

**ÇANKAYA UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
INDUSTRIAL ENGINEERING
MASTER THESIS**

**A NEW CUSTOMER ORDER SCHEDULING PROBLEM
ON A SINGLE-MACHINE WITH JOB SETUP TIMES**

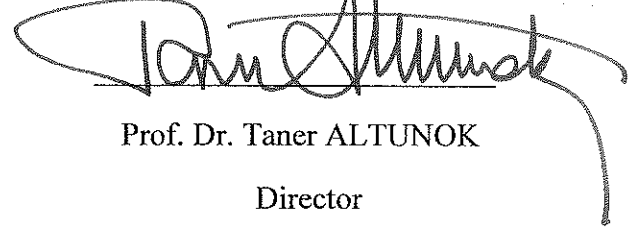
HALE AKKOCAOĞLU

FEBRUARY 2014

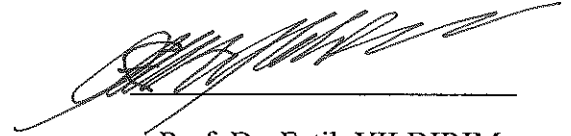
Title of Thesis: **A New Customer Order Scheduling Problem on a Single-machine with Job Setup Times**

Submitted by: **Hale AKKOCAOĞLU**


Approval of the Graduate School of Natural and Applied Sciences, Çankaya University

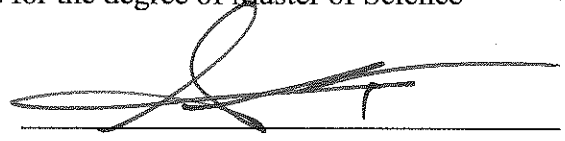

Prof. Dr. Taner ALTUNOK
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science


Prof. Dr. Fetih YILDIRIM
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science


Assist. Prof. Dr. Abdül Kadir GÖRÜR
Co-supervisor


Assoc. Prof. Dr. Ferda Can ÇETİNKAYA
Supervisor

Examination Date: 05.02.2014

Examining Committee Members

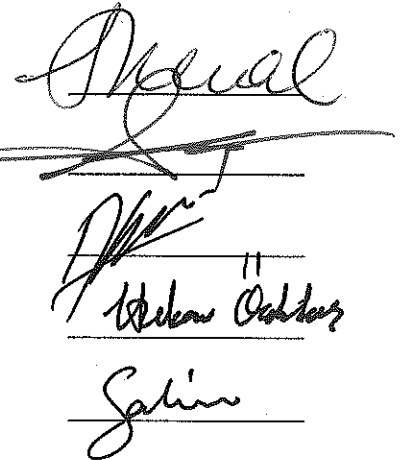
Assoc. Prof. Dr. Sedef MERAL (METU)

Assoc. Prof. Dr. Ferda Can ÇETİNKAYA (Çankaya Univ.)

Assist. Prof. Dr. Abdül Kadir GÖRÜR (Çankaya Univ.)

Assist. Prof. Dr. Hakan ÖZAKTAŞ (Çankaya Univ.)

Assist. Prof. Dr. Sakine BATUN (METU)


Sedef Meral
Ferda Can Çetinkaya
Abdul Kadir Görür
Hakan Ozaktas
Sakine Batun

STATEMENT OF NON-PLAGIARISM

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name : Hale AKKOCAOĞLU

Signature : 

Date : 05.02.2014

ABSTRACT

A NEW CUSTOMER ORDER SCHEDULING PROBLEM ON A SINGLE-MACHINE WITH JOB SETUP TIMES

AKKOCAOĞLU, Hale

M.Sc., Department of Industrial Engineering

Supervisor: Assoc. Prof. Dr. Ferda Can ÇETİNKAYA

Co-Supervisor: Assist. Prof. Dr. Abdül Kadir GÖRÜR

February 2014, 72 pages

In this study, we consider a relatively new class of the customer order scheduling (COS) problem where each order consists of one or more individual jobs. All jobs in the same customer order are processed successively and delivered at the same time to the customer. Thus, the completion time of the last job processed in each customer order defines the completion time of the order. A sequence independent setup is required before the processing of each job in a customer order. However, no setup is necessary before the processing of the first job of a customer order if this first job is the same as the last job of the immediately preceding customer order. We investigate the single-machine problem for two cases in which the makespan, which is the time to complete all customer orders, is minimized in the first case while the total completion time, which is the sum of the completion time of the orders, is minimized in the second case. For some special cases of both problems, we derive the properties of the optimal solution, which can be obtained by priority rules. We show that the makespan problem is polynomially solvable. For the total completion time problem, we develop a mixed integer programming model capable of solving small-sized

problem instances optimally and propose a constructive heuristic algorithm that obtains optimal and near-optimal solutions for medium and large sized problem instances. Computational experiments are done to evaluate the performance of our solution approaches in terms of both quality and time. The results show that the mixed integer linear programming model does not seem to be a useful alternative, especially for large-sized problem instances. However, the proposed heuristic algorithms find near-optimal solutions in very short time.

Keywords: Scheduling, Customer Order Scheduling, Group Scheduling, Makespan, Total Completion Time

ÖZ

HAZIRLIK SÜRESİ GEREKTİREN TEK BİR MAKİNEDEN YENİ BİR “MÜŞTERİ SİPARİŞLERİ ÇİZELGELEMESİ” PROBLEMİ

AKKOCAOĞLU, Hale

Yüksek Lisans, Endüstri Mühendisliği Anabilim Dalı

Tez Yöneticisi: Doç. Dr. Ferda Can ÇETİNKAYA

Ortak Tez Yöneticisi: Yrd. Doç. Dr. Abdül Kadir GÖRÜR

Şubat 2014, 72 sayfa

Bu çalışmada, müşteri sipariş problemlerine yakınlık gösteren yeni bir problem ele alınmıştır. Her müşteri siparişi bir ya da birden fazla işten oluşmaktadır. Bir siparişteki bütün işler ardışık işlenmeli ve müşteriye birlikte gönderilmelidir. Bu sebeple herhangi bir siparişteki son işin tamamlanma süresi, aynı zamanda o siparişin tamamlanma süresine eşittir. Bir müşteri siparişindeki her bir işin işlenmesinden önce iş sırasından bağımsız bir hazırlık zamanı gerekmektedir. Fakat eğer birbirini takip eden iki müşteri siparişindeki son ve ilk işler aynıysa, sıralamada sonra gelen müşteri siparişinin ilk işinden önce hazırlık gerekmez. Bu çalışmada, tek makineli problem iki farklı durum için irdelenmiştir. Birinci durumda amaç, tüm siparişlerin bitirilme süresinin en küçüklenmesi; ikinci durumda ise amaç, siparişlerin tamamlanma süreleri toplamının en küçüklenmesidir. Her iki problem için bazı özel durumlar dikkate alınmış ve bu durumlarda en iyi çözümün öncelik kurallarıyla elde edilmesini sağlayan özellikleri belirlenmiştir. Tüm siparişlerin bitirilme süresinin en küçüklenmesi probleminin polinom zamanda çözülebildiği gösterilmiştir. Siparişlerin tamamlanma süreleri toplamının en küçüklenmesi problemi için de küçük ölçekli

problemleri çözebilen matematiksel model geliştirilmiş ve orta ve büyük ölçekli problemlerin eniyi ya da eniyiye yakın çözülebilmesi için sezgisel algoritmalar geliştirilmiştir. Geliştirilen bu algoritmaların performansını çözüm kalitesi ve süresi açısından incelemek için sayısal deney setleri tasarlanmıştır. Matematiksel model çok uzun çözüm sürelerine ihtiyaç duyarken algoritmalar saniyeler içinde iyi sonuçlar verdiği için zaman karşılaştırması anlamlı bulunmamıştır. Bu nedenle algoritmaların, özellikle büyük ölçekli problemler için, matematiksel modele göre daha iyi performans sergiledikleri gözlemlenmiştir.

Anahtar Kelimeler: Çizelgeleme, Müşteri Sipariş Çizelgelemesi, Grup Çizelgelemesi, Tüm Siparişlerin Bitirilme Süresi, Siparişlerin Tamamlanma Süreleri Toplamı

ACKNOWLEDGEMENTS

First, I would like to thank my advisors Assoc. Prof. Dr. Ferda Can ÇETİNKAYA and Assist. Prof. Dr. Abdül Kadir GÖRÜR, for their endless support, motivating guidance and especially their trust and patience during this study.

I would also like to thank my examining committee members, Assoc. Prof. Dr. Sedef MERAL, Assist. Prof. Dr. Hakan ÖZAKTAŞ and Assist. Prof. Dr. Sakine BATUN for their precious opinions and contributions to this study.

I am grateful to my parents Handan AKKOCAOĞLU and Aslan AKKOCAOĞLU for their support in my whole life. I am especially grateful to my sister Hande AKKOCAOĞLU for being my first teacher and advisor.

I would like to thank my colleagues at the IE department for sharing my stress and supporting during the all stages of my thesis.

I would also like to thank all my professors at the IE department for their motivation and tolerance.

Finally, I am grateful to all my friends for their faithfulness, technical and morale support whenever I need.

TABLE OF CONTENTS

STATEMENT OF NON-PLAGIARISM	Hata! Yer işareti tanımlanmamış.
ABSTRACT.....	iv
ÖZ	vi
ACKNOWLEDGEMENTS	viii
TABLE OF CONTENTS	ix
LIST OF TABLES	xi
LIST OF FIGURES	xiii
CHAPTERS:	
1. INTRODUCTION	1
2. PROBLEM DEFINITION AND PRELIMINARY RESULTS	4
2.1 Problem Statement	4
2.2 Some Observations.....	6
2.3 Some Structural Properties of the Optimal Schedules	6
3. LITERATURE REVIEW	11
3.1 Group Scheduling.....	12
3.2 Customer Order Scheduling	13
4. MAKESPAN MINIMIZATION PROBLEM	16
4.1 Shortest Path Network.....	16
4.2 Some Special Cases Solvable by Priority Rules	19
5. TOTAL COMPLETION TIME MINIMIZATION PROBLEM.....	22
5.1 Mathematical Model.....	22
5.2 Lower and Upper Bounds on the Total Completion Time	25

5.3	Some Special Cases Solvable by Priority Rules	26
5.4	Heuristic Algorithms	29
6.	COMPUTATIONAL EXPERIMENTS	45
6.1	Computational Setting for Test Problems	45
6.2	Performance Measures	46
6.3	Discussion of the Results	48
6.3.1	Performance of the MILP Model.....	48
6.3.2	Performance of the Heuristic Algorithms.....	50
7.	CONCLUSIONS AND FUTURE RESEARCH DIRECTIONS	60
	REFERENCES.....	62
	APPENDICES:	
A.	COMPARISON TABLES	65
B.	AVERAGE PERCENT DEVIATIONS FROM THE OPTIMAL WHEN $K=15$	67
C.	AVERAGE PERCENT DEVIATIONS FROM THE OPTIMAL WHEN $K=20$	68
D.	AVERAGE PERCENT DEVIATIONS CALCULATED FOR $K=5, 10, 15, 20$.69
E.	CURRICULUM VITAE.....	72

LIST OF TABLES

Table 2.1 Setup and Processing times of jobs in Example 1.....	5
Table 4.1 Flow costs between intermediate nodes of the network for the problem considered in Example 1 when k equals 1 or 2.....	18
Table 5.1 Jobs and initial total times of the customer orders in Example 2.....	33
Table 5.2 Setup and processing times for all jobs in Example 2.....	33
Table 5.3 Sequence $\sigma_{STT} = \{O_1, O_5, O_4, O_3\}$ obtained for the set $O'' \cup O'_R$ in Step 1	34
Table 5.4 Improved sequence $\{O_1, O_5, O_4, O_3\}$ obtained for the set $O'' \cup O'_R$ in Step 2	35
Table 5.5 Improved sequence $\{O_1, O_4, O_3, O_5\}$ obtained for the set $O'' \cup O'_R$ in Step 3	36
Table 5.6 Final sequence σ_F of all customer orders and jobs in Example 2	36
Table 5.7 Sequence $\{O_1, O_4, O_3, O_5\}$ obtained for the set $O'' \cup O'_R$ in Step 2.....	39
Table 5.8 Sequence $\{O_5, O_1, O_4, O_3\}$ obtained for the set $O'' \cup O'_R$ in Step 3.....	41
Table 5.9 Final sequence σ_F of all customer orders and jobs in Example 4	42
Table 6.1 Performance of the MILP model.....	48
Table 6.2 Number of optimally solved problems by the MILP model	49
Table 6.3 Number of optimally solved problems by the MILP model and heuristic algorithms	50
Table 6.4 Average percent deviation of Algorithm 4 from the optimal solution for VARIABLE case	51
Table 6.5 Average Percent deviation of Algorithm 4 from optimal solutions for CONSTANT case	53

Table 6.6 Average percent deviation of Algorithm 4 from the best integer solutions for CONSTANT case	53
Table 6.7 Average percent deviation of Algorithms 1, 2 and 3 for VARIABLE case	58
Table 6.8 Average percent deviation of Algorithms 1, 2 and 3 for CONSTANT case	58

LIST OF FIGURES

Figure 2.1 A feasible schedule of customer orders and jobs within the orders.....	5
Figure 4.1 Shortest path network for the problem considered in Example 1.....	17
Figure 5.1 Flowchart Diagram for the Heuristic Algorithms 1, 2 and 3.....	43
Figure 6.1 Difference between the average percent deviations	52
Figure 6.2 Percent deviation of Algorithm 4 with respect to $K = 20$	54
Figure 6.3 Percent deviations from the optimal for $K = 5$	55
Figure 6.4 Percent deviations from the optimal for $K = 10$	56
Figure 6.5 General average percent deviations for $K = 5, K = 10, K = 15$ and $K = 20$	57
Figure 6.6 General average percent deviations of each algorithm for $K = 5, K = 10,$ $K = 15$ and $K = 20$	59

CHAPTER 1

INTRODUCTION

Scheduling is the allocation of resources to complete a given set of tasks over time. In production systems, resources and tasks are usually referred to as *machines* and *jobs*, respectively. Scheduling is an important issue, because determining the sequence of jobs on a machine is affected by a lot of factors such as processing times, setup times, due dates, precedence relations among jobs, etc. Generally, these factors cannot be handled without a systematic approach. Researchers investigate scheduling problems to satisfy the need of a systematic approach since 1950s. Scheduling problems vary with the concern of increasing efficiency in different manufacturing and service systems. This concern also leads to an increase in studies with setup time considerations. Setup activity may include obtaining tools, positioning, work-in-process material, returning tools, cleaning up, setting the required jigs and fixtures, adjusting tools and inspecting material (Allahverdi and Soroush, 2008). A setup can be a *sequence independent* activity depending only on the job to be processed, or it can be a *sequence dependent* one depending on both the job to be processed and the immediately preceding job. Setup times are ignored or considered as a part of process times in some studies. However, setup times are treated separately from processing times because in this case operations can be performed simultaneously which lead to an increase in resource utilization. Therefore, reducing the effect of setup times or costs has been investigated by many researches with different objectives and various shop structures. Some of the researches take into account only one objective or shop structure, while some take into account more than one objectives or shop structures. Setup time is one of the most important factors that affect the efficiency and utilization of resources. Hence,

most of the studies in the literature have focused on reducing the impact of setup time.

Customer order scheduling (COS) problem, is a kind of scheduling problem which considers order-based production environments. COS problems are widely studied problems in which there are K customer orders, each of which consists of a different number of jobs from N different job families. The setup time required depends on the family to be processed immediately after it. Therefore, grouping jobs in each customer order which are from the same family and processing these jobs successively have an advantage to reduce the setup time. The COS and the group (batch) scheduling (GS) problems are two closely related problems. Within the context of GS problems, a *customer order* and a *product ordered by a customer* may correspond to a *group* (string of jobs) and a *job in the group*, respectively.

In this study, we address a relatively new variant of the customer order scheduling problem, which differs from the other studies in the literature, because all jobs in each customer order are processed together (group technology assumption) and delivered at the same time to the customer (i.e., the completion time of the last job processed in each order defines the completion time of the order), and no setup time is necessary before the processing of the first job of a customer order if this first job is same as the last job of immediately preceding customer order. We investigate the problem for two scheduling criteria. Makespan (the time to complete all customer orders) is minimized in the first case, while the total completion time (the sum of the completion times of the customer orders) is minimized in the second case. The first case, which is equivalent to minimizing total required setup time, is focused on improving resource utilization and productivity whereas the second one, which is equivalent to minimizing total work-in-process inventory, is focused on increasing customer satisfaction. We show that the makespan can be solved in polynomial time. For the total completion time problem, we develop a mathematical model that obtains optimal solutions for small-sized problem instances and constructive heuristic algorithms that obtain near-optimal solutions for medium and large-sized instances, respectively.

The motivation for this type of customer order scheduling problem comes from several environments such as metal processing, food processing, and data processing.

An application can be found in any kind of manufacturing system, especially in manufacturing cells, in which there is a multi-purpose machine or a Computer Numerical Control (CNC) machine capable of processing several jobs. Suppose that the machine has a turret holding various machine tools. Each job uses some tools in performing an operation (e.g., drilling, turning, punching, etc.). Each tool change takes a different time in general to setup the machine for processing a job. If two consecutive jobs scheduled on the machine are the same, then there is no need to make a tool change and the setup time before the next job is zero or negligibly small. Another application can be observed in a printing company that receives orders from several customers with each order consisting of several printing jobs.

To the best of our knowledge, there is no study that deals with this variant of the COS problem. We wish that our study will provide a new starting point for future research in customer order scheduling since the setup time consideration of ours is different from the studies in the literature.

The remainder of this study is organized as follows. In Chapter 2, we define the problems under consideration, and then investigate some structural properties of the optimal schedules for both makespan and total completion time minimization problems. Chapter 3 presents a literature review on customer order scheduling and group scheduling problems. Chapter 4 is devoted to the makespan minimization problem, and proposes a network flow model that can be solved in polynomial time. Chapter 5 investigates the total completion time minimization problem, and proposes a mixed-integer linear programming model and four heuristic algorithms. The computational tests to evaluate the performance of the mathematical model and the proposed heuristic algorithms are presented in Chapter 6. Finally, our main findings and several directions for future research are discussed in Chapter 7.

CHAPTER 2

PROBLEM DEFINITION AND PRELIMINARY RESULTS

In this chapter, we first define our problem under consideration and then investigate some structural properties of the optimal schedules for both makespan and total completion time minimization problems.

2.1 Problem Statement

There is a set $O = \{O_1, O_2, \dots, O_K\}$ of K customer orders to be processed on a single machine, which is ready at time zero. Each order consists of one or more jobs from a set $J = \{J_1, J_2, \dots, J_N\}$ of N jobs ready for processing at time zero, and each job requires a sequence independent setup before processing. In contrast with existing studies on customer order scheduling, this study regards *group technology assumption*. In other words, all jobs in a customer order should be processed successively rather than scheduling the same jobs from different customer orders together. All jobs in a customer order are delivered at the same time to the customer and hence the completion time of the last job in a customer order is also the completion time of that order. Each job J_j has a constant processing time p_j and a constant setup time s_j . The setup time required before processing the job is inevitable inside a customer order, however no setup is required before the first job of a customer order if this job is the same as the last job of the immediately preceding customer order. Moreover, only one job can be processed on a machine at a time and preemption is not allowed, i.e., the processing or setup of any job cannot be interrupted at any time and resumed at a later time. In our study, we assume that each customer order consists of several different jobs but each job occurs only once in an order, i.e., multiple identical jobs do not exist in a customer order.

Before we proceed with our analysis, it seems appropriate to illustrate the problem by the following numerical example.

Example 1 Consider a simple instance of the problem in which there are three customer orders. Order 1 has jobs 1 and 3, Order 2 has jobs 1, 2 and 4, and Order 3 has only job 3. Setup and processing times of all jobs are given in the following table.

Table 2.1 Setup and Processing times of jobs in Example 1

Job	1	2	3	4
Setup time	3	2	4	1
Processing time	3	4	7	5

A feasible schedule of these three orders and four types of jobs is $O_2(J_4 - J_2 - J_1) - O_1(J_1 - J_3) - O_3(J_3)$, and the makespan value is 39 time units, as shown in Figure 2.1, and the total completion time of the customer orders is $18 + 32 + 39 = 60$. As it is illustrated in Figure 2.1, there is no need for a setup for job 1 in Order 1 since the last job of the previous order (Order 2) is the same as the first job of Order 1. Similarly, there is no need for a setup for job 3 in Order 3, since the last job of the previous order (Order 1) is job 3.

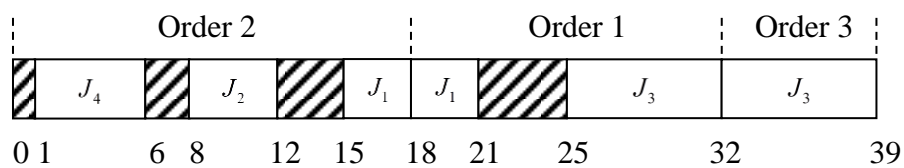


Figure 2.1 A feasible schedule of customer orders and jobs within the orders

2.2 Some Observations

Let us now give some preliminary definitions before discussing our some observations.

Definition 1 *Total time* (TT_i) of a customer order O_i is the sum of setup and processing times of all jobs in this customer order, i.e. $TT_i = \sum_{J_j \in O_i} (s_j + p_j)$.

Definition 2 The *shortest total time* (STT) sequence is a sequence in which the customer orders are sequenced in non-decreasing order of their total time.

When the setup times are omitted, we observe that the problem reduces to the scheduling of K customer orders (groups) with several common and uncommon jobs in each group. In this reduced problem, the makespan minimization problem becomes trivial since any sequence of customer orders gives the same objective value. On the other hand, processing the customer orders in *STT* sequence minimizes the total completion time of the customer orders, where the total time of a customer order is only the sum of the processing times of the jobs in this customer order since the setup times are omitted. However, the structure of the problem changes dramatically when the setup times are introduced. Depending on the composition of the customer orders, the makespan minimization problem is not straightforward as in the case of no-setup times, and the shortest total time rule may or may not perform well in the total completion time minimization problem.

2.3 Some Structural Properties of the Optimal Schedules

We now give some structural properties of the optimal schedules, which will be used in the development of the solution procedures for both makespan and total completion time minimization problems.

Property 1 *For both makespan and total completion time minimization problems, there exists an optimal schedule without inserted idle time.*

Proof If an idle time exists on the machine, then subsequent customer orders along with their jobs may be shifted left on the machine without increasing the objective of the current schedule. \square

The set of customer orders O can be divided into two disjoint sets O' and O'' ($O = O' \cup O''$ and $O' \cap O'' = \emptyset$) where the set O' is composed of customer orders having no common job with other customer orders, and its complement set O'' is composed of remaining customer orders.

Property 2 *There exists at least one optimal schedule of the makespan minimization problem in which the customer orders in the set O' are not intermingled with the customer orders in the set O'' . That is, the customer orders in the set O' are processed consecutively in any order before or after the optimal sequence of the customer orders in the set O'' .*

Proof Suppose that O_i is a customer order in the set O' . Furthermore, suppose that O_k and O_l are two customer orders in the set O'' . It is obvious that processing the customer order O_i between O_k and O_l may prevent the earlier completion of the customer order O_l if the customer orders O_k and O_l have a common job that may reduce the makespan. Thus, the customer order O_i in the set O' should not be intermingled with the customer orders in the set O'' . \square

Definition 3 Let

$$O'_B = \left\{ O_i \mid TT_i \leq \min_{O_l \in O''} \left\{ TT_l - \max_{J_j \in O_l} \{s_j\} \right\} \right\},$$

$$O'_R = \left\{ O_i \mid \min_{O_l \in O''} \left\{ TT_l - \max_{J_j \in O_l} \{s_j\} \right\} < TT_i < \max_{O_l \in O''} \{TT_l\} \right\}, \text{ and}$$

$$O'_A = \left\{ O_i \mid \max_{O_l \in O''} \{TT_l\} \leq TT_i \right\}$$

be three disjoint subsets of the set O' which is composed of customer orders having no common job with other customer orders.

Property 3 For the total completion time minimization problem, there is an optimal schedule with the following properties:

- (a) The customer orders of the subset O'_B precede all other customer orders, and the customer orders of the subset O'_A succeed all other customer orders.
- (b) The customer orders in the subsets O'_B and O'_A are scheduled in STT sequence.

That is,

$$\sigma_{STT}(O'_B) \rightarrow \{\text{Optimal schedule of } O'' \cup O'_R\} \rightarrow \sigma_{STT}(O'_A).$$

Proof Without loss of generality, we assume $\sigma = \{O_1, O_2, \dots, O_{i-1}, O_i, O_{i+1}, \dots, O_n\}$ is any sequence of all customer orders. Suppose that a customer order O_i is the first customer order of type O'_B in the sequence σ . That is, all the customer orders preceding the customer order O_i are the members of the other sets O'' , O'_R and O'_A . Let C_i and $TC(\sigma)$ be the completion time of the customer order i in the sequence σ , and the total completion time of the sequence σ , respectively. Then, we have

$$\begin{aligned} TC(\sigma) &= C_1 + C_2 + \dots + C_{i-1} + C_i + C_{i+1} + \dots + C_n \\ &= C_1 + C_2 + \dots + C_{i-1} + (C_{i-1} + TT_i) + C_{i+1} + \dots + C_n \\ &> C_1 + C_2 + \dots + C_{i-1} + (i-1)TT_i + TT_i + C_{i+1} + \dots + C_n \\ &> C_1 + C_2 + \dots + C_{i-1} + i \times TT_i + C_{i+1} + \dots + C_n, \end{aligned}$$

by Definition 3. Moving the job O_i to the beginning of all the customer orders in the sets O'' , O'_R and O'_A currently preceding the job O_i in the sequence σ and shifting backward all these customer orders in the sets O'' , O'_R and O'_A currently preceding the job O_i yields a new sequence $\sigma' = \{O_i, O_1, O_2, \dots, O_{i-1}, O_{i+1}, \dots, O_n\}$. Let $TC(\sigma')$ be the total completion time of the sequence σ' . Then, we have

$$\begin{aligned}
TC(\sigma') &= TT_i + (C_1 + TT_i) + (C_2 + TT_i) + \dots + (C_{i-1} + TT_i) + C_{i+1} + \dots + C_n \\
&= C_1 + C_2 + \dots + C_{i-1} + (i-1)TT_i + TT_i + C_{i+1} + \dots + C_n \\
&= C_1 + C_2 + \dots + C_{i-1} + i \times TT_i + C_{i+1} + \dots + C_n.
\end{aligned}$$

Thus, it is clear that the total completion time of the current sequence σ is improved by this change since $TC(\sigma) > TC(\sigma')$. Repetition of this argument for all remaining customer orders in the subset O'_B shows that customer orders of the subset O'_B precede all other customer orders.

Suppose that a customer order O_i is the last customer order of type O'_A in the sequence σ . That is, all the customer orders succeeding the customer order O_i are the members of the other sets O'_B , O'' and O'_R . Then, we have

$$\begin{aligned}
TC(\sigma) &= C_1 + C_2 + \dots + C_{i-1} + C_i + C_{i+1} + \dots + C_n \\
&= C_1 + C_2 + \dots + C_{i-1} + (C_{i-1} + TT_i) + C_{i+1} + \dots + C_n \\
&> C_1 + C_2 + \dots + C_{i-1} + (i-1)TT_i + TT_i + C_{i+1} + \dots + C_n \\
&> C_1 + C_2 + \dots + C_{i-1} + i \times TT_i + C_{i+1} + \dots + C_n,
\end{aligned}$$

by Definition 3. Moving the job O_i to the end of all the customer orders in the sets O'_B , O'' and O'_R currently succeeding the job O_i in the sequence σ and shifting forward all these customer orders in the sets O'_B , O'' and O'_R currently succeeding the job O_i yields a new sequence $\sigma' = \{O_1, O_2, \dots, O_{i-1}, O_{i+1}, \dots, O_n, O_i\}$. Then, we have

$$\begin{aligned}
TC(\sigma') &= C_1 + C_2 + \dots + C_{i-1} + (C_{i+1} - TT_i) + \dots + (C_n - TT_i) + C_n \\
&= C_1 + C_2 + \dots + C_{i-1} + C_{i+1} + \dots + C_n - (n-i)TT_i + C_n \\
&= C_1 + C_2 + \dots + C_{i-1} + C_{i+1} + \dots + C_n - n \times TT_i + i \times TT_i + C_n \\
&< C_1 + C_2 + \dots + C_{i-1} + C_{i+1} + \dots + C_n - n \times TT_i + i \times TT_i + n \times TT_i
\end{aligned}$$

$$< C_1 + C_2 + \dots + C_{i-1} + C_{i+1} + \dots + C_n + i \times TT_i.$$

Thus, it is clear that the total completion time of the current sequence σ is improved by this change since $TC(\sigma) > TC(\sigma')$. Repetition of this argument for all remaining customer orders in the subset O'_A shows that customer orders of the subset O'_A succeed all other customer orders.

The proof of the Property 3(b) follows from the result by Smith (1956), who showed that processing the jobs in SPT-order minimizes the total completion time for the classical one-machine problem in which there are n jobs. In our problem it is clear that the customer orders in the subsets O'_A and O'_B may each be treated as pseudo-jobs in the classical one-machine problem, yielding an optimal sequence characterized by a customer order-based version of SPT, which we refer to as the STT sequence in which the customer orders in the subsets O'_A and O'_B are sequenced in non-decreasing order of their total time TT_i . \square

CHAPTER 3

LITERATURE REVIEW

Scheduling problems in production environments are classified according to shop structures, job characteristics and performance measures. Several shop structures are considered in scheduling studies, but generally single machine, parallel machine, job shop and flow shop environments are studied extensively in the literature. Performance measures considered in these studies diversify depending on the shop structure. For instance, there is an increasing number of studies considering job shop environment that aim to minimize makespan, total completion time, maximum lateness, maximum tardiness, total tardiness, number of tardy jobs and so on. Therefore, different characteristics and performance measures are taken into account according to observations of real life applications in production systems.

Variety of the problems leads to an increase in the number of studies mostly in recent years. Those studies show that setup time is considered frequently in scheduling studies. Until mid-1960s, benefits of incorporating setup times have been mentioned by many researchers for different manufacturing and service systems. A review of studies on scheduling problems considering setup times or costs is given first by Allahverdi et al (1999). This review was expanded with the growth of interest in reducing the effect of setup times or costs in Allahverdi et al (2008). In this survey, studies are classified with respect to shop type, shop characteristic, setup information (i.e. whether the setup time is sequence dependent or independent) and performance measure. Single machine, parallel machine and flow shop problems with setup consideration have dominance over the other shop structures. In order to reduce the effect of setup times or setup costs, several scheduling policies have been developed by the researchers. Whether the setup time is sequence dependent or not, processing

jobs in the same family as a batch is one of the way to avoid setup times. Therefore, there has been significant interest in scheduling problems involving *batching* issue. Studies with setup times or costs clearly indicate the advantage of batching and group technology assumption.

The problem under consideration can be associated with *group scheduling* and *customer order scheduling*. In this chapter, we briefly review the most relevant work to our study on group scheduling and customer order scheduling, especially for single machine problems.

3.1 Group Scheduling

In this section, we will give a review of studies on group scheduling, but definition of group technology should be given first. Janiak et al (2005) defines *group technology* as an approach to manufacturing and engineering management that seeks to achieve the efficiency of high-volume production by exploiting similarities of different products and activities in their production or execution. In this article, single machine problem to minimize total weighted resource consumption is studied with sequence independent setup times which precede processing of each group, and it is assumed that two external resources can be used to compress setup and processing times by each resource, respectively. Wang et al (2012) considers single machine problem with fixed group setup times, release dates and changeable processing times. It is shown that total completion time minimization problem can be solved in polynomial time for a special case only. Fixed group setup times considered by Wang et al (2012) can be regarded as job setup times and groups as customer orders in our study. The difference is that processing times are constant and ready times are not considered in our study. A similar problem is presented by He and Sun (2012) who consider the single machine group scheduling with deterioration without ready times. The aim of their study is to minimize the sum of completion times and they show that their problem can be polynomially solvable only under some conditions. In the case of jointly compressible setup and processing times, a polynomial time algorithm to find the optimal solution is presented in Ng et al. (2004). The objective of this problem is minimizing total job completion time on a single machine. Furthermore, a real life application of group technology is studied in PCB manufacturing (Sabouni and Logendram, 2013). This paper considers the problem of minimizing the

makespan on a single machine with carryover sequence dependent setup times. They propose a branch-and-bound algorithm and lower bound for their problem. A branch-and-bound procedure is described in Azizoglu and Webster (2001) in order to solve minimizing total weighted completion times problem on a batch processing machine with incompatible job families. Another study which proposes a branch-and-bound algorithm for solving minimizing flow times is proposed by Mazdeh et al. (2007). Single machine with batch delivery problem is defined and structural properties of the problem are investigated. Also these properties are used to devise a branch-and-bound solution scheme. Computational experiments show significant improvements over an existing dynamic programming approach.

Gupta and Chantaravarapan (2008) considers single machine group scheduling problem with family setups. A MILP is proposed and heuristic algorithms are developed in order to minimize total tardiness. Greedy algorithm, swap algorithm, simulated annealing are used to solve this problem. Performance of proposed algorithms is presented for small, medium and large size problems.

A different approach to group scheduling problem is investigated in Gerodimos et al. (1999). In this study there are jobs each requiring multiple operations. According to group technology terminology, jobs are referred to as groups and operations are referred to as jobs in the groups. All operations should be processed successively, and also all jobs in a group should also be processed together. Minimizing the sum of completion times, minimizing maximum lateness and minimizing the weighted number of late jobs are examined separately. Complexity of each type of problem is analyzed and a dynamic programming approach, which requires pseudo-polynomial running time, is presented to minimize the number of late jobs. A similar algorithm is also used to minimize the sum of completion time.

3.2 Customer Order Scheduling

Customer order scheduling is another closely related area to our problem environment under consideration.

Gupta et al. (1997) consider single machine scheduling problems to minimize makespan and total carrying cost of the customer orders where jobs are from different classes with sequence independent class setup times and there are customer

orders consisting of at least one job from each of the classes. They propose constructive polynomial time algorithms for the two hierarchical scheduling problems.

Customer order scheduling with family setup times that is required whenever production switches from one family to another is considered by Erel and Ghosh (2007). Their study considers a situation where C customers order various quantities of products from P different families, which can be produced on a continuously available machine in any sequence. The time from the start to completion of a customer order is called the order lead time and they consider total customer order lead time as the performance measure to be minimized. They show for the first time that the problem is strongly NP-hard and propose dynamic programming based exact solution algorithms for the general problem and a special case where number of customer orders is fixed. Solution of the special case shows that the problem can be solvable in polynomial time where number of customers is fixed and they expect to solve problem instances with number of jobs up to 30 or up to 200 when number of customers is less than 5 and the number of job families is less than 500.

Hazır (2008) considers customer order scheduling problem which is defined as to determining the sequence of tasks to satisfy the demand of customers who order several types of products produced on a single machine. In their study, setup is required whenever a product type is launched and the objective is to minimize the average customer order flow time. They propose four major metaheuristics and compare the performance of these heuristics which are simulated annealing, genetic algorithms, tabu search and ant colony optimization. They generate a set of problem and compare the solution quality and computational efforts of these heuristics and show that tabu search and ant colony perform better for large-sized problems whereas simulated annealing performs best in small-sized problems.

Gupta and Sivakumar (2005) considers single machine environment with two job families and more than one objective such as cycle time, machine utilization and due-date accuracy. Pareto optimal solution is obtained combining analytically optimal and conjunctive simulated scheduling approach, and results are compared with the solutions obtained by common dispatching rules (EDD and SPT). Improvements in percentage are given in the computational results of this article.

Minimizing the maximum job lateness on a single machine with job families are also investigated (Baker and Magazine, 2000). A heuristic based on EDD is proposed in this paper and computational results are presented. Similar problem is taken into account by Chen (2008). The only difference is the objective of the problem being minimization of maximum tardiness. A heuristic is presented in order to solve large-sized problems. A branch-and-bound algorithm utilizing several theorems is also proposed to find the optimal schedules for the problem. Computational results are provided to demonstrate the effectiveness of the heuristic.

In the literature, there are several studies considering group scheduling and customer order scheduling. However, to the best of our knowledge, the customer order scheduling problem, in which all jobs in a customer order should be processed successively, is studied for the first time.

CHAPTER 4

MAKESPAN MINIMIZATION PROBLEM

In this chapter, we analyze our customer order scheduling problem for minimizing the makespan problem, describe a shortest path formulation by constructing a network, and show that the problem can be solved in polynomial time.

It is clear that the application of Property 2 given in Section 2.3 will reduce the size of the problem by dropping the customer orders in the set O' from the problem before applying the shortest path network approach to the customer orders in the set O'' .

4.1 Shortest Path Network

The shortest path network consists of a set of nodes and a set of arcs connecting certain pairs of the nodes, as shown in Figure 4.1. The node set in the network includes:

- a dummy beginning (source) node (0),
- a dummy ending (sink) node ($K + 1$),
- node (k, O_i, J_j) , $k = 1, \dots, K$, $i = 1, \dots, K$, $\forall J_j \in O_i$: Each such node represents that customer order O_i is assigned to position k of the sequence of the customer orders and job J_j in order O_i is assigned to the last position of this customer order.

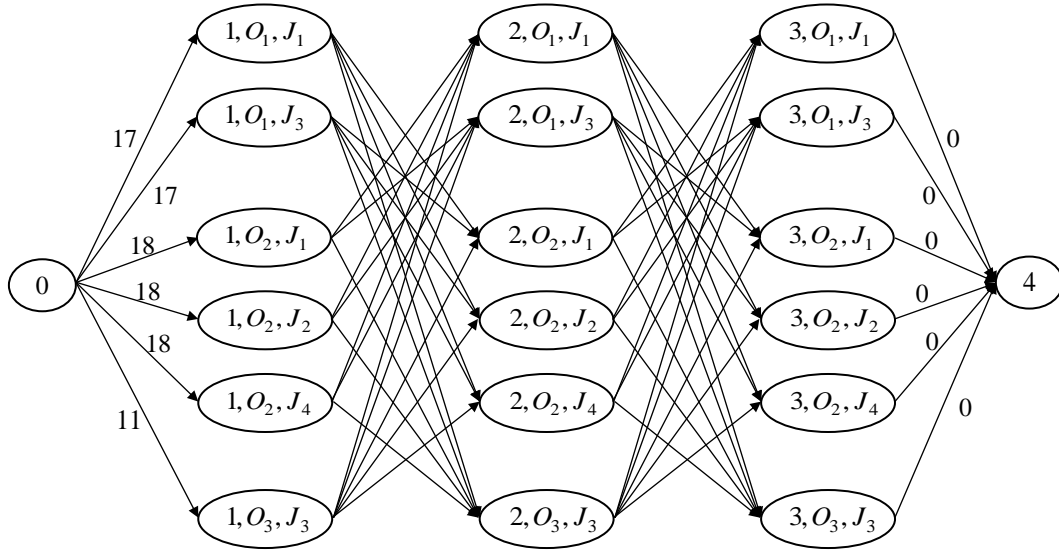


Figure 4.1 Shortest path network for the problem considered in Example 1

For each position, which corresponds to a stage in the shortest path network, in the sequence of customer orders, we create $N_T = \sum_{i=1}^K N_i$ nodes, where N_T is the total number of jobs in all customer orders and N_i is the number of jobs in the customer order i . The directed arc set is generated as follows:

- An arc from the beginning node (0) to node $(1, O_i, J_r)$ with flow cost $\sum_{J_j \in O_i} (s_j + p_j)$.
- An arc from node (k, O_i, J_c) to node $(k+1, O_l, J_c)$ with flow cost $\sum_{J_j \in O_l} (s_j + p_j)$, where $k=1, \dots, K-1$, $i \neq l$, and J_c is a common job in customer orders O_i and O_l .
- An arc from node (k, O_i, J_r) to node $(k+1, O_l, J_v)$ with flow cost $\sum_{\substack{J_j \in O_l \\ j \neq r}} s_j + \sum_{J_j \in O_l} p_j$, where $k=1, \dots, K-1$, $i \neq l$, $r \neq v$.
- An arc from node (K, O_i, J_r) to the ending node $(K+1)$ with flow cost of zero.

It is clear that the maximum possible number of nodes based on K and N_T is $(K \times N_T) + 2$.

The flow costs between intermediate nodes of the network for the problem considered in Example 1 are given in Table 4.1.

Table 4.1 Flow costs between intermediate nodes of the network for the problem considered in Example 1 when k equals 1 or 2

to from	$k+1, O_1, J_1$	$k+1, O_1, J_3$	$k+1, O_2, J_1$	$k+1, O_2, J_2$	$k+1, O_2, J_4$	$k+1, O_3, J_3$
k, O_1, J_1	-	-	18	15	15	11
k, O_1, J_3	-	-	18	18	18	7
k, O_2, J_1	17	14	-	-	-	11
k, O_2, J_2	17	17	-	-	-	11
k, O_2, J_4	17	17	-	-	-	11
k, O_3, J_3	13	17	18	18	18	-

Thus, the following pure-integer linear programming model needs to be solved to find the shortest path in the network.

$$\begin{aligned}
 \text{(SP) Minimize} \quad & \sum_f \sum_t c_{ft} x_{ft} \\
 \text{Subject to} \quad & \sum_f x_{ft} - \sum_u x_{uf} = \begin{cases} 1 & \text{if } f = 0 \\ 0 & \text{if } f \neq 0 \text{ or } K+1 \\ -1 & \text{if } f = K+1 \end{cases} \\
 & x_{ft} \in \{0,1\} \quad \forall f, t
 \end{aligned}$$

where c_{ft} is the cost flow from node f to node t , and x_{ft} is the amount of flow on the arc (f, t) .

It is known that the node-arc incidence matrix associated with the conservation equations given above is totally unimodular. Hence, there exists at least one optimal solution to the linear programming (LP) relaxation of the above model in which all decision variables are integer, i.e., $x_{ft} = 0$ or 1. Such a solution can be found by replacing $x_{ft} \in \{0,1\}$ by $x_{ft} \geq 0$ and solving the resulting LP. However, the solution can be determined even more efficiently by Dijkstra's algorithm.

4.2 Some Special Cases Solvable by Priority Rules

The following theorems give the properties of an optimal schedule for some special cases of the makespan minimization problem which are solvable by priority rules.

Special Case 1: Each customer order has only one job, and all jobs are distinct.

Theorem 1 *If each customer order has only one job and all jobs are distinct, then any sequence of the customer orders is optimal.*

Proof If each customer order has only one job, and all jobs are distinct, then a setup is required before processing each job. Thus, all sequences of the customer orders give the same minimum makespan value which is equivalent to $C_{\max} = \sum_{J_j \in J} (s_j + p_j)$

This completes the proof. \square

Special Case 2: Each customer order has only one job, and some jobs are common.

Theorem 2 *If each customer order has only one job and some jobs are common, then*

- *the customer orders having a common job are processed consecutively, and*
- *the blocks of the customer orders and the customer orders having no common job are processed in any sequence*

in the optimal schedule.

Proof Suppose that K_1 customer orders have job J_1 , K_2 customer orders have job J_2 , etc. Processing K_1 customer orders having job J_1 consecutively requires only one setup. Similarly, processing K_2 customer orders having job J_2 consecutively requires only one setup. Making such blocks of customer orders having a common job reduces the number of setups to a minimum value. It is clear that any sequence of the blocks of customer orders along with the customer orders having no common job gives the same minimum makespan value. This completes the proof. \square

Special Case 3: Each customer order has two distinct jobs, which exist in all customer orders.

Theorem 3 *If each customer order has two distinct jobs, say jobs J_1 and J_2 , which exist in all customer orders, then the job with long (short) setup time is processed as the second job of the customer orders in each odd-numbered (even-numbered) position in the optimal schedule.*

Proof In this case, the sequence of the customer orders are not important since all customer orders have common two jobs. Thus, we need to determine whether job J_1 or job J_2 is processed in the first position within each position of customer orders. If the first job in the first customer order is job J_1 , then the first job in the second customer order should be job J_2 to get the advantage of no-setup when job J_2 is the last and the first job of the first and second customer orders, respectively. Similarly, the first job in the third customer order should be job J_1 to get the advantage of no-setup when job J_1 is the last and the first job of the second and third customer orders, respectively. Repeating this process for the remaining positions of the customer orders, we obtain the minimum makespan schedule when job J_1 is selected as the first job in the first customer order. The resulting schedule is as follows:

$$\text{Schedule 1: } O_1(J_1 - J_2) - O_2(J_2 - J_1) - O_3(J_1 - J_2) - O_4(J_2 - J_1) - \dots$$

On the other hand, if the first job in the first customer order is job J_2 , then the first job in the second customer order should be job J_1 to get the advantage of no-setup when job J_1 is the last and the first job of the first and second customer orders, respectively. Similarly, the first job in the third customer order should be job J_2 to get the advantage of no-setup when job J_2 is the last and the first job of the second and third customer orders, respectively. Repeating this process for the remaining positions of the customer orders, we obtain the minimum makespan schedule when job J_2 is selected as the first job in the first customer order. The resulting schedule is as follows:

$$\text{Schedule 2: } O_1(J_2 - J_1) - O_2(J_1 - J_2) - O_3(J_2 - J_1) - O_4(J_1 - J_2) - \dots$$

It is obvious that Schedules 1 and 2 have an equal makespan value if there are odd-numbered of customer orders. However, the makespan value is different when there

are even-numbered of customer orders. In this case, the schedule having the job with long setup time in the second position of the first customer order will give the larger reduction in makespan. For example, if job J_2 has greater setup time (i.e., $s_2 \geq s_1$), then the makespan of Schedule 1 is $s_2 - s_1$ smaller than that of Schedule 2. The reverse is true when job J_1 has greater setup time (i.e., $s_1 \geq s_2$). That is, the makespan of Schedule 2 is $s_1 - s_2$ smaller than that of Schedule 1. Therefore, the job with long (short) setup time is processed as the second job of the customer order in each odd-numbered (even-numbered) position in the optimal sequence. This completes the proof. \square

CHAPTER 5

TOTAL COMPLETION TIME MINIMIZATION PROBLEM

In this chapter, the single-machine customer order scheduling problem is studied for minimizing the total completion time of the customer orders. The decision is to determine the sequence of the customer orders as well as the first and last jobs processed in each customer order. The complexity of this problem is open; but it is most likely NP-hard. We first develop a mixed integer linear programming (MILP) model, and then give the optimal solutions of some polynomial-time solvable cases of the problem. Finally, we close this chapter by proposing several heuristic algorithms.

5.1 Mathematical Model

The following indices, sets, parameters and variables are used in this model.

Parameters, indices and sets

K Number of customer orders

i Index for customer orders ($i = 1, 2, \dots, K$)

j Index for jobs ($j = 1, 2, \dots, N$)

k Index for position of customer orders in the sequence ($k = 1, 2, \dots, K$)

D_{ij} $D_{ij} = 1$ if customer order O_i has job J_j ; otherwise, $D_{ij} = 0$

p_j Processing time for job J_j

s_j Setup time for job J_j

N_i Set of different jobs in customer O_i

A Set of customer orders having more than one job to be processed

Using p_j and s_j , we compute the total (sum of setup and processing) time of all jobs in the customer order O_i and the setup time between two successive jobs as

TT_i Total (sum of setup and processing) time of all jobs in customer order O_i ,
where $TT_i = \sum_{J_j \in O_i} (s_j + p_j)$

ST_{hj} Setup time between jobs J_h and J_j if job J_j immediately follows job J_h ,
where $ST_{hj} = s_j$ if $j \neq h$; otherwise, $ST_{hj} = 0$

Decision variables

$X_{ik} = \begin{cases} 1 & \text{if customer order } O_i \text{ is assigned to position } k \\ 0 & \text{otherwise} \end{cases}$

$F_{jik} = \begin{cases} 1 & \text{if job } J_j \text{ is the first job in customer order } O_i \text{ assigned to position } k \\ 0 & \text{otherwise} \end{cases}$

$L_{jik} = \begin{cases} 1 & \text{if job } J_j \text{ is the last job in customer order } O_i \text{ assigned to position } k \\ 0 & \text{otherwise} \end{cases}$

$Y_{hijk} = \begin{cases} 1 & \text{if both } L_{hik} \text{ and } F_{j_l, k+1} \text{ are equal to 1 (i.e., last job of a customer order and} \\ & \text{the first job of the immediately following customer order are not same.)} \\ 0 & \text{otherwise} \end{cases}$

RT_{ik} = Realized total (sum of setup and processing) time of customer orders O_i
assigned to position k

TC = Total completion time of customer orders

MILP model

$$\text{Minimize } TC = \sum_{k=1}^K (K - k + 1) \sum_{i=1}^K RT_{ik} \quad (1)$$

$$\text{Subject to } \sum_{i=1}^K X_{ik} = 1 \quad \text{for } k = 1, 2, \dots, K \quad (2)$$

$$\sum_{k=1}^K X_{ik} = 1 \quad \text{for } i = 1, 2, \dots, K \quad (3)$$

$$\sum_{j \in N_i} \sum_{i=1}^K F_{jik} = 1 \quad \text{for } k = 1, 2, \dots, K \quad (4)$$

$$\sum_{j \in N_i} \sum_{i=1}^K L_{jik} = 1 \quad \text{for } k = 1, 2, \dots, K \quad (5)$$

$$F_{jik} \leq D_{ij} X_{ik} \quad \text{for } j \in N_i; i = 1, 2, \dots, K; k = 1, 2, \dots, K \quad (6)$$

$$L_{jik} \leq D_{ij} X_{ik} \quad \text{for } j \in N_i; i = 1, 2, \dots, K; k = 1, 2, \dots, K \quad (7)$$

$$F_{hik} + L_{jlk+1} - 1 \leq Y_{hijlk} \quad \text{for } j \in N_i; h \in N_i; j \neq h \\ i = 1, 2, \dots, K; l = 1, 2, \dots, K; l \neq i; k = 1, 2, \dots, K \quad (8)$$

$$F_{jik} + L_{jik} \leq 1 \quad \text{for } j \in A; i = 1, 2, \dots, K; k = 1, 2, \dots, K \quad (9)$$

$$RT_{i1} \geq TT_i X_{i1} \quad \text{for } i = 1, 2, \dots, K \quad (10)$$

$$RT_{ik} \geq TT_i X_{ik} - \sum_{j \in N_i} s_j F_{jik} + \sum_{h \in N_i} \sum_{j \in N_i} \sum_{l=1}^K ST_{hj} Y_{hijlk-1} \quad \text{for } i = 1, 2, \dots, K; k \geq 2 \quad (11)$$

$$RT_{ik} \geq 0 \quad \text{for } \forall i, k \quad (12)$$

$$X_{ik}, F_{jik}, L_{jik}, Y_{hijlk} \in \{0, 1\} \quad \text{for } \forall h, i, j, k, l \quad (13)$$

In the above MILP model, the objective in (1) is to minimize the total completion time. Constraint sets (2) and (3) ensure that each position in the sequence of customer orders is occupied by one customer only and each customer order is assigned to one position only, respectively. Constraint sets (4) and (5) guarantee only one job in each customer order can be processed as the first or last job in its customer order, respectively. Constraint sets (6) and (7) ensure that a job cannot be the first or last job of a customer order assigned to a position if this customer order does not include the job. Constraint set (8) satisfies the condition that no setup time is necessary before the processing of the first job of a customer order if this first job is

same as the last job of the immediately preceding customer order. Constraint set (9) guarantees that each job in a customer order can be the first, immediate or last job of this customer order. Constraint sets (10) and (11) define the realized total (sum of setup and processing) time of the customer orders assigned to the first and other positions, respectively. Constraint sets (12) and (13) impose non-negativity and binary restrictions on the decision variables, respectively.

5.2 Lower and Upper Bounds on the Total Completion Time

To improve efficiency of the MILP model we introduce a lower bound on the total completion time value. A lower bound LB is found by assuming that

- the job with the longest setup time in each customer order is assigned to the first position of its group,
- setup time of the job with the longest setup time in each customer order is canceled by assuming that this job is same as the one in the last position of the previous customer orders, and
- all customer orders are sequenced in non-decreasing order of their revised total time $TT_i - \max_{J_j \in O_i} \{s_j\}$.

Then we add the constraint $LB \leq TC$ to the mathematical model.

Instead of assuming an initial upper bound on the total completion time as infinity, the total completion time value obtained sequencing all customer orders in non-decreasing order of their total time TT_i can be used as an upper bound UB . Let $TT_{STT}[k]$ be the total time of the customer order in the k th position of the sequence in which customer orders are sequenced in non-decreasing order of their total time (i.e., shortest total time rule). Then, the total completion time $\sum_{k=1}^K (K - k + 1) TT_{STT}[k]$

becomes the upper bound, i.e., $UB = \sum_{k=1}^K (K - k + 1) TT_{STT}[k]$. Therefore, we add the constraint $TC \leq UB$ to the mathematical model.

In the MILP model including the lower and upper bounds on the total completion time, one set of the decision variables is a continuous variable, and other

$K^2[1 + N_T(K + 2)]$ decision variables, where $N_T = \sum_{i=1}^K N_i$, are 0–1 type. On the other hand, the MILP model has $4K + K^2[1 + |A| + N_T(N_T(K - 1) - K + 1)] + 2$ constraints. It is expected that the MILP model cannot be solved for large-sized problem instances so that we will propose heuristic solution procedures.

5.3 Some Special Cases Solvable by Priority Rules

The following theorems give the properties of an optimal schedule for some special cases of the total completion time minimization problem which are solvable by priority rules.

Special Case 1: Each customer order has only one job, and all jobs are distinct.

Theorems 4 *If each customer order has only one job and all jobs are distinct, then the customer orders are processed in STT sequence in an optimal schedule.*

Proof If each customer order has only one job and all jobs are distinct, then the problem reduces to the traditional single-machine scheduling problem, where the minimum total completion time is obtained by sequencing the customer orders in non-decreasing order of $TT_i = \sum_{J_j \in O_i} (s_j + p_j)$ time (i.e, in STT sequence). This

completes the proof. \square

Special Case 2: Each customer order has two distinct jobs, which exist in all customer orders.

Theorem 5 *If each customer order has two distinct jobs, say jobs J_1 and J_2 , which exist in all customer orders, then the job with long (short) setup time is processed as the second job of the customer orders in each odd-numbered (even-numbered) position in the optimal sequence.*

Proof As it is stated in Theorem 3, the following two schedules need to be considered again:

Schedule 1: $O_1(J_1 - J_2) - O_2(J_2 - J_1) - O_3(J_1 - J_2) - O_4(J_2 - J_1) - \dots$

Schedule 2: $O_1(J_2 - J_1) - O_2(J_1 - J_2) - O_3(J_2 - J_1) - O_4(J_1 - J_2) - \dots$

Now, let $TT = (s_1 + p_1) + (s_2 + p_2)$ be the total time of a customer order. Furthermore, let $TC(1|K)$ and $TC(2|K)$ be the total completion times of Schedules 1 and 2, respectively, given K customer orders. Then we have,

$$\begin{array}{ll}
TC(1|K = 1) = TT & TC(1|K = 1) = TT \\
TC(1|K = 2) = 3TT - s_2 & TC(1|K = 2) = 3TT - s_1 \\
TC(1|K = 3) = 6TT - s_1 - 2s_2 & TC(1|K = 3) = 6TT - 2s_1 - s_2 \\
TC(1|K = 4) = 10TT - 2s_1 - 4s_2 & TC(1|K = 4) = 10TT - 4s_1 - 2s_2 \\
TC(1|K = 5) = 15TT - 4s_1 - 6s_2 & TC(1|K = 5) = 15TT - 6s_1 - 4s_2 \\
TC(1|K = 6) = 21TT - 6s_1 - 9s_2 & TC(1|K = 6) = 21TT - 9s_1 - 6s_2 \\
\vdots & \vdots
\end{array}$$

It is obvious that $TC(1|K) \leq TC(2|K)$ if $s_2 \geq s_1$. Thus, the schedule having the job with long setup time in the second position of the first customer order will give the larger reduction in total completion time. For example, if job J_2 has greater setup time (i.e., $s_2 \geq s_1$), then the total completion time of Schedule 1 is smaller than that of Schedule 2. The reverse is true when job J_1 has greater setup time (i.e., $s_1 \geq s_2$). That is, the total completion time of Schedule 2 is smaller than that of Schedule 1. Therefore, the job with long (short) setup time is processed as the second job of the customer order in each odd-numbered (even-numbered) position in the optimal sequence. This completes the proof. \square

Special Case 3: There are two customer orders only.

Theorem 6 *If there are two customer orders only, say customer orders O_1 and O_2 , and $TT_1 \leq TT_2$, then*

- *the customer order O_1 is processed earlier in the optimal schedule, and*
- *the common job, if any, which has the longest setup time, should be assigned to the last and first positions of the customer orders O_1 and O_2 , respectively.*

Proof We use the method of pair-wise interchange of adjacent customer orders. Suppose that S is a schedule in which the customer orders O_1 and O_2 are processed in the first and second positions, respectively. Furthermore, assume that they have a common job J_c having the longest setup time. Now construct a new schedule S' , in which the customer orders O_1 and O_2 are interchanged in sequence. That is, O_2 and O_1 are processed in the first and second positions, respectively. It is obvious that assigning the common job J_c having the longest setup time to the last and first positions of the first and second customer orders, respectively, will reduce the completion time of the second customer order in any sequence of customer orders. Sequences S and S' are represented as:

$$\text{Schedule } S : O_1(\cdots - J_c) - O_2(J_c - \cdots)$$

$$\text{Schedule } S' : O_2(\cdots - J_c) - O_1(J_c - \cdots)$$

Let $C_1(S)$ and $C_2(S)$ be the completion times of the customer orders O_1 and O_2 , respectively, in Schedule S . It is clear that

$$C_1(S) = TT_1 = \sum_{J_j \in O_1} (s_j + p_j) \quad C_2(S) = TT_1 + TT_2 - s_c$$

where s_c is the setup time of the common job J_c having the longest setup time.

Similarly, let $C_1(S')$ and $C_2(S')$ be the completion times of the customer orders O_1 and O_2 , respectively, in Schedule S' . It is clear that

$$C_2(S') = TT_2 = \sum_{J_j \in O_2} (s_j + p_j) \quad C_1(S') = TT_2 + TT_1 - s_c$$

Let $TC(S)$ and $TC(S')$ be the total completion times of Schedules S and S' , respectively. Then, we have

$$TC(S) = C_1(S) + C_2(S) = TT_1 + TT_1 + TT_2 - s_c = 2TT_1 + TT_2 - s_c$$

and

$$TC(S') = C_2(S') + C_1(S') = TT_2 + TT_2 + TT_1 - s_c = 2TT_2 + TT_1 - s_c.$$

The difference between $TC(S)$ and $TC(S')$ is

$$TC(S) - TC(S') = 2TT_1 + TT_2 - s_c - (2TT_2 + TT_1 - s_c) = TT_1 - TT_2 \leq 0$$

since $TT_1 \leq TT_2$. In other words, $TC(S') > TC(S)$. That is, the interchange of the customer orders O_1 and O_2 in Schedule S increases the total completion time. It follows that processing the customer order with smaller total time in the first position must be optimal.

5.4 Heuristic Algorithms

The size of the MILP model discussed in the previous section increases drastically as the number of customer orders and jobs increases. Therefore, optimal solution for large-sized problems may not be determined within reasonable computational times. Moreover, for most real life problems sub-optimal results can be satisfactory. This motivates us to develop fast algorithms which provide optimal or near-optimal solutions within acceptable computational times.

Our first proposed heuristic algorithm consists of three steps. In the first step, an initial sequence is created without setup savings obtained by assigning a common job of a pair of customer orders processed successively. Step 2 attempts to improve this initial sequence by taking into account the setup savings. In Step 3, the schedule obtained in Step 2 is improved, if possible, using a pair-wise interchange the positions of the customer orders. The stepwise description of our proposed algorithm is as follows:

Algorithm-1:

Step 1. [Initial Schedule Generation without Setup Savings]

(a) For each customer order O_i , calculate the total time as:

$$TT_i = \sum_{J_j \in O_i} (s_j + p_j)$$

(b) Decompose the customer order set O into two disjoint sets O' and O'' ($O = O' \cup O''$ and $O' \cap O'' = \emptyset$) where the set O' consists of customer orders having no common job with other customer orders, and its complement set O'' consists of remaining customer orders.

(c) Decompose the customer order set O' into three disjoint subsets O'_B , O'_R and O'_A , where

$$O'_B = \left\{ O_i \mid TT_i \leq \min_{O_l \in O''} \left\{ TT_l - \max_{J_j \in O_l} \{s_j\} \right\} \right\},$$

$$O'_R = \left\{ O_i \mid \min_{O_l \in O''} \left\{ TT_l - \max_{J_j \in O_l} \{s_j\} \right\} < TT_i < \max_{O_l \in O''} \{TT_l\} \right\},$$

$$O'_A = \left\{ O_i \mid \max_{O_l \in O''} \{TT_l\} \leq TT_i \right\}.$$

(d) Sequence all customer orders of each subset O'_B , O'_A and $O'' \cup O'_R$ in non-decreasing order of their total times. Denote the resulting sequences as $\sigma_{STT}(O'_B)$, $\sigma_{STT}(O'_A)$ and $\sigma_{STT}(O'' \cup O'_R)$. Let K_U be the number of customer orders in the set $\sigma_{STT}(O'' \cup O'_R)$. Set $i = 1$.

(e) Let σ be the initial sequence of all customer orders without setup savings as

$$\sigma: \sigma_{STT}(O'_B) \rightarrow \sigma_{STT}(O'' \cup O'_R) \rightarrow \sigma_{STT}(O'_A)$$

(f) Calculate the associated total completion time CT of the initial schedule of all customer orders.

Step 2. [Improved Schedule with Setup Savings]

- (a) If $i > K_U - 1$ then go to Step 2(c); otherwise, let $O_{[i]}$ and $O_{[i+1]}$ are two adjacent customer orders in positions i and $i+1$ of the sequence $\sigma_{STT}(O'' \cup O'_R)$. If the set of common jobs in this pair is empty, then go to Step 2(b); otherwise, determine an unassigned job having the maximum setup time from the set of common jobs, assign this job to the last and the first job positions of these linked customer orders $O_{[i]}$ and $O_{[i+1]}$, respectively.
- (b) Set $i = i + 1$, and repeat Step 2(a).
- (c) Let σ_I be the first improved schedule of all customer orders. Calculate the associated total completion time CT_I of the improved schedule σ_I .

Step 3. [Improved Schedule by Pair-wise Interchange of Customer Orders]

- (a) Set $i = 1$ and $l = 2$.
- (b) If $i > K_U - 1$ then go to Step 3(f); otherwise, let $O_{[i]}$ and $O_{[l]}$ are two customer orders in positions i and l of the initial feasible sequence σ_I obtained in Step 1.
- (c) If there is at least one unassigned common job in the pair of the customer orders $O_{[i]}$ and $O_{[l]}$, then
 - Determine a common job having the maximum setup time from the set of unassigned common jobs.
 - Assign the common job to the last and the first job positions of the customer orders $O_{[i]}$ and $O_{[l]}$, respectively.
 - Go to Step 3(e).Otherwise, Go to Step 3(d).
- (d) If there is a common job, which is the single job, in the customer order $O_{[i]}$ or $O_{[l]}$, then

- If the common job is the single job of the customer order $O_{[l]}$, then assign the common job to the last job position of the customer order $O_{[i]}$; otherwise (i.e. the common job is the single job of the customer order $O_{[i]}$), assign the common job to the first job position of the customer order $O_{[l]}$.
- Go to Step 3(e).

Otherwise, set $l = l + 1$ and Go to Step 3(b).

- (e) Pairwise interchange the position of the string of all customer orders between positions i and l with the position of the string of customer orders starting at position l and ending at position k .
- Set $i = i + 1 + k - l$, $l = i + 1$. If $l = K_U$, then set $i = i + 1$.
 - Go to Step 3(b).
- (f) Let σ_{II} be the second improved schedule of all customer orders. Calculate the associated total completion time CT_{II} of the schedule σ_{II} .

Step 4. [Final schedule]

- (a) If $CT_{II} < CT_I$, then select the schedule σ_{II} as the final schedule, and set $\sigma_F = \sigma_{II}$; otherwise, select the schedule σ_I , and set $\sigma_F = \sigma_I$.
- (b) Stop.

We illustrate Algorithm-1 with a numerical example.

Example 2 Consider a simple instance of the problem in which there are seven orders. Jobs in each customer order and the setup and processing times for all jobs are given in Table 5.1 and Table 5.2, respectively. The total time for each customer order, which is obtained in Step 1(a), is already included in the data shown in Table 5.1.

Table 5.1 Jobs and initial total times of the customer orders in Example 2

Customer	O_1		O_2		O_3			O_4			O_5		O_6		O_7
Job (J_j)	J_1	J_2	J_9	J_{11}	J_4	J_5	J_8	J_1	J_2	J_5	J_7	J_8	J_3	J_6	J_{10}
Initial Total Time (TT_i)	38		86		79			65			47		80		20

Table 5.2 Setup and processing times for all jobs in Example 2

Job (J_j)	J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8	J_9	J_{10}	J_{11}
Setup Time (s_j)	13	17	25	12	16	13	14	20	13	12	28
Processing Time (p_j)	7	1	13	8	11	29	1	12	32	8	13

Step 1(b) of the algorithm gives us the set $O' = \{O_2, O_6, O_7\}$, which is composed of customer orders having no common job with other customer orders, and the set $O'' = \{O_1, O_3, O_4, O_5\}$. Step 1(c) of the algorithm decomposes the set O' into $O'_B = \{O_7\}$, $O'_A = \{O_2, O_6\}$ and $O'_R = \emptyset$ since $TT_7 = 20 < 21$, where

$$\begin{aligned}
 & \min_{O_i \in O''} \left\{ TT_i - \max_{J_j \in O_i} \{s_j\} \right\} \\
 &= \min \left\{ TT_1 - \max_{J_j \in O_1} \{s_j\}, TT_3 - \max_{J_j \in O_3} \{s_j\}, TT_4 - \max_{J_j \in O_4} \{s_j\}, TT_5 - \max_{J_j \in O_5} \{s_j\} \right\} \\
 &= \min \{ 38 - \max \{13, 17\}, 79 - \max \{12, 16, 20\}, 65 - \max \{13, 17, 16\}, \\
 &\quad 47 - \max \{14, 20\} \} \\
 &= \min \{ 21, 59, 48, 27 \} \\
 &= 21
 \end{aligned}$$

and $TT_2 = 86 > 79$, $TT_6 = 80 > 79$ where

$$\max_{O_i \in O''} \{TT_i\} = \max \{TT_1, TT_3, TT_4, TT_5\} = \max \{38, 79, 65, 47\} = 79.$$

The following sequences are obtained by Step 1(d) of the algorithm

$$\sigma_{STT}(O'_B) = \{O_7\}$$

$$\sigma_{STT}(O'_A) = \{O_6, O_2\}$$

$$\sigma_{STT}(O'' \cup O'_R) = \{O_1, O_5, O_4, O_3\}$$

Table 5.3 illustrates the sequence $\sigma_{STT} = \{O_1, O_5, O_4, O_3\}$ obtained by Step 1 for the set $O'' \cup O'_R = \{O_1, O_3, O_4, O_5\}$.

Table 5.3 Sequence $\sigma_{STT} = \{O_1, O_5, O_4, O_3\}$ obtained for the set $O'' \cup O'_R$ in Step 1

Customer Order Sequence	O_1		O_5		O_4			O_3		
Job Sequence	J_1	J_2	J_7	J_8	J_1	J_2	J_5	J_4	J_5	J_8
Total Time	38		47		65			79		

In Step 1(e), the initial sequence of all customer orders without setup savings is constructed as:

$$\sigma: \{O_7\} \rightarrow \{O_1, O_5, O_4, O_3\} \rightarrow \{O_6, O_2\}$$

The associated total completion time of this sequence is obtained in Step 1(f) as

$$CT = 20 \times 7 + 38 \times 6 + 47 \times 5 + 65 \times 4 + 79 \times 3 + 80 \times 2 + 86 \times 1 = 1,346 \text{ time units.}$$

In Step 2, we first consider the adjacent customer orders in positions 1 and 2 of the sequence $\sigma_{STT}(O'' \cup O'_R) = \{O_1, O_5, O_4, O_3\}$. This first pair of customer orders are O_1 and O_5 , and they have no common job. Thus, we consider the adjacent customer orders in positions 2 and 3, which are O_5 and O_4 . This second pair of customer orders has also no common job. When we consider the adjacent customer orders in positions 3 and 4, which are O_4 and O_3 , we observe that job J_5 is the common job, and it is assigned to the last and the first job positions of the customer orders O_4 and

O_3 , respectively. Table 5.4 illustrates the improved sequence for the set $O'' \cup O'_R = \{O_1, O_3, O_4, O_5\}$, and the associated total completion time CT_I of all customer orders is

$$CT_I = 20 \times 7 + 38 \times 6 + 47 \times 5 + 65 \times 4 + 63 \times 3 + 80 \times 2 + 86 \times 1 = 1,298 \text{ time units.}$$

Table 5.4 Improved sequence $\{O_1, O_5, O_4, O_3\}$ obtained for the set $O'' \cup O'_R$ in Step 2

Customer Order Sequence	O_1		O_5		O_4			O_3		
Job Sequence	J_1	J_2	J_7	J_8	J_1	J_2	J_5	J_5	J_4	J_8
Modified Total Time	38		47		65			63		

In Step 3, the customer order O_4 in position 3 ($l = 3$) has two unassigned common jobs J_1 and J_2 with the customer order O_1 in position 1 ($i = 1$). The job having the maximum setup time from the set of common jobs is J_2 , and it is assigned to the last and the first job positions of the customer orders O_1 and O_4 , respectively. The customer order O_5 is between the customer orders in positions 1 and 3, and the customer orders O_4 and O_3 form a string of customer orders since they are linked by job J_5 . We interchange the positions of the customer order O_5 and the string of customer orders O_4 and O_3 . Thus, the modified sequence of the orders becomes $O_1 - O_4 - O_3 - O_5$. The new values for indices i and l become $i = i + 1 + k - l = 1 + 1 + 4 - 3 = 3$ and $l = i + 1 = 3 + 1 = 4$, respectively. The customer order O_5 in position 4 ($l = 4$) has one unassigned common job, which is J_8 , with the customer order O_3 in position 3 ($i = 3$). Thus, job J_8 is assigned to the last and the first job positions of the customer orders O_3 and O_5 , respectively. The improved sequence obtained for the set $O'' \cup O'_R = \{O_1, O_3, O_4, O_5\}$ is illustrated by Table 5.5, and the associated total completion time CT_{II} of all customer orders is

$$CT_{II} = 20 \times 7 + 38 \times 6 + 48 \times 5 + 63 \times 4 + 27 \times 3 + 80 \times 2 + 86 \times 1 = 1,187 \text{ time units.}$$

Table 5.5 Improved sequence $\{O_1, O_4, O_3, O_5\}$ obtained for the set $O'' \cup O'_R$ in Step 3

Customer Order Sequence	O_1		O_4			O_3			O_5	
Job Sequence	J_1	J_2	J_2	J_1	J_5	J_5	J_4	J_8	J_8	J_7
Modified Total Time	38		48			63			27	

From Step 4, we obtain the best (final) sequence $\sigma_F = \{O_7, O_1, O_4, O_3, O_5, O_6, O_2\}$ of all customer orders and jobs with a total completion time of $CT_F = 1,187$ time units (equivalent to the optimal total completion time value) since $CT_{II} = 1,187 < CT_I = 1,298$. Table 5.6 illustrates the final sequence.

Table 5.6 Final sequence σ_F of all customer orders and jobs in Example 2

Customer Order Sequence	O_7	O_1		O_4			O_3			O_5	O_6		O_2		
Job Sequence	J_{10}	J_1	J_2	J_2	J_1	J_5	J_5	J_4	J_8	J_8	J_7	J_3	J_6	J_9	J_{11}
Modified Total Time	20	38		48			63			27	80		86		

Our second proposed heuristic algorithm consists of two steps. In the first step, an initial sequence without setup savings is created as in Step 1 of Algorithm-1. Step 2 then attempts to improve this initial sequence by taking into account the setup savings. The stepwise description of our proposed algorithm is as follows:

Algorithm-2:

Step 1. [Initial Schedule without Setup Savings] Apply Step 1 of Algorithm-1.

Step 2. [Improved Schedule with Setup Savings]

- (a) Among the customer orders in positions 1 through K_U of the sequence $\sigma_{STT}(O'' \cup O'_R)$, set $c = 1$. Let $O_{[c]}$ be this customer order in position c of the sequence $\sigma_{STT}(O'' \cup O'_R)$. Set $i = 1$.

- (b) Determine a job having the maximum setup time from the set of unassigned jobs in the customer order $O_{[c]}$. Let J_j be this job. Assign job J_j to the last job position of the customer order $O_{[c]}$.
- (c) Move the customer order $O_{[c]}$ to position i of the sequence $\sigma_{STT}(O'' \cup O'_R)$.
- (d) Among the customer orders in positions $i+1$ through K_U of the sequence $\sigma_{STT}(O'' \cup O'_R)$, determine the customer order having the minimum total time TT with job J_j . Let $O_{[c]}$ be this customer order in position c of the sequence $\sigma_{STT}(O'' \cup O'_R)$. If there is no such customer order, then set $c = i+1$, and go to Step 2(b); otherwise, go to Step 2(e).
- (e) Assign job J_j to the first job position of the customer order $O_{[c]}$.
- (f) Set $i = i+1$. If $i > K_U - 1$ then go to Step 3; otherwise, go to Step 2(b).

Step 3. [Final schedule]

- (a) Let σ_F be the improved (final) schedule, obtained by Step 2, of all customer orders.
- (b) Calculate the associated total completion time CT_F of the final sequence of all customer orders.

We illustrate Algorithm-2 with a numerical example.

Example 3 Consider the problem given in Example 2.

In Step 1(e) of Algorithm-1, the initial sequence of all customer orders without setup savings is constructed as:

$$\sigma : \{O_7\} \rightarrow \{O_1, O_5, O_4, O_3\} \rightarrow \{O_6, O_2\}$$

The associated total completion time of this sequence is obtained in Step 1(f) as

$$CT = 20 \times 7 + 38 \times 6 + 47 \times 5 + 65 \times 4 + 79 \times 3 + 80 \times 2 + 86 \times 1 = 1,346 \text{ time units.}$$

In Step 2(a) of Algorithm-2, we set $c = 1$ and consider O_1 as it is the first customer order in the sequence $\sigma_{STT}(O'' \cup O'_R) = \{O_1, O_5, O_4, O_3\}$. That is, $O_{[1]} = O_1$ since $c = 1$. The job having the maximum setup from the set of jobs in this customer order is J_2 , which is assigned to the last job position of the customer order O_1 . In Step 2(c), no need to move O_1 to position 1 of the sequence $\{O_1, O_5, O_4, O_3\}$ since O_1 is already in the first position. Among the customer orders in positions 2 through 4 of the sequence $\{O_1, O_5, O_4, O_3\}$, the customer order having the minimum total time with job J_2 is determined as O_4 , which is in position 3 (i.e., $c = 3$). In Step 2(e), job J_2 is assigned to the first job position of the customer order O_4 . In Step 2(b), job J_5 is determined as the job having the maximum setup time from the set of unassigned jobs J_1 and J_5 in the customer order O_4 , and job J_5 is assigned to the last job position of the customer order O_4 . In Step 2(c), the customer order O_4 is moved to position 2 of the sequence $\{O_1, O_5, O_4, O_3\}$, and the new sequence becomes $\{O_1, O_4, O_5, O_3\}$. Among the customer orders in positions 3 and 4 of the new sequence $\{O_1, O_4, O_5, O_3\}$, the customer order having the minimum total time with job J_5 is determined as O_3 , which is in position 4 (i.e., $c = 4$). In Step 2(e), job J_5 is assigned to the first job position of the customer order O_3 . In Step 2(b), job J_5 is determined as the job having the maximum setup time from the set of unassigned jobs J_4 and J_8 in the customer order O_3 , and job J_8 is assigned to the last job position of the customer order O_3 . In Step 2(c), the customer order O_3 is moved to position 3 of the sequence $\{O_1, O_4, O_5, O_3\}$, and the new sequence becomes $\{O_1, O_4, O_3, O_5\}$. The customer order in position 4 (i.e., $c = 4$) in the new sequence $\{O_1, O_4, O_3, O_5\}$ is O_5 , and it is the only customer order having the minimum total time with J_8 . In Step 2(e), job J_8 is assigned to the first job position of the customer order O_5 . The improved sequence $\{O_1, O_4, O_3, O_5\}$ obtained by Step 2 for the set $O'' \cup O'_R = \{O_1, O_3, O_4, O_5\}$ is illustrated by Table 5.7.

Table 5.7 Sequence $\{O_1, O_4, O_3, O_5\}$ obtained for the set $O'' \cup O'_R$ in Step 2

Customer Order Sequence	O_1		O_4			O_3			O_5	
Job Sequence	J_1	J_2	J_2	J_1	J_5	J_5	J_4	J_8	J_8	J_7
Modified Total Time	38		48			63			27	

The final sequence of all customer orders and jobs is $\sigma_F = \{O_7, O_1, O_4, O_3, O_5, O_6, O_2\}$, which is same as the sequence found by Algorithm-1 and illustrated by Table 5.6. The associated total completion time of this sequence is

$$CT_F = 20 \times 7 + 38 \times 6 + 48 \times 5 + 63 \times 4 + 27 \times 3 + 80 \times 2 + 86 \times 1 = 1,187 \text{ time units.}$$

Our third proposed heuristic algorithm consists of three steps. First two steps of Algorithm-3 are same as the first two steps of Algorithm-1. However, in the third step of Algorithm-3, the pair-wise interchange starts from the customer order in the last position and customer orders are moved backward instead of forward. This moving direction has an advantage over Algorithm-1 when processing times are longer than setup times.

Algorithm-3:

Step 1. [Initial Schedule Generation without Setup Savings] Apply Step 1 of Algorithm-1.

Step 2. [Improved Schedule with Setup Savings] Apply Step 2 of Algorithm-1.

Step 3. [Improved Schedule by Pair-wise Interchange of Customer Orders]

- (a) Set $i = K_U$ and $l = K_U - 1$.
- (b) If $i = 1$ then go to Step 3(f); otherwise, let $O_{[i]}$ and $O_{[l]}$ are two customer orders in positions i and l of the initial feasible sequence σ_I obtained in Step 1.
- (c) If there is at least one unassigned common job in the pair of the customer orders $O_{[i]}$ and $O_{[l]}$, then

- Determine a common job having the maximum setup time from the set of unassigned common jobs.
- Assign the common job to the first and the last job positions of the customer orders $O_{[i]}$ and $O_{[l]}$, respectively.
- Go to Step 3(e).

Otherwise, Go to Step 3(d).

(d) If there is a common job, which is the single job, in the customer order $O_{[i]}$ or $O_{[l]}$, then

- If the common job is the single job of the customer order $O_{[l]}$, then assign the common job to the first job position of the customer order $O_{[i]}$; otherwise (i.e. the common job is the single job of the customer order $O_{[i]}$), assign the common job to the last job position of the customer order $O_{[l]}$.
- Go to Step 3(e).

Otherwise, set $l = l - 1$ and Go to Step 3(b).

(e) Pairwise interchange the position of the string of all customer orders between positions i and l with the position of the string of customer orders starting at position l and ending at position k .

- Set $i = i - 1 + k - l$, $l = i - 1$. If $l = 1$, then set $i = i - 1$.
- Go to Step 3(b).

(f) Let σ_{II} be the second improved schedule of all customer orders. Calculate the associated total completion time CT_{II} of the schedule σ_{II} .

Step 4. [Final schedule] Apply Step 4 of Algorithm-1.

Example 4 Consider the previous example. First two steps of Algorithm-3 are same as the ones in Algorithm-1. Therefore, the improved sequence for the set $O'' \cup O'_R = \{O_1, O_3, O_4, O_5\}$, and the associated total completion time CT_I of all customer orders is

$$CT_I = 20 \times 7 + 38 \times 6 + 47 \times 5 + 65 \times 4 + 63 \times 3 + 80 \times 2 + 86 \times 1 = 1,298 \text{ time units.}$$

In Step 3, set $i = 4$ and $l = 3$. Since these two customer orders are linked, indices i and l are updated until $i = 3$ and $l = 1$. The customer order O_4 in position 3 ($i = 3$) has two unassigned common jobs J_1 and J_2 with the customer order O_1 in position ($l = 1$). The job having the maximum setup time from the set of common jobs is J_2 , and it is assigned to the first and the last positions of the customer orders O_4 and O_1 , respectively. Positions of customer orders O_1 and O_5 are interchanged. Thus, the modified sequence of the orders becomes $O_5 - O_1 - O_4 - O_3$. The new values for indices i and l become $i = i - 1 + k - l = 3 - 1 + 1 - 1 = 2$ and $l = i - 1 = 2 - 1 = 1$, respectively. This pair of customer orders has no common job. Hence, i becomes 1 and Step 3 is completed. The improved sequence obtained for the set $O'' \cup O'_R = \{O_5, O_1, O_4, O_3\}$ is illustrated by Table 5.8, and the associated total completion time CT_{II} of all customer orders is

$$CT_{II} = 20 \times 7 + 47 \times 6 + 38 \times 5 + 48 \times 4 + 63 \times 3 + 80 \times 2 + 86 \times 1 = 1,239 \text{ time units.}$$

Table 5.8 Sequence $\{O_5, O_1, O_4, O_3\}$ obtained for the set $O'' \cup O'_R$ in Step 3

Customer Order Sequence	O_5		O_1		O_4			O_3		
Job Sequence	J_7	J_8	J_1	J_2	J_2	J_1	J_5	J_5	J_4	J_8
Total Time	47		38		48			63		

From Step 4, we obtain the best (final) sequence $\sigma_F = \{O_7, O_5, O_1, O_4, O_3, O_6, O_2\}$ of all customer orders and jobs with a total completion time of $CT_F = 1,239$ time units (equivalent to the optimal total completion time value) since $CT_{II} = 1,239 < CT_I = 1,298$. Table 5.9 illustrates the final sequence.

Table 5.9 Final sequence σ_F of all customer orders and jobs in Example 4

Customer Order Sequence	O_7	O_5		O_1	O_4			O_3		O_6		O_2			
Job Sequence	J_{10}	J_7	J_8	J_1	J_2	J_2	J_1	J_5	J_5	J_4	J_8	J_3	J_6	J_9	J_{11}
Modified Total Time	20	47		38	48			63		80		86			

Initial step of Algorithms 1, 2 and 3, which is a decomposition of customer orders into subsets, is common, but improvement step of these algorithms differs from each other. Figure 5.1 illustrates the flow of same and different steps of these algorithms.

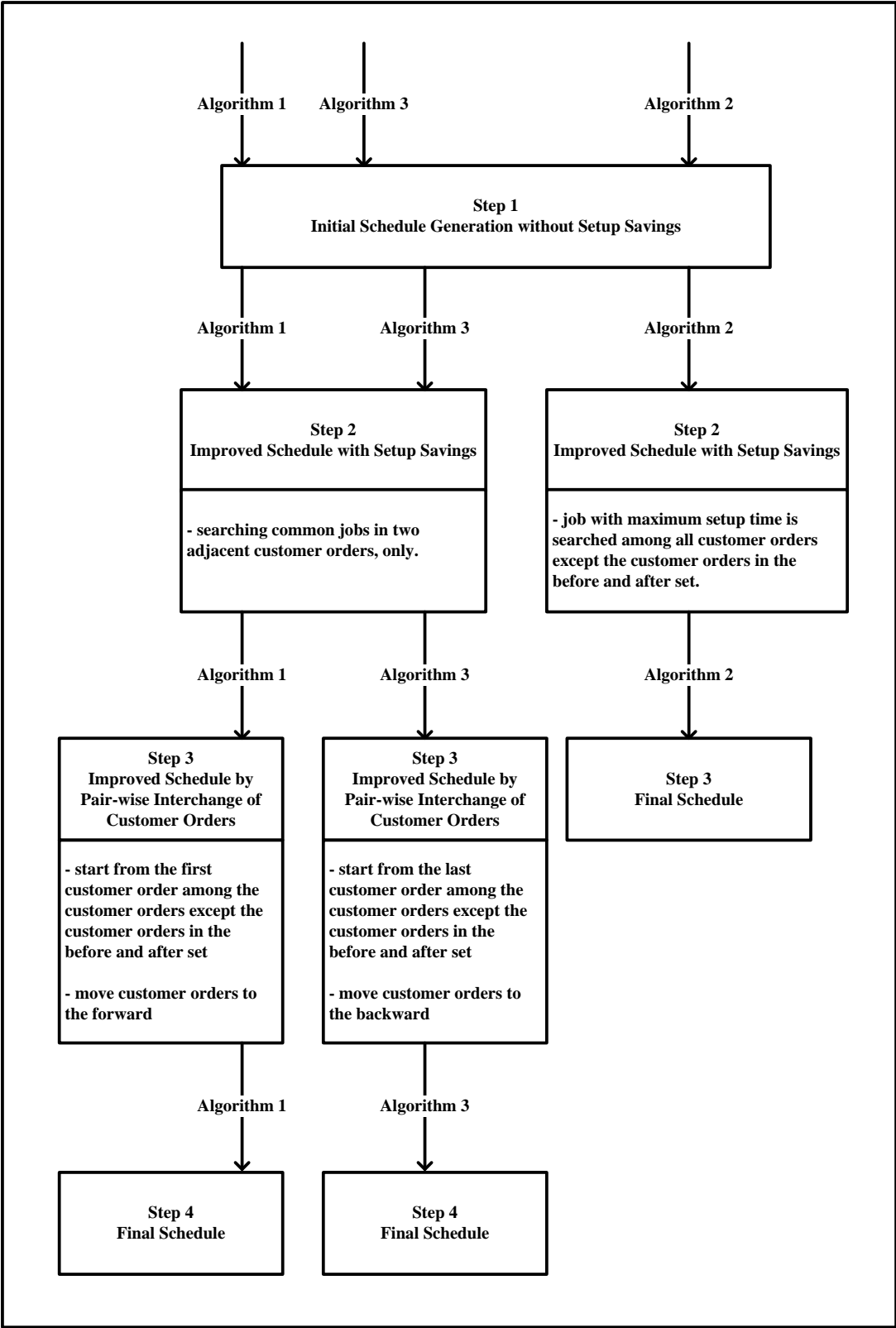


Figure 5.1 Flowchart Diagram for the Heuristic Algorithms 1, 2 and 3

Finally, an algorithm is proposed to find the best solution among the solutions of previously presented algorithms.

Algorithm-4:

Step 1. Let CT_a be the total completion time obtained by Algorithm a .

Run Algorithms 1, 2, and 3.

Step 2. Calculate $CT_{Best} = \min\{CT_1, CT_2, CT_3\}$, and stop.

Example 5 Consider Example 2. Application of Algorithm-4 gives the following steps.

Step 1. Algorithms 1, 2, and 3 give solutions with $CT_1 = 1,187$ time units, $CT_2 = 1,187$ time units and $CT_3 = 1,239$ time units, respectively.

Step 2. Total completion time is calculated as

$$CT_{Best} = \min\{CT_1, CT_2, CT_3\} = \min\{1,187; 1,187; 1,239\} = 1,187$$

CHAPTER 6

COMPUTATIONAL EXPERIMENTS

In this chapter, we describe our computational tests to evaluate the effectiveness and efficiency of the MILP model and the proposed heuristic algorithms in finding the optimal schedules for the total completion time minimization problem. The mathematical model is coded in GAMS 22.6 and solved by using CPLEX 11.0. The proposed heuristic algorithms are coded in C++. All computational experiments are conducted on a personal computer with Intel Core i7 Dual-Core 2.20 GHz CPU and 4 GB RAM under Windows 7 operating system.

6.1 Computational Setting for Test Problems

The values of the parameters used in our experiments are generated as follows:

1. *Number of customer orders (K)*: 5, 10, 15, 20
2. *Number of job types (N)*: 5, 10, 15, 20
3. *Number of jobs within each customer order*: They are generated from four discrete uniform distributions

Variable: $DU[1, N]$

Constant: $DU[2, N-1]$

To measure the effectiveness of the heuristic algorithms for the cases in which an optimal solution is obtained by the MILP model, we calculate the percent deviation of the total completion time obtained by each heuristic algorithm from the total completion time of the optimal solution. Let PD^O be this percent deviation, which can be calculated by

$$PD^O = \frac{TC^H - TC^O}{TC^H} \times 100 \quad (16)$$

where TC^H = Total completion time of the solution obtained by the heuristic algorithm, and

TC^O = Total completion time of the optimal solution obtained by the MILP model.

Similarly, for the cases in which an optimal solution is not guaranteed (but a best-integer solution exists) by the MILP model, we calculate the percent deviation of the total completion time obtained by each heuristic algorithm from the total completion time of the best-integer solution. Let PD^B be this percent deviation, which can be calculated by

$$PD^B = \frac{TC^H - TC^B}{TC^H} \times 100 \quad (17)$$

where TC^B = Total completion time of the best integer solution obtained by the MILP model.

The efficiency measure of the MILP model and the heuristic algorithms is the computational time required to solve the problem. The computational time for the heuristic algorithms was not measured since it was relatively very small, less than 1 second, for all heuristic algorithms. Also note that the computational time required for solving a problem instance increases as the number of customer orders and the number of job types increases. But the computational time is again very small which is less than 1 second generally.

6.3 Discussion of the Results

In this section the performances of our solution approaches are discussed. We first examine the performance of the MILP model, and then discuss the performances of the heuristic algorithms.

6.3.1 Performance of the MILP Model

The performance of the MILP model is given in the Table 6.1. From this table, it is clear that all problem instances can be solved optimally when the number of customer orders is five. However, among the problem instances having ten customer orders there is only one problem instance that cannot be solved optimally. As mentioned earlier, the number of optimally solved problem instances decreases to 249 and 95 for the set of problems having fifteen and twenty customer orders, respectively.

Table 6.1 Performance of the MILP model

K	TOTAL NUMBER OF PROBLEM INSTANCES CONSIDERED	MILP		
		NUMBER OF OPTIMUM INTEGER SOLUTIONS OBTAINED	NUMBER OF BEST INTEGER SOLUTIONS OBTAINED	NUMBER OF UNSOLVED PROBLEM INSTANCES
5	320	320	0	0
10	320	319	1	0
15	320	249	65	6
20	320	95	175	50

The detailed analysis on the number of optimal solutions and best-integer solutions obtained by the MILP model for different combinations of the processing and setup times are given in Table 6.2.

To emphasize the performance of MILP model, we should investigate the quality of solutions which are not optimal. It is a common phenomenon that MILP model ends up with a gap between solution found and the best possible. Therefore, gap values are examined in order to indicate the percentage difference of integer solution from the theoretical optimum. Gap values are analyzed for 240 nonoptimally solved problem instances under three circumstances; best case, worst case, and average case.

Table 6.2 Number of optimally solved problems by the MILP model

K	N	TOTAL NUMBER OF PROBLEM INSTANCES CONSIDERED	RANDOM												CONSTANT																			
			LOW						HIGH						LOW						HIGH													
			LL	LH	HL	HH	LL	HH	LL	LH	HL	HH	LL	HH	LL	LH	HL	HH	LL	LH	HL	HH												
NUMBER OF SOLVED	NUMBER OF OPTIMAL	NUMBER OF SOLVED	NUMBER OF OPTIMAL	NUMBER OF SOLVED	NUMBER OF OPTIMAL	NUMBER OF SOLVED	NUMBER OF OPTIMAL	NUMBER OF SOLVED	NUMBER OF OPTIMAL	NUMBER OF SOLVED	NUMBER OF OPTIMAL	NUMBER OF SOLVED	NUMBER OF OPTIMAL	NUMBER OF SOLVED	NUMBER OF OPTIMAL	NUMBER OF SOLVED	NUMBER OF OPTIMAL	NUMBER OF SOLVED	NUMBER OF OPTIMAL	NUMBER OF SOLVED	NUMBER OF OPTIMAL													
5	5	80	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5											
	10	80	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5										
	15	80	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5									
	20	80	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5									
10	5	320	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20									
	10	80	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5									
	15	80	5	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5									
	20	80	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5									
15	5	320	20	19	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20									
	10	80	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5									
	15	80	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5									
	20	80	5	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5									
20	5	320	20	19	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20									
	10	80	5	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5									
	15	80	5	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5									
	20	80	2	0	4	0	2	0	3	0	2	0	3	0	2	0	3	0	2	0	3	0	2	0	3	0								
		320	17	10	19	3	16	7	19	4	18	12	17	5	18	7	11	7	16	2	16	0	14	4	18	1	16	5	16	1	17	6	16	2

For some of the problem instances, so many iterations are done and integer solutions found become closer to the theoretical optimum after each iteration. However, GAMS is terminated because of time limitation before reaching the optimum solution. But, this case is the best case since until 3-hour time limitation is completed, gap values are very close to zero. In the worst case analysis, we focused on the problem instances whose solution procedure (branching) is terminated due to memory errors that occur after few iterations are completed. In this case, integer solutions found are very raw, hence gap values are high. Maximum gap value, which represents the worst case, is found as 22.92%. On the other hand, for some problem instances branching becomes very difficult and time consuming. When branching is slow, the number of iterations is moderate which leads to higher gap values than the best case and lower gap values than the worst case. In the average case analysis, gap values are found as 0.56% on the average.

6.3.2 Performance of the Heuristic Algorithms

In this section, we discuss the effects of changes in the problem parameters on the performance of heuristic algorithms. To understand the comparison tables given in the following pages, we use abbreviations which are given in Appendix A. For the individual performances of the heuristic algorithms, we first report the number of times each heuristic gives the optimal solution in Table 6.3.

Table 6.3 Number of optimally solved problems by the MILP model and heuristic algorithms

K	NUMBER OF PROBLEM	MILP		ALG 1	ALG 2	ALG 3	ALG 4
		OPTIMAL	BEST INTEGER	OPTIMAL	OPTIMAL	OPTIMAL	OPTIMAL
5	320	320	0	202	108	205	232
10	320	319	1	93	52	96	113
15	320	249	65	42	37	47	65
20	320	95	175	7	6	5	10

While we are analyzing the performance of heuristic algorithms, we focused on the performance Algorithm 4. Many comparisons shown in this section is based on Algorithm 4 because we know that the performance of Algorithm 4 is better than other three algorithms, since Algorithm 4 takes the best solution among the solutions of Algorithm 1, 2 and 3. For each combination of experiment parameters, 5 problem instances are generated and we take the average of percent deviations of these 5 problem instances. Table 6.4 shows the percent deviations of Algorithm 4 from the optimal solution for 640 problem instances where number of jobs in each customer order is variable. For instance, the average percent deviation of 5 problem instances is calculated as 1.887 when the number of customer order is 5, the number of job type is 5, processing times are short, mean and variance of setup times are low and the number of jobs within each customer order is variable. Empty cells in Table 6.4 indicate that the average percent deviation cannot be calculated for the related problem sets since MILP solutions do not exist for those problem sets due to lack of memory.

Table 6.4 Average percent deviation of Algorithm 4 from the optimal solution for VARIABLE case

K	N	VARIABLE										
		SHORT					LONG					AVG
		LL	LH	HL	HH	AVG	LL	LH	HL	HH	AVG	
5	5	1.887	2.365	0.000	0.000	1.063	0.000	0.660	0.846	0.341	0.462	0.762
	10	0.066	0.000	1.008	0.000	0.268	0.000	0.000	0.000	0.038	0.009	0.139
	15	0.700	0.948	0.835	0.000	0.621	0.000	0.029	0.000	0.003	0.008	0.314
	20	0.000	0.437	0.015	0.224	0.169	0.000	0.008	0.000	0.000	0.002	0.085
AVG_5		0.663	0.937	0.464	0.056	0.530	0.000	0.174	0.211	0.096	0.120	0.325
10	5	5.707	2.869	1.843	2.826	3.311	0.740	0.526	0.826	1.556	0.912	2.112
	10	1.157	1.716	2.747	1.632	1.813	0.238	0.227	0.212	0.389	0.266	1.040
	15	0.910	0.175	1.531	1.025	0.910	0.045	0.000	0.000	0.050	0.024	0.467
	20	1.323	2.062	0.436	1.776	1.400	0.042	0.016	0.001	0.187	0.062	0.731
AVG_10		2.274	1.706	1.639	1.815	1.858	0.266	0.192	0.260	0.545	0.316	1.087
15	5	3.498	2.209	1.876	5.908	3.373	0.340	1.156	1.449	0.881	0.956	2.165
	10	1.736	2.787	2.856	1.885	2.316	0.223	0.338	0.373	0.636	0.392	1.354
	15	1.321	0.793	1.388	1.229	1.183	0.152	0.090	0.064	0.139	0.112	0.647
	20	1.182	3.902	0.559	0.693	1.584	0.023	0.127	0.014	0.156	0.080	0.832
AVG_15		1.934	2.423	1.670	2.429	2.114	0.185	0.428	0.475	0.453	0.385	1.249
20	5	0.943	1.873	1.271	0.849	1.234	1.077	0.515	1.588	0.763	0.986	1.110
	10	1.602	1.266	2.825		1.898	0.126	0.271	0.774	0.609	0.445	1.171
	15	3.404		2.519		2.961	0.103	0.107	0.630	0.338	0.295	1.628
	20					0.000					0.000	0.000
AVG_20		1.983	1.570	2.205	0.849	1.652	0.435	0.297	0.997	0.570	0.575	1.113
AVG_TOTAL		1.714	1.659	1.495	1.287	1.539	0.221	0.273	0.486	0.416	0.349	0.944

Table 6.4 shows that average percent deviations of the solutions obtained by Algorithm 4 when the processing times are short (denoted by SHORT) are greater than the solutions obtained when the processing times are long (denoted by LONG). Figure 6.1 illustrates the difference between the averages of SHORT and LONG for each number of customer orders. From figure, percent deviations of LONG for each number of customer order is significantly less than percent deviations of SHORT. This situation is expected, because, in LONG problem instances, processing times are relatively greater than setup times. Hence, the effect of setup reduction is less significant than SHORT case since the process times are very small relative to setup times. This behavior is valid for the problem instances in which the customer orders have constant number of jobs.

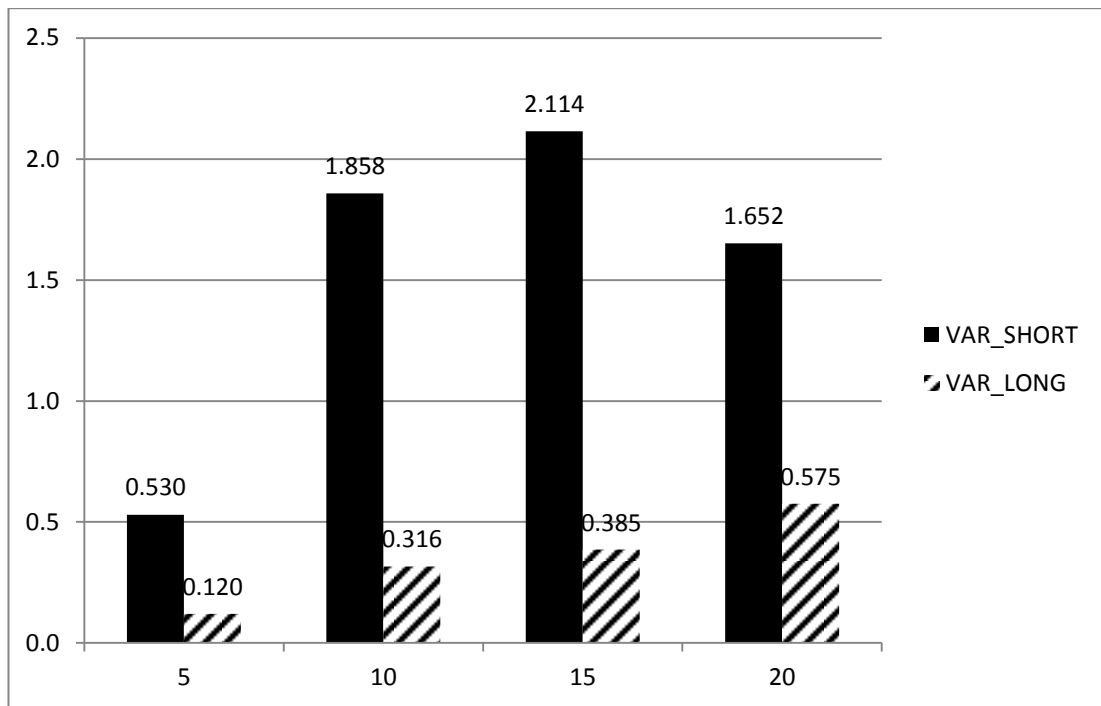


Figure 6.1 Difference between the average percent deviations

A similar table is constructed for the problem instances in which the number of jobs in customer orders is taken as constant. As can be seen in Table 6.5, many of the problem instances cannot be solved when the number of customer orders is twenty.

Table 6.5 Average Percent deviation of Algorithm 4 from optimal solutions for CONSTANT case

K	N	CONSTANT										
		SHORT					LONG					AVG
		LL	LH	HL	HH	AVG	LL	LH	HL	HH	AVG	
5	5	1.487	0.974	1.472	0.055	0.997	0.061	0.273	0.010	0.008	0.088	0.542
	10	0.155	0.077	2.407	2.096	1.184	0.201	0.175	0.151	0.474	0.250	0.717
	15	0.170	0.529	3.739	0.828	1.316	0.162	0.018	0.040	0.215	0.109	0.713
	20	2.504	0.817	1.344	0.314	1.245	0.013	0.100	0.180	0.098	0.098	0.671
	AVG_5	1.079	0.599	2.241	0.823	1.186	0.109	0.142	0.095	0.199	0.136	0.661
10	5	0.024	0.153	0.744	1.129	0.513	0.317	0.125	0.027	0.891	0.340	0.426
	10	0.152	0.403	0.000	1.303	0.465	0.317	0.528	0.514	0.392	0.438	0.451
	15	0.275	0.693	3.490	0.717	1.294	0.284	0.144	0.260	0.381	0.267	0.780
	20	0.539	1.831	0.344	1.620	1.084	0.079	0.240	0.145	0.099	0.141	0.612
	AVG_10	0.248	0.770	1.145	1.192	0.839	0.249	0.259	0.236	0.441	0.296	0.568
15	5	0.069	0.000	0.318	0.523	0.228	0.057	0.284	0.128	0.754	0.306	0.267
	10	0.031	0.000	1.872	1.985	0.972	0.148	0.170	0.309	0.096	0.181	0.576
	15	0.049	0.021	0.854	0.011	0.233	0.017	0.090	0.131	0.013	0.063	0.148
	20	0.163		0.259	0.032	0.151	0.087	0.007	0.158	0.003	0.064	0.108
	AVG_15	0.078	0.007	0.826	0.638	0.387	0.077	0.138	0.182	0.216	0.153	0.270
20	5	0.000				0.000			0.420		0.420	0.210
	10	0.062		0.029	0.006	0.032	0.002	0.001	0.017	0.012	0.008	0.020
	15					0.000	0.021		0.400		0.210	0.105
	20					0.000					0.000	0.000
	AVG_20	0.031	0.000	0.029	0.006	0.017	0.012	0.001	0.279	0.012	0.076	0.046
AVG_TOTAL	0.359	0.344	1.060	0.665	0.607	0.112	0.135	0.198	0.217	0.165	0.386	

Percent deviations of the solutions obtained by Algorithm 4 when processing times are short are greater than percent deviations when processing times are long except for twenty customer orders. When the number of customer orders is twenty and the number of jobs in each customer order is constant, most of the problem instances cannot be solved optimally. Hence, for those problem instances, analysis of Algorithm 4's performance, which is based on its percent deviation from optimal solution, would not be enough and may lead to wrong results. Thus, the best integer solutions obtained by the MILP model should be considered for comparison when the optimal solution does not exist. Table 6.6 shows the percent deviations of Algorithm 4 solutions from the best integer solutions obtained by the MILP model.

Table 6.6 Average percent deviation of Algorithm 4 from the best integer solutions for CONSTANT case

K	N	CONSTANT										
		SHORT					LONG					AVG
		LL	LH	HL	HH	AVG	LL	LH	HL	HH	AVG	
20	5	0.908	-0.064	2.300	-0.206	0.734	-0.003	-0.031	0.115	0.286	0.092	0.413
	10	-0.030	-1.734	0.004	-0.622	-0.596	1.000	-0.198	0.329	0.133	0.316	-0.140
	15	0.326	-1.776	-1.336	-3.528	-1.578	-0.048	-0.146	1.734	-0.044	0.374	-0.602
	20	-0.992	-3.169	*	2.163	-0.666	-0.732	0.758	0.143	-1.737	-0.392	-0.529
	AVG_20	0.053	-1.686	0.323	-0.548	-0.465	0.054	0.096	0.580	-0.341	0.097	-0.184

From Figure 6.2, it is clear that Algorithm 4 gives better solutions than best integer solutions, and the percent deviations obtained for the case, in which processing times are short, are significantly smaller than those obtained for the case, in which processing times are long, since the total completion time becomes sensitive to the setup reduction when processing times are relatively smaller than the setup times. The solutions obtained by Algorithm 4 are compared with the best integer solutions obtained by the MILP model are given in detail in Appendix A.

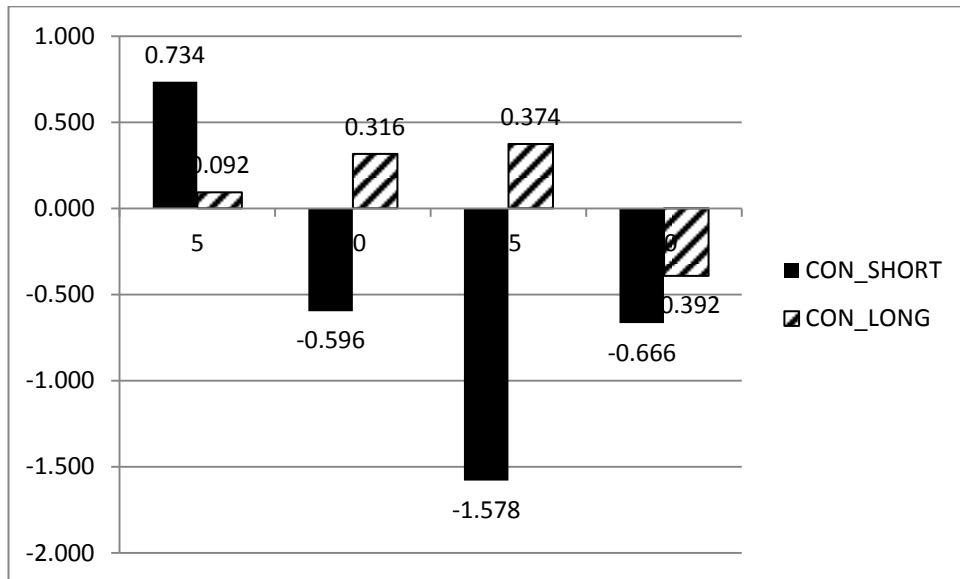


Figure 6.2 Percent deviation of Algorithm 4 with respect to $K = 20$

We also investigate the behavior of Algorithm 4 for each case of different number of customer orders. In Figure 6.3, the behavior of algorithm is presented for 5-order case, based on the number of jobs within each customer order and the processing times.

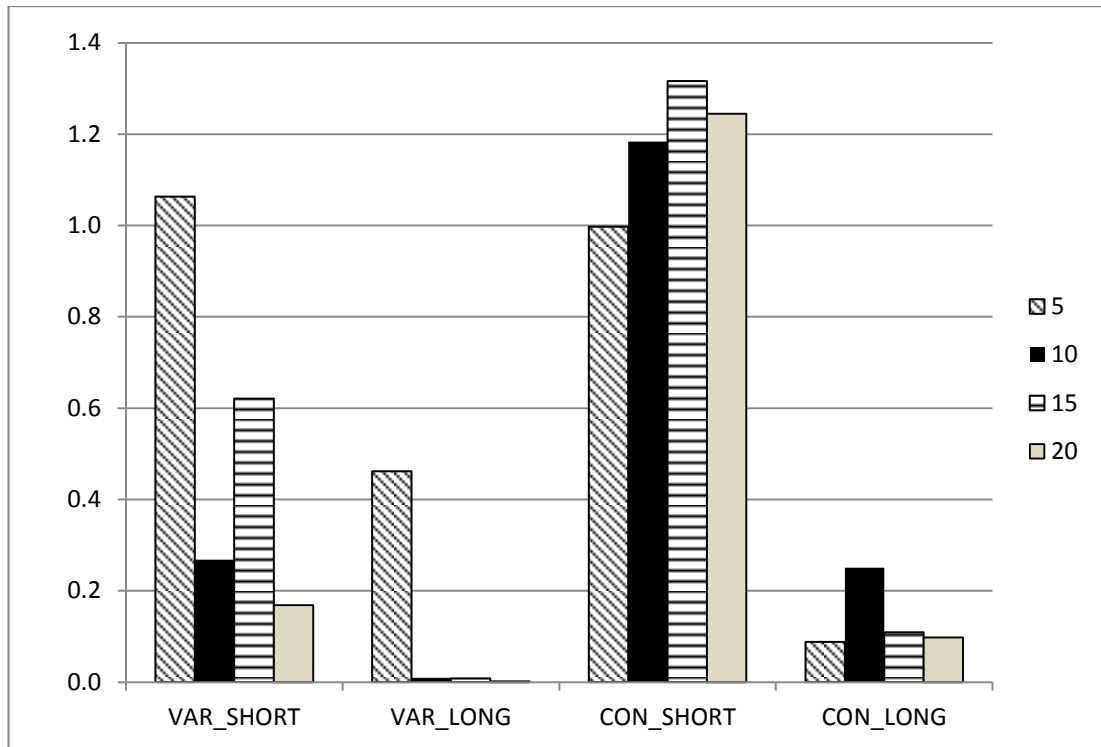


Figure 6.3 Percent deviations from the optimal for $K = 5$

As can be seen from Figure 6.3, Algorithm 4 efficiently finds optimal and near optimal solutions for the problem instances when processing times are long whether the number of jobs in each order is taken as variable or constant. Furthermore, the solution quality of Algorithm 4 also increases as the number of jobs increases. Figure 6.4 shows the average percent deviations of the solutions obtained by Algorithm 4 from the optimal solution obtained by MILP when the number of jobs in each customer order is variable or constant and the processing times are short and long. One can observe that the percent deviation is dramatically large for the case in which the number of job is five, the number of jobs in each customer order is variable and processing times are short, as compared to the cases where there are 10, 15 or 20 customer orders.

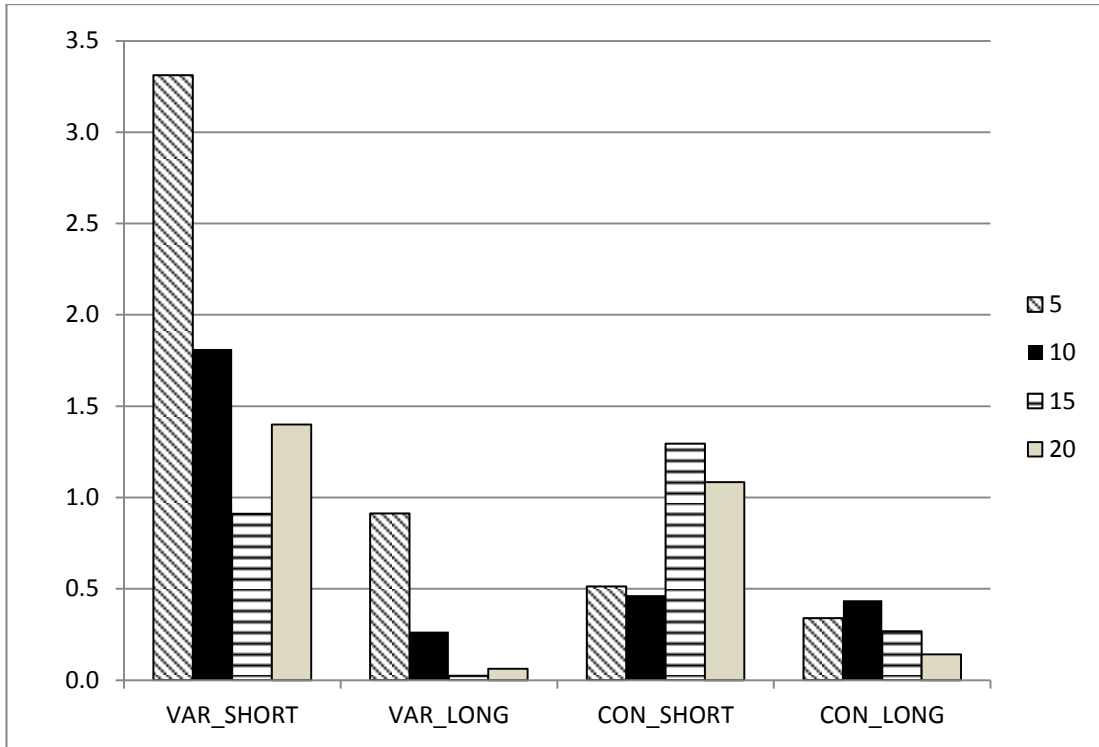


Figure 6.4 Percent deviations from the optimal for $K = 10$

The percent deviations of Algorithm 4 solutions from those obtained by MILP model for problem instances having 15 or 20 customer orders are very similar to the ones obtained for the problem instances with 5 or 10 customer orders, and they are presented in Appendix B and C, respectively. Besides these percent deviation analyses, Figure 6.5 illustrates the general averages of percent deviations for each case of different number of customer orders.

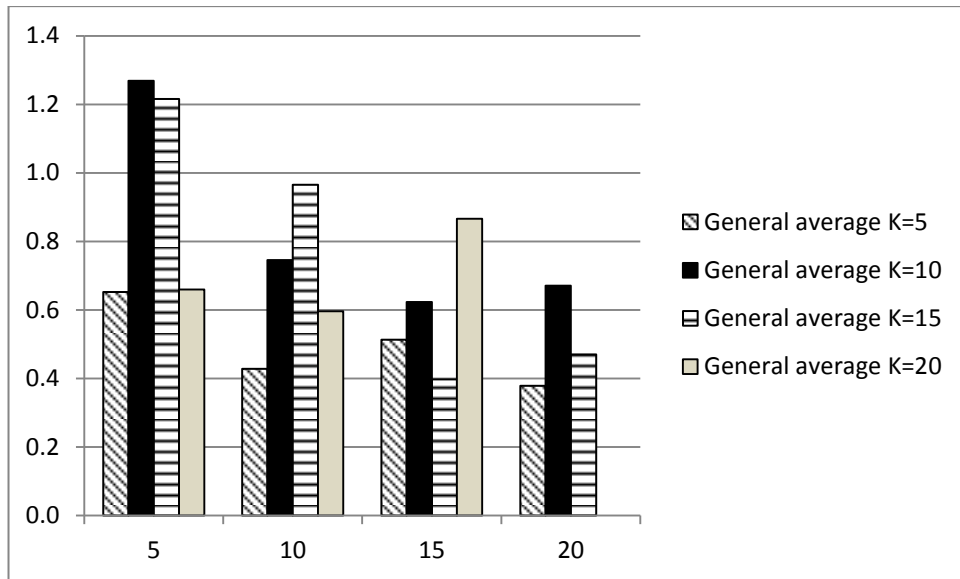


Figure 6.5 General average percent deviations for $K=5$, $K=10$, $K=15$ and $K=20$

As can be seen from Figure 6.5, Algorithm 4 gives better solutions when the number of customer orders is 5. However, no trend or pattern related with the number of customer orders is observed. On the other hand, performance of Algorithm 4 increases while the number of jobs increases. Detailed analysis of average percent deviations obtained for problem instances generated by different combinations of the number of jobs within each customer order and the processing times of the jobs are given in Appendix D.

Table 6.7 shows the performance of the three algorithms when the number of jobs in each customer order is variable. For some problem instances, we observe that Algorithms 1 and 3 give optimal or near optimal solutions whereas Algorithm 2 gives solutions considerably far from the optimum. However, we observe the opposite results, in which the solutions obtained by Algorithms 1 and 3 are considerably far from the optimal solutions, when Algorithm 2 gives optimal or near optimal solutions. This shows us that Algorithm 2 behaves different than the other two algorithms while Algorithms 1 and 3 behave similarly. The performance of Algorithm 2 is worse than the performance of other two algorithms especially when the number of jobs in each customer order is variable.

Table 6.7 Average percent deviation of Algorithms 1, 2 and 3 for VARIABLE case

K	N	VARIABLE								
		SHORT			LONG			AVG_VAR		
		AVG_SHORT			AVG_LONG					
		ALG1	ALG2	ALG3	ALG1	ALG2	ALG3	ALG1	ALG2	ALG3
5	5	3.575	8.760	1.796	0.744	10.280	0.610	2.159	9.520	1.203
	10	1.845	5.830	1.383	0.009	8.384	0.009	0.927	7.107	0.696
	15	0.626	4.630	0.626	0.027	20.427	0.008	0.326	12.529	0.317
	20	0.169	17.159	0.169	0.002	14.356	0.002	0.085	15.758	0.085
AVG_5		1.554	9.095	0.993	0.195	13.362	0.157	0.875	11.228	0.575
10	5	12.203	5.288	10.447	1.060	5.495	0.936	6.632	5.392	5.692
	10	2.662	13.737	2.242	0.443	25.474	0.359	1.553	19.606	1.300
	15	1.102	12.851	1.102	0.059	14.184	0.024	0.580	13.517	0.563
	20	1.833	17.714	1.459	0.062	21.587	0.062	0.947	19.650	0.760
AVG_10		4.450	12.398	3.813	0.406	16.685	0.345	2.428	14.541	2.079
15	5	17.188	3.784	13.896	1.731	4.631	1.400	9.460	4.208	7.648
	10	3.807	13.086	3.679	0.584	8.041	0.461	2.195	10.564	2.070
	15	1.584	12.203	1.321	0.268	17.195	0.211	0.926	14.699	0.766
	20	1.939	21.889	1.588	0.165	22.011	0.080	1.052	21.950	0.834
AVG_15		6.130	12.740	5.121	0.687	12.970	0.538	3.408	12.855	2.829
20	5	13.667	1.234	11.980	2.014	2.495	1.644	7.841	1.864	6.812
	10	4.922	4.983	2.902	0.597	10.318	0.463	2.759	7.650	1.682
	15	3.642	21.151	3.248	0.395	14.102	0.324	2.018	17.626	1.786
	20									
AVG_20		9.345	5.455	8.096	1.002	8.972	0.810	5.174	7.213	4.453
AVG_TOTAL		5.370	9.922	4.506	0.573	12.997	0.463	2.971	11.459	2.484

In contrary, as it can be seen in Table 6.8, the difference among the percent deviations of the solutions obtained by all algorithms is not significant when number of jobs in each customer order is constant.

Table 6.8 Average percent deviation of Algorithms 1, 2 and 3 for CONSTANT case

K	N	CONSTANT									AVG_TOTAL		
		SHORT			LONG			AVG_CON					
		AVG_SHORT			AVG_LONG								
		ALG1	ALG2	ALG3	ALG1	ALG2	ALG3	ALG1	ALG2	ALG3	ALG1	ALG2	ALG3
5	5	1.809	2.157	1.809	0.327	0.381	0.327	1.068	1.269	1.068	1.614	5.394	1.136
	10	2.380	2.072	2.725	0.357	0.673	0.300	1.369	1.372	1.513	1.148	4.240	1.104
	15	1.663	2.464	2.112	0.144	1.031	0.152	0.903	1.748	1.132	0.615	7.138	0.724
	20	1.921	2.940	2.003	0.113	0.637	0.205	1.017	1.789	1.104	0.551	8.773	0.595
AVG_5		1.943	2.408	2.162	0.235	0.681	0.246	1.089	1.544	1.204	0.982	6.386	0.890
10	5	3.379	0.696	3.672	0.732	0.562	0.737	2.055	0.629	2.205	4.344	3.010	3.948
	10	1.175	0.751	1.890	0.719	0.678	0.950	0.947	0.715	1.420	1.250	10.160	1.360
	15	2.005	1.529	2.195	0.374	0.379	0.458	1.190	0.954	1.326	0.885	7.236	0.945
	20	1.376	2.117	2.149	0.141	0.593	0.243	0.758	1.355	1.196	0.853	10.503	0.978
AVG_10		1.984	1.273	2.476	0.492	0.553	0.597	1.238	0.913	1.537	1.833	7.727	1.808
15	5	0.673	0.286	0.836	0.700	0.335	0.899	0.686	0.311	0.867	5.073	2.259	4.258
	10	2.164	0.979	4.105	0.363	0.286	0.454	1.264	0.632	2.280	1.729	5.598	2.175
	15	0.451	0.383	0.659	0.075	0.094	0.075	0.263	0.239	0.367	0.595	7.469	0.567
	20	0.479	0.157	0.477	0.222	0.329	0.075	0.351	0.243	0.276	0.701	11.097	0.555
AVG_15		0.914	0.442	1.492	0.340	0.261	0.376	0.627	0.352	0.934	2.018	6.603	1.882
20	5	0.000	0.000	0.000	0.826	0.420	0.826	0.413	0.210	0.413	4.127	1.037	3.613
	10	0.213	0.033	0.359	0.051	0.019	0.051	0.132	0.026	0.205	1.446	3.838	0.944
	15				0.306	0.363	0.349	0.000	0.363	0.349	1.009	8.995	1.067
	20							0.000			0.000		
AVG_20		0.142	0.017	0.251	0.167	0.112	0.174	0.154	0.065	0.212	2.664	3.639	2.333
AVG_TOTAL		1.246	1.035	1.595	0.308	0.402	0.348	0.777	0.718	0.972	1.874	6.089	1.728

In Figure 6.6, average percent deviations of the solutions obtained by Algorithms 1, 2, and 3 from the optimal solutions are analyzed, and the performance of these algorithms are compared. It can be observed that Algorithms 1 and 3 gives better results than Algorithm 2 for all cases in which the number of customer orders is taken as 5, 10, 15 or 20, and their average percent deviations from the optimal solution is less than 3%. The average percent deviations of Algorithms 1, 2 and 3 from the optimal solutions are calculated as 1.874%, 6.089% and 1.728%, respectively.

As a summary, Algorithms 1 and 3 outperform Algorithm 2 for most of the problem instances but Algorithm 2 gives better results for large-sized problem instances especially when the number of jobs ordered by each customer is constant. On the other hand, Algorithm 4 has an average percent deviation of 0.665% over all problem instances since it takes the best solution among Algorithms 1, 2 and 3.

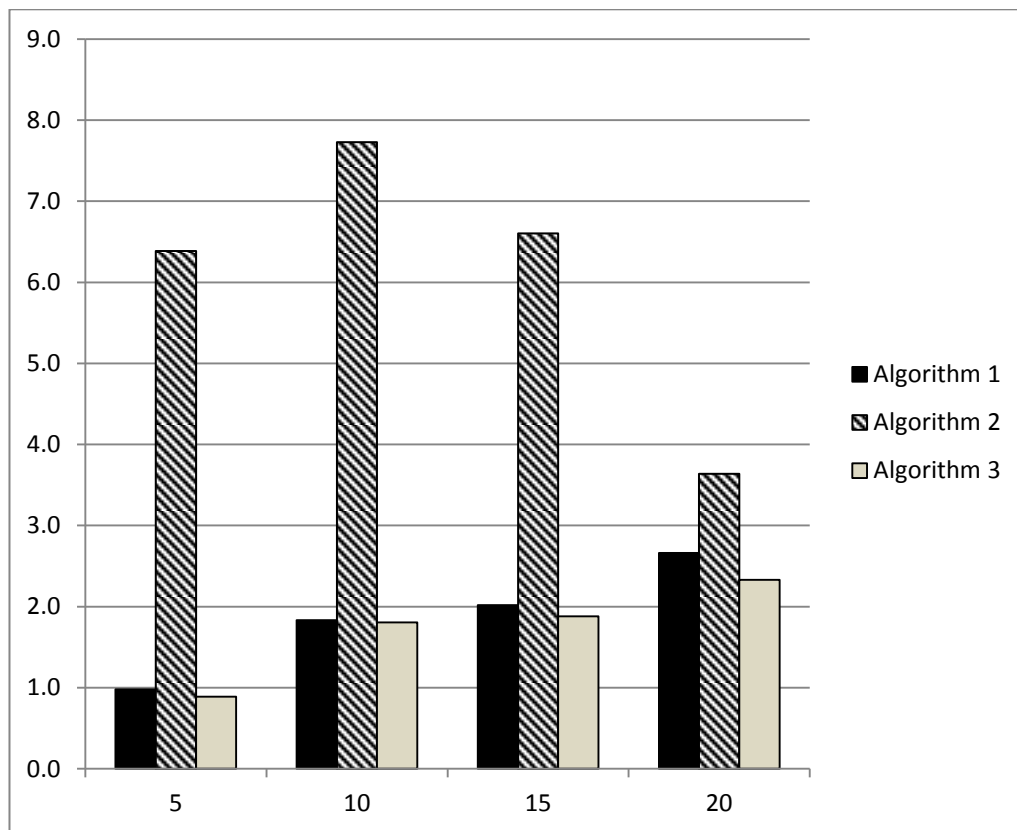


Figure 6.6 General average percent deviations of each algorithm for $K = 5$, $K = 10$, $K = 15$ and $K = 20$

CHAPTER 7

CONCLUSIONS AND FUTURE RESEARCH DIRECTIONS

In this study, a new customer order scheduling problem on a single-machine with job setup times is considered. It is assumed that all jobs in the same customer order are processed successively and delivered to the customer at the same time, and no setup time is necessary before the processing of the first job of a customer order if this first job is the same as the last job of the immediately preceding customer order. We have investigated the problem for two objectives, one is minimizing the makespan and the other one is minimizing the total completion time of customer orders.

For the makespan problem, we have shown that the problem is polynomially solvable. However, for the total completion time problem, we have developed a mathematical programming model and heuristic algorithms that obtain optimal and near-optimal solutions, respectively.

We observed from our experiments that the proposed heuristic algorithms developed for the total completion time minimization problem find promising results as it solves small-and medium-sized problem instances optimally and finds near-optimal solutions for large-sized instances in very short computational time. The results also reveal that solving the problem by a standard MILP solver seems not to be useful alternative, especially for large-sized problem instances.

Order scheduling problems are not yet extensively studied. Thus, there is considerable number of issues remaining open for future research. Several extensions of our study can be investigated. First, development of additional heuristic algorithms, including meta-heuristics could be useful to improve the quality of solutions obtained in this study. Second, the study of the same problem considered in

this study for different problem characteristics, such as ready times, precedence relations among the jobs, and the performance measures concerning due dates of the customer orders, especially the total tardiness, the maximum lateness, and the number of tardy customer orders, would be other extensions. Third, the study of same problem for more complex machining environments such as parallel machines or multiple stages may be another future research issue. Finally, the assumption of non-existence of multiple identical jobs in a customer order may be relaxed, and the future study issues mentioned above would be investigated.

REFERENCES

- ALLAHVERDI, A., & SOROUSH, H. (2008).** The significance of reducing setup times/setup costs. *European Journal of Operational Research* 187, pp. 978-984.
- ALLAHVERDI, A., GUPTA, J. N., & ALDOWAISAN, T. (1999).** A review of scheduling research involving setup considerations. *Omega, International Journal of Management Science* 27, pp. 219-239.
- ALLAHVERDI, A., NG, C., CHENG, T., & KOVALYOV, M. Y. (2008).** A survey of scheduling problems with setup times or costs. *European Journal of Operational Research* 187, pp. 985-1032.
- AZIZOĞLU, M., & WEBSTER, S. (2001).** Scheduling a batch processing machine with incompatible job families. *Computers & Industrial Engineering* 39, pp. 325-335.
- BAKER, K. R., & MAGAZINE, M. J. (2000).** Minimizing maximum lateness with job families. *European Journal of Operational Research* 127, pp. 126-139.
- BOZORGIRAD, M., & LOGENDRAN, R. (2012).** Sequence-dependent group scheduling on unrelated parallel machines. *Expert Systems with Applications* 39, pp. 9021-9030.
- CHEN, W.-J. (2008).** Single-machine scheduling with family setup times in a manufacturing system. *Engineering Optimization* 40, pp. 579-589.
- EREL, E., & GHOSH, J. B. (2007).** Customer order scheduling on a single machine with setup times: Complexity and algorithms. *Applied Mathematics and Computation* 185, pp. 11-18.
- GERODIMOS, A., GLASS, C., POTTS, C. N., & TAUTENHAHN, T. (1999).** Scheduling multi-operation jobs on a single machine. *Annals of Operations Research* 92, pp. 87-105.
- GRUNDEL, S., ÇİFTÇİ, B., BORM, P., & HAMERS, H. (2013).** Family Sequencing and cooperation. *European Journal of Operational Research* 226, pp. 414-424.
- GUPTA, A., & SIVAKUMAR, A. (2005).** Multi-objective scheduling of two-job families on a single machine. *Omega, International Journal of Management Science* 33, pp. 399-405.

- GUPTA, J. N., & CHANTARAVARAPAN, S. (2008).** Single machine group scheduling with family setups to minimize total tardiness. *International Journal of Production Research* 46, pp. 1707-1722.
- GUPTA, J. N., HO, J. C., & VAN DER VEEN, J. A. (1997).** Single machine hierarchical scheduling with customer orders and multiple job classes. *Annals of Operations Research* 70, pp. 127-143.
- HE, Y., & SUN, L. (2012).** Single-machine group scheduling problems with deterioration to minimize the sum of completion times. *Mathematical Problems in Engineering* 2012, pp. 1-9.
- JANIAK, A., KOVALYOV, M. Y., & PORTMANN, M.-C. (2005).** Single machine group scheduling with resource dependent setup and processing times. *European Journal of Operational Research* 162, pp. 112-121.
- LIAEE, M. M., & EMMONS, H. (1997).** Scheduling families of jobs with setup times. *International Journal of Production Economics* 51, pp. 165-176.
- LIU, C.-H. (2010).** A coordinated scheduling system for customer orders scheduling problem in job shop environments. *Expert Systems with Applications* 37, pp. 7831-7837.
- LOGENDRAN, R., GELOGULLARI, C. A., & SRISKANDARAJAH, C. (2003).** Minimizing the mean flow time in a two-machine group-scheduling problem with carryover sequence dependency. *Robotics and Computer Integrated Manufacturing* 19, pp. 21-33.
- MAZDEH, M. M., SARHADI, M., & HINDI, K. S. (2007).** A branch-and-bound algorithm for single machine scheduling with batch delivery minimizing flow times and delivery costs. *European Journal of Operational Research* 183, pp. 74-86.
- NG, D., CHENG, E. T., & KOVALYOV, M. Y. (2004).** Single machine batch scheduling with jointly compressible setup and processing times. *European Journal of Operational Research* 162, pp. 211-219.
- POTTS, C. N., & KOVALYOV, M. Y. (2000).** Scheduling with batching: A review. *European Journal of Operational Research* 120, pp. 228-249.
- SABOUNI, Y. M., & LOGENDRAN, R. (2013).** A single machine carryover sequence-dependent group scheduling in PCB manufacturing. *Computers & Operations Research* 40, pp. 236-247.
- SCHALLER, J. E., & GUPTA, J. N. (2008).** Single machine scheduling with setups to minimize total earliness and tardiness. *European Journal of Operational Research* 187, pp. 1050-1068.
- SCHUTTEN, J., & LEUSSINK, R. (1996).** Parallel machine scheduling with release dates, due dates and family setup times. *International Journal of Production Economics* 46-47, pp. 119-125.

- SMITH, W. (1956).** Various optimizers for single stage production. *Naval Research Logistics Quarterly* 3, pp. 59-66.
- SU, L.-H., CHEN, P.-S., & CHEN, S.-Y. (2013).** Scheduling on parallel machines to minimise maximum lateness for the customer order problem. *International Journal of Systems Science* 44, pp. 926-936.
- VAN DER VEEN, J. A., WOEGINGER, G. J., & ZHANG, S. (1998).** Sequencing jobs that require common resources on a single machine: A solvable case of the TSP. *Mathematical Programming* 82, pp. 235-254.
- WANG, G., & CHENG, E. T. (2007).** Customer order scheduling to minimize total weighted completion time. *Omega, International Journal of Management Science* 35, pp. 623-626.
- WANG, J.-B., HUANG, X., WU, Y.-B., & JI, P. (2012).** Group scheduling with independent setup times, ready times, and deteriorating job processing times. *International Journal of Advanced Manufacturing Technology* 60, pp. 643-649.
- YANG, J. (2011).** Customer order scheduling in a two machine flowshop. *International Journal of Management Science* 17, pp. 95-116.

APPENDICES
APPENDIX A – COMPARISON TABLES

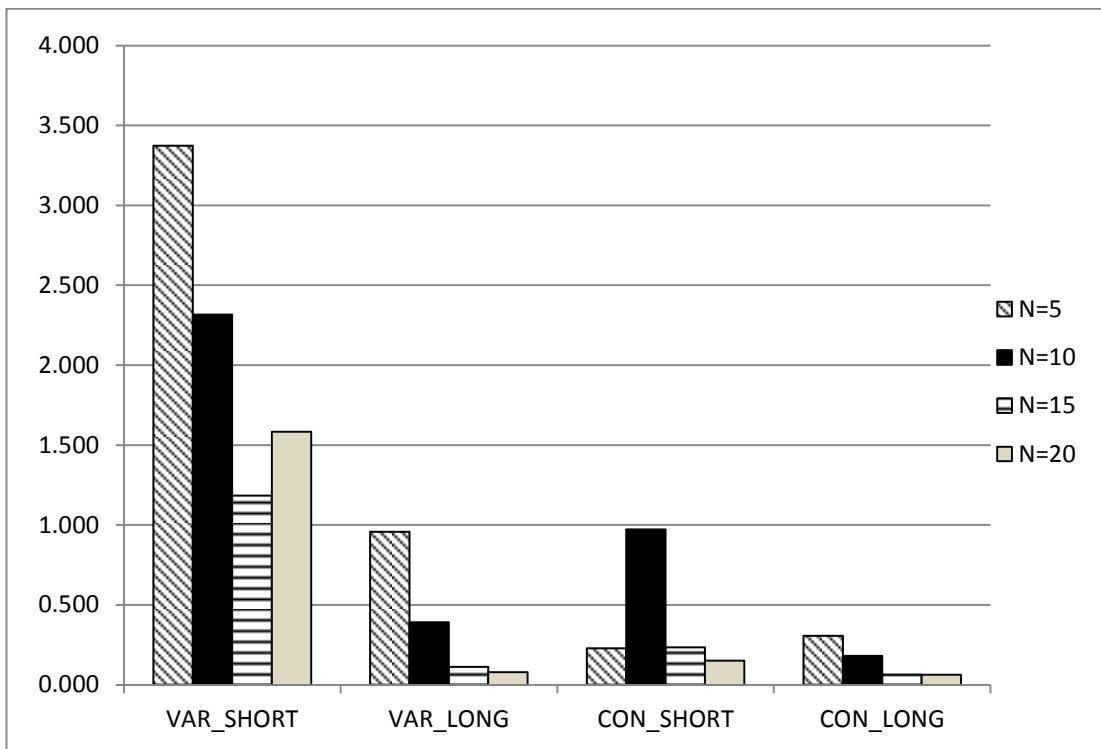
TABLE A. 1 - ABBREVIATIONS USED IN COMPARISON TABLES

ABBREVIATION	EXPLANATION
K	Number of customer orders
N	Number of different jobs
VARIABLE or VAR	Number of jobs within each customer order is variable, and determined by $DU[1,N]$
CONSTANT or CNST	Number of jobs within each customer order is constant (fixed), and determined by $DU[2, N-1]$
SHORT	Processing times are determined by $DU[1,10]$
LONG	Processing times are determined by $DU[100,200]$
LL	Setup times have low mean and low variance, and are determined by $DU[25,35]$
LH	S Setup times have low mean and high variance, and are determined by $DU[10,50]$
HL	Setup times have high mean and low variance, and are determined by $DU[55,65]$
HH	Setup times have high mean and high variance, and are determined by $DU[40,80]$

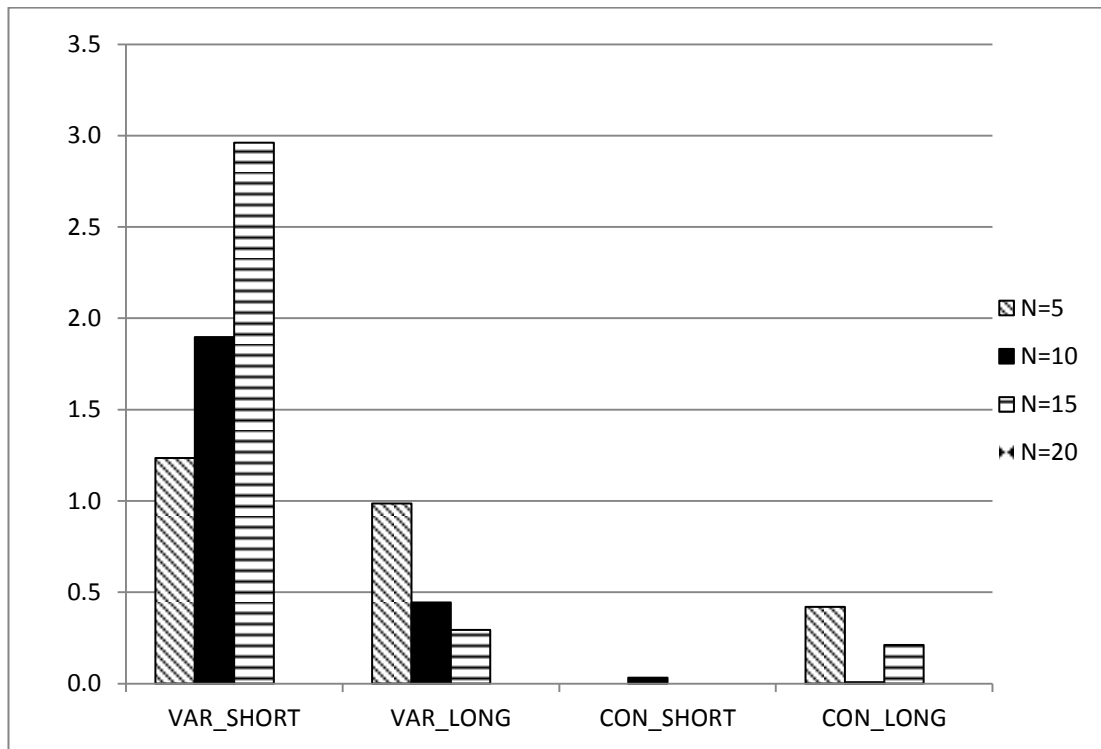
TABLE A. 2 - AVERAGE PERCENT DEVIATIONS OF ALGORITHM 4 FROM THE BEST-INTEGGER SOLUTIONS

K	N	VARIABLE												CONSTANT												AVG_TOTAL
		SHORT						LONG						SHORT						LONG						
		LL	LH	HL	HH	AVG	LL	LH	HL	HH	AVG	LL	LH	HL	HH	AVG	LL	LH	HL	HH	AVG					
15	5	*	*	*	*	0.000	*	*	*	0.000	*	*	*	0.000	3.975	0.213	0.000	-0.002	1.046	*	0.144	*	0.068	0.106	0.576	0.288
	10	*	0.307	*	*	0.307	*	*	*	0.000	*	*	*	0.000	1.677	0.585	0.000	-0.382	0.470	*	0.027	*	0.682	0.354	0.412	0.283
	15	*	0.998	*	*	0.998	*	*	*	0.000	*	*	*	0.000	1.306	1.617	*	0.003	0.975	-0.153	0.260	0.000	-0.157	-0.012	0.482	0.490
	20	1.505	0.646	*	1.446	1.199	*	0.006	*	0.230	0.118	0.658	-3.590	0.936	*	0.515	-0.713	-0.113	0.388	*	0.865	0.380	-0.166	0.246		
	AVG_15	1.505	0.650	0.000	1.446	0.900	0.000	0.006	0.230	0.059	0.480	0.842	0.838	0.000	0.033	0.428	-0.133	0.205	0.000	0.364	0.109	0.269	0.374			
20	5	1.440	1.905	*	4.571	2.639	0.222	*	0.198	0.210	1.424	0.908	-0.064	2.300	-0.206	0.734	-0.003	-0.081	0.115	0.286	0.092	0.413	0.919			
	10	5.888	1.325	*	2.736	3.316	*	0.296	1.038	0.667	1.992	-0.030	-1.734	0.004	-0.622	-0.596	1.000	-0.198	0.329	0.133	0.316	-0.140	0.926			
	15	1.391	2.381	0.866	1.124	1.441	0.102	0.198	0.480	0.141	0.791	0.326	-1.776	-1.336	-3.528	-1.578	-0.048	-0.146	1.734	-0.044	0.374	-0.602	0.094			
	20	-3.001	-2.657	-3.590	-3.406	-3.164	-0.684	-0.978	-1.219	-1.101	-2.132	-0.992	-3.169	*	2.163	-0.666	-0.732	0.758	0.143	-1.737	-0.392	-0.529	-1.331			
	AVG_20	1.429	0.739	-1.362	1.256	0.516	-0.120	-0.161	0.124	-0.256	0.130	0.053	-1.686	0.323	-0.548	-0.465	0.054	0.096	0.580	-0.341	0.097	-0.184	-0.027			
	AVG_TOTAL	1.467	0.694	-0.681	1.351	0.708	-0.060	-0.077	0.177	-0.099	0.305	0.447	-0.424	0.161	-0.258	-0.018	-0.039	0.150	0.290	0.012	0.103	0.043	0.174			

**APPENDIX B – AVERAGE PERCENT DEVIATIONS FROM THE
OPTIMAL WHEN $K = 15$**



**APPENDIX C – AVERAGE PERCENT DEVIATIONS FROM THE
OPTIMAL WHEN $K = 20$**



**APPENDIX D – AVERAGE PERCENT DEVIATIONS CALCULATED
FOR $K = 5, 10, 15, 20$**

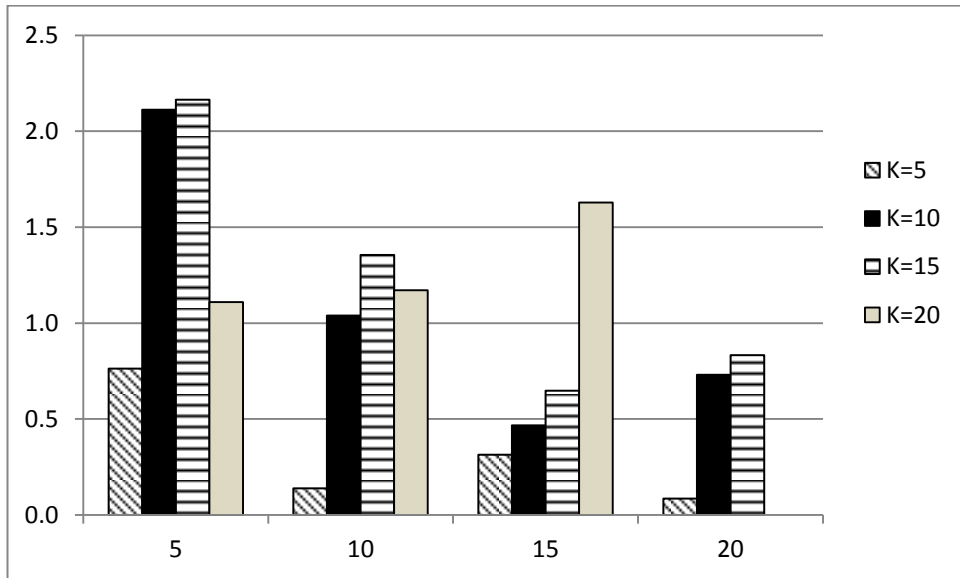


Figure D- 1 Average percent deviations for VARIABLE case

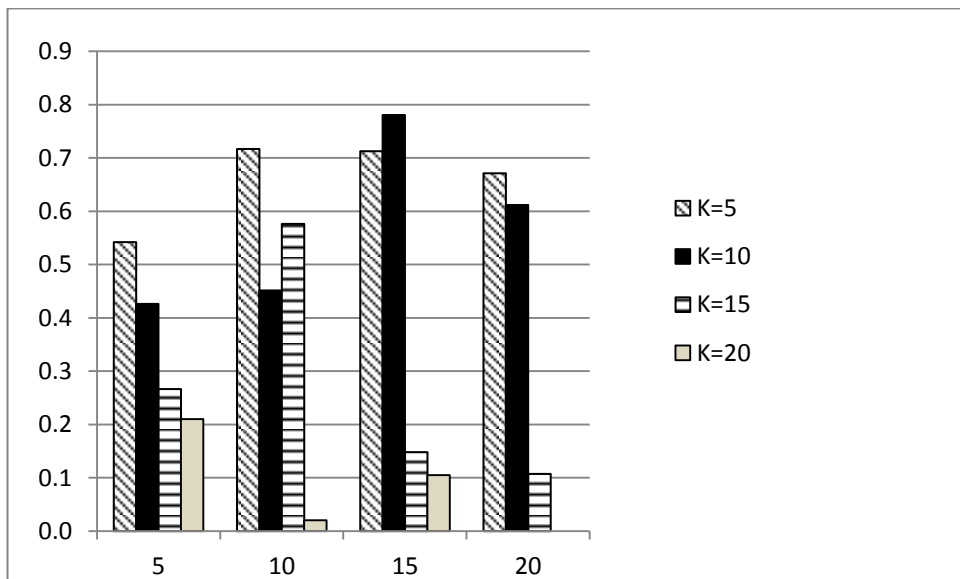


Figure D- 2 Average percent deviations for CONSTANT case

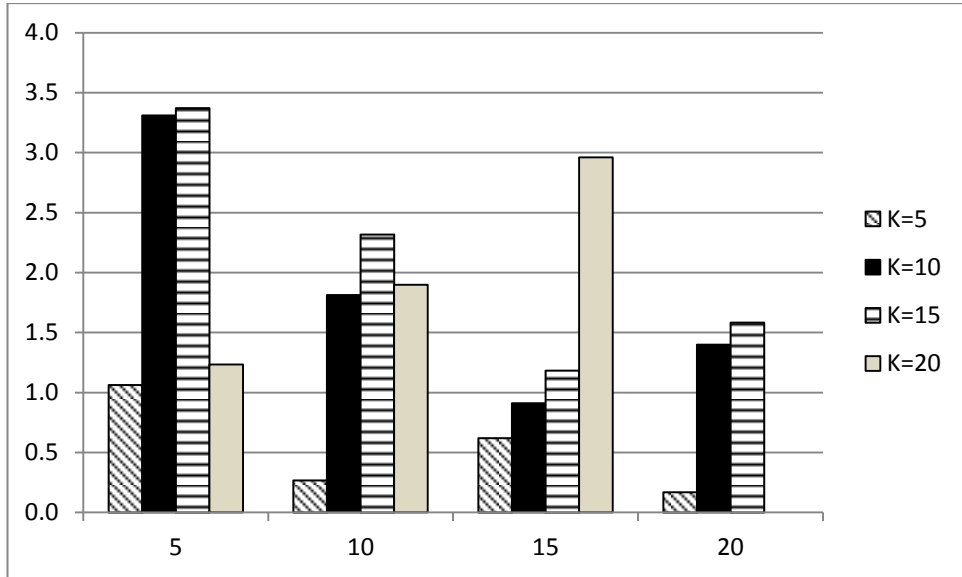


Figure D- 3 Average percent deviations for VAR_SHORT case

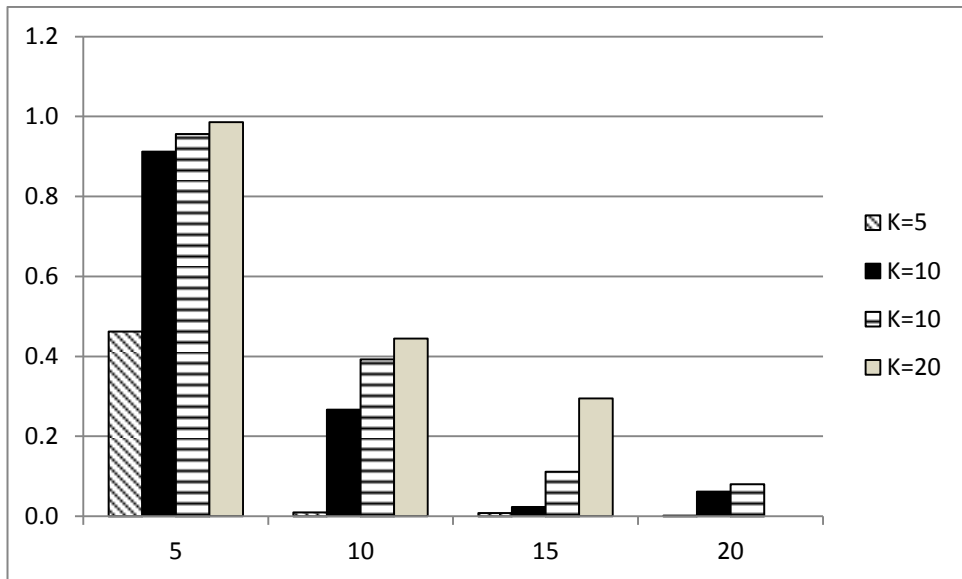


Figure D- 4 Average percent deviations for VAR_LONG case

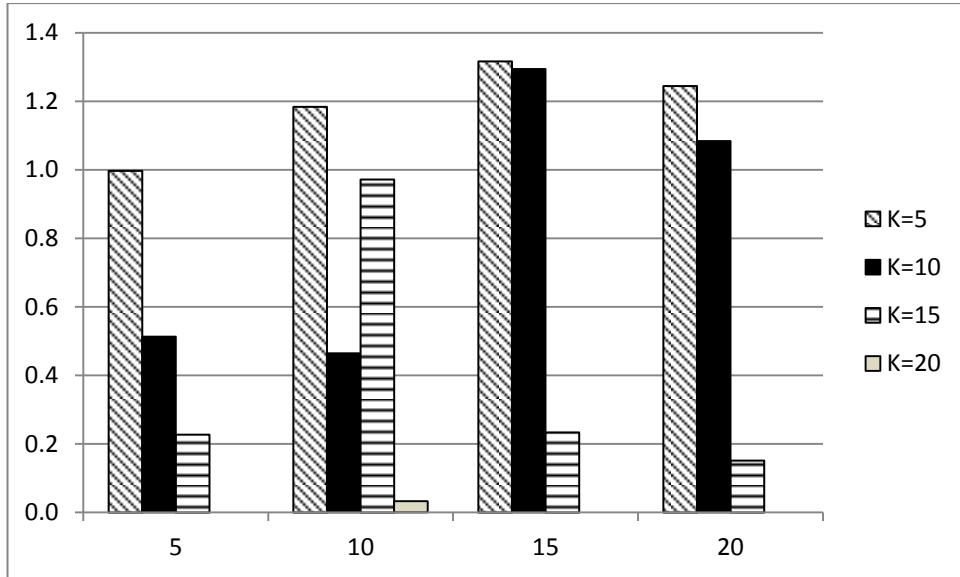


Figure D- 5 Average percent deviations for CON_SHORT case

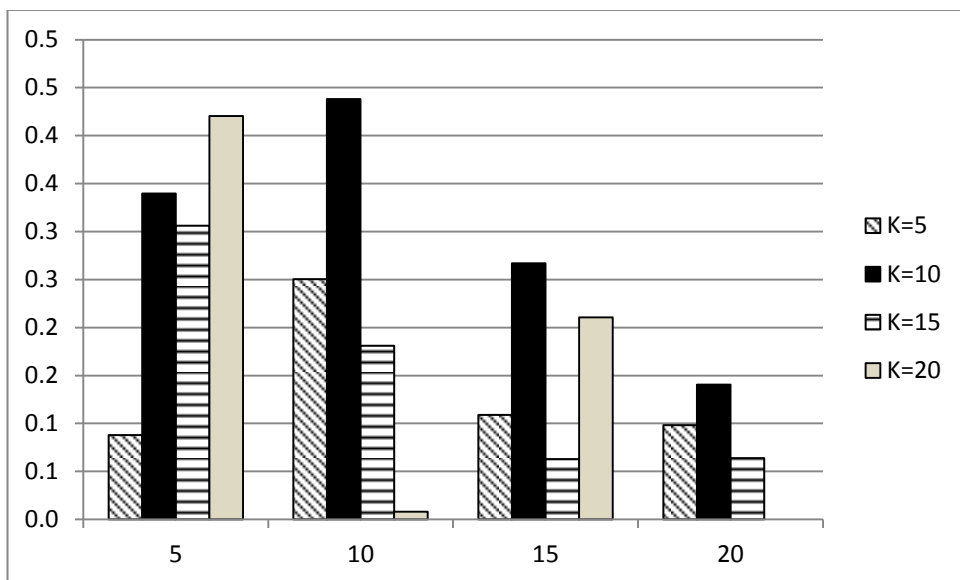


Figure D- 6 Average percent deviations for CON_LONG case

APPENDIX E

CURRICULUM VITAE

PERSONAL INFORMATION

Surname, Name: Akkocaoğlu, Hale

Nationality: Turkish (TC)

Date and Place of Birth: 29 May 1990, Ankara

Marital Status: Single

Phone: +90 312 233 13 70

Fax: +90 312 233 10 26

email: hale@cankaya.edu.tr

EDUCATION

Degree	Institution	Year of Graduation
MS	Çankaya University/ Industrial Engineering	2014
BS	Çankaya University/ Industrial Engineering	2010
BS	Çankaya University/ Political Science and International Relations	2010
High School	Cumhuriyet Anatolian High School, Ankara	2006

WORK EXPERIENCE

Year	Place	Enrollment
2012-present	Çankaya University/ Dept. of Industrial Engineering	Expert
2011-2012	Atılım University/ Dept. of Industrial Engineering	Research Assistant
2008 July	TEMSAN	Intern Engineering Student
High School	MKE MAKSAM	Intern Engineering Student

FOREIGN LANGUAGES

Advanced English