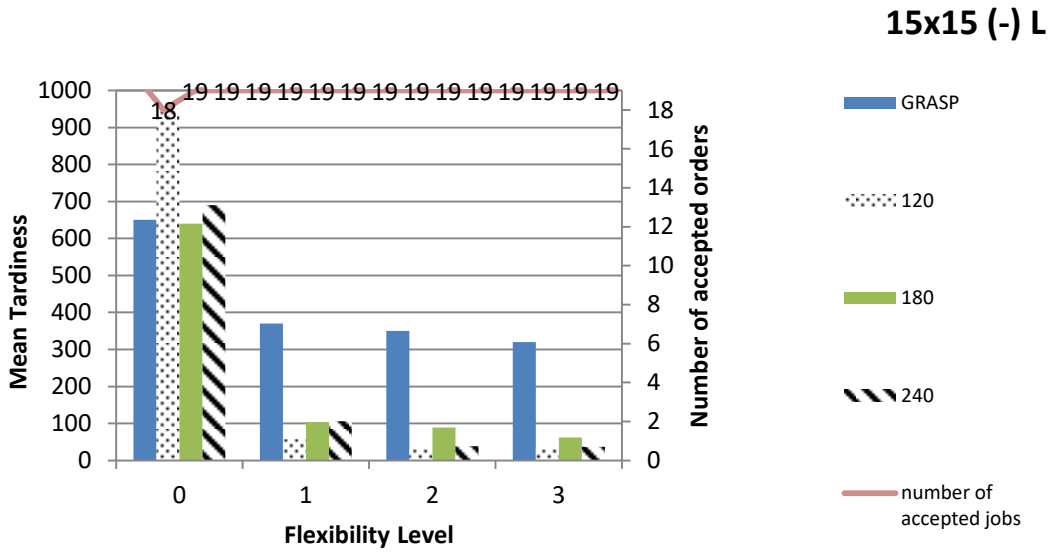
****(-) low resource utilization, (+) high resource utilization, T-Tight due date, L-loose due date.*

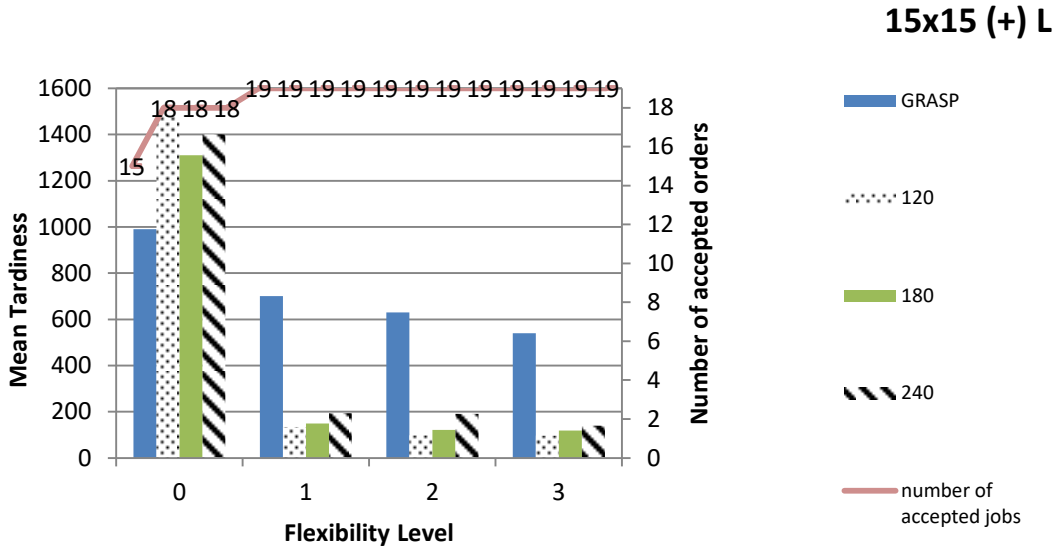
Figure 5.5 Comparison of the event-driven and the periodical rescheduling policies for 10x10 problems (continue)



****(-) low resource utilization, (+) high resource utilization, T-Tight due date, L-loose due date.*

Figure 5.6 Comparison of the event-driven and the periodical rescheduling policies for 15x15 problems





****(-) low resource utilization, (+) high resource utilization, T-Tight due date, L-loose due date.*

Figure 5.6 Comparison of the event-driven and the periodical rescheduling policies for 15x15 problems (continue).

5.3 Chapter Conclusion

In this chapter, a novel GRASP based metaheuristic algorithm is developed for modeling and solving FJSSP with dynamic events such as new order arrivals, machine breakdowns, arrival of urgent orders, order cancellations, and change of due dates under event-driven rescheduling policy. Moreover, machine capacity constraints, sequence dependent setup times, order acceptance/rejection decision and an ORR mechanism are also considered in order to better reflect the scheduling factors that are encountered in real manufacturing environments. A preemptive goal programming logic is used to assess the relative performance of the generated schedules. Extensive computational studies are conducted along with statistical performance analyses. Firstly, the performance of the proposed algorithm is evaluated for solving static benchmark problems. Satisfactory results are obtained. Then, the proposed algorithm is employed to solve the dynamic scheduling problems over varying degrees of machine flexibility, shop utilization and due date tightness with event-driven rescheduling policy. Results are compared with dispatching rules. Finally, event driven and periodical rescheduling policies which are utilized within the proposed algorithm are compared.

CHAPTER SIX
CONCLUSION

6.1 Summary of the Thesis

This thesis focuses on reflecting real-life like manufacturing environments to scheduling. The dynamic flexible job shop scheduling problem is handled by taking into consideration the sequence dependent setup times and machine capacity.

In Chapter 2, dynamic events (new order arrival, machine breakdowns order cancellation, due date change, rush order) are detailed. The rescheduling policies, order acceptance/rejection decision, ORR mechanism and objective function are explained. The literature review of the dynamic job shop scheduling and dynamic flexible job shop scheduling is detailed.

In Chapter 3, the proposed constructive and multi-start algorithm is developed. The aim is to construct a feasible, efficient and applicable schedules rapidly.

In Chapter 4, the proposed DJSSP is introduced and solved by the proposed constructive and multi-start algorithm.

In Chapter 5, the presented DFJSSP is conducted by the proposed constructive and multi-start algorithm. The effects of the flexibility, machine utilization and due date tightness are analyzed under event-driven and periodical rescheduling policies.

6.2 Contributions of the Thesis

Depending on today's competitive conditions in production systems, the importance of effective and precise scheduling is known in order to increase operational responsiveness. More complex problems have arisen with the widespread use of multipurpose automatic machines in production workshops. The flexible job shop scheduling problem is one of them.

The real manufaturing system is dynamic in nature. In order to be ahead in a competitive environment, an approach that effectively reflects this dynamism to the production system is aimed in this thesis. For this purpose, most of the dynamic events such as new order arrival, machine breakdown, rush order, order cancellation, due date change that took place in real production systems are concurrently involved in the flexible job shop scheduling system under event-driven rescheduling policy. Unlike classical scheduling problems, it is aimed to create more effective schedules by approaching real-life problems by taking into account sequence dependent setup times. Machine capacity has also been taken into account in the dynamic flexible job shop scheduling problem. Additionly, we defined and considered simultaneously order acceptance/rejection decisions and an ORR mechanism in a dynamic flexible job shop environment. Besides, in order to cope with the multi-objectivity, a preemptive goal programming approach is proposed. The proposed approach was not dealt in the literature before.

A constructive and multi-start algorithm is proposed. G&T algorithm is embedded into the proposed algorithm to construct feasible schedule at each time. The proposed algorithm is suitable for the time-varying problems. The advantages of the constructive and multi-start strategy are: The lost of diversity problem is overcome with the proposed algorithm. The proposed algorithm doesn't require any response mechanism to dynamic events. The length of the representation dimension is changeable and this makes possible for adapting the dynamic changes to the representation. The proposed algorithm can easily obtain fast and efficient solutions.

First of all, the proposed algorithm is implemented for job shop scheduling and flexible job shop scheduling benchmark problems to test the suitability of the proposed approach. Then, this is implemented to the generated DJSSP and DFJSSP. Finally, the effects of the due date, shop utilization and flexibility on the results are inspected. The developed method can be used for many scheduling applications.

6.3 Future Research Directions

These future researches can be classified as follows. According to the literature review that is given in Chapter 2, there is no paper, which handles the process plan flexibility under the proposed approach. This thesis can be extended by integrating the process plan flexibility.

In most of the real-life manufacturing environments, machines may be unavailable due to machine breakdowns (stochastic unavailability) or preventive maintenance (PM) activities (deterministic unavailability). Besides, production scheduling and maintenance activities are planned separately. However, there are a strict interdependent between production scheduling and maintenance scheduling. We aim to extend our study by integrating maintenance scheduling into production scheduling.

We also aim to provide a better capacity management strategy by making use of the revenue management principles, along with machine scheduling.

REFERENCES

Adibi, M. A., & Shahrabi, J. (2014). A clustering-based modified variable neighborhood search algorithm for a dynamic job shop scheduling problem. *The International Journal of Advanced Manufacturing Technology*, 70 (9-12), 1955-1961.

Aiex, R. M., Binato, S., & Resende, M. G. (2003). Parallel GRASP with path-relinking for job shop scheduling. *Parallel Computing*, *29*(4), 393-430.

Al-Hinai, N., & ElMekkawy, T. Y. (2011). Robust and stable flexible job shop scheduling with random machine breakdowns using a hybrid genetic algorithm. *International Journal of Production Economics*, 132 (2), 279-291.

Ali, A., Hackney, P., Bell, D., & Birkett, M. (2015). Genetic Algorithms for Solving Bicriteria Dynamic Job Shop Scheduling Problems with Alternative Routes. In *Proceedings of the The International Conference on Engineering & MIS 2015* (p. 31). ACM.

Allahverdi, A., & Soroush, H. M. (2008). The significance of reducing setup times/setup costs. *European Journal of Operational Research*, 187 (3), 978-984.

Argüello, M. F., Bard, J. F., & Yu, G. (1997). A GRASP for aircraft routing in response to groundings and delays. *Journal of Combinatorial Optimization*, 1 (3), 211-228.

Aydin, M. E., & Öztemel, E. (2000). Dynamic job-shop scheduling using reinforcement learning agents. *Robotics and Autonomous Systems*, 33 (2), 169-178.

Aytug, H., Lawley, M. A., McKay, K., Mohan, S., & Uzsoy, R. (2005). Executing production schedules in the face of uncertainties: A review and some future directions. *European Journal of Operational Research,* 161 (1), 86-110.

Bagheri, A., Zandieh, M., Mahdavi, I., & Yazdani, M. (2010). An artificial immune algorithm for the flexible job-shop scheduling problem. *Future Generation Computer Systems*, 26 (4), 533-541.

Baker, K. R. (1984). The effects of input control in a simple scheduling model. *Journal of Operations Management*, 4 (2), 99-112.

Bard, J. F., Venkatraman, K., & Feo, T. A. (1993). Single machine scheduling with flow time and earliness penalties. *Journal of Global Optimization*, 3 (3), 289-309.

Baykasoğlu, A. (2002). Linguistic-based meta-heuristic optimization model for flexible job shop scheduling. *International Journal of Production Research*, 40 (17), 4523-4543.

Baykasoğlu, A., Özbakır, L., & Sönmez, A. I. (2004). Using multiple objective tabu search and grammars to model and solve multi-objective flexible job shop scheduling problems. *Journal of Intelligent Manufacturing*, 15 (6), 777-785.

Baykasoğlu, A. (2005). Preemptive goal programming using simulated annealing. *Engineering Optimization*, 37 (1), 49-63.

Baykasoğlu, A., & Özbakır, L. (2010). Analyzing the effect of dispatching rules on the scheduling performance through grammar based flexible scheduling system. *International Journal of Production Economics*, 124 (2), 369-381.

Baykasoğlu, A., Hamzadayi, A., & Köse, S. Y. (2014). Testing the performance of teaching–learning based optimization (TLBO) algorithm on combinatorial

problems: Flow shop and job shop scheduling cases. *Information Sciences*, *276*, 204-218.

Baykasoglu, A., & Ozsoydan, F. B. (2015). A constructive search algorithm for combinatorial dynamic optimization problems. In *Evolving and Adaptive Intelligent Systems (EAIS), 2015 IEEE International Conference on* (pp. 1-7). IEEE.

Bergamaschi, D., Cigolini, R., Perona, M., & Portioli, A. (1997). Order review and release strategies ina job shop environment: A review and a classification. *International Journal of Production Research*, 35 (2), 399-420.

Bierwirth, C., & Mattfeld, D. C. (1999). Production scheduling and rescheduling with genetic algorithms. *Evolutionary Computation*, 7 (1), 1-17.

Binato, S., Hery, W. J., Loewenstern, D. M., & Resende, M. G. (2002). A GRASP for job shop scheduling. In *Essays and Surveys in Metaheuristics* (pp. 59-79). Springer US.

Blum, C., & Roli, A. (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys (CSUR)*, 35 (3), 268-308.

Bobrowski, P. M., & Park, P. S. (1989). Work release strategies in a dual resource constrained job shop. *Omega*, 17 (2), 177-188.

Brandimarte, P. (1993). Routing and scheduling in a flexible job shop by tabu search. *Annals of Operations Research*, 41 (3), 157-183.

Branke, J., & Mattfeld, D. C. (2005). Anticipation and flexibility in dynamic scheduling. *International Journal of Production Research*, 43 (15), 3103-3129.

Brucker, P., & Schlie, R. (1990). Job-shop scheduling with multi-purpose machines. *Computing*, 45 (4), 369-375.

Chryssolouris, G., & Subramaniam, V. (2001). Dynamic scheduling of manufacturing job shops using genetic algorithms. *Journal of Intelligent Manufacturing*, 12 (3), 281-293.

Corberán, A., Martı, R., & Sanchis, J. M. (2002). A GRASP heuristic for the mixed Chinese postman problem. *European Journal of Operational Research*, 142 (1), 70-80.

Cowling, P., & Johansson, M. (2002). Using real time information for effective dynamic scheduling. *European Journal of Operational Research*, 139 (2), 230-244.

Dauzère-Pérès, S., & Paulli, J. (1997). An integrated approach for modeling and solving the general multiprocessor job-shop scheduling problem using tabu search. *Annals of Operations Research*, 70, 281-306.

Derrac, J., García, S., Molina, D., & Herrera, F. (2011). A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation*, 1 (1), 3-18.

Dewan, P., & Joshi, S. (2002). Auction-based distributed scheduling in a dynamic job shop environment. *International Journal of Production Research*, 40 (5), 1173-1191.

Dominic, P. D., Kaliyamoorthy, S., & Kumar, M. S. (2004). Efficient dispatching rules for dynamic job shop scheduling. *The International Journal of Advanced Manufacturing Technology*, 24 (1-2), 70-75.

Duffuaa, S. O., Raouf, A., & Campbell, J. D. (2000). Planning and control of maintenance systems. *Willey & Sons*, 31-32, Switzerland: Springer.

Duguay, C. R., Landry, S., & Pasin, F. (1997). From mass production to flexible/agile production. *International Journal of Operations & Production Management*, 17 (12), 1183-1195.

Fang, J., & Xi, Y. (1997). A rolling horizon job shop rescheduling strategy in the dynamic environment. *The International Journal of Advanced Manufacturing Technology*, 13 (3), 227-232.

Fattahi, P., & Fallahi, A. (2010). Dynamic scheduling in flexible job shop systems by considering simultaneously efficiency and stability. *CIRP Journal of Manufacturing Science and Technology*, 2 (2), 114-123.

Feo, T. A., & Resende, M. G. (1989). A probabilistic heuristic for a computationally difficult set covering problem. *Operations Research Letters*, 8 (2), 67-71.

Feo, T. A., & Bard, J. F. (1989). Flight scheduling and maintenance base planning. *Management Science*, 35 (12), 1415-1432.

Feo, T. A., Resende, M. G., & Smith, S. H. (1994). A greedy randomized adaptive search procedure for maximum independent set. *Operations Research*, 42 (5), 860-878.

Feo, T. A., & Resende, M. G. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6 (2), 109-133.

Framinan, J. M., Leisten, R., & García, R. R. (2014). *Manufacturing scheduling systems*. Berlin: Springer.

Garey, M. R., Johnson, D. S., & Sethi, R. (1976). The complexity of flowshop and jobshop scheduling. *Mathematics of Operations Research*, 1 (2), 117-129.

Gao, J., Sun, L., & Gen, M. (2008). A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems. *Computers & Operations Research*, 35 (9), 2892-2907.

Gao, K. Z., Suganthan, P. N., Chua, T. J., Chong, C. S., Cai, T. X., & Pan, Q. K. (2015). A two-stage artificial bee colony algorithm scheduling flexible job-shop scheduling problem with new job insertion. *Expert Systems with Applications*, 42 (21), 7652-7663.

Gao, K. Z., Suganthan, P. N., Pan, Q. K., Tasgetiren, M. F., & Sadollah, A. (2016). Artificial bee colony algorithm for scheduling and rescheduling fuzzy flexible job shop problem with new job insertion. *Knowledge-Based Systems*, 109, 1-16.

Gao, K. Z., Suganthan, P. N., Pan, Q. K., Chua, T. J., Cai, T. X., & Chong, C. S. (2016). Discrete harmony search algorithm for flexible job shop scheduling problem with multiple objectives. *Journal of Intelligent Manufacturing*, 27 (2), 363-374.

Ge, H. W., Sun, L., Liang, Y. C., & Qian, F. (2008). An effective PSO and AIS-based hybrid intelligent algorithm for job-shop scheduling. *IEEE Transactions on systems, Man, and Cybernetics-Part A: Systems and Humans*, 38 (2), 358-368.

Gen, M., Gao, J., & Lin, L. (2009). Multistage-based genetic algorithm for flexible job-shop scheduling problem. In *Intelligent and Evolutionary Systems* (pp. 183-196). Berlin: Springer.

Geyik, F., & Dosdoğru, A. T. (2013). Process plan and part routing optimization in a dynamic flexible job shop scheduling environment: an optimization via simulation approach. *Neural Computing and Applications*, 23 (6), 1631-1641.

Gholami, M., & Zandieh, M. (2009). Integrating simulation and genetic algorithm to schedule a dynamic flexible job shop. *Journal of Intelligent Manufacturing*, 20 (4), 481-498.

Giffler, B., & Thompson, G. L. (1960). Algorithms for solving production-scheduling problems. *Operations Research*, 8 (4), 487-503.

Glover, F. W., & Kochenberger, G. A. (Eds.). (2006). *Handbook of metaheuristics* (Vol. 57). New York, Boston, Dordrecht, London, Moscow: Springer Science & Business Media.

González, M. A., Vela, C. R., & Varela, R. (2015). Scatter search with path relinking for the flexible job shop scheduling problem. *European Journal of Operational Research*, 245 (1), 35-45.

Gupta, S. R., & Smith, J. S. (2006). Algorithms for single machine total tardiness scheduling with sequence dependent setups. *European Journal of Operational Research*, 175 (2), 722-739.

He, W., & Sun, D. H. (2013). Scheduling flexible job shop problem subject to machine breakdown with route changing and right-shift strategies. *The International Journal of Advanced Manufacturing Technology*, 66 (1-4), 501-514.

Herroelen, W., & Leus, R. (2005). Project scheduling under uncertainty: Survey and research potentials. *European Journal of Operational Research*, 165 (2), 289-306.

Hendry, L. C., & Wong, S. K. (1994). Alternative order release mechanisms: a comparison by simulation. *The International Journal of Production Research*, 32 (12), 2827-2842.

Holthaus, O. (1999). Scheduling in job shops with machine breakdowns: an experimental study. *Computers & Industrial Engineering*, 36 (1), 137-162.

Huang, R. H. (2010). Multi-objective job-shop scheduling with lot-splitting production. *International Journal of Production Economics*, 124 (1), 206-213.

Hutchison, J. (1991). Current and future issues concerning FMS scheduling. *Omega*, 19 (6), 529-537.

Iman, R. L., & Davenport, J. M. (1980). Approximations of the critical region of the fbietkan statistic. *Communications in Statistics-Theory and Methods*, 9 (6), 571-595.

Jin, L., Zhang, C., Shao, X., & Yang, X. (2016). A study on the impact of periodic and event-driven rescheduling on a manufacturing system: An integrated process planning and scheduling case. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 1-15.

Kacem, I., Hammadi, S., & Borne, P. (2002). Approach by localization and multiobjective evolutionary optimization for flexible job-shop scheduling problems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 32 (1), 1-13.

Kang, K., feng Zhang, R., & qing Yang, Y. (2007). MAS equipped with ant colony applied into dynamic job shop scheduling. In *International Conference on Intelligent Computing* (pp. 823-835). Berlin : Springer.

Kundakcı, N., & Kulak, O. (2016). Hybrid genetic algorithms for minimizing makespan in dynamic job shop scheduling problem. *Computers & Industrial Engineering*, 96, 31-51.

Kutanoglu, E. (1999). An analysis of heuristics in a dynamic job shop with weighted tardiness objectives. *International Journal of Production Research*, 37 (1), 165-187.

Kutanoglu, E., & Sabuncuoglu, I. (2001). Experimental investigation of iterative simulation-based scheduling in a dynamic and stochastic job shop. *Journal of Manufacturing Systems*, 20 (4), 264-279.

Lee, Y. H., Jeong, C. S., & Moon, C. (2002). Advanced planning and scheduling with outsourcing in manufacturing supply chain. *Computers & Industrial Engineering*, 43 (1), 351-374.

Lin, Z. L., & Huang, Y. S. (2010). Nonperiodic preventive maintenance for repairable systems. *Naval Research Logistics*, 57 (7), 615-625.

Liu, S. Q., Ong, H. L., & Ng, K. M. (2005). Metaheuristics for minimizing the makespan of the dynamic shop scheduling problem. *Advances in Engineering Software*, 36 (3), 199-205.

Liu, B., Fan, Y., & Liu, Y. (2015). A fast estimation of distribution algorithm for dynamic fuzzy flexible job-shop scheduling problem. *Computers & Industrial Engineering*, 87, 193-201.

Lou, P., Liu, Q., Zhou, Z., Wang, H., & Sun, S. X. (2012). Multi-agent-based proactive–reactive scheduling for a job shop. *The International Journal of Advanced Manufacturing Technology*, 59 (1-4), 311-324.

Lu, M. S., & Romanowski, R. (2013). Multicontextual dispatching rules for job shops with dynamic job arrival. *The International Journal of Advanced Manufacturing Technology*, 67 (1-4), 19-33.

Maccarthy, B. L., & Liu, J. (1993). Addressing the gap in scheduling research: a review of optimization and heuristic methods in production scheduling. *The International Journal of Production Research*, 31 (1), 59-79.

Mehta, S. V., & Uzsoy, R. M. (1998). Predictable scheduling of a job shop subject to breakdowns. *IEEE Transactions on Robotics and Automation*, 14 (3), 365-378.

Mehta, S. V. (1999). Predictable scheduling of a single machine subject to breakdowns. *International Journal of Computer Integrated Manufacturing*, 12 (1), 15-38.

Morton, T., & Pentico, D. W. (1993). *Heuristic scheduling systems: with applications to production systems and project management* (Vol. 3). United States of America: John Wiley & Sons.

Nelson, R. T. & Holloway, C. A., 1977, Centralized scheduling and priority implementation heuristics for a dynamic job shop model with due dates and variable processing times. *AIIE Transactions*, 9 (1), 95–102.

Nie, L., Gao, L., Li, P., & Li, X. (2013). A GEP-based reactive scheduling policies constructing approach for dynamic flexible job shop scheduling problem with job release dates. *Journal of Intelligent Manufacturing*, 24 (4), 763-774.

Ning, T., Huang, M., Liang, X., & Jin, H. A novel dynamic scheduling strategy for solving flexible job-shop problems. *Journal of Ambient Intelligence and Humanized Computing*, 1-9.

Nobibon, F. T., Herbots, J., & Leus, R. (2009). Order acceptance and scheduling in a single-machine environment: exact and heuristic algorithms. *Available at:* http://ssrn.com/abstract=1472957 .

Qi, J. G., Burns, G. R., & Harrison, D. K. (2000). The application of parallel multipopulation genetic algorithms to dynamic job-shop scheduling. *The International Journal of Advanced Manufacturing Technology*, 16 (8), 609-615.

Qiu, X., & Lau, H. Y. (2013). An AIS-based hybrid algorithm with PDRs for multi-objective dynamic online job shop scheduling problem. *Applied Soft Computing*, 13 (3), 1340-1351.

Onur, L., & Fabrycky, W. J. (1987). An input/output control system for the dynamic job shop. *IIE Transactions*, 19 (1), 88-97.

Pardalos, L., & Resende, M. (1994). A greedy randomized adaptive search procedure for the quadratic assignment problem. *Quadratic Assignment and Related Problems, DIMACS Series on Discrete Mathematics and Theoretical Computer Science*, 16, 237-261.

Pezzella, F., Morganti, G., & Ciaschetti, G. (2008). A genetic algorithm for the flexible job-shop scheduling problem. *Computers & Operations Research*, 35 (10), 3202-3212.

Pinedo, M., & Chao, X. (1999). *Operations scheduling*. Singapore: McGraw Hill.

Pinedo, M. (2005). *Planning and scheduling in manufacturing and services* (Vol. 24). New York: Springer.

Prabhaharan, G., Khan, B. S. H., & Rakesh, L. (2006). Implementation of grasp in flow shop scheduling. *The International Journal of Advanced Manufacturing Technology*, 30 (11-12), 1126-1131.

Rajabinasab, A., & Mansour, S. (2011). Dynamic flexible job shop scheduling with alternative process plans: an agent-based approach. *The International Journal of Advanced Manufacturing Technology*, 54 (9-12), 1091-1107.

Rajkumar, M., Asokan, P., & Vamsikrishna, V. (2010). A GRASP algorithm for flexible job-shop scheduling with maintenance constraints. *International Journal of Production Research*, 48 (22), 6821-6836.

Rajkumar, M., Asokan, P., Anilkumar, N., & Page, T. (2011). A GRASP algorithm for flexible job-shop scheduling problem with limited resource constraints. *International Journal of Production Research*, 49 (8), 2409-2423.

Rangsaritratsamee, R., Ferrell, W. G., & Kurz, M. B. (2004). Dynamic rescheduling that simultaneously considers efficiency and stability. *Computers & Industrial Engineering*, 46 (1), 1-15.

Resende, M. G., & Ribeiro, C. C. (2003). A GRASP with path relinking for private virtual circuit routing. *Networks*, 41 (2), 104-114.

Robertson, A. J. (2001). A set of greedy randomized adaptive local search procedure (GRASP) implementations for the multidimensional assignment problem. *Computational Optimization and Applications*, 19 (2), 145-164.

Sabuncuoglu, I., & Karapınar, H. Y. (1999). Analysis of order review/release problems in production systems. *International Journal of Production Economics*, 62 (3), 259-279.

Sabuncuoglu, I., & Bayız, M. (2000). Analysis of reactive scheduling problems in a job shop environment. *European Journal of Operational Research*, 126 (3), 567-586.

Sabuncuoglu, I., & Kizilisik, O. B. (2003). Reactive scheduling in a dynamic and stochastic FMS environment. *International Journal of Production Research*, 41 (17), 4211-4231.

Saidi-Mehrabad, M., & Fattahi, P. (2007). Flexible job shop scheduling with tabu search algorithms. *The International Journal of Advanced Manufacturing Technology*, 32 (5-6), 563-570.

Sarker, R., Omar, M., Hasan, S. K., & Essam, D. (2013). Hybrid Evolutionary Algorithm for job scheduling under machine maintenance. *Applied Soft Computing*, 13 (3), 1440-1447.

Scholz-Reiter, B., Hildebrandt, T., & Tan, Y. (2013). Effective and efficient scheduling of dynamic job shops—Combining the shifting bottleneck procedure with variable neighbourhood search. *CIRP Annals-Manufacturing Technology*, 62 (1), 423-426.

Sharma, P., & Jain, A. (2015). Performance analysis of dispatching rules in a stochastic dynamic job shop manufacturing system with sequence-dependent setup times: Simulation approach. *CIRP Journal of Manufacturing Science and Technology*, 10, 110-119.

Shen, X. N., & Yao, X. (2015). Mathematical modeling and multi-objective evolutionary algorithms applied to dynamic flexible job shop scheduling problems. *Information Sciences*, 298, 198-224.

Shnits, B., Rubinovitz, J., & Sinreich, D. (2004). Multicriteria dynamic scheduling methodology for controlling a flexible manufacturing system. *International Journal of Production Research*, 42 (17), 3457-3472.

Shukla, C. S., & Chen, F. F. (1996). The state of the art in intelligent real-time FMS control: a comprehensive survey. *Journal of intelligent Manufacturing*, 7 (6), 441-455.

Singh, M. R., & Mahapatra, S. S. (2016). A quantum behaved particle swarm optimization for flexible job shop scheduling. *Computers & Industrial Engineering*, 93, 36-44.

Stoop, P. P., & Wiers, V. C. (1996). The complexity of scheduling in practice. *International Journal of Operations & Production Management*, 16 (10), 37-53.

Subramaniam, V., Lee, G. K., Ramesh, T., Hong, G. S., & Wong, Y. S. (2000). Machine selection rules in a dynamic job shop. *The International Journal of Advanced Manufacturing Technology*, 16 (12), 902-908.

Subramaniam, V., Raheja, A. S., & Rama Bhupal Reddy, K. (2005). Reactive repair tool for job shop schedules. *International Journal of Production Research*, 43 (1), 1-23.

Suresh, V., & Chaudhuri, D. (1993). Dynamic scheduling—a survey of research. *International Journal of Production Economics*, 32 (1), 53-63.

Suwa, H., & Sandoh, H. (2007). Capability of cumulative delay based reactive scheduling for job shops with machine breakdowns. *Computers & Industrial Engineering*, 53 (1), 63-78.

Suwa, H., & Sandoh, H. (2012). *Online scheduling in manufacturing: a cumulative delay approach*. London : Springer Science & Business Media.

Uzsoy, R., Lee, C. Y., & Martin-Vega, L. A. (1994). A review of production planning and scheduling models in the semiconductor industry part II: shop-floor control. *IIE Transactions*, 26 (5), 44-55.

Vieira, G. E., Herrmann, J. W., & Lin, E. (2000). Analytical models to predict the performance of a single-machine system under periodic and event-driven rescheduling strategies. *International Journal of Production Research*, 38 (8), 1899-1915.

Vieira, G. E., Herrmann, J. W., & Lin, E. (2003). Rescheduling manufacturing systems: a framework of strategies, policies, and methods. *Journal of Scheduling*, 6 (1), 39-62.

Vinod, V., & Sridharan, R. (2008). Dynamic job-shop scheduling with sequence-dependent setup times: simulation modeling and analysis. *The International Journal of Advanced Manufacturing Technology*, 36 (3-4), 355-372.

Vinod, V., & Sridharan, R. (2009). Simulation-based metamodels for scheduling a dynamic job shop with sequence-dependent setup times. *International Journal of Production Research*, 47 (6), 1425-1447.

Vinod, V., & Sridharan, R. (2011). Simulation modeling and analysis of due-date assignment methods and scheduling decision rules in a dynamic job shop production system. *International Journal of Production Economics*, 129 (1), 127-146.

Wang, Y. M., Yin, H. L., & Da Qin, K. (2013). A novel genetic algorithm for flexible job shop scheduling problems with machine disruptions. *The International Journal of Advanced Manufacturing Technology*, 68 (5-8), 1317-1326.

Watanabe, M., Ida, K., & Gen, M. (2005). A genetic algorithm with modified crossover operator and search area adaptation for the job-shop scheduling problem. *Computers & Industrial Engineering*, 48 (4), 743-752.

Wight, O. (1970). Input/output control a real handle on lead time. *Production and Inventory Management*, 11 (3), 9-31.

Xia, W., & Wu, Z. (2005). An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problems. *Computers & Industrial Engineering*, 48 (2), 409-425.

Xiang, W., & Lee, H. P. (2008). Ant colony intelligence in multi-agent dynamic manufacturing scheduling. *Engineering Applications of Artificial Intelligence*, 21 (1), 73-85.

Xiong, J., Xing, L. N., & Chen, Y. W. (2013). Robust scheduling for multi-objective flexible job-shop problems with random machine breakdowns. *International Journal of Production Economics*, 141 (1), 112-126.

Xiong, H., Fan, H., Jiang, G., & Li, G. (2017). A simulation-based study of dispatching rules in a dynamic job shop scheduling problem with batch release and extended technical precedence constraints. *European Journal of Operational Research*, 257 (1), 13-24.

Xu, Y., Wang, L., Wang, S. Y., & Liu, M. (2015). An effective teaching–learning-based optimization algorithm for the flexible job-shop scheduling problem with fuzzy processing time. *Neurocomputing*, 148, 260-268.

Yang, J. H., Sun, L., Lee, H. P., Qian, Y., & Liang, Y. C. (2008). Clonal selection based memetic algorithm for job shop scheduling problems. *Journal of Bionic Engineering*, 5 (2), 111-119.

Yazdani, M., Gholami, M., Zandieh, M., & Mousakhani, M. (2009). A simulated annealing algorithm for flexible job-shop scheduling problem. *Journal of Applied Sciences*, 9 (4), 662-670.

Yazdani, M., Amiri, M., & Zandieh, M. (2010). Flexible job-shop scheduling with parallel variable neighborhood search algorithm. *Expert Systems with Applications*, 37 (1), 678-687.

Yen, G. G., & Ivers, B. (2009). Job shop scheduling optimization through multiple independent particle swarms. *International Journal of Intelligent Computing and Cybernetics*, 2 (1), 5-33.

Yu, X., & Ram, B. (2006). Bio-inspired scheduling for dynamic job shops with flexible routing and sequence-dependent setups. *International Journal of Production Research*, 44 (22), 4793-4813.

Yuan, Y., & Xu, H. (2013). An integrated search heuristic for large-scale flexible job shop scheduling problems. *Computers & Operations Research*, 40 (12), 2864-2877.

Zambrano Rey, G., Bekrar, A., Prabhu, V., & Trentesaux, D. (2014). Coupling a genetic algorithm with the distributed arrival-time control for the JIT dynamic scheduling of flexible job-shops. *International Journal of Production Research*, 52 (12), 3688-3709.

Zandieh, M., & Adibi, M. A. (2010). Dynamic job shop scheduling using variable neighbourhood search. *International Journal of Production Research*, 48 (8), 2449-2458.

Zhang, G., Gao, L., & Shi, Y. (2011). An effective genetic algorithm for the flexible job-shop scheduling problem. *Expert Systems with Applications*, 38 (4), 3563-3573.

Zhang, L., Gao, L., & Li, X. (2013). A hybrid genetic algorithm and tabu search for a multi-objective dynamic job shop scheduling problem. *International Journal of Production Research*, 51 (12), 3516-3531.

Zhou, R., Nee, A. Y. C., & Lee, H. P. (2009). Performance of an ant colony optimisation algorithm in dynamic job shop scheduling problems. *International Journal of Production Research*, 47 (11), 2903-2920.

APPENDICES

A1 Data set of DJSSP Introduced in Chapter 4

Table A1.1 Data for 5x5 DJSSP (small scaled instances)

Jobs	Machine Sequences and Processing Times (min)					Dynamic events	Occurence Time (min)	Event Type[a]	Information about dynamic events
J1	M2(21)	M3(35)	M1(53)	M5(21)	M4(55)	1	50	1	J1- Order 6
J2	M5(55)	M3(31)	M1(12)	M2(42)	M4(39)	2	100	1	J3- Order 7
J3	M4(34)	M3(64)	M2(19)	M5(92)	M1(83)	3	120	2	M1-repair time 30 min
J4	M3(87)	M1(69)	M2(87)	M5(93)	M4(60)	4	240	2	M2-repair time 42min
J5	M3(98)	M4(44)	M5(49)	M1(96)	M2(19)	5	310	2	M3-repair time 51min
Machine Capacity(machine-min)			M1-450 M2-420 M3-450 M4-440 M5-450			6	210	2	M4-repair time 22 min
						7	300	2	M5-repair time 43 min
						8	350	3	Order 4
						9	100	4	Order 7
						10	120	5	Order 6-Due date 800min

[a] 1:New order arrival, 2: Machine breakdown, 3: Order cancellation, 4: Rush order, 5: Change of due date.

Table A1.2 Data for 10x10 DJSSP (medium scaled instances)

Jobs	Machine Sequences and Processing Times(min)										Dynamic events	Occurence Time (min)	Event Type[a]	Information about dynamic events
J1	M5(89)	M9(79)	M7(99)	M6(53)	M2(66)	M3(70)	M10(62)	M8(81)	M1(61)	M4(72)	1	100	1	J5-Order 11
J2	M6(74)	M4(89)	M7(84)	M5(91)	M3(91)	M9(65)	M1(91)	M2(64)	M8(79)	M10(71)	2	150	1	J8- Order 12
J3	M10(78)	M9(88)	M1(76)	M2(68)	M7(62)	M6(51)	M8(73)	M5(92)	M3(89)	M4(81)	3	200	1	J6- Order 13
J4	M8(58)	M3(81)	M2(94)	M5(61)	M4(87)	M7(62)	M6(63)	M1(70)	M10(84)	M9(75)	4	120	2	M1-repair time 30 min
J5	M4(59)	M5(57)	M10(60)	M9(81)	M1(80)	M3(89)	M7(90)	M10(64)	M4(61)	M1(67)	5	240	2	M2-repair time 42 min
J6	M2(91)	M5(56)	M6(85)	M7(83)	M9(68)	M3(59)	M8(56)	M10(60)	M4(77)	M1(78)	6	310	2	M3-repair time 51 min
J7	M8(80)	M2(84)	M5(62)	M4(86)	M1(76)	M9(70)	M3(99)	M6(54)	M7(97)	M10(90)	7	210	2	M4-repair time 22 min
J8	M5(95)	M7(53)	M4(67)	M3(78)	M2(69)	M6(92)	M8(89)	M10(97)	M9(96)	M1(93)	8	300	2	M5-repair time 43 min
J9	M1(70)	M7(57)	M4(93)	M8(54)	M10(61)	M3(53)	M5(100)	M6(85)	M9(86)	M2(62)	9	150	2	M6-repair time 42 min
J10	M4(81)	M1(59)	M2(61)	M9(57)	M8(65)	M10(77)	M6(70)	M5(53)	M3(68)	M7(52)	10	210	2	M7-repair time 36 min
Machine Capacity(machine-min)				M1-400 M2-420 M3-450 M4-440 M5-450 M6-460 M7-450 M8-440 M9-450 M10-420							11	420	2	M8-repair time 65 min
											12	320	2	M9-repair time 80 min
Order Arrival Time (job-min)				J1-0 J2-0 J3-0 J4-0 J5-0 J6-20 J7-40 J8-55 J9-63 J10-80							13	520	2	M10-repair time 45 min
											14	550	3	Order 11
											15	80	4	Order 10
											16	100	5	Order 8-Due date 2500 min

Table A1.3 Data for 15x15 DJSSP (large scaled instances)

Jobs	Machine Sequences and Processing Times(min)														
J1	M12(80)	M7(99)	M8(91)	M11(57)	M1(61)	M3(92)	M2(74)	M5(99)	M13(63)	M10(96)	M9(77)	M15(95)	M14(61)	M6(59)	M4(73)
J2	M5(80)	M10(77)	M4(75)	M8(67)	M14(98)	M3(66)	M15(52)	M2(89)	M9(74)	M11(96)	M12(84)	M7(99)	M13(52)	M1(68)	M6(87)
J3	M3(100)	M12(66)	M15(80)	M1(76)	M14(69)	M5(53)	M13(88)	M2(57)	M6(96)	M10(90)	M7(74)	M8(100)	M4(72)	M11(96)	M9(56)
J4	M3(57)	M9(73)	M1(51)	M11(97)	M13(52)	M14(58)	M12(75)	M15(56)	M2(82)	M5(76)	M10(99)	M7(64)	M4(88)	M6(88)	M8(66)
J5	M11(61)	M9(62)	M15(77)	M12(67)	M10(59)	M14(75)	M6(64)	M4(87)	M7(54)	M1(60)	M13(79)	M3(73)	M5(78)	M8(64)	M2(74)
J6	M8(53)	M13(95)	M14(97)	M2(58)	M5(76)	M3(87)	M1(58)	M15(65)	M11(70)	M9(68)	M6(72)	M12(62)	M4(80)	M7(90)	M10(63)
J7	M4(84)	M10(95)	M13(88)	M8(60)	M15(82)	M3(78)	M5(71)	M1(76)	M12(54)	M6(54)	M14(91)	M7(64)	M9(51)	M11(57)	M2(69)
J8	M7(79)	M12(100)	M15(57)	M1(51)	M6(54)	M4(73)	M10(68)	M8(55)	M2(51)	M13(55)	M9(84)	M3(90)	M5(85)	M14(63)	M11(95)
J9	M5(76	M3(62)	M6(86)	M11(76)	M13(94)	M2(71)	M14(92)	M7(53)	M12(61)	M1(78)	M9(96)	M15(69)	M4(51)	M10(66)	M8(57)
J10	M2(84)	M12(5)	M3(90)	M13(87)	M7(61)	M5(74)	M8(94)	M6(84)	M15(74)	M14(83)	M11(59)	M1(56)	M4(85)	M10(66)	M9(85)
J11	M15(68)	M14(67)	M13(89)	M7(75)	M3(96)	M12(81)	M1(67)	M2(89)	M10(74)	M6(81)	M8(81)	M4(83)	M9(96)	M5(64)	M11(64)
J12	M7(86)	M6(82)	M15(85)	M11(61)	M3(72)	M4(67)	M1(54)	M8(85)	M9(61)	M12(63)	M14(88)	M5(79)	M10(51)	M2(51)	M13(94)
J13	M9(95)	M12(95)	M5(96)	M7(90)	M8(97)	M14(63)	M3(90)	M2(81)	M1(57)	M10(51)	M6(92)	M13(70)	M11(86)	M15(63)	M4(56)
J14	M15(77)	M1(52)	M10(64)	M11(57)	M7(77)	M3(88)	M9(83)	M4(59)	M8(77)	M12(68)	M2(70)	M5(85)	M13(53)	M14(94)	M6(58)
J15	M12(52)	M6(51)	M10(64)	M4(86)	M11(80)	M3(77)	M9(59)	M1(98)	M14(63)	M7(94)	M8(51)	M5(89)	M15(89)	M2(51)	M13(99)
Machine Capacity (machine-min)			M1-400 M2-420 M3-450 M4-440 M5-450 M6-460 M7-450 M8-440 M9-450 M10-420 M11-450 M12-455 M13-470 M14-460 M15-470												
Order Arrival Time (job-min)			J1-0 J2-0 J3-0 J4-0 J5-0 J6-20 J7-40 J8-0 J9-23 J10-156 J11-165 J12-170 J13-182 J14-210 J15-250												

Dynamic Events	Occurence Time (min)	Event Types	Information about dynamic events
1	300	1	J4-Order 16
2	450	1	J5-Order 17
3	500	1	J6-Order 18
4	650	1	J7-Order 19
5	800	1	J12-Order 20
6	120	2	M1-repair time 30 min
7	240	2	M2-repair time 42 min
8	310	2	M3-repair time 51 min
9	210	2	M4-repair time 22 min
10	300	2	M5-repair time 43 min
11	150	2	M6-repair time 42 min
12	210	2	M7-repair time 36 min
13	420	2	M8-repair time 65 min
14	320	2	M9-repair time 80 min
15	520	2	M10-repair time 45 min
16	120	2	M11-repair time 52 min
17	210	2	M12-repair time 46 min
18	250	2	M13-repair time 35 min
19	400	2	M14-repair time 65 min
20	500	2	M15-repair time 53 min
21	750	3	Order 17
22	455	4	Order 12
23	520	5	Order 10- Due date 3900 min

A2 Data set of DFJSSP Introduced in Chapter 5

Table A2.1 Data for 5x5 DFJSSP (small scaled instances)

Jobs	Processing Times (min)					Dynamic events	Occurrence Time (min)	Event Type[a]	Information about dynamic events
J1	U(21,45)	U(35,41)	U(53,68)	U(21,45)	U(55,82)	1	50	1	J1- Order 6
J2	U(55,75)	U(31,55)	U(12,31)	U(42,60)	U(39,55)	2	100	1	J3- Order 7
J3	U(34,42)	U(64,78)	U(19,30)	U(92,100)	U(83,110)	3	120	2	M1-repair time 30 min
J4	U(87,100)	U(69,82)	U(87,95)	U(93,110)	U(60,75)	4	240	2	M2-repair time 42min
J5	U(98,115)	U(44,60)	U(49,66)	U(96,102)	U(19,25)	5	310	2	M3-repair time 51min
Machine Capacity(machine-min)			M1-450 M2-420 M3-450 M4-440 M5-450			6	210	2	M4-repair time 22 min
						7	300	2	M5-repair time 43 min
						8	350	3	Order 4
						9	100	4	Order 7
						10	120	5	Order 6-Due date 800min

[a] 1:New order arrival, 2: Machine breakdown, 3: Order cancellation, 4: Rush order, 5: Change of due date.

Table A2.2 Data for 10x10 DFJSSP (medium scaled instances)

Jobs	Processing Times(min)										Dynamic events	Occurrence Time (min)	Event Type[a]	Information about dynamic events
J1	U(89,114)	U(79,96)	U(99,107)	U(53,57)	U(66,75)	U(70,90)	U(62,71)	U(81,96)	U(61,88)	U(72,85)	1	100	1	J5-Order 11
J2	U(74,85)	U(89,102)	U(84,100)	U(91,108)	U(91,101)	U(65,75)	U(91,96)	U(64,66)	U(79,93)	U(71,84)	2	150	1	J8- Order 12
J3	U(7890)	U(88,92)	U(76,88)	U(68,74)	U(62,73)	U(51,74)	U(73,108)	U(92,118)	U(89,113)	U(81,96)	3	200	1	J6- Order 13
J4	U(58,69)	U(81,92)	U(94,102)	U(61,71)	U(87,94)	U(62,74)	U(63,68)	U(70,81)	U(84,97)	U(75,82)	4	120	2	M1-repair time 30 min
J5	U(59,64)	U(57,62)	U(60,70)	U(81,92)	U(80,92)	U(89,103)	U(90,105)	U(64,70)	U(61,73)	U(67,79)	5	240	2	M2-repair time 42 min
J6	U(91,98)	U(56,63)	U(85,97)	U(83,96)	U(68,78)	U(59,68)	U(56,67)	U(60,69)	U(77,87)	U(78,93)	6	310	2	M3-repair time 51 min
J7	U(80,86)	U(84,96)	U(62,73)	U(86,100)	U(76,85)	U(70,79)	U(99,109)	U(54,60)	U(97,108)	U(90,107)	7	210	2	M4-repair time 22 min
J8	U(95,99)	U(53,62)	U(67,74)	U(78,93)	U(69,81)	U(92,105)	U(89,104)	U(97,105)	U(96,110)	U(93,109)	8	300	2	M5-repair time 43 min
J9	U(70,74)	U(57,65)	U(93,109)	U(54,64)	U(62,70)	U(53,63)	U(100,114)	U(85,91)	U(86,90)	U(62,71)	9	150	2	M6-repair time 42 min
J10	U(81,86)	U(59,64)	U(61,66)	U(57,65)	U(65,69)	U(77,87)	U(70,78)	U(53,60)	U(68,79)	U(52,57)	10	210	2	M7-repair time 36 min
Machine Capacity(machine-min)				M1-400 M2-420 M3-450 M4-440 M5-450 M6-460 M7-450 M8-440 M9-450 M10-420							11	420	2	M8-repair time 65 min
											12	320	2	M9-repair time 80 min
Order Arrival Time (job-min)				J1-0 J2-0 J3-0 J4-0 J5-0 J6-20 J7-40 J8-55 J9-63 J10-80							13	520	2	M10-repair time 45 min
											14	550	3	Order 11
											15	80	4	Order 10
											16	100	5	Order 8-Due date 2500 min

Table A2.3 Data for 15x15 DJSSP (large scaled instances)

Jobs	Machine Sequences and Processing Times(min)														
J1	U(80,95)	U(99,114)	U(91,105)	U(57,67)	U(61,72)	U(92,107)	U(74,88)	U(99,117)	U(63,73)	U(96,110)	U(77,90)	U(95,113)	U(61,73)	U(59,68)	U(73,87)
J2	U(80,92)	U(77,90)	U(75,86)	U(67,77)	U(98,113)	U(66,75)	U(52,61)	U(89,106)	U(74,89)	U(96,113)	U(84,100)	U(99,117)	U(52,61)	U(68,80)	U(87,104)
J3	U(100,113)	U(66,79)	U(80,95)	U(76,91)	U(69,82)	U(53,62)	U(88,101)	U(57,67)	U(96,113)	U(90,106)	U(74,88)	U(100,117)	U(72,88)	U(96,114)	U(56,66)
J4	U(57,65)	U(73,87)	U(51,61)	U(97,114)	U(52,62)	U(58,63)	U(75,81)	U(56,61)	U(82,90)	U(76,82)	U(99,108)	U(64,69)	U(88,96)	U(88,94)	U(66,71)
J5	U(61,72)	U(62,73)	U(77,88)	U(67,78)	U(59,70)	U(75,82)	U(64,68)	U(87,94)	U(54,59)	U(60,64)	U(79,86)	U(73,80)	U(78,85)	U(64,68)	U(74,81)
J6	U(53,63)	U(95,112)	U(97,116)	U(58,68)	U(76,89)	U(87,94)	U(58,62)	U(65,70)	U(70,76)	U(68,74)	U(72,77)	U(62,67)	U(80,86)	U(90,96)	U(63,68)
J7	U(84,98)	U(95,110)	U(88,104)	U(60,71)	U(82,97)	U(78,85)	U(71,78)	U(76,83)	U(54,58)	U(54,58)	U(91,100)	U(64,70)	U(51,56)	U(57,62)	U(69,75)
J8	U(79,93)	U(100,119)	U(57,68)	U(51,60)	U(54,63)	U(73,79)	U(68,74)	U(55,60)	U(51,56)	U(55,60)	U(84,91)	U(90,98)	U(85,93)	U(63,68)	U(95,103)
J9	U(76,90)	U(62,74)	U(86,103)	U(76,90)	U(94,109)	U(71,77)	U(92,99)	U(53,57)	U(61,66)	U(78,84)	U(96,100)	U(69,75)	U(51,55)	U(66,72)	U(57,60)
J10	U(84,99)	U(55,63)	U(90,105)	U(87,104)	U(61,67)	U(74,81)	U(94,102)	U(84,92)	U(74,80)	U(83,90)	U(59,64)	U(56,61)	U(85,93)	U(66,72)	U(85,92)
J11	U(68,79)	U(67,79)	U(89,105)	U(75,87)	U(96,115)	U(81,88)	U(67,73)	U(89,97)	U(74,80)	U(81,87)	U(81,87)	U(83,91)	U(96,105)	U(64,69)	U(64,69)
J12	U(86,99)	U(82,95)	U(85,99)	U(61,72)	U(72,86)	U(67,80)	U(54,59)	U(85,92)	U(61,66)	U(63,68)	U(88,96)	U(79,85)	U(51,55)	U(51,55)	U(94,102)
J13	U(95,113)	U(95,112)	U(96,112)	U(90,102)	U(97,115)	U(63,75)	U(90,98)	U(81,86)	U(57,62)	U(51,55)	U(92100)	U(70,75)	U(86,93)	U(63,68)	U(56,60)
J14	U(77,88)	U(52,61)	U(64,75)	U(57,67)	U(77,90)	U(88,104)	U(83,90)	U(59,64)	U(77,84)	U(68,74)	U(70,76)	U(85,90)	U(53,58)	U(94,101)	U(58,63)
J15	U(52,62)	U(51,60)	U(64,75)	U(86,101)	U(80,94)	U(77,90)	U(59,64)	U(98,107)	U(63,69)	U(94,103)	U(51,55)	U(89,94)	U(89,97)	U(51,56)	U(99,108)

Machine Capacity (machine-min) M1-400 M2-420 M3-450 M4-440 M5-450 M6-460 M7-450 M8-440 M9-450 M10-420 M11-450 M12-455 M13-470 M14-460 M15-470

Order Arrival Time (job-min) J1-0 J2-0 J3-0 J4-0 J5-0 J6-20 J7-40 J8-0 J9-23 J10-156 J11-165 J12-170 J13-182 J14-210 J15-250

Dynamic Events	Occurence Time (min)	Event Types	Information about dynamic events
1	300	1	J4-Order 16
2	450	1	J5-Order 17
3	500	1	J6-Order 18
4	650	1	J7-Order 19
5	800	1	J12-Order 20
6	120	2	M1-repair time 30 min
7	240	2	M2-repair time 42 min
8	310	2	M3-repair time 51 min
9	210	2	M4-repair time 22 min
10	300	2	M5-repair time 43 min
11	150	2	M6-repair time 42 min
12	210	2	M7-repair time 36 min
13	420	2	M8-repair time 65 min
14	320	2	M9-repair time 80 min
15	520	2	M10-repair time 45 min
16	120	2	M11-repair time 52 min
17	210	2	M12-repair time 46 min
18	250	2	M13-repair time 35 min
19	400	2	M14-repair time 65 min
20	500	2	M15-repair time 53 min
21	750	3	Order 17
22	455	4	Order 12
23	520	5	Order 10- Due date 3900 min