8-2009

Scheduling for Service Stability and Supply Chain Coordination

Yuerong Chen
University of Tennessee - Knoxville

To the Graduate Council:

I am submitting herewith a dissertation written by Yuerong Chen entitled "Scheduling for Service Stability and Supply Chain Coordination." I have examined the final electronic copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Industrial Engineering.

Xueping Li, Major Professor

We have read this dissertation and recommend its acceptance:

Rapinder S. Sawhney, Denise F. Jackson, Frank M. Guess

Accepted for the Council:

Carolyn R. Hodges

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

To the Graduate Council:

I am submitting herewith a dissertation written by Yuerong Chen entitled "Scheduling for Service Stability and Supply Chain Coordination." I have examined the final electronic copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Industrial Engineering.

Xueping Li, Major Professor

We have read this dissertation
and recommend its acceptance:

Rapinder S. Sawhney

Denise F. Jackson

Frank M. Guess

Accepted for the Council:

Carolyn R. Hodges

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

Scheduling for Service Stability and Supply Chain Coordination

A Dissertation
Presented for the
Doctor of Philosophy
Degree
The University of Tennessee, Knoxville

Yuerong Chen
August 2009

Dedication

This dissertation is dedicated to my parents, Yicheng Chen and Tianying Chen, who always selflessly provide their support and encouragement to me.

Acknowledgments

First and foremost, I am deeply indebted to my family, especially my parents, Yicheng Chen and Tianying Chen, and my sister, Xuerong Chen. Their everlasting love and support give me tremendous strength to conquer life's challenges. I would like to thank my uncle, Shuicheng Chen, who provided strong support to several of my most important decisions in my life. Definitely, I owe a lot to the other family members including my grandpas and grandmas.

I additionally would like to thank my advisor, Dr. Xueping Li, who provides excellent guidance to my research and always encourages me when I run into difficulties. I also thank him for the favorable lab environment he brings about. I also would like to thank the other members of my committee: Dr. Rapinder S. Sawhney, Dr. Denise F. Jackson and Dr. Frank M. Guess. I greatly appreciate their time and input to this dissertation. I especially thank Dr. Rapinder S. Sawhney for his sense of humor that helps relieve the sometimes stressful life of graduate school.

I also would like to thank my fellow graduate students: Dengfeng Yang, Laigang Song, Jiao Wang, Qingzhu Yao and Qi Yuan. Thank them for sharing the graduate school life together here and the supportive team environment. I would further like to thank the other graduate students in my department, especially Sasikurmar Naidu, for the favorable research atmosphere.

Finally, I must express my appreciation to the many friends outside of my studies. I thank Xin Wang, Ling Liu and Meiling Chen for their full support during my hard time. I am deeply grateful for their invaluable friendships. I additionally thank Yueying Wu and Xiaoyan Yu for taking part in extracurricular activities together.

Abstract

This dissertation studies scheduling for service stability and for supply chain coordination as well. The scheduling problems for service stability are studied from the single perspective of a firm itself, while the scheduling problems for supply chain coordination are investigated from the perspective of a supply chain. Both the studies have broad applications in real life.

In the first study, several job scheduling problems are addressed, with the measure of performance being job completion time variance (CTV). CTV minimization is used to represent service stability, since it means that jobs are completed in a relative concentrated period of time. CTV minimization also conforms to the Just-in-time philosophy. Two scheduling problems are studied on multiple identical parallel machines. The one problem does not restrict the idle times of machines before their job processing, while the other does. For these two scheduling problems, desirable properties are explored and heuristic algorithms are proposed. Computational results show the excellent performances of the proposed algorithms. The third scheduling problem in the first study is considered on a single machine and from the users' perspective rather than the system's perspective. The performance measure is thus class-based completion time variance (CB-CTV). This problem is shown to be able to be transformed into multiple CTV problems. Therefore, the well-developed desirable properties of the CTV problem can be applied to solve the CB-CTV problem. The tradeoff between the CB-CTV problem and the CTV problem is also investigated.

The second study deals with scheduling coordination in a supply chain, since supply chain coordination is increasingly critical in recent years. Usually, different standpoints prevent decision makers in a supply chain from having agreement on a certain scheduling decision. Therefore conflicts arise. In pursuit of excellent performance of the whole supply chain, coordination among decision makers is needed. In this study, the scheduling conflicts are measured and analyzed from different perspectives of decision makers, and cooperation mechanisms are proposed based on different scenarios of the relative bargaining power among decision makers. The cooperation savings are examined as well.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Introduction to Scheduling

Scheduling, by definition, is a decision-making process that deals with the allocation of scarce resources (or machines) to tasks (or jobs), with the goal of optimizing one or more objectives. The resources and tasks have many forms in real life. For example, the resources may be machines in a workstation, tellers at a bank, runways at an airport, processors in a computing system, and so on. The corresponding tasks may be operations in a production process, transactions of customers, take-offs and landings of planes, executions of computer programs, and so on. Each task may have a certain priority level, a due date, etc. The objectives may also take many forms. For instance, the objective may be the minimization of the completion time of the last task, or the maximization of the number of the completed tasks during a period of time. In this dissertation, we exclusively denote resources and tasks by machines and jobs, respectively.

Scheduling has broad applications in real life. Take airport schedule as an example. At a major airport, there are dozens of gates and hundreds of airplanes arriving and departing each day. Generally, the sizes of the gates are different. Neither are the planes' sizes. The gates with small space can only accommodate small planes. This makes the assignment of planes to gates a difficult task. Another difficulty lies in the uncertainty that may be weather-related or due to the influences of the events in other airports. For example, if it is known in advance that a plane cannot land at its next destination because of anticipated congestion at the scheduled arrival time, then in order to conserve fuel the plane will postpone its take-off. The consequence is that the boarding is delayed and that passengers are kept in the terminal waiting. Moreover, the plane may remain at the gate for an extended period of time, therefore preventing other planes from using the gate. In view of these factors (but not limited to), the airport needs a good schedule to assign planes to the appropriate gates, with the objective of minimizing the workload of airline personnel and (or) minimizing airplane delays, and so on.

Because of its numerous applications, scheduling has received extensive attention from researchers and practitioners for several decades. There is a tremendous body of literature in this field. Representative are the works of Brucker (2004); Leung (2004); Pinedo (2002). Many effective scheduling theory and algorithms have been developed. They are widely employed in lots of manufacturing and production systems, in transportation and distribution systems, as well as in many information-processing environments.

Conventionally, a scheduling problem is represented by a triplet $\alpha|\beta|\gamma$ introduced by Graham et al. (1979). The α field determines the machine environment (*e.g.*, single machine, parallel machines, flow shop, job shop, and so on), the β field specifies the processing characteristics and constraints (*e.g.*, preemptions, release dates, setup times, and so on), and the γ field describes the objective function (*e.g.*, makespan, total weighted completion time, total weighted tardiness, and so on). There are several classification methods of scheduling problems. We introduce two kinds of methods below. First classification method is based on the machine environment. So there are two categories: single-machine scheduling problems and multiple-machine scheduling problems. In the previous literature, single-machine scheduling problems are well studied. However, due to the relatively complicated nature, multiple-machine scheduling problems are less investigated. The second classification method is based on the characteristics of job processing times. Using this method, scheduling problems can be categorized into three kinds: deterministic, stochastic, and online scheduling. With regard to deterministic scheduling, job processing times are deterministic and are known in advance of their processing. Vast literature focuses on deterministic scheduling, see (Al-Turki et al., 2001; Eilon and Chowdhury, 1977; Manna and Prasad, 1999; Merten and Muller, 1972; Schrage, 1975). With respect to stochastic scheduling, job processing times are subject to uncertainty. In other words, job processing times are random variables. The actual processing times become known only upon the completion of jobs. It is generally assumed, though, that the first moments of these random variables are known beforehand. For the literature of stochastic scheduling, refer to (Mittenthal and Raghavachari, 1993; Prasad and Manna, 1997; Vani and Raghavachari, 1987). The most difficult is online scheduling, in which the instance is presented to the scheduler only piecewise. Jobs are arriving either one-by-one (sequence model), or over time (time-stamp model). Job processing times are usually known upon the arrival of jobs and decisions must be made without any knowledge of the jobs to come. See (Anderson and Potts, 2002; Hoogeveen and Vestjens, 1996; Megow and Schulz, 2004) for reference.

As mentioned above, the objectives (also called as performance measures) of scheduling have various forms. There are generally, though, two kind of performance measures: regular and non-regular. By definition, regular performance measures are nondecreasing in job completion times. Other performance measures fall into the non-regular category. Regular performance measures include mean completion time, mean lateness, mean tardiness, and so on. In particular, the mean tardiness has been a standard way of measuring conformance to due dates, even though it ignores the consequence of jobs completing early. Non-regular performance measures arise from the increasing interest in Just-In-Time (JIT) philosophy, which espouses the notion that both earliness and tardiness should be penalized (Baker and Scudder, 1990). Examples of non-regular performance measures include mean squared deviation (MSD) of completion times, waiting time variance (WTV), and so forth.

1.2 Motivation of the Study

1.2.1 Service Stability

For many firms, the quality of service (QoS) is what they always run after because customers are expecting better and better service level. A firm with higher level of service has a stronger competitive advantage in the market. Qos is widely applied in the field of computer networking, in which the goal of QoS is to provide guarantees on the ability of a network to deliver predictable results. Elements of network performance within the scope of QoS often include availability (uptime), bandwidth (throughput), latency (delay), and error rate. QoS is especially important for the new generation of Internet applications such as VoIP, video-on-demand and other consumer services.

Service stability is included in the scope of QoS. Consumers always desire to be serviced in a stable way. Take a network server as an example. Users who are browsing a web site expect a stable speed to open a new web page. If the network server can not provide stable service (*i.e.*, the connection speed is fast sometimes while slow other times), the users will be dissatisfied with the service and will turn to other network server providers. So service stability is very crucial in the market competition. The significance of service stability can be noticed in other service industries as well.

In this dissertation, several scheduling problems are considered for service stability. The performance measure used is job completion time variance (CTV), the minimization of which is used to represent service stability. The completion time of a job is defined as the point of time at which the job is completed, and CTV is the variance of all jobs' completion times. CTV minimization means that all jobs are completed within a relatively concentrated time period. Neither earliness nor tardiness is desired. Therefore, CTV minimization is related to service stability if each service request is regarded as a job.

1.2.2 Supply Chain Coordination

In the recent decade, we have seen an explosion of publications on supply chain management. Many articles have appeared in academic and popular magazines and numerous books have been published. Interest in supply chain management, both in industry and in academia, has been evoked by part from the discovered enormous magnitude of savings that can be achieved by effectively planning and managing supply chains. A striking example is Wal-Mart's success, which is partly attributed to implementing a new logistics strategy called cross-docking. Another cause of the enthusiastic research interest in supply chain management is the emerging and development of information technology and communication systems, which provide access to various data from all components of the supply chain. Typical examples are business giants such as Dell Computers and Amazon.com. They enable customers to order products over the Internet and thus sell products without relying on third-party distributors or physical stores. The information technology dramatically decreases the operating costs of these companies.

Simchi-Levi et al. (2003) define supply chain management as "a set of approaches utilized to efficiently integrate suppliers, manufacturers, warehouses, and stores, so that merchandise is produced and distributed at the right quantities, to the right locations, and at the right time, in order to minimize systemwide costs while satisfying service level requirements". A company can manage its supply chain from strategic, tactical, and operational levels. The strategic level deals with decisions that have a long-run effect on the company. This includes supplier selection, decisions on the number, location, and capacity of plants and warehouses, and the like. The tactical level deals with decisions that are typically updated anywhere between once every quarter and once every year. This includes purchasing and production decisions, inventory policies, and so on. The operational level deals with day-to-day decisions such as scheduling, routing, and so forth.

Supply chain management has several key issues, including inventory management (Axsäter, 2006; Silver et al., 1998), information sharing (Lee et al., 1997, 2000; Lee and Whang, 2000), disruption risk management (Chopra and Sodhi, 2004; Tomlin, 2006), to name a few. Thomas and Griffin (1996) provide an extensive review on supply chain management research.They identify several research streams, including coordinated planning in inventory-distribution systems (Mittenthal and Raghavachari, 1993), coordination in production-distribution systems (Chandra and Fisher, 1994), and buyer-vendor coordination (Anupindi and Akella, 1993). These research streams reflect the importance of studying the coordination issues in supply chains. Furthermore, Sarmiento and Nagi (1999) conduct a survey of integrated production and distribution models, pointing out that the trend towards reduced inventory levels creates a need for greater coordination between decisions at different stages of a supply chain. Banker and Khosla (1995) provide general motivation of coordinated decision making in supply chains.

In this dissertation we will study an important facet of supply chain coordination: supply chain scheduling. It is concerned with the coordination of scheduling decisions among different decision makers in a supply chain. The study on this area is inspired by the phenomena that there are conflicts with respect to decisions on some scheduling problem(s) that simultaneously confront(s) two or more decision makers in a supply chain. The conflicts are usually caused by the inconsistency of decision makers' individual optimal schedules determined by their respective standpoints. For instance, consider a supplier and a manufacturer. The supplier provides parts to the manufacturer. Assume that the production of each part is time consuming and thus each part will be immediately shipped to the manufacturer for use once it is finished producing. As for the parts' production at the supplier's, the manufacturer has its own optimal schedule that meets its need. However, the supplier may desire a different production schedule that reduces its cost. The conflict arises accordingly. Simply adopting the optimal schedule of a decision maker will not optimize the cost of the whole supply chain. Therefore, the scheduling decision needs to be coordinated. Different scenarios of relative bargaining power among decision makers determine different coordinated scheduling decisions.

1.3 Contributions and Document Organization

For service stability, this dissertation studies two scheduling problems on identical parallel machines and a class-based scheduling problem on a single machine. The first two scheduling problems take the same performance measure: job completion time variance minimization. The difference lies in that the former considers the unrestricted case of the problem while the latter addresses the restricted case. The properties of the two problems are explored and heuristic algorithms are proposed for solving the problems. The last single-machine scheduling problem takes the minimization of class-based completion time variance as the objective. It treats the problem from the users' perspective instead of the conventional system's perspective. The property of the problem is investigated. These scheduling problems for service stability are addressed in Chapter 2.

The above scheduling problems only focus on the optimal schedule within a firm. They do not consider other components of a supply chain. However, in the current market with fierce competition, the performance of the whole supply chain is more important than that of a component itself. Chapter 3 takes into account the scheduling coordination problem in a supply chain. The specific investigated problem instance in this chapter occurs in a production-distribution environment. The manufacturer produces the products that are delivered to customers by the distributor. Due to different standpoints, the manufacturer and the distributor have different opinions regarding the production scheduling policy of the customers' orders that the manufacturer receives. This chapter analyzes the conflict, proposes cooperation mechanisms, and investigates the cost saving provided by the cooperation.

In Chapter 4, we summarize the work and demonstrate its potential applications. Future research directions are also pointed out.

Chapter 2

Scheduling for Service Stability

2.1 Introduction

Service stability is an aspect of quality of service (QoS), which plays an increasingly important role in service industries. QoS is widely employed in computer networking. QoS requirements in Web services include availability, accessibility, integrity, performance, reliability, regulatory, and security. Consumers always hope to obtain stable service. A service provider who can provide highly stable service has an enormous body of customers. In this chapter, we study job scheduling problems for service stability. This is realized by adopting the performance measure of completion time variance (CTV). CTV is defined as the variance of job completion times. Since CTV is not nondecreasing in job completion times, it is a non-regular performance measure.

CTV minimization punishes both earliness and tardiness. This is because that CTV is the variance of job completion times. If CTV minimization is desirable, then all jobs are desired to be completed in a concentrated period of time. Therefore, it is undesirable that a job completes too early or too late, compared to most jobs. Therefore, CTV minimization is related to service stability. It also pertains to the Just-in-Time (JIT) philosophy. In a JIT scheduling environment, jobs that complete early must be held in finished goods inventory which incurs inventory holding cost, while jobs that complete after their due dates may cause customer dissatisfaction and penalty of agreement violations. Therefore, an ideal schedule is the one in which all jobs finish exactly on their assigned due dates, or the one in which a certain performance measure is stabilized according to (Chen et al., 1998). In the JIT philosophy, both earliness and tardiness are penalized as well.

Job scheduling with CTV minimization has a wide range of applications in real life. It can be used in such areas as production scheduling, Internet data packet dispatching, and so on. Take a hamburger store as an example. Assume that a number of consumers arrive at the store almost simultaneously. Each consumer places an order of hamburgers. Different orders may request different preparation times. In pursuit of service stability, the store manager desires that these consumers receive their orders within a relative centralized period of time. How should the manager assign these orders to its cooks so as to achieve his/her goal? This problem is in essence a job scheduling problem on multiple parallel machines with CTV minimization.

The remainder of this chapter is organized as follows. Section 2.2 is the literature review of CTV minimization problems. Sections 2.3 and 2.4 are concerned with the CTV minimization problems on identical parallel machines, without and with the restriction that machine idle times are zero before starting processing jobs, respectively. Section 2.5 studies a class-based CTV minimization problem on a single machine, which comes from a very different viewpoint.

2.2 Literature Review

2.2.1 Under Single-Machine Environment

The CTV minimization problem on a single machine is denoted by $1||CTV$ and it has been widely studied. It is first introduced by Merten and Muller (1972) to minimize the response time variance in computer file organization problems. Merten and Muller (1972) show that for the $1||CTV$ problem, the optimal sequence with CTV minimization is antithetical to that with waiting time variance (WTV) minimization. They also prove that there exists a dual optimal sequence, which is obtained by keeping the first job intact while reversing the order of the remaining jobs. Many other dominant properties about CTV minimization have been discovered in the past decades. Eilon and Chowdhury (1977) prove that the optimal sequence with the minimum CTV is V-shaped, which means that the jobs before the smallest job are scheduled in descending order of processing times and the jobs after the smallest job are scheduled in ascending order of processing times. Schrage (1975) shows that for the $1||CTV$ problem, the largest job should be placed on the first position. The author also makes conjecture on the positions of the next three largest jobs. Hall and Kubiak (1991) verify Schrage's conjecture about the placement of the second and the third largest jobs, *i.e.*, they should be placed on the last and the second positions respectively, or the second and the last positions respectively according to the dual rule. Manna and Prasad (1999) exhibit the bounds for the position of the smallest job in an optimal sequence.

Despite so many favorable properties, there exist no polynomial time algorithms to obtain an optimal sequence with CTV minimization. In fact, Kubiak (1993) proves that the CTV problem is NP-hard. However, countless heuristics have been proposed to obtain a near-optimal schedule, such as those in (Eilon and Chowdhury, 1977; Kanet, 1981; Manna and Prasad, 1997, 1999; Vani and Raghavachari, 1987; Ye et al., 2007). Some famous algorithms are involved in developing efficient heuristics for the $1||CTV$ problem as well. For instance, De et al. (1992) develop a dynamic programming algorithm. Other examples include a genetic algorithm in (Gupta et al., 1993), a simulated annealing method in (Mittenthal et al., 1993), a tabu search method in (Al-Turki et al., 2001), a branch and bound method in (Viswanathkumar and Srinivasan, 2003), and an ant-colony optimization algorithm in (Gajpal and Rajendran, 2006).

2.2.2 Under Parallel-Machine Environment

In this dissertation, parallel machines are assumed to be identical. The CTV minimization problem on identical parallel machines is denoted by $Pm||CTV$. By far, few research has been conducted on the $Pm||CTV$ problem. Cheng and Sin (1990) present that most of the previous work on parallel-machine job scheduling does not consider the performance measure of CTV. They mainly focus on other measures of performance such as total completion time, mean completion time, weighted completion time, maximum completion time, and so on. To the best of our knowledge, Cai and Cheng (1998) first discuss the problem of $Pm||CTV$.

They derive some properties of optimal solutions and show that the problem is NP-complete in the strong sense when number of machines m is arbitrary and in the ordinary sense when m is fixed. Xu and Ye (2007) is mainly concerned with $Pm||WTV$, but they show that the optimal value of $Pm||WTV$ is equal to that of $Pm||CTV$ and that a feasible schedule for $Pm||WTV$ can be transformed into a feasible schedule for $Pm||CTV$.

2.3 The Unrestricted Case of $Pm||CTV$: $Pm|Unres|CTV$

As seen in literature review, there is very few existing work related to the $Pm||CTV$ problem. Therefore, in the current and immediately following sections the $Pm||CTV$ problem is targeted. Two cases of this problem are addressed. The difference of these two cases lies in whether there is a restriction on machine idle times that exist before machines start to process jobs. One case has no restriction, while the other case restricts such idle times to be zero, in other words, there are no idle times for machines and machines must start processing jobs at the very beginning.

In this section, we deal with the unrestricted case of job scheduling on identical and parallel machines with CTV minimization. This problem is denoted by $Pm|Unres|CTV$. Several properties of the $Pm|Unres|CTV$ problem are explored and an efficient heuristic algorithm is proposed. The performance of the proposed heuristic is compared with the optimal schedules when problem instances are small and is compared with some existing algorithms when problem instances are large.

2.3.1 Problem Definition and Notation

In this case, machines' idle times before they start to process jobs can be a time period of any length. This is reasonable in practice, because machines may wait some time before they begin processing jobs, in order to optimize some objective. For instance, in JIT production system, earliness is undesirable, so machines may have some idle times before processing jobs to avoid earliness.

Some assumptions are made as follows. First, job processing times are known in advance. Second, each machine can only process one job at a time. Third, all jobs are available at time zero. Fourth, no setup time exists between two consecutive jobs. In addition, preemption is not allowed, that is, a job cannot be interrupted once the machine starts to process it. Similar assumptions are made in (Eilon and Chowdhury, 1977; Merten and Muller, 1972; Ye et al., 2007). The notation to be used is defined as follows:

n : the total number of jobs

m : the total number of machines

I : the index set of machines, *i.e.*, $I = \{1, \cdots, m\}$

n_i : the total number of jobs assigned to the i^{th} machine, $i \in I$, $\sum\limits_{i=1}^{m} n_i = n$

λ : a schedule

λ^* : an optimal schedule

MCT : mean completion time

$d_i(\lambda)$: the idle time that exists before the i^{th} machine's job processing under a schedule λ, $i \in I$

$p_{ij}(\lambda)$: the processing time of the j^{th} job on the i^{th} machine under a schedule λ, $i \in I, j \in \{1, 2, \cdots, n_i\}$

$C_{ij}(\lambda)$: the completion time of the j^{th} job on the i^{th} machine under a schedule λ, $i \in I, j \in \{1, 2, \cdots, n_i\}$

$\bar{C}_i(\lambda)$: the job MCT on the i^{th} machine under a schedule λ, $i \in I$

$MCT_i(\lambda)$: same as $\bar{C}_i(\lambda)$

$\bar{\bar{C}}(\lambda)$: the MCT of all n jobs under a schedule λ

$CTV_i(\lambda)$: the job CTV on the i^{th} machine under a schedule λ, $i \in I$

$CTV(\lambda)$: the CTV of all n jobs under a schedule λ

The completion time of a job, by definition, is the time when the job is completed. Therefore, under the unrestricted case, when calculating a job's completion time, one needs to consider the idle time that exists before the corresponding machine's job processing. Using the notation, we have

$$C_{ij}(\lambda) = d_i(\lambda) + \sum_{k=1}^{j} p_{ik}(\lambda), \quad i \in I,\ j \in \{1, 2, \cdots, n_i\}.$$

For example, assume that a schedule is of the following form:

M1:	(2.5)	8	25	13	11	7
M2:	(3.0)	23	12	9	6	10

where M1 and M2 represent two machines, the numbers in the parenthesis represent d_1 and d_2, and the other numbers represent the processing times of jobs. Then the corresponding job completion times are as follows:

M1:	10.5	35.5	48.5	59.5	66.5
M2:	26	38	47	53	63

Take the calculation of C_{12} for example. It is equal to $C_{12} = d_1 + p_{11} + p_{12} = 2.5 + 8 + 25 = 35.5$.

The objective of the $Pm|Unres|CTV$ problem is to find a schedule λ so that

$$CTV(\lambda) = \frac{1}{n-1} \sum_{i=1}^{m} \sum_{j=1}^{n_i} \left(C_{ij}(\lambda) - \bar{\bar{C}}(\lambda)\right)^2 \tag{2.1}$$

is the minimum, where

$$\bar{\bar{C}}(\lambda) = \frac{1}{n} \sum_{i=1}^{m} \sum_{j=1}^{n_i} C_{ij}(\lambda) \ . \tag{2.2}$$

Also, the other two important computation formulas are as follows:

$$\bar{C}_i(\lambda) = \frac{1}{n_i} \sum_{j=1}^{n_i} C_{ij}(\lambda), \tag{2.3}$$

$$CTV_i(\lambda) = \frac{1}{n_i - 1} \sum_{j=1}^{n_i} \left(C_{ij}(\lambda) - \bar{C}_i(\lambda)\right)^2. \tag{2.4}$$

2.3.2 Dominant Properties

Lemma 1. *For a given schedule λ, the following equality holds:*

$$(n-1)CTV(\lambda) = \sum_{i=1}^{m} (n_i - 1) CTV_i(\lambda) + \sum_{i=1}^{m} n_i (\bar{C}_i(\lambda) - \bar{\bar{C}}(\lambda))^2.$$

Proof. For simplicity, we neglect λ's in the notation in the proof below when there is no confusion.

$$
\begin{aligned}
&(n-1)CTV \\
=&\sum_{i=1}^{m}\sum_{j=1}^{n_i}(C_{ij}-\bar{\bar{C}})^2 \\
=&\sum_{j=1}^{n_1}(C_{1j}-\bar{C}_1+\bar{C}_1-\bar{\bar{C}})^2+\sum_{j=1}^{n_2}(C_{2j}-\bar{C}_2+\bar{C}_2-\bar{\bar{C}})^2+\cdots+\sum_{j=1}^{n_m}(C_{mj}-\bar{C}_m+\bar{C}_m-\bar{\bar{C}})^2 \\
=&\sum_{j=1}^{n_1}(C_{1j}-\bar{C}_1)^2+\sum_{j=1}^{n_1}(\bar{C}_1-\bar{\bar{C}})^2+2(\bar{C}_1-\bar{\bar{C}})\sum_{j=1}^{n_1}(C_{1j}-\bar{C}_1)+\sum_{j=1}^{n_2}(C_{2j}-\bar{C}_2)^2 \\
&+\sum_{j=1}^{n_2}(\bar{C}_2-\bar{\bar{C}})^2+2(\bar{C}_2-\bar{\bar{C}})\sum_{j=1}^{n_2}(C_{2j}-\bar{C}_2)+\cdots+\sum_{j=1}^{n_m}(C_{mj}-\bar{C}_m)^2+\sum_{j=1}^{n_m}(\bar{C}_m-\bar{\bar{C}})^2 \\
&+2(\bar{C}_m-\bar{\bar{C}})\sum_{j=1}^{n_m}(C_{mj}-\bar{C}_m) \\
=&\sum_{j=1}^{n_1}(C_{1j}-\bar{C}_1)^2+\sum_{j=1}^{n_1}(\bar{C}_1-\bar{\bar{C}})^2+0+\sum_{j=1}^{n_2}(C_{2j}-\bar{C}_2)^2+\sum_{j=1}^{n_2}(\bar{C}_2-\bar{\bar{C}})^2+0+\cdots \\
&+\sum_{j=1}^{n_m}(C_{mj}-\bar{C}_m)^2+\sum_{j=1}^{n_m}(\bar{C}_m-\bar{\bar{C}})^2+0 \\
&\quad \text{(since by Equality (2.3), } \sum_{j=1}^{n_i}C_{ij}=n_i\bar{C}_i,\ i\in I) \\
=&\sum_{i=1}^{m}(n_i-1)CTV_i+\sum_{i=1}^{m}n_i(\bar{C}_i-\bar{\bar{C}})^2 \\
&\quad \text{(since by Equality (2.4), } \sum_{j=1}^{n_i}(C_{ij}-\bar{C}_i)^2=(n_i-1)CTV_i,\ i\in I)
\end{aligned}
$$

This completes the proof. □

Lemma 1 states the CTV of a $Pm|Unres|CTV$ problem can be partitioned into the weighted sum of CTV_i on each machine i and the weighed sum of squares of the differences between the MCT_i on each machine i and the grand MCT.

Lemma 2. *Consider a sequence of numbers* $\{h_1, h_2, \dots, h_n\}$ *and a number* d*. The following equalities hold:*

$$\begin{aligned} Mean(h_1 + d, h_2 + d, \dots, h_n + d) &= Mean(h_1, h_2, \dots, h_n) + d; \\ Var(h_1 + d, h_2 + d, \dots, h_n + d) &= Var(h_1, h_2, \dots, h_n) \end{aligned}$$

where $Mean(\{a_n\})$ *and* $Var(\{a_n\})$ *represent the mean value and variance of the array* $\{a_n\}$*, respectively.*

Proof. It can be easily seen that,

$Mean(h_1+d, h_2+d, \dots, h_n+d) = \frac{1}{n}\sum_{i=1}^{n}(h_i+d) = \frac{1}{n}\sum_{i=1}^{n} h_i + d = Mean(h_1, h_2, \dots, h_n) + d;$

Denoting $Mean(h_1, h_2, \dots, h_n)$ by $\bar{h}$, we have

$Var(h_1 + d, h_2 + d, \dots, h_n + d) = \frac{1}{n-1}\sum_{i=1}^{n}[h_i + d - (\bar{h} + d)]^2 = \frac{1}{n-1}\sum_{i=1}^{n}(h_i - \bar{h})^2 = Var(h_1, h_2, \dots, h_n).$

This completes the proof. □

Lemma 3. *Consider a problem of* $1 \parallel CTV$ *with* n *jobs, then the following hold:*

i) The specific value of the processing time of the first job has no influence on the CTV;

ii) Only increasing(or decreasing) the processing time of a single job other than the first job will increase(or decrease) the CTV.

Proof. Denote the processing times of these n jobs by $\{p_1, p_2, \dots, p_n\}$.

i) It is easy to see that the corresponding completion times are $C = \{p_1, p_1+p_2, p_1+p_2+p_3, \ldots, p_1+p_2+\cdots+p_n\}$. By Lemma 2,

$$CTV = Var(C) = Var(0, p_2, p_2+p_3, \ldots, p_2+\cdots+p_n).$$

The most right hand side is unrelated to p_1. This completes the proof of i).

ii) Denote by $j (j \neq 1)$ the position of the job whose processing time increases (or decreases). Separate the completion times into two groups, with one group consisting of the first $j-1$ ones and the other group consisting of the last $n-j+1$ ones. Denote the mean values and variances of the completion times in the two groups by $\bar{C}_1$, $\bar{C}_2$, V_1, and V_2. According to Vani and Raghavachari (1987), the pooled variance of the two groups is given by:

$$(n-1)CTV = (j-2)V_1 + (n-j)V_2 + \frac{(j-1)(n-j+1)}{n}(\bar{C}_1 - \bar{C}_2)^2.$$

When only increasing (or decreasing) the processing time p_j, it is clear that $\bar{C}_1$, V_1 will not change and by Lemma 2, $\bar{C}_2$ will increase (or decrease) and V_2 will not change. On the other hand, it is obvious that $\bar{C}_1 < \bar{C}_2$ any time. Thus, when only p_j increases (or decreases), $(\bar{C}_1 - \bar{C}_2)^2$ and thereby CTV will increase (or decrease). This completes the proof of ii). $\square$

Property 1. *Under λ^*, the MCT on each machine is the same. That is, $\bar{C}_i(\lambda^*) = \bar{C}_j(\lambda^*), i \neq j,\ i, j \in I$.*

Proof. (By contradiction.)

Suppose that under λ^* there exist at least two machines whose job MCTs are not equal. Let K be the index set of the machines whose job MCTs achieve the maximum under λ^*, *i.e.*, $K = \arg\max_{i\in I} \bar{C}_i(\lambda^*)$. Let T be the maximum job MCT, *i.e.*, $T = \max_{i\in I} \bar{C}_i(\lambda^*)$. We adjust the schedule λ^* as follows. For each machine $i \notin K$, add an additional idle time $d_i = T - \bar{C}_i(\lambda^*)$ before the machine starts to process jobs. It is obvious that $d_i > 0, i \notin K$. Denote by λ' the new schedule. Then, by Lemma 2, we have

$$\forall i \notin K,\ \bar{C}_i(\lambda') = \bar{C}_i(\lambda^*) + d_i = T, \quad \text{and} \quad CTV_i(\lambda') = CTV_i(\lambda^*).$$

On the other hand, since no changes are made on machine $i \in K$, job completion times on these machines keep the same and therefore, so do the MCTs and the CTVs on these machines. That is, $\forall i \in K$,

$$C_{ij}(\lambda') = C_{ij}(\lambda^*),\ j = 1, \ldots, n_i; \quad \bar{C}_i(\lambda') = \bar{C}_i(\lambda^*) = T; \quad \text{and} \quad CTV_i(\lambda') = CTV_i(\lambda^*).$$

Thus,

$$\begin{aligned}
\bar{\bar{C}}(\lambda') =& \frac{1}{n}\sum_{i=1}^{m}\sum_{j=1}^{n_i} C_{ij}(\lambda') = \frac{1}{n}\sum_{i=1}^{m} n_i\bar{C}_i(\lambda') = \frac{1}{n}\sum_{i=1}^{m} n_i T = T \\
\bar{\bar{C}}(\lambda^*) =& \frac{1}{n}\sum_{i=1}^{m}\sum_{j=1}^{n_i} C_{ij}(\lambda^*) = \frac{1}{n}\sum_{i=1}^{m} n_i\bar{C}_i(\lambda^*) = \frac{1}{n}\Big(\sum_{i\notin K} n_i\bar{C}_i(\lambda^*) + \sum_{i\in K} n_i\bar{C}_i(\lambda^*)\Big) \\
=& \frac{1}{n}\Big(\sum_{i\notin K} n_i(T - d_i) + \sum_{i\in K} n_i T\Big) = T - (\sum_{i\notin K} n_i d_i)/n.
\end{aligned}$$

Then, by Lemma 1,

$$\begin{aligned}
(n-1)CTV(\lambda') =& \sum_{i=1}^{m}(n_i - 1)CTV_i(\lambda') + \sum_{i=1}^{m} n_i\big(\bar{C}_i(\lambda') - \bar{\bar{C}}(\lambda')\big)^2 \\
=& \sum_{i=1}^{m}(n_i - 1)CTV_i(\lambda') + \sum_{i=1}^{m} n_i(T - T)^2 \\
=& \sum_{i=1}^{m}(n_i - 1)CTV_i(\lambda') \qquad (2.5)
\end{aligned}$$

$$
\begin{aligned}
(n-1)CTV(\lambda^*) =& \sum_{i=1}^{m}(n_i-1)CTV_i(\lambda^*) + \sum_{i=1}^{m} n_i\big(\bar{C}_i(\lambda^*) - \bar{\bar{C}}(\lambda^*)\big)^2 \\
=& \sum_{i=1}^{m}(n_i-1)CTV_i(\lambda^*) + \sum_{i\notin K} n_i\big(\bar{C}_i(\lambda^*) - \bar{\bar{C}}(\lambda^*)\big)^2 + \sum_{i\in K} n_i\big(\bar{C}_i(\lambda^*) - \bar{\bar{C}}(\lambda^*)\big)^2 \\
=& \sum_{i=1}^{m}(n_i-1)CTV_i(\lambda^*) + \sum_{i\notin K} n_i\big((\sum_{j\notin K} n_j d_j)/n - d_i\big)^2 + \sum_{i\in K} n_i\big((\sum_{j\notin K} n_j d_j)/n\big)^2 \\
>& \sum_{i=1}^{m}(n_i-1)CTV_i(\lambda^*) \qquad (\text{since } d_j > 0,\ \forall j \notin K) \\
=& \sum_{i=1}^{m}(n_i-1)CTV_i(\lambda') \qquad (\text{since } CTV_i(\lambda') = CTV_i(\lambda^*),\ \forall i \in I) \\
=& (n-1)CTV(\lambda') \qquad (\text{by the expression (2.5)})
\end{aligned}
$$

That is, $CTV(\lambda^*) > CTV(\lambda')$, which violates the assumption that λ^* is an optimal schedule. Therefore Property 1 holds. $\square$

Corollary 1. *$\bar{C}_i(\lambda^*) = \bar{\bar{C}}(\lambda^*)$, $\forall i \in I$, and $(n-1)CTV(\lambda^*) = \sum_{i=1}^{m}(n_i-1)CTV_i(\lambda^*)$.*

Proof. By Property 1, $\bar{C}_i(\lambda^*) = \bar{C}_j(\lambda^*), i \neq j,\ i, j \in I$. Assume that $\bar{C}_i(\lambda^*) = T, \forall i \in I$. Hence, $\bar{\bar{C}}(\lambda^*) = \frac{1}{n}\sum_{i=1}^{m}\sum_{j=1}^{n_i} C_{ij}(\lambda^*) = \frac{1}{n}\sum_{i=1}^{m} n_i\bar{C}_i(\lambda^*) = \frac{1}{n}\sum_{i=1}^{m} n_i T = T = \bar{C}_i(\lambda^*),\ \forall i \in I$.
Thus, by Lemma 1,

$$(n-1)CTV(\lambda^*) = \sum_{i=1}^{m}(n_i-1)CTV_i(\lambda^*) + \sum_{i=1}^{m} n_i(\bar{C}_i(\lambda^*) - \bar{\bar{C}}(\lambda^*))^2 = \sum_{i=1}^{m}(n_i-1)CTV_i(\lambda^*).$$

This completes the proof. $\square$

Property 2. *Under λ^*, the schedule on each machine is optimal.*

Proof. (By contradiction.)

Suppose that the schedules on some machines are not optimal under λ^*. Denote Q as the index set of these machines. We reschedule the jobs on these machines so that the schedules become optimal. Since an optimal schedule for the $Pm|Unres|CTV$ problem should satisfy Property 1, we further adjust idle times on each machine so that the MCT on each machine keeps the same. Denote the adjusted schedule by λ'. It is clear that $CTV_i(\lambda') < CTV_i(\lambda^*)$, $\forall i \in Q$, and $CTV_i(\lambda') = CTV_i(\lambda^*)$, $\forall i \notin Q$, which is based on Lemma 2.

Since Corollary 1 is derived by the condition that the MCT on each machine is the same, we likewise have

$$(n-1)CTV(\lambda') = \sum_{i=1}^{m}(n_i - 1)CTV_i(\lambda'). \tag{2.6}$$

On the other hand, by Corollary 1,

$$\begin{aligned}(n-1)CTV(\lambda^*) =& \sum_{i=1}^{m}(n_i-1)CTV_i(\lambda^*)\\ =& \sum_{i\in Q}(n_i-1)CTV_i(\lambda^*) + \sum_{i\notin Q}(n_i-1)CTV_i(\lambda^*)\\ >& \sum_{i\in Q}(n_i-1)CTV_i(\lambda') + \sum_{i\notin Q}(n_i-1)CTV_i(\lambda')\\ =& \sum_{i=1}^{m}(n_i-1)CTV_i(\lambda')\\ =&(n-1)CTV(\lambda'). \quad \text{(by the expression (2.6))}\end{aligned}$$

That is, $CTV(\lambda^*) > CTV(\lambda')$, which violates the assumption that λ^* is an optimal schedule. So Property 2 holds. □

Corollary 2. *Under λ^*, the job sequence on each machine is V-shaped.*

Proof. This corollary naturally holds, since Eilon and Chowdhury (1977) prove the V-shaped property of an optimal schedule on a single machine. □

Property 3. *Under λ^*, the m largest jobs should be placed on the respective first positions of the m machines.*

Proof. (By contradiction.)

Let $M = \{n, n-1, \cdots, n-m+1\}$ be the set of the largest m jobs. Suppose the property does not hold, then under an optimal schedule λ^*, there must exist at least 3 jobs $r, k, j (r \notin M, k, j \in M)$ such that the job r is scheduled first on some machine t and the jobs k, j are scheduled on a same machine denoted by q. Obviously, r is the largest job among those assigned to the machine t, since the schedule on each machine is optimal under an optimal schedule(*i.e.*, Property 2). Assume $k > j$. The job j is therefore not scheduled on the first position of the machine q by Property 2. Below we prove that interchanging the jobs r and j leads to the decrease of CTV.

Interchange the jobs r and j. Although the processing time of the first job on the machine t increases, by Lemma 3i) CTV_t does not change. On the other hand, interchanging r with j can be regarded as the decrease of the processing time of the job j on the machine q. By Lemma 3ii) this decreases CTV_q. Note that an optimal schedule should satisfy Property 1, so we need to adjust idle times so that the MCT on each machine is the same. By Lemma 2, adjusting idle times does not change CTV on each machine. Thus, if denoting the new schedule after interchange and adjustment by λ', we have $\bar{C}_i(\lambda') = \bar{C}_j(\lambda') = \bar{\bar{C}}(\lambda')$, $i, j \in I, i \neq j$, and $CTV_q(\lambda') < CTV_q(\lambda^*)$, $CTV_i(\lambda') = CTV_i(\lambda^*),\ i \neq q$.

Thus by Lemma 1,

$$\begin{aligned}(n-1)CTV(\lambda') =& \sum_{i=1}^{m}(n_i-1)CTV_i(\lambda') + \sum_{i=1}^{m} n_i\big(\bar{C}_i(\lambda') - \bar{\bar{C}}(\lambda')\big)^2 \\ =& \sum_{i=1}^{m}(n_i-1)CTV_i(\lambda') \\ =& \sum_{i\neq q}(n_i-1)CTV_i(\lambda') + (n_q-1)CTV_q(\lambda') \\ <& \sum_{i=1}^{m}(n_i-1)CTV_i(\lambda^*) \\ =& (n-1)CTV(\lambda^*). \quad \text{(by Corollary 1)}\end{aligned}$$

That is, $CTV(\lambda') < CTV(\lambda^*)$, which violates the assumption that λ^* is an optimal schedule. So Property 3 holds. $\square$

Property 4. *There exist countless optimal schedules for a $Pm|Unres|CTV$ problem. However, there exists an optimal schedule under which there are no idle times before the job processing of some machine(s).*

Proof. Given an optimal schedule λ^*, simultaneously add the same amount of idle time before each machine to obtain a new schedule denoted by λ'. Then by Lemma 2, Property 1 still holds under λ' and $CTV_i(\lambda') = CTV_i(\lambda^*)$, $i \in I$. We can also know that $(n-1)CTV(\lambda') = \sum_{i=1}^{m}(n_i-1)CTV_i(\lambda') = \sum_{i=1}^{m}(n_i-1)CTV_i(\lambda^*) = (n-1)CTV(\lambda^*)$, which implies $CTV(\lambda') = CTV(\lambda^*)$, *i.e.*, λ' is also optimal. Therefore, a different amount of added idle time corresponds to a different optimal schedule, which results in countless optimal schedules.

On the other hand, we can simultaneously subtract the smallest idle time on each machine to obtain an optimal schedule under which at least one machine has zero idle time. □

2.3.3 The Heuristic Algorithm

Hereafter in this section, the optimal schedules we consider refer to the schedules especially mentioned in Property 4, *i.e.*, those under which there exist at least one machine that doesn't have an idle time before it starts to process jobs. One can obtain such schedules by first enumerating all possible schedules without considering idle times, then adding idle times to the appropriated machines so that each schedule satisfies Property 1, and finally choosing among them the one with the minimum CTV.

We first simulate 30 small problem instances with $n = 9$ and the job processing times following the uniform distribution of $Uniform(1, 60)$. We then derive the optimal schedules of these problem instances on $m = 2$ identical parallel machines. Table 2.1 shows some of the computational outputs, from which we observe an interesting phenomenon about an optimal schedule: the sum of job processing times plus idle time on each machine keeps as close to each other as possible.

Without loss of generality, we assume that the n jobs $J_1, J_2, \cdots, J_n$ are sorted in descending order of their processing times. Based on the observation from Table 2.1 and the dominant properties in Subsection 2.3.2, we propose a heuristic algorithm as follows:

Step 1: Following a wavy pattern, assign the jobs J_i, J_{2m-i+1}, J_{2m+i}, J_{4m-i+1}, $\cdots$ to the machine i, $i = 1, 2, ..., m$. The specific assignment pattern is illustrated in Figure 2.1.

Table 2.1: Optimal schedules for 5 small problem instances by exhaustive enumeration, where "DbtS" stands for "Difference between two Sums".

No.	Problem instances	Optimal schedules		Sum	DbtS
1	26, 23, 10, 50, 51, 28, 58, 9, 53	M1:	58, 50, 23, 26	157	1.3
		M2:	(4.70) 53, 51, 9, 10, 28	155.7	
2	25, 28, 37, 5, 19, 11, 38, 15, 36	M1:	38, 28, 15, 25	106	6.75
		M2:	(4.75) 37, 19, 11, 5, 36	112.75	
3	31, 44, 19, 7, 27, 29, 1, 40, 17	M1:	(4.50) 40, 27, 17, 19	107.5	4.5
		M2:	44, 29, 7, 1, 31	112	
4	14, 43, 32, 56, 27, 11, 59, 22, 3	M1:	59, 27, 14, 32	132	7.25
		M2:	(4.25) 56, 22, 11, 3, 43	139.25	
5	13, 28, 5, 52, 34, 20, 23, 53, 12	M1:	53, 23, 13, 34	123	3.55
		M2:	(2.45) 52, 20, 12, 5, 28	119.45	

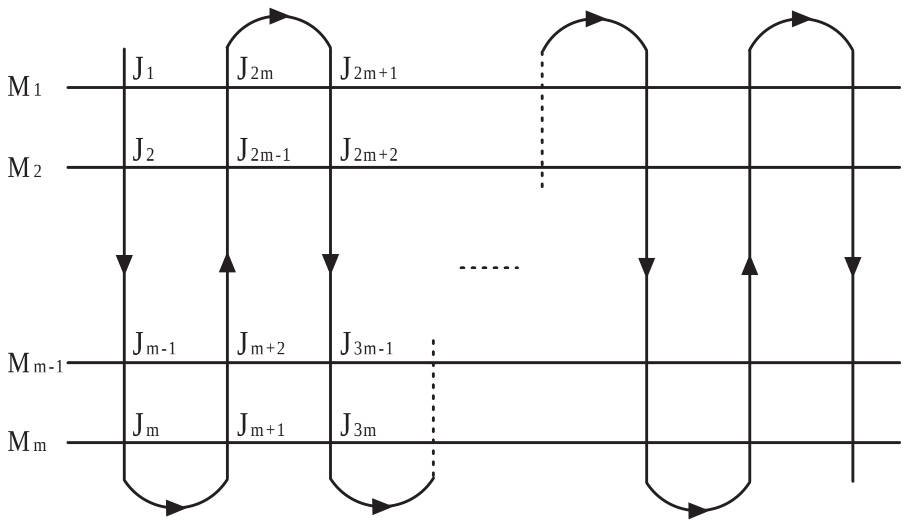


Figure 2.1: The initial assignment of jobs to machines. M$_i$ stands for the i^{th} machine, $i = 1, 2, \ldots, m$. The arrows represent the direction of the assignment.

Step 2: On each machine $i \in I$, schedule the jobs assigned to it as follows:

i) Sort the jobs in descending order of their processing times. Denote these sorted jobs by $\{J_{i1},\ J_{i2},\ \cdots,\ J_{in_i}\}$.

ii) Place J_{i1} on the first position, J_{i2} on the second position, J_{i3} on the last position, and J_{in_i} on the last-but-one position. The current job sequence is $\{J_{i1}, J_{i2}, J_{in_i}, J_{i3}\}$. The remaining jobs include $\{J_{i4}, J_{i5}, \cdots, J_{i(n_i-1)}\}$.

iii) Place the largest one among the remaining jobs to either exactly before or exactly after J_{in_i}, depending on which position achieves a smaller CTV of the current job sequence. If there is a tie, we arbitrarily place it to the left of J_{in_i}.

iv) Repeat iii) until all jobs have been scheduled.

Step 3: Calculate the MCT on each machine. Let $MCT^* = \max_{i \in I} MCT_i$. Then insert an idle time $MCT^* - MCT_i$ before the job processing of each machine i such that the MCT on each machine is the same. The schedule is now complete.

We name the heuristic algorithm as WAVS (Wavy Assignment, Verified Schedule), since we first assign jobs to machines in a wavy pattern and then decide the specific positions of the jobs on each machine by a verification method. Step 2 is similar to the Verified Spiral (VS) method proposed by Ye et al. (2007) for large $1||WTV$ problem instances. Note that when there are a small number (*e.g.*, less than 10) of jobs on each machine after Step 1, we can use exhaustive enumeration to replace Step 2 to find an optimal sequence on each machine as if it were a $1||CTV$ problem. Step 1 in the algorithm WAVS is based on Property 3 and the observation from Table 2.1. Step 2 is inspired by Property 2, and the foundation of Step 3 is Property 1.

2.3.4 Computational Experiments

We test the performance of WAVS by conducting experiments on both small and large problem instances. These problem instances are generated by simulating the processing times of a batch of jobs. Due to the diversity of the distributions of job processing times in real life, we consider four types of probabilistic distributions to simulate job processing times: uniform distribution, normal distribution, exponential distribution, and Pareto distribution. Note that a normal distribution may generate a negative number, which is unrealistic to be a job's processing time. So in this case we take its absolute value. In addition, for the feasibility of comparing the computational outputs, we set the mean value of job processing times from each distribution to be the same.

2.3.4.1 Small Problem Instances

For small problem instances, job processing times are set to follow the following four kinds of distributions: the uniform distribution of $Uniform(1, 59)$, the normal distribution of $Normal(30, 10^2)$, the exponential distribution of $Exponential(30)$, and the Pareto distribution of $Pareto(1.0345, 1)$. The mean job processing time is 30 in these four scenarios.

Since the optimal schedules for small problem instances can be obtained by exhaustive enumeration, we measure the performance of WAVS by comparing CTV_W with CTV_O, where CTV_W and CTV_O denote the CTV's obtained by WAVS and an optimal schedule, respectively. The comparison criterion we take is competitive ratio (CR), which is denoted by CR_O here and is calculated as follows:

$$CR_O = \frac{CTV_W - CTV_O}{CTV_O}$$

The smaller the CR_O, the closer WAVS is to obtain an optimal schedule. Tables 2.2 - 2.5 are some typical computational outputs for small problem instances from the four kinds of distributions.

Table 2.2: The competitive ratios of WAVS versus the optimal solutions for small problem instances from $Uniform(1, 59)$.

No.	Optimal schedules		CTV_O	Schedules by WAVS		CTV_W	CR_O
1	M1:	(8.20) 51, 37, 26, 45	1512.85	M1:	57, 45, 7, 6, 47	1515.84	0.0020
	M2:	57, 48, 6, 7, 47		M2:	(5.25) 51, 37, 26, 48		
2	M1:	(12.90) 45, 22, 21, 34	831.28	M1:	58, 26, 7, 4, 34	840.40	0.0110
	M2:	58, 26, 7, 4, 44		M2:	(8.40) 45, 22, 21, 44		
3	M1:	(4.35) 48, 17, 11, 20	322.74	M1:	(0.30) 49, 20, 4, 1, 22	324.98	0.0069
	M2:	49, 25, 1, 4, 22		M2:	48, 17, 11, 25		
4	M1:	56, 47, 10, 15	737.50	M1:	58, 30, 9, 6, 31	737.50	0.0000
	M2:	(4.00) 58, 30, 9, 6, 31		M2:	(12.00) 56, 15, 10, 47		
5	M1:	51, 30, 10, 29	630.59	M1:	51, 30, 4, 2, 31	632.53	0.0031
	M2:	(4.75) 44, 35, 2, 4, 31		M2:	(4.90) 44, 29, 10, 35		

Table 2.3: The competitive ratios of WAVS versus the optimal solutions for small problem instances from $Normal(30, 10^2)$.

No.	Optimal schedules	CTV_O	Schedules by WAVS	CTV_W	CR_O
1	M1: 57, 30, 22, 33 M2: (6.75) 51, 28, 12, 15, 27	948.09	M1: 57, 28, 15, 12, 30 M2: (8.70) 51, 27, 22, 33	948.98	0.0009
2	M1: (0.65) 45, 38, 29, 39 M2: 40, 36, 22, 25, 32	1719.99	M1: 45, 36, 25, 22, 38 M2: (16.95) 40, 32, 29, 39	1732.44	0.0072
3	M1: (8.15) 34, 29, 20, 30 M2: 41, 28, 11, 18, 21	849.74	M1: 41, 28, 18, 11, 29 M2: (17.15) 34, 21, 20, 30	858.49	0.0103
4	M1: (3.15) 45, 31, 24, 32 M2: 43, 39, 10, 13, 30	1073.74	M1: 45, 31, 13, 10, 32 M2: (0.75) 43, 30, 24, 39	1074.59	0.0008
5	M1: (3.70) 39, 34, 30, 36 M2: 38, 33, 21, 22, 32	1525.98	M1: 39, 33, 22, 21, 34 M2: (7.80) 38, 32, 30, 36	1528.85	0.0019

Table 2.4: The competitive ratios of WAVS versus the optimal solutions for small problem instances from $Exponential(30)$.

No.	Optimal schedules	CTV_O	Schedules by WAVS	CTV_W	CR_O
1	M1: 55, 21, 7, 23 M2: (21.60) 29, 28, 3, 8, 10	333.90	M1: 55, 21, 7, 3, 23 M2: (34.30) 29, 10, 8, 28	335.23	0.0040
2	M1: (70.90) 58, 29, 20, 47 M2: 132, 24, 17, 6, 43	1161.78	M1: 132, 29, 17, 6, 43 M2: (78.65) 58, 24, 20, 47	1162.74	0.0008
3	M1: (73.35) 68, 25, 13, 52 M2: 133, 39, 9, 4, 42	1102.99	M1: 133, 39, 9, 4, 42 M2: (73.35) 68, 25, 13, 52	1102.99	0.0000
4	M1: (46.50) 96, 86, 29 M2: 123, 85, 4, 5, 20, 23	2250.94	M1: 123, 29, 5, 4, 85 M2: (23.05) 96, 23, 20, 86	2255.94	0.0022
5	M1: 110, 22, 9, 21 M2: (77.45) 33, 20, 6, 3, 25	362.19	M1: 110, 21, 6, 3, 22 M2: (77.25) 33, 20, 9, 25	362.34	0.0004

Table 2.5: The competitive ratios of WAVS versus the optimal solutions for small problem instances from $Pareto(1.0345, 1)$.

No.	Optimal schedules	CTV_O	Schedules by WAVS	CTV_W	CR_O
1	M1: (285.07) 153, 51 M2: 449, 8, 4, 3, 2, 5, 6	225.5	M1: 449, 6, 3, 2, 8 M2: (286.50) 153, 5, 4, 51	316.38	0.4028
2	M1: (5.55) 12, 11, 5, 6 M2: 18, 9, 3, 2, 10	70.19	M1: 18, 9, 3, 2, 10 M2: (8.05) 12, 6, 5, 11	70.19	0.0000
3	M1: 53, 5, 4, 16 M2: (27.35) 25, 7, 3, 2, 11	78.99	M1: 53, 7, 3, 2, 11 M2: (28.65) 25, 5, 4, 16	78.99	0.0000
4	M1: (13.25) 28, 15, 2, 18 M2: 40, 17, 3, 4, 5	137.84	M1: 40, 15, 3, 2, 17 M2: (19.75) 28, 5, 4, 18	139.09	0.0091
5	M1: 108, 8, 4, 6 M2: (98.50) 10, 7, 2, 3, 5	41.13	M1: 108, 6, 3, 2, 7 M2: (99.05) 10, 5, 4, 8	41.19	0.0017

Except for $Pareto(1.0345, 1)$, the outputs show the extremely good performance of WAVS, which can generate a near-optimal schedule. For example, for the problem instances tested for $Exponential(30)$ in Table 2.4, the CR_O values are less than 0.004. As to the distribution of $Pareto(1.0345, 1)$, the output of the first instance is abnormal since the resulting CR_O is far greater than the others'. We calculate 30 small problem instances from $Pareto(1.0345, 1)$ and there are only 3 instances whose CR_O's are greater than 0.10. This implies that in general WAVS is able to efficiently reduce CTV for small problem instances from $Pareto(1.0345, 1)$. In sum, WAVS shows great performance for small problem instances, compared to the optimal schedules.

2.3.4.2 Large Problem Instances

For large problem instances, job processing times are set to follow these distributions: $Uniform(1, 999)$, $Normal(500, 100^2)$, $Exponential(500)$, and $Pareto(1.11, 50)$. The mean job processing time is set to 500. Due to the size of large problem instances, it is computationally costly if not impossible to use exhaustive enumeration to obtain the optimal schedules. Thus, we compare the CTV obtained by WAVS with those obtained by some existing scheduling algorithms, including First-Come-First-Served (FCFS), Longest-Processing-Time-First (LPT), Smallest-Processing-Time-First (SPT), and Dynamic Verified Spiral (DVS, refer to (Xu and Ye, 2007)). Since FCFS, LPT, or SPT is generally used for the single-machine scheduling problem, we adjust them so as to accommodate to the multiple identical parallel machine environment. Suppose that the jobs enter the system in the order in which the jobs are generated. Then the Adjusted-FCFS (A-FCFS) algorithm is as follows:

Step 1: Assign the first m jobs to the m machines, with one job on one machine.

Step 2: Assign the next job to the machine that has the smallest sum of processing times of the jobs already assigned to it. According to this rule, assign the remaining jobs to the machines.

Step 3: Adjust the MCT on each machine by inserting appropriate idle times such that the MCT on each machine is the same. The method to calculate the idle times is the same as the Step 3 in WAVS.

The Adjusted-LPT (A-LPT) or Adjusted-SPT (A-SPT) algorithm is as follows:

Step 1: Sort the jobs in descending (or ascending if A-SPT) order of their processing times.

Step 2: Ragard the order of the sorted jobs as the order these jobs enter the system and schedule them according to the above A-FCFS algorithm.

Although DVS is proposed for $Pm||WTV$, Xu and Ye (2007) prove that the optimal value of $Pm||WTV$ is equal to that of $Pm||CTV$ and that any feasible schedule for $Pm||WTV$ can be transformed into a feasible schedule for $Pm||CTV$. Hence, we can directly apply DVS to calculate CTV and compare it with the CTV obtained by WAVS.

The performance tests of WAVS for large problem instances are conducted with the various machine numbers ($m = 2, \ldots, 20$) and job numbers ($n = 100, 200, 300, 400$), for each of the four distributions of job processing times. Specifically, for each combination of a machine number, a job number, and a distribution, we calculate the competitive ratios of A-FCFS (A-LPT, and A-SPT) versus WAVS for 1000 large instances and then average them. While for DVS, we refine $n = 100$ since DVS has very high time complexity as shown later. In a similar way, we calculate and average the competitive ratios of DVS versus WAVS for 1000 large instances. Note that the competitive ratios are computed as follows:

$$CR_Z = \frac{CTV_Z - CTV_W}{CTV_W}, \qquad Z \in \{A\text{-}FCFS, A\text{-}LPT, A\text{-}SPT, DVS\}$$

where CR_Z stands for the competitive ratio of some algorithm Z, while CTV_Z stands for the CTV obtained by the algorithm Z. The larger the CR_Z, the better the WAVS is than the compared algorithms. The computational outputs are presented in Figure 2.2 and Figure 2.3.

Figures 2.2(a), (b), (c) clearly demonstrate that for the problem instances generated from the uniform, normal, and exponential distributions, WAVS is significantly better than (since $CR \gg 0$) these three scheduling algorithms: A-FCFS, A-LPT, and A-SPT. Figure

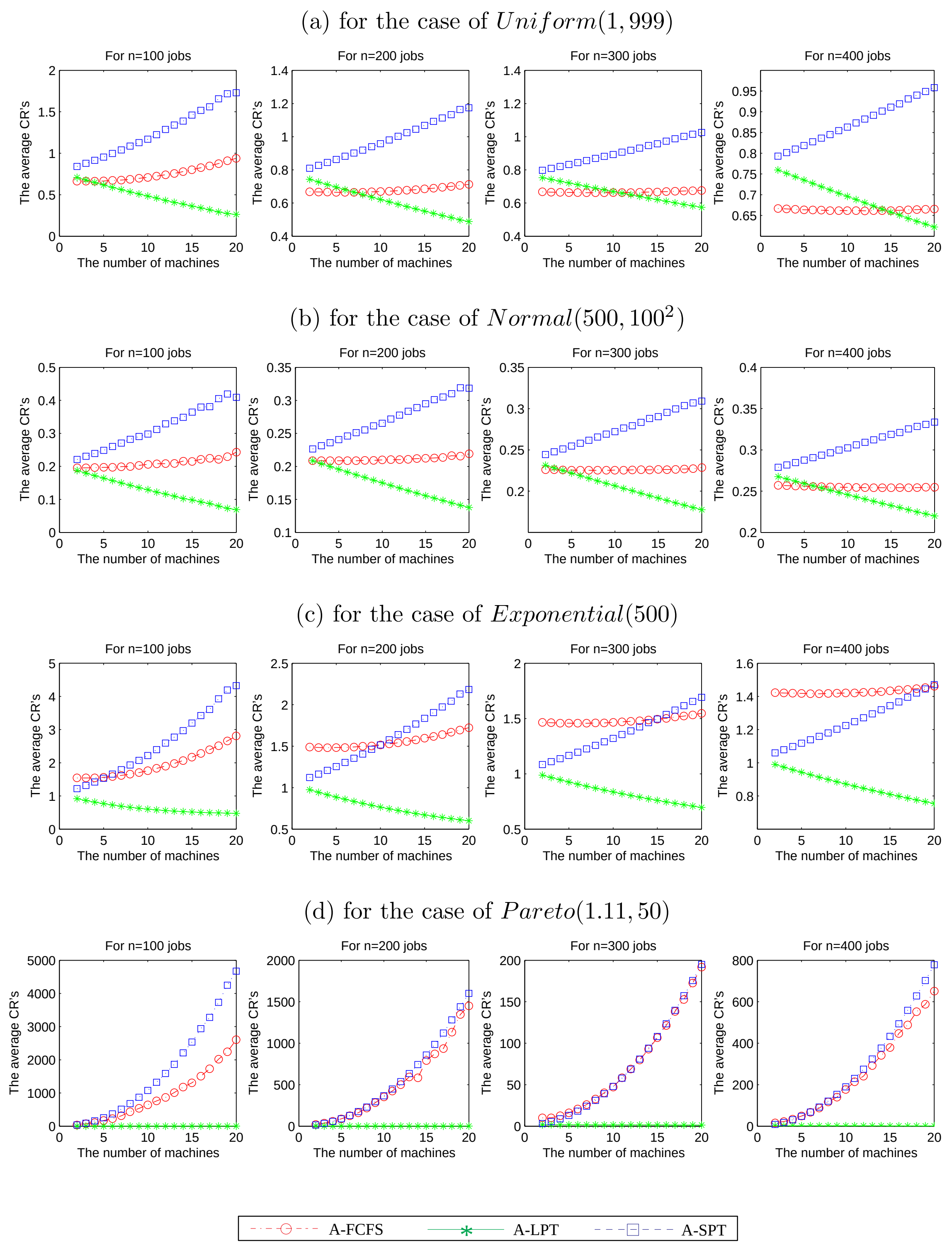


Figure 2.2: The performance of WAVS versus A-FCFS, A-LPT, and A-SPT

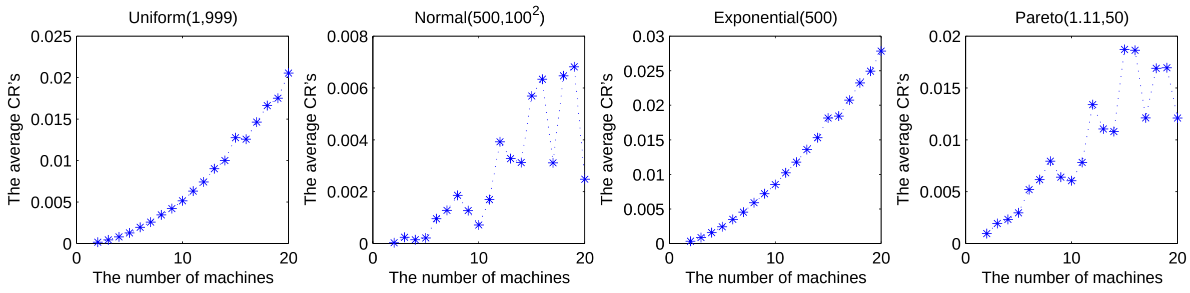


Figure 2.3: The performance of WAVS versus DVS

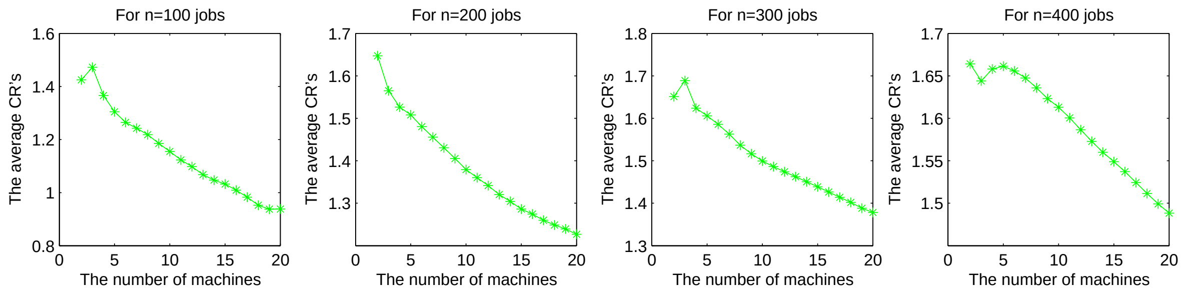


Figure 2.4: The performance of WAVS versus A-LPT (enlarged for the case of $Pareto(1.11, 50)$)

2.2(d) shows the performance of WAVS with respect to the problem instances generated from Pareto distribution. It clearly shows that WAVS dramatically outperforms the algorithms A-FCFS and A-SPT. The largest average CR is even close to 5000. However, simply from Figure 2.2(d) the average competitive ratios of A-LPT versus WAVS seem to be equal to 0. In fact, they are much greater than 0. Figure 2.4 is the enlarged plot of the performance of WAVS versus A-LPT for the Pareto distribution case. It is obvious that WAVS greatly outperforms A-LPT.

Figure 2.3 shows that WAVS consistently but not significantly outperforms DVS with respect to CTV reduction. However, from the perspective of time consumption, WAVS is much better than DVS. It can be easily derived that the time complexity of WAVS is

$O(n^2/m)$ while the time complexity of DVS is $O(n^3/m^2)$. In addition, we record the average computational times of WAVS and DVS for a large problem instance of $n = 100$, regardless of the distribution of job processing times. The record is based on different numbers of machines and is shown in Table 2.6. Each item in Table 2.6 is obtained based on 4000 problem instances. The implementation uses MATLAB on a PC with a 3.20GHz CPU and 1.99GB RAM. As can be seen, the computational time of WAVS is much less than that of DVS. Furthermore, the time consumption for WAVS has a decreasing trend with the increase of the number of machines, while the opposite is for DVS.

In sum, for large problem instances, the proposed algorithm WAVS outperforms the existing algorithms A-FCFS, A-LPT, A-SPT, and DVS with respect to CTV reduction.

2.3.5 Summary

In this section, we consider a multiple identical parallel machine scheduling problem with the objective of minimizing job completion time variance, which is closely related to the JIT philosophy and the service stability concept. It can be applied to many areas such as the Internet data package dispatching and production planning. We investigate the unrestricted case of the problem, denoted by $Pm|Unres|CTV$, in which idle times are allowed to exist before machines start to process jobs. We first prove a dominant property that under an

Table 2.6: The computational time comparison of WAVS versus DVS for a large problem instance of $n = 100$(unit: s)

m	2	5	8	10	15	20
WAVS	0.0207	0.016	0.0141	0.0131	0.0104	0.0083
DVS	1.61	2.9155	3.639	3.8803	4.0175	4.1503

optimal schedule the mean completion time on each machine is the same. We further prove that under an optimal schedule, the schedule on each machine is optimal and the largest m jobs should be scheduled on the respective first positions of the m machines. In addition, from the optimal schedules of some small problem instances, we observe that under an optimal schedule, the difference between the sums of job processing times plus idle time on two machines is rather small. Using this observation and the proven favorable properties, we develop a heuristic algorithm named WAVS, which is shown to generate near optimal schedules for small problem instances and dramatically outperform some existing scheduling algorithms for large problem instances.

2.4 The Restricted Case of $Pm||CTV$: $Pm|Res|CTV$

In this section, we deal with the restricted case of job scheduling on identical and parallel machines, denoted by $Pm|Res|CTV$. The restricted case does not permit the existence of machines' idle times before they start to process jobs. This restricted case is more complicated than its unrestricted peer, since it imposes zero idle times. Similarly, in this section several properties of the $Pm|Res|CTV$ problem are explored and an efficient heuristic algorithm is proposed. The proposed heuristic is compared with the optimal schedules when problem instances are small and is compared with some existing algorithms when problem instances are large.

2.4.1 Problem Statement

This section considers the scheduling problem of n jobs on m identical parallel machines with the objective of minimizing job completion time variance. We tackle the case where no idle times are admitted to exist before machines begin processing jobs. That is, all machines must start to process jobs at time zero. In practice, such requirement may be necessary in order to improve utilization of machines.

Several assumptions are made as follows. First, all jobs are assumed to be ready for scheduling at time zero. Second, no setup time exists between two consecutive jobs or setup times are included in the processing times of jobs. Third, each machine only processes one job at a time. Fourth, machines run continuously, *i.e.*, machines process the next job immediately after finishing the processing of a job. In addition, preemption is not allowed, that is, a job cannot be interrupted once the machine starts to process it.

The notations to be used are defined as follows:

n : the total number of jobs

m : the total number of machines

n_i : the number of jobs assigned to the i^{th} machine, $i = 1, \ldots, m$, $\sum_{i=1}^{m} n_i = n$

λ : a schedule

λ^* : an optimal schedule

$p_{ij}(\lambda)$: the processing time of the j^{th} job on the i^{th} machine under a schedule λ

$C_{ij}(\lambda)$: the completion time of the j^{th} job on the i^{th} machine under a schedule λ

$\bar{C}_i(\lambda)$: the job mean completion time (MCT) on the i^{th} machine under a schedule λ

$\bar{\bar{C}}(\lambda)$: the MCT of all the n jobs under a schedule λ

$CTV(\lambda)$: the CTV of all the n jobs under a schedule λ

With the above notation, the objective of a $Pm|Res|CTV$ problem is to find a schedule λ such that

$$CTV(\lambda) = \frac{1}{n-1}\sum_{i=1}^{m}\sum_{j=1}^{n_i}(C_{ij}(\lambda) - \bar{\bar{C}}(\lambda))^2 \tag{2.7}$$

is the minimum, where

$$\bar{\bar{C}}(\lambda) = \frac{1}{n}\sum_{i=1}^{m}\sum_{j=1}^{n_i} C_{ij}(\lambda)$$

and

$$C_{ij}(\lambda) = \sum_{k=1}^{j} p_{ik}(\lambda), \quad i = 1, \ldots, m; \quad j = 1, \ldots, n_i.$$

The following example illustrates how to calculate the CTV. Suppose that a schedule is of the following form:

M1:	8	25	13	11	7
M2:	23	12	9	6	10

where M1, M2 represent two machines and the numbers represent the processing times of jobs. Then, the corresponding job completion times are calculated as follows:

M1:	8	33	46	57	64
M2:	23	35	44	50	60

CTV can then be easily calculated as $Var(8, 33, 46, 57, 64, 23, 35, 44, 50, 60) = 307.11$.

For many years, scheduling research focuses on regular performance measures, which are nondecreasing in job completion times C_j. Such regular measures include mean completion time, mean lateness, and mean tardiness. In particular, the mean tardiness has been a standard way of measuring conformance to due dates, even though it ignores the consequence of jobs completing early. However, with the increasing interest in Just-In-Time (JIT) concept which espouses the notion that both earliness and tardiness should be discouraged, researchers have begun to study objective functions that are not regular, or nonregular. For example, an objective function as $\sum E_j + \sum T_j$ is nonregular, where $E_j = max(d_j - C_j, 0) = (d_j - C_j)^+$ denotes the earliness and $T_j = max(C_j - d_j, 0) = (C_j - d_j)^+$ denotes the tardiness. d_j and C_j are due date and completion time of job j, respectively. In a JIT scheduling environment, jobs that complete early must be held in finished goods inventory which incurs inventory holding cost, while jobs that complete after their due dates may cause customer dissatisfaction and penalty of agreement violations. So, an ideal schedule is one in which all jobs finish exactly on their assigned due dates, or one in which the performance measure is stabilized according to (Baker and Scudder, 1990) and (Chen et al., 1998). Apparently, CTV is a nonregular performance measure that penalizes both earliness and tardiness.

2.4.2 The Proposed Algorithm

2.4.2.1 Preliminary Results

Let us first observe the structure an optimal schedule should have. We take into account small-sized problem instances, since for such size of problem instances, an optimal schedule can be obtained through exhaustive enumeration. We randomly generate a batch of job sets with job processing times following four kinds of different probability distributions: uniform, triangular, lognormal, and weibull distributions. Then by enumeration, we obtain an optimal schedule for each job set on two identical and parallel machines. If there are more than one optimal schedules, we choose the first one with the minimum CTV obtained from our procedure. Table 2.7 presents the optimal schedules for eight small-sized problem instances, with two from each kind of probability distribution.

Table 2.7: The optimal schedules for eight small-sized job sets by exhaustive enumeration, where "Dbs" stands for "Difference between sums".

No.	Job sets	Optimal schedules	Sums	Dbs
1 (Uniform, 9 jobs)	3, 98, 74, 67, 30, 98, 32, 90, 79	M1: 98, 74, 32, 79 M2: 98, 67, 30, 3, 90	283 288	5
2 (Uniform, 10 jobs)	16, 25, 73, 75, 45, 69, 54, 6, 4, 38	M1: 73, 45, 16, 4, 69 M2: 75, 38, 25, 6, 54	207 198	9
3 (Triangular, 9 jobs)	95, 44, 66, 60, 87, 76, 58, 20, 81	M1: 87, 81, 58, 66 M2: 95, 60, 44, 20, 76	292 295	3
4 (Triangular, 10 jobs)	77, 53, 29, 34, 93, 61, 43, 56, 81, 37	M1: 93, 53, 43, 34, 61 M2: 81, 77, 29, 37, 56	284 280	4
5 (Lognormal, 9 jobs)	76, 72, 53, 48, 59, 57, 52, 65, 76	M1: 76, 72, 57, 65 M2: 76, 53, 48, 52, 59	270 288	18
6 (Lognormal, 10 jobs)	52, 65, 62, 51, 42, 59, 66, 79, 54, 64	M1: 66, 65, 52, 51, 62 M2: 79, 59, 42, 54, 64	296 298	2
7 (Weibull, 9 jobs)	129, 33, 2, 101, 47, 97, 74, 96, 83	M1: 101, 97, 47, 83 M2: 129, 74, 33, 2, 96	328 334	6
8 (Weibull, 10 jobs)	40, 29, 144, 118, 57, 76, 141, 33, 112, 74	M1: 144, 112, 29, 57, 74 M2: 141, 118, 33, 40, 76	416 408	8

From Table 2.7, we can observe three patterns about optimal schedules, regardless of what probability distribution their job processing times follow. First, each of the largest two jobs is placed on the first position of each machine. Second, the job sequence on each machine is of V shape. That is, the jobs before the smallest job are scheduled at the LPT rule, while the jobs after the smallest job are scheduled at the SPT rule. Third, if job processing times are summed up on each machine, then the difference between these two sums is so small. Based on these observations substantiated by numerous problem instances, we make the following conjectures:

Conjecture 1. *Under λ^*, the m largest jobs should be placed on the first positions of the m machines, respectively.*

Conjecture 2. *Under λ^*, the job sequence on each machine is V-shaped.*

Conjecture 3. *Under λ^*, the sum of job processing times on each machine is very close to each other.*

Conjectures 1 and 2 are the proven properties of $Pm|Unres|CTV$ in (Li et al., 2009). It is reasonable to make such conjectures since $Pm|Res|CTV$ can be deemed as a special case of $Pm|Unres|CTV$. If idle times are restricted to zero, $Pm|Unres|CTV$ becomes $Pm|Res|CTV$. Conjectures 3 is similar to the property of $Pm|Unres|CTV$ that $\bar{C}_i(\lambda^*)$ is equal to $\bar{C}_j(\lambda^*)$, $i \neq j$, which is the key to prove other dominant properties of $Pm|Unres|CTV$. However, $\bar{C}_i(\lambda^*)$ does not have to equal $\bar{C}_j(\lambda^*)$ for $Pm|Res|CTV$, which makes the proofs of these conjectures difficult.

2.4.2.2 The Heuristic Algorithm

The $Pm|Res|CTV$ problem is NP-hard since it can be easily reduced to the $1||CTV$ problem which is proven to be NP-hard by Kubiak (1993). So, there is no polynomial time algorithm for this problem. Here we propose a heuristic algorithm, which is inspired by the conjectures in Subsection 2.4.2.1. The algorithm is described as follows.

Algorithm 1: Balanced Assignment, Verified Schedule (BAVS)

```
Sort jobs such that $p_1 \geq p_2 \geq \cdots \geq p_n$.  /* $p_i$ is the processing time of job $i$. */
$S(i) \leftarrow \emptyset$, $i = 1, \ldots, m$; /* $S(i)$ is the job sequence on machine $i$. No jobs are assigned at the beginning. */
for $i \leftarrow 1$ to $m$ do
    $S(i) \leftarrow [p_i]$;  /* Assign the largest $m$ jobs to the first positions of the $m$ machines. */
end
for $j \leftarrow m + 1$ to $n$ do
    for $k \leftarrow 1$ to $m$ do
        $T(k) \leftarrow \sum(S(k))$; /* Calculate the sum of job processing times on machine $k$. */
    end
    $t \leftarrow \mathrm{argmin}\{T(k) : k = 1, \ldots, m\}$;
    $S(t) \leftarrow [S(t), p_j]$;   /* Assign job $j$ to the end of the job sequence on machine $t$. */
end
for $i \leftarrow 1$ to $m$ do
    $S(i) \leftarrow VS(S(i))$;   /* Schedule the jobs on machine $i$ by the Verified Schedule (VS) algorithm. */
end
```

```
Procedure Verified Schedule($VS$)
    input: $J_1, J_2, \cdots, J_u \in S(i)$
        /* $J_1, J_2, \cdots, J_u$ are the jobs assigned to machine $i$.  Assume $u > 4$
    since it is trivial if $u \leq 4$.  */.
 1  Without loss of generality, assume $J_1 \geq J_2 \geq \cdots \geq J_u$.
 2  $L \leftarrow \{J_1, J_2\}$;
        /* $L$ is the left-hand-side job sequence of the smallest job $J_u$ in
    the current sequence so far.  */
 3  $R \leftarrow \{J_3\}$;
        /* $R$ is the right-hand-side job sequence of the smallest job $J_u$ in
    the current sequence so far.  */
 4  for $k \leftarrow 4$ to $u - 1$ do
 5      if $CTV(L, J_k, J_u, R) \leq CTV(L, J_u, J_k, R)$ then
 6          $L \leftarrow [L, J_k]$;        /* The job $J_k$ is assigned to the end of $L$.  */
 7      else
 8          $R \leftarrow [J_k, R]$;        /* The job $J_k$ is assigned in front of $R$.  */
 9      end
10  end
11  $VS(S(i)) \leftarrow [L, J_u, R]$;
```

In the proposed algorithm, we first sort jobs in descending order of job processing times as in Line 1. From Line 3 to Line 5, we assign the first m jobs to m machines respectively. This is based on the Conjecture 1. From Line 6 to 12, we assign the next job to the machine that has the smallest sum of job processing times so far. Repeat this step till all jobs are assigned. Such assignment is based on Conjecture 3 that the sums of the job processing times on different machines are balanced. From Line 13 to Line 15, we schedule jobs on each machine using the Verified Schedule (VS) algorithm (Ye et al., 2007). The VS algorithm is a heuristic algorithm for the $1||WTV$ problem and thereby for the $1||CTV$ problem (due to the interchangeability of these two problems, see (Merten and Muller, 1972)). It generates a V-shaped sequence. Such schedule is based on Conjecture 2. Here completes the proposed algorithm. Since this algorithm involves a balanced assignment and verified schedule, we name it BAVS for simplicity. It can be easily seen that the computational complexity of BAVS is $O[n^2(\lceil \frac{n}{m} + 1 \rceil + 2m)]$.

2.4.3 Computational Results

2.4.3.1 The Deterministic Case

In the deterministic case of job scheduling, job processing times are deterministic and are known in advance. In this subsection, we will evaluate the performance of the BAVS heuristic in the deterministic environment. We consider both small-sized and large-sized problem instances. Problem instances are generated through four kinds of probability distributions. They are uniform distribution, triangular distribution, lognormal distribution, and weibull distribution, respectively.

A) Small-sized Problem Instances

Small-sized problem instances are generated by letting job processing times follow Uniform(1, 119), Triangular(10, 60, 110), Lognormal(60, 10^2), and Weibull(2, 67.7), respectively. For the sake of uniformness, the mean values of these 4 kinds of probability distributions are set to be the same and equal to 60. We also set $m = 2$. For small-sized problem instances, the BAVS heuristic is compared to the optimal schedules obtained by exhaustive enumeration. For a job set of length n, the exhaustive enumeration includes $n!\lfloor \frac{n}{2} \rfloor$ cases when $m = 2$, where $\lfloor x \rfloor$ denotes the biggest integer not greater than x. Therefore, due to computational costs, we only consider $n = 9$ and $n = 10$. Using MATLAB on a Pentium 4 PC with a 3.2GHz CPU and 2GB RAM, the average running time of obtaining an optimal schedule for a problem instance is 2.14 minutes for $n = 9$ and 25.23 minutes for $n = 10$. Both are based on 50 cases, respectively.

The comparison criterion for small-sized problem instances is competitive ratio (CR), which can be calculated by:

$$CR_O = \frac{CTV_B - CTV_O}{CTV_O} \times 100\%$$

where CTV_B and CTV_O denote the CTV's under BAVS and under an optimal schedule, respectively. The smaller the CR_O, the closer the BAVS is to be optimal. Table 2.8 is eight typical computational outputs for small-sized problem instances.

For further illustration, we randomly choose 50 outputs from each kind of probability distribution and plot CR_O over instance as in Figure 2.5.

Table 2.8: The comparison of the BAVS heuristic with optimal schedules for eight small-sized instances

No.	Optimal schedules	CTV_O	BAVS schedules	CTV_B	CR_O
1 (Uniform, 9 jobs)	M1: 102, 55, 25, 101 M2: 104, 45, 38, 10, 67	3919.28	M1: 104, 67, 10, 38, 55 M2: 102, 101, 25, 45	3969.50	1.28%
2 (Uniform, 10 jobs)	M1: 112, 75, 53, 26, 89 M2: 100, 87, 50, 32, 82	7528.04	M1: 112, 87, 32, 50, 75 M2: 100, 89, 26, 53, 82	7556.99	0.38%
3 (Triangular, 9 jobs)	M1: 87, 81, 58, 66 M2: 95, 60, 44, 20, 76	5614.36	M1: 95, 76, 58, 60 M2: 87, 81, 20, 44, 66	5623.44	0.16%
4 (Triangular, 10 jobs)	M1: 93, 53, 43, 34, 61 M2: 81, 77, 29, 37, 56	4813.17	M1: 93, 61, 29, 43, 56 M2: 81, 77, 34, 37, 53	4818.67	0.11%
5 (Lognormal, 9 jobs)	M1: 70, 62, 51, 60 M2: 63, 55, 42, 49, 57	5138.78	M1: 70, 60, 49, 55 M2: 63, 62, 42, 51, 57	5257.50	2.31%
6 (Lognormal, 10 jobs)	M1: 70, 60, 50, 49, 59 M2: 76, 56, 48, 51, 57	6227.60	M1: 76, 59, 48, 50, 56 M2: 70, 60, 49, 51, 57	6230.62	0.05%
7 (Weibull, 9 jobs)	M1: 148, 66, 49, 82 M2: 107, 93, 35, 45, 73	6794.00	M1: 148, 82, 45, 66 M2: 107, 93, 35, 49, 73	6862.11	1.00%
8 (Weibull, 10 jobs)	M1: 60, 56, 38, 9, 58 M2: 70, 41, 34, 31, 47	3117.88	M1: 70, 56, 31, 34, 41 M2: 60, 58, 9, 38, 47	3219.51	3.26%

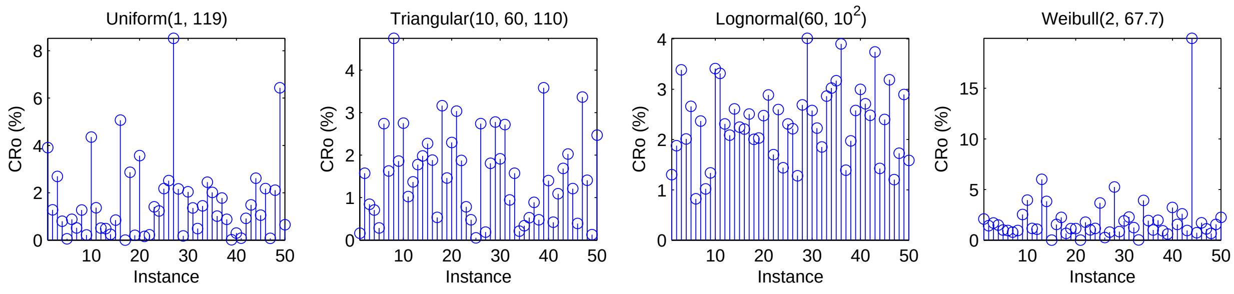


Figure 2.5: The CRo's for small-sized problem instances with different probability distributions

From Table 2.8 and Figure 2.5, we can easily see that for small-sized problem instances, the effect of the BAVS heuristic is rather good when compared to the optimal schedules.

B) Large-sized Problem Instances

For large-sized problem instances, we let job processing times follow the four kinds of probability distributions: Uniform(1, 999), Triangular(10, 550, 940), Lognormal(500, 100^2), and Weibull(2, 564.19), respectively. The mean value is set to be 500 for each kind of probability distribution. We consider four different numbers of jobs: $n = 100, 200, 300, 400$. For each combination of distribution type and n value, we generate 1000 large-sized problem instances respectively. Furthermore, for each problem instance, we take into account various different numbers of machines: m takes values from 2 to 20.

To evaluate the performance, the BAVS heuristic is compared to some existing algorithms, including three basic dispatching rules and three extended existing heuristics. The three dispatching rules are: first-come-first-served(FCFS) rule, longest-processing-time-first(LPT) rule, and shortest-processing-time-first(SPT) rule. The three existing heuristics compared here include EC (Eilon and Chowdhury, 1977), MP (Manna and Prasad, 1999), and GGB

(Gupta et al., 1990). These three heuristics are not originally developed for the $Pm||CTV$ problem, but for the $1||CTV$ problem. Therefore, in order for the comparison with BAVS, they are extended to the multiple-machine environment. Take the heuristic of EC for example. We first apply the EC heuristic to derive an optimal schedule for the $1||CTV$ problem. Then according to FCFS rule, these jobs are assigned to the machines. That is, when some machine is available, the current first job in the generated sequence is assigned to that machine for processing. If $m = 1$, the extended heuristics become the original ones.

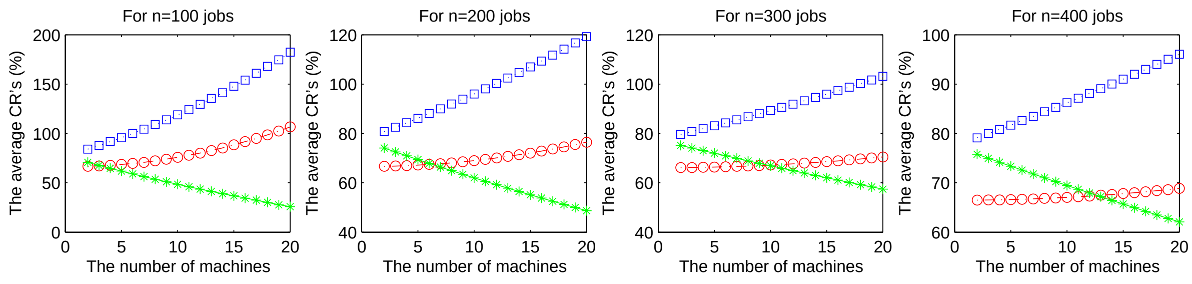
The comparison criteria are still competitive ratios, but the formula change as follows:

$$CR_Z = \frac{CTV_Z - CTV_B}{CTV_B} \times 100\%, Z \in \{FCFS, LPT, SPT, E\_EC, E\_MP, E\_GGB\}$$

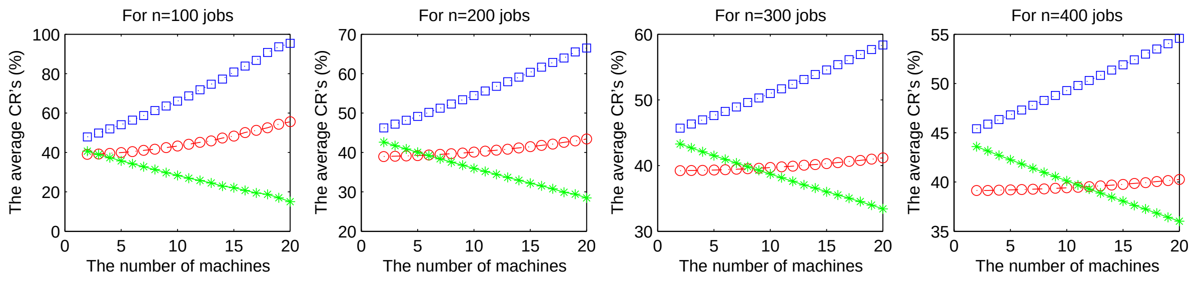
where CR_Z stands for the competitive ratio of the algorithm Z to the BAVS heuristic, CTV_Z stands for the CTV under the algorithm Z, and E_EC, E_MP, E_GGB refer to the extended EC, MP, GGB heuristics, respectively. The larger the CR_Z, the better the BAVS is than the algorithm Z. By computation, we find that the CRs are greater than 0 for all considered problem instances, which indicates that BAVS is always better than these rules and heuristics for the tested problem instances. Since the computational complexity of E_EC, E_MP, E_GGB is much higher than that of those three dispatching rules, we separate the comparison of BAVS with the three dispatching rules and the three heuristics.

For the comparison with FCFS, LPT, and SPT, we calculate and plot the average of CRs of 1000 instances for each combination of m value, n value, and distribution type. Figures 2.6(a) through 2.6(d) are the plots of the average CR versus the number of machines,

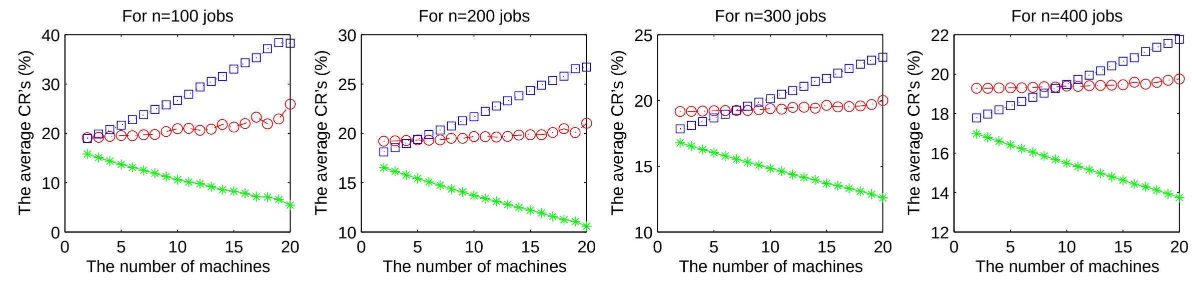
(a) For the case of $Uniform(1, 999)$



(b) For the case of $Triangular(10, 550, 940)$



(c) For the case of $Lognormal(500, 100^2)$



(d) For the case of $Weibull(2, 564.19)$

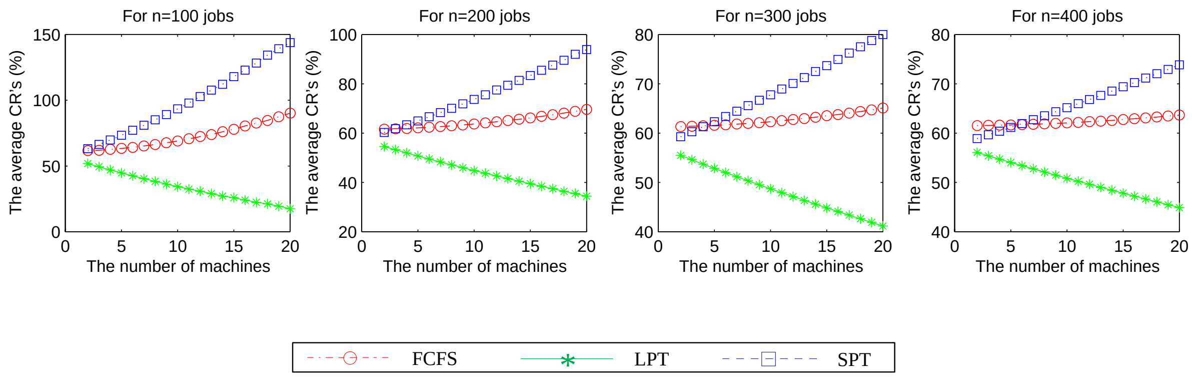


Figure 2.6: The performance of BAVS versus FCFS, LPT, and SPT

according to different distribution types and different numbers of jobs. These plots clearly show that BAVS is remarkably better than FCFS, LPT, and SPT rules. In some case, CR_Z is even close to 200%, which means that the CTV under the algorithm Z is almost three times that under BAVS. Moreover, we can observe three trends: BAVS is increasingly better than SPT and FCFS and is decreasingly better than LPT when m increases.

For the comparison with E_EC, E_MP, and E_GGB, we also consider four kinds of probability distributions, but only consider the case of $n = 100$ jobs. For each kind of probability distribution, we generate 100 problem instances, based on which the average CR's are calculated. The value of m is from 2 to 20 as well. Figure 2.7 is the obtained

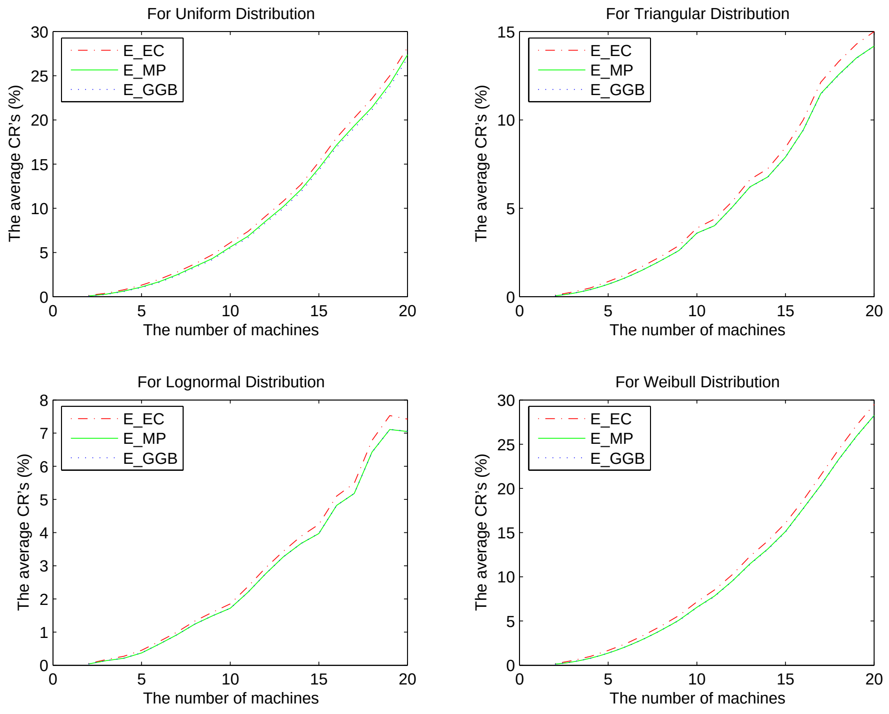


Figure 2.7: The performance of BAVS versus E_EC, E_MP, and E_GGB

CR plot. We observe that BAVS also outperforms these extended heuristics. The more machines, the larger the average CR's. This means that the advantage of BAVS over the three heuristics is more significant when the number of machines increases. We also observe that from each subplot, corresponding to a same m, the average CR's are very close to each other, regardless of what kind of heuristic is compared. This may be because that FCFS is applied in the extensions of all three heuristics, which reduces the difference among the heuristics. Although so, the comparison result demonstrates the advantage of BAVS.

2.4.3.2 Special Scenario where $m = 1$

If we reduce the number of machines to one, the proposed heuristic is actually the VS algorithm for $1||CTV$ problems (Ye et al., 2007). The VS algorithm is presented to outperform FCFS, SPT, and Methods 1.1 & 1.2 of Eilon and Chowdhury (1977) in the single machine situation. Furthermore, the mean CTV derived by the VS algorithm is shown to be extremely close to the lower bound developed in (Kubiak et al., 2002). This lower bound is for the $1||CTV$ problem and is as follows:

$$LB = \begin{cases} \frac{1}{n-1}\sum\limits_{i=1}^{(n-1)/2}(p_1+p_2+\cdots+p_{2i})^2/2 & \text{if } n \text{ is odd,} \\ \frac{1}{n-1}\sum\limits_{i=1}^{(n-2)/2}(p_1+p_2+\cdots+p_{2i+1})^2/2 & \text{if } n \text{ is even.} \end{cases}$$

We compare the performance of VS with MP and GGB in this subsection, in order to assess the proposed heuristic from another perspective. Four kinds of probability distributions and two cases of $n = 20$ and $n = 100$ are taken into account. For each combination of n and distribution type, 1000 problem instances are generated. For each problem instance, the CTV's under VS, MP, and GGB are calculated and compared. In Table 2.9, we present the comparison result, where N_{VS_M} and N_{VS_G} stand for the numbers of problem instances (out of 1000) for which VS is better than MP and GGB, respectively. N_{MP} and N_{GGB} stand for the numbers of problem instances (out of 1000) for which MP and GGB outperform VS, respectively. N_{EQ_M} and N_{EQ_G} are the numbers of problem instances (out of 1000) for which VS and the compared heuristic have the same performance. For example, corresponding to the scenario of $n = 20$ and uniform distribution, among 1000 problem instances, there are 176 problem instances for which VS is better than MP, 158 instances for which MP is better than VS, and 666 instances for which VS and MP have the equal CTVs. Table 2.9 shows that VS and MP have the approximate performances, but the computational complexity of VS is apparently lower than that of MP (Manna and Prasad, 1999; Ye et al., 2007). Table 2.9 also shows that VS significantly outperforms GGB for large job sets and is worse than GGB for small job sets. These indicate the effectiveness of VS.

Table 2.9: The comparison of VS with MP and GGB, which is based on 1000 problem instances for each combination of n and distribution type.

n	Distribution	VS versus MP			VS versus GGB		
		N_{VS_M}	N_{MP}	N_{EQ_M}	N_{VS_G}	N_{GGB}	N_{EQ_G}
20	Uniform	176	158	666	237	661	102
	Triangular	191	199	610	165	699	136
	Lognormal	187	209	604	213	499	288
	Weibull	169	187	644	200	633	167
100	Uniform	352	316	332	946	54	0
	Triangular	325	304	371	691	309	0
	Lognormal	327	379	294	626	374	0
	Weibull	312	322	366	780	220	0

2.4.3.3 Special Problem Instances

In the case that a job's processing time is far larger than those of other jobs, the LPT algorithm appears to be optimal for the $Pm|Res|CTV$ problem. The following are two examples.

Example 1: Schedule a job set $J = \{2, 9, 3, 17, 27, 4, 24, 10, 392\}$ on 2 identical and parallel machines. An optimal schedule with the minimum CTV is:

```
M1:   392
M2:    27    24    17    10    9    4    3    2
```

where the CTV is 11752.5. Clearly, this schedule is of LPT rule.

Example 2: Schedule a job set $J = \{19, 11, 22, 579, 17, 6, 7, 8, 15\}$ on 2 identical and parallel machines. An optimal schedule with the minimum CTV is:

```
M1:   579
M2:    22    19    17    15    11    8    7    6
```

where the CTV is 29341.61. This schedule is of LPT rule, too.

2.4.4 Summary

This section considers the scheduling problem of n jobs on m identical and parallel machines so as to minimize job completion time variance (CTV), which is closely related to service uniformness and stability. We focus on the restricted case, which does not allow idle time insertion before machines start to process jobs. By observing the patterns of optimal schedules for various problem instances, we conjecture three properties of the optimal schedules and propose a simple but efficient heuristic algorithm, denoted by BAVS. The performance of the BAVS heuristic is tested in the deterministic environment. Both small-sized and large-sized problem instances are considered. These problem instances are generated by four kinds of probability distributions: uniform, triangular, lognormal, and weibull distributions. The computational results show that the BAVS heuristic is near-optimal for small-sized problem instances and significantly outperforms some existing scheduling algorithms such as FCFS, LPT, SPT, and the extended EC, MP, GGB heuristics for large-sized problem instances. The performance of BAVS is studied as well in the special scenario where $m = 1$. We also discuss the optimal scheduling method for some special problem instances.

2.5 Class-based CTV on a Single Machine: $1||CB\text{-}CTV$

2.5.1 Motivation

Previous literature on CTV minimization deals with problems mainly from the viewpoint of the system. In this point of view, jobs are assumed to be independent of each other, which is often not practical in the real world. In general, some jobs are related to some other jobs. For instance, jobs requested by the same user, such as multiple requests to a web server from the same client, often need to be considered together as a class. Thus, the system CTV performance measure may result in the dissatisfaction of a certain user with the service, because CTV of the jobs belonging to this user may be very large even though the overall CTV of all jobs in the system has reached the minimum. Consequently, the dissatisfaction with the service may cause the user to leave a system and turn to its rival. Such a result is undesirable to service providers. It is necessary, therefore, to investigate CTV minimization problems from the viewpoint of users. The Class-based CTV (CB-CTV) minimization problem arises accordingly. CB-CTV is closely related to service stability since it penalizes both earliness and tardiness, and it is further related to customer satisfaction because it takes into account customer preferences. CB-CTV minimization has wide applications in many areas such as packet scheduling for Internet communications and reservation systems, modern manufacturing systems, supply chain management, and others where it is desirable to achieve service stability while considering customer preference. Since CTV is important from the perspective of the system, reducing the overall CTV is taken as the secondary objective in this section.

In the following we present several dominant properties for CTV problems and prove that CB-CTV problems can be transformed into a series of CTV problems on a single machine. In addition, computational results are presented for both small and large problem instances and a trade-off relationship between CB-CTV and the overall CTV is revealed.

2.5.2 Problem Definition

In this section, we consider the problem of scheduling L-class jobs on a single machine. All jobs are released at time zero and their processing times are known deterministically. Preemption is not allowed, *i.e.*, jobs cannot be interrupted during their processing. Also, we assume that there is no setup time between two consecutive jobs. These assumptions are the same as those adopted by Eilon and Chowdhury (1977); Kanet (1981); Merten and Muller

(1972). Our objective is to find an optimal scheduling sequence that minimizes

$$CB\text{-}CTV = \sum_{i=1}^{L} \frac{n_i}{n} CTV_i \tag{2.8}$$

where L is the number of classes, n is the total number of all jobs, n_i is the number of jobs in the i^{th} class, and CTV_i is the CTV of jobs in the i^{th} class. CTV_i is computed as follows:

$$CTV_i = \frac{1}{n_i - 1} \sum_{j=1}^{n_i} (C_{ij} - \bar{C}_i)^2 \tag{2.9}$$

where C_{ij} is the completion time of the j^{th} job in the i^{th} class and $\bar{C}_i$ is the mean completion time of the jobs in the i^{th} class.

For illustration, we give an example as follows. Suppose that there are three classes of jobs required to be scheduled on a single machine. These jobs are as follows:

Class I: *20, 5*; Class II: *14, 2, 12*; Class III: *8, 4, 1, 16*

Here and throughout the section, we denote jobs by their processing times. Assume they are scheduled in the following way: *12, 1, 4, 20, 14, 8, 5, 16, 2.* Then the completion times of the jobs in these three classes are 37, 64; 51, 82, 12; and 59, 17, 13, 80 respectively. Hence, the CTVs of the three classes are 364.5, 1230.3, and 1066.3, respectively. Thus, the CB-CTV of this scheduling sequence is $\frac{2}{9} * 364.5 + \frac{3}{9} * 1230.3 + \frac{4}{9} * 1066.3 = 965$.

Using exhaustive enumeration, we can obtain an optimal sequence of the above example that minimizes CB-CTV and takes reducing the overall CTV as the secondary objective. This optimal schedule is *20, 5, 16, 4, 1, 8, 14, 2, 12.* The obtained minimum CB-CTV is

35.07 and the respective CTVs of the three classes are 12.5, 57.33, and 29.67. On the other hand, the overall CTV of this optimal sequence is 426.36. If we schedule these jobs without the consideration of the classes, we can obtain an optimal sequence which has the minimum overall CTV through enumeration. This sequence is *20, 16, 8, 5, 2, 1, 4, 12, 14* and the corresponding overall CTV is 314.36, while the CB-CTV of this sequence is 208.74 with the respective inner-class CTVs of 420.5, 241, and 78.67. We summarize these results into Table 2.10, where * denotes the optimal value.

From Table 2.10, we observe that when the overall minimum CTV is desired, CB-CTV is not minimized. The corresponding CB-CTV (208.74) has a large deviation from the possible minimum CB-CTV (35.07). The jobs from the same class receive the greatly different treatment, which are represented by their large inner-class CTVs (420.5, 241, 78.67) compared with (12.5, 57.33, 29.67). This implies that the jobs of the same class under CB-CTV minimization gain stabler services than under the overall CTV minimization without the consideration of classes. This inner-class CTV reduction leads to user satisfaction in the viewpoint of users with regard to service stability. It is the difference between the overall CTV and CB-CTV that motivates our research on the CB-CTV minimization problem.

Table 2.10: An example of a small problem instance for CB-CTV and overall CTV minimization.

	optimal sequence	CTV_1	CTV_2	CTV_3	*CB-CTV*	*CTV*
Class-based	20, 5, 16, 4, 1, 8, 14, 2, 12	12.5*	57.33*	29.67*	35.07*	426.36
Non-class-based	20, 16, 8, 5, 2, 1, 4, 12, 14	420.5	241	78.67	208.74	314.36*

2.5.3 Dominant Properties for CTV and CB-CTV Problems

The CTV problem has been discovered to have a number of dominant properties. The following properties are summarized from the literature.

Property 5. *For any scheduling sequence R, CTV of R is equal to WTV of R', where R' is the antithetical schedule of R (Theorem H in (Merten and Muller, 1972)).*

Property 6. *The scheduling sequence that minimizes WTV is antithetical to the scheduling sequence that minimizes CTV (Corollary H.1 in (Merten and Muller, 1972)).*

Property 7. *CTV remains unchanged when reversing the order of the last $n-1$ jobs (Theorem K in (Merten and Muller, 1972)).*

Property 8. *For CTV minimization problems, an optimal scheduling sequence is of the form of $(n, n-2, \ldots, n-1)$. That is, the largest job is arranged at the first position, the second longest job is arranged at the last position, and the third longest job is arranged at the second position (Theorem 1 in (Hall and Kubiak, 1991)).*

Property 9. *The optimal sequence for a WTV minimization problem is V-shaped (Theorem B in (Eilon and Chowdhury, 1977)).*

Property 10. *The optimal sequence for a CTV minimization problem is V-shaped. (the combination of Property 6 and Property 9)*

In view of these properties, it will be very desirable if a CB-CTV minimization problem can be transformed into CTV minimization problems. If so, we can apply these properties to solve the CB-CTV minimization problem. We will prove this property later.

We use the following notation to represent job scheduling sequences:

p_{ij} = the processing time of the j^{th} processed job in the i^{th} class;

X_i = the i^{th} job block that separates the jobs in a certain class.

To illustrate our notation, consider only the q^{th} class. A possible scheduling sequence may be of the following form:

$$p_{21}, p_{31}, p_{q1}, p_{q2}, p_{22}, p_{11}, p_{32}, p_{q3}, p_{L1}, p_{23}, p_{q4}, p_{q5}, p_{q6}, \ldots, p_{Ln_L}, p_{qn_q}, p_{36}, p_{2n_2}, \ldots$$

Then we can denote this scheduling sequence by the following:

$$\boxed{X_0}, p_{q1}, p_{q2}, \boxed{X_1}, p_{q3}, \boxed{X_2}, p_{q4}, p_{q5}, p_{q6}, \boxed{X_3}, \ldots, \boxed{X_{s-1}}, p_{qn_q}, \boxed{X_s} \tag{2.10}$$

where s is an appropriate integer.

Lemma 4. *CTV_q is smaller in the following schedule than in Schedule (2.10):*

$$\boxed{X_0}, p_{q1}, p_{q2}, \ldots, p_{qn_q}, \boxed{X_1}, \boxed{X_2}, \boxed{X_3}, \ldots, \boxed{X_{s-1}}, \boxed{X_s} \tag{2.11}$$

Proof. First, we prove a special case: there is only one block among the q^{th} class in Schedule (2.10). That is, CTV_q is smaller in the schedule

$$\boxed{X_0}, p_{q1}, p_{q2}, \ldots, p_{q(m-1)}, p_{qm}, \ldots, p_{qn_q}, \boxed{X_1}, \boxed{X_2} \tag{2.12}$$

than in the schedule

$$\boxed{X_0}, p_{q1}, p_{q2}, \ldots, p_{q(m-1)}, \boxed{X_1}, p_{qm}, \ldots, p_{qn_q}, \boxed{X_2} \tag{2.13}$$

where $2 \leq m \leq n_q$.

The following notation is used in the proof of this special case:

k: the sum of processing times of jobs in the block X_1;

V: CTV_q by Schedule (2.12);

V': CTV_q by Schedule (2.13);

C_i: completion time of job p_{qi} in Schedule (2.12), $i = 1, 2, \ldots, n_q$;

C'_i: completion time of job p_{qi} in Schedule (2.13), $i = 1, 2, \ldots, n_q$.

It is easy to show that

$$\begin{aligned} C'_i &= C_i, & i &= 1, 2, \ldots, m-1 \\ C'_i &= C_i + k, & i &= m, m+1, \ldots, n_q \end{aligned}$$

According to Kanet (1981), CTV has an alternative form: $V = \sum_{i=1}^{n_q} \sum_{j=i+1}^{n_q} (C_j - C_i)^2$. Hence,

$$\begin{aligned} V' - V &= \sum_{i=1}^{n_q} \sum_{j=i+1}^{n_q} (C'_j - C'_i)^2 - \sum_{i=1}^{n_q} \sum_{j=i+1}^{n_q} (C_j - C_i)^2 \\ &= \sum_{i=1}^{m-1} \sum_{j=m}^{n_q} (C_j + k - C_i)^2 - \sum_{i=1}^{m-1} \sum_{j=m}^{n_q} (C_j - C_i)^2 \\ &= \sum_{i=1}^{m-1} \sum_{j=m}^{n_q} [(C_j + k - C_i)^2 - (C_j - C_i)^2] \\ &= \sum_{i=1}^{m-1} \sum_{j=m}^{n_q} [k(2C_j - 2C_i + k)] \\ &> 0 \ (\textit{since}\ \ k > 0\ \ \textit{and}\ \ C_j > C_i) \end{aligned}$$

So, CTV_q is smaller in Schedule (2.12) than in Schedule (2.13).

Next we prove the lemma. First we move the block X_{s-1} in Schedule (2.10) to the exact back of p_{qn_q}. According to the above special case, the new schedule has a smaller CTV_q. Again, by moving the block X_{s-2} to the exact back of p_{qn_q}, we obtain a schedule in which CTV_q is smaller than in the last schedule. Keep moving blocks in this fashion until the block X_1 is moved and Schedule (2.11) is obtained. Since the schedule after each move produces a smaller CTV_q than in the former schedule, Schedule (2.11) has a smaller CTV_q than Schedule (2.10). $\square$

Lemma 5. *CTV_q keeps a constant, as long as the scheduling form satisfies: i)No jobs from other classes are scheduled among the q^{th} class, i.e., no blocks exist among the q^{th} class; and ii)The inner-class scheduling order of the q^{th} class keeps unchanged.*

Proof. Let S_1 and S_2 be any two schedules that satisfy the above two conditions. Denote job completion times of the q^{th} class in S_1 by $\{C_1, C_2, \ldots, C_{n_q}\}$. Then job completion times of the q^{th} class in S_2 will be $\{C_1 + h, C_2 + h, \ldots, C_{n_q} + h\}$, where h is an appropriate real number. Let $\bar{C}'$, CTV' and $\bar{C}$, CTV be mean completion times and CTVs of the q^{th} class in S_1 and S_2 respectively. Then

$$\begin{aligned}
\bar{C}' &= \frac{1}{n_q}\sum_{i=1}^{n_q}(C_i + h) = \frac{1}{n_q}\sum_{i=1}^{n_q} C_i + h = \bar{C} + h \\
CTV' &= \frac{1}{n_q - 1}\sum_{i=1}^{n_q}(C_i + h - \bar{C}')^2 = \frac{1}{n_q - 1}\sum_{i=1}^{n_q}(C_i - \bar{C})^2 = CTV
\end{aligned}$$

This completes the proof. $\square$

Theorem 1. *Regardless of the intra-class scheduling order, the scheduling form*

$$\boxed{p_{11}, \ldots, p_{1n_1}}, \boxed{p_{21}, \ldots, p_{2n_2}}, \ldots, \boxed{p_{L1}, \ldots, p_{Ln_L}} \tag{2.14}$$

has a smaller CB-CTV than any scheduling form that has the same inner-class scheduling order as Schedule (2.14) and in which there exists at least one class whose jobs are not scheduled consecutively.

Proof. Consider a schedule in which there is at least one class whose jobs are not scheduled consecutively. Gather the jobs of the same class at the position where that class first appears, for scheduling consecutively and without changing inner-class scheduling order. Then the scheduling form will become Schedule (2.14) or a similar schedule that only changes intra-class scheduling order, compared with Schedule (2.14). Lemma 5 guarantees that the change of intra-class order does not change every class's CTV. Thus, Schedule (2.14) and similar schedules have the same CB-CTV. On the other hand, Lemma 4 indicates that, every class's CTV in Schedule (2.14) or a similar schedule is smaller than in the original schedule. Hence, by definition, CB-CTV of Schedule (2.14) or a similar schedule is smaller. □

Corollary 3. *A CB-CTV minimization problem can be transformed into a series of CTV minimization problems. That is, the following equation holds:*

$$\underset{\lambda \in \Lambda}{Min}\left(\sum_{i=1}^{L} \frac{n_i}{n} CTV_i(\lambda)\right) = \sum_{i=1}^{L}\left(\frac{n_i}{n} \underset{\lambda_i \in \Lambda_i}{Min}(CTV_i(\lambda_i))\right) \tag{2.15}$$

where $\lambda, \Lambda,$ *and* $CTV_i(\lambda)(i = 1, \ldots, L)$ *are respectively a schedule of all jobs of* L *classes, the schedule set composed of all possible* λ*, and the CTV of the* i^{th} *class under the schedule* λ*, while* $\lambda_i(i = 1, \ldots, L), \Lambda_i(i = 1, \ldots, L),$ *and* $CTV_i(\lambda_i)(i = 1, \ldots, L)$ *are respectively a schedule of all jobs of the* i^{th} *class, the schedule set composed of all possible* λ_i*, and the CTV of the* i^{th} *class under the schedule* λ_i*.*

Proof. Since Schedule (2.14) or a similar schedule that only changes intra-class scheduling order has a smaller CB-CTV, as long as every class's jobs are further scheduled in the way such that the class's CTV is at a minimum, a minimal CB-CTV is achieved. □

According to Corollary 3, to obtain the optimal scheduling sequence with the minimum CB-CTV, we only need to schedule jobs by their classes and schedule jobs of each class in the way that their inner-class CTVs are the minimum. This transformation dramatically simplifies the problem since there have been a lot of heuristics that can be used for CTV minimization problems, such as those in (Eilon and Chowdhury, 1977; Kanet, 1981; Manna and Prasad, 1997, 1999; Vani and Raghavachari, 1987).

2.5.4 Computational Results

We have bridged CB-CTV minimization problems with CTV minimization problems. Nevertheless, it is necessary to investigate the relationship between the overall CTV and CB-CTV. In this section, we compute scheduling sequences for the overall CTV and CB-CTV minimization respectively for the same small or large program instances. Assume that the processing times of jobs follow a uniform distribution, taking values from the integers between 1 and 20 for small problem instances and the integers between 1 to 150 for large problem instances.

2.5.4.1 Small Problem Instances

For small instances, we consider two instances from $n = 9$ jobs, $L = 3$ classes and two instances from $n = 10$ jobs, $L = 5$ classes respectively. In Table 2.11 there are four small problem instances and job classes are distinguished by semicolons.

Similar to the example in subsection 2.5.2, the overall CTV and CB-CTV of these problem instances cannot be minimized at the same time. Because of their small size, we can use exhaustive enumeration to obtain optimal sequences with CTV minimization. The computation of the minimum CB-CTV is based on Corollary 3 and realized by using enumeration to obtain the minimum CTV of each class and combining them. Since there is a total of $(2^L \cdot L!)$ optimal sequences for CB-CTV minimization, we choose the one with the smallest CTV. The computational output is shown in Table 2.12, where * denotes the optimal value, `"N/A"` stands for `"not applicable"`, CB means that the optimal sequences are obtained under CB-CTV minimization, and NCB means that the optimal sequences are obtained under the overall CTV minimization (*i.e.*, non-class-based CTV minimization).

Table 2.11: Four small problem instances.

Instances	Job processing times
1	3, 13; 6, 5, 15; 11, 19, 7, 12
2	18, 15; 8, 4, 20; 16, 13, 1, 6
3	6 10; 2 20; 12 9; 11 7; 5 16
4	20 5; 13 10; 18 16; 1 17; 9 19

Table 2.12: CB-CTV vs. CTV for four small problem instances. It shows consistently smaller CTV for individual class under CB-CTV minimization than under the overall CTV minimization.

Optimal sequences	CTV_1	CTV_2	CTV_3	CTV_4	CTV_5	$CB\text{-}CTV$	CTV
1) CB: 19, 12, 7, 11, 13, 3, 15, 6, 5	4.5*	30.33*	158.25*	N/A	N/A	81.44*	644.11
NCB: 19, 15, 11, 7, 3, 5, 6, 12, 13	648	289.33	588.33	N/A	N/A	501.93	476.78*
2) CB: 20, 4, 8, 16, 13, 1, 6, 18, 15	112.5*	37.33*	70.92*	N/A	N/A	68.96*	761.19
NCB: 20, 16, 15, 6, 4, 1, 8, 13, 18	1250	710.33	372.33	N/A	N/A	680.04	572.61*
3) CB: 20, 2, 16, 5, 10, 6, 11, 7, 12, 9	18*	2*	40.5*	24.5*	12.5*	19.5*	716.1
NCB: 20, 12, 11, 9, 5, 2, 6, 7, 10, 16	144.5	760.5	200	420.5	840.5	473.2	533.78*
4) CB: 19, 9, 20, 5, 17, 1, 13, 10, 18, 16	12.5*	50*	128*	0.5*	40.5*	46.3*	1226.01
NCB: 20, 18, 17, 10, 9, 1, 5, 13, 16, 19	1800	392	2520.5	200	1458	1274.1	1021.34*

From Table 2.12, we can observe that there is a trade-off between CB-CTV and the overall CTV. That is, if the overall CTV is minimized, inner-class CTVs may be large. Namely, CB-CTV is large. Conversely, if CB-CTV is minimized, inner-class CTVs will be optimal, while the overall CTV deviates from its optimum. In other words, the improvement of CB-CTV performance is obtained at the cost of sacrificing the overall CTV performance. Let $CTV_{Im}^i (i = 1, 2, \ldots, 5)$, $CB\text{-}CTV_{Im}$, and CTV_S denote the performance improvement of inner-class CTVs, CB-CTV and the performance sacrifice of the overall CTV respectively. Then they can be measured by respective decrease or increase percentages as follows:

$$CTV_{Im}^i = \frac{CTV_{NCB}^i - CTV_{CB}^i}{CTV_{NCB}^i} * 100\% \qquad i = 1, 2, \ldots, 5 \tag{2.16}$$

$$CB\text{-}CTV_{Im} = \frac{CB{-}CTV_{NCB} - CB{-}CTV_{CB}}{CB{-}CTV_{NCB}} * 100\% \tag{2.17}$$

$$CTV_S = \frac{CTV_{CB} - CTV_{NCB}}{CTV_{NCB}} * 100\% \tag{2.18}$$

where $CTV_{CB}^i (i = 1, 2, \ldots, 5), CB\text{-}CTV_{CB}, CTV_{CB}, CTV_{NCB}^i (i = 1, 2, \ldots, 5), CB\text{-}CTV_{NCB}$, and CTV_{NCB} denote inner-class CTVs, CB-CTV and the overall CTV of optimal sequences under the class-based and the non-class-based situations respectively. Through the calculation with regard to the above four small problem instances, the values of these performance indices are listed in Table 2.13.

According to Table 2.13, the overall CTV performance sacrifices when the objective is to minimize CB-CTV. However, the CTV performance of an individual class is improved dramatically, which is more significant from a user's perspective. For example, in Table 2.12, the CTV of class 4 of instance 4 under CB-CTV minimization is equal to 1/400 of that under the overall CTV minimization. Since users are independent of each other, they receive a better service under CB-CTV minimization than under the overall CTV minimization. Also, the rate of the overall CTV performance sacrifice is much smaller than that of CB-CTV performance improvement. It indicates that the objective defined by us is desirable.

2.5.4.2 Large Problem Instances

For large instances, we consider 4 instances from $L = 3$ classes and $n = 100$ jobs and 4 instances from $L = 5$ classes and $n = 100$ jobs. For the first 4 instances, the job number of each class is 20, 30, and 50 respectively. For the last 4 instances, the job number of each class is the same 20. Because of their large size, these instances can not be listed. For the same reason, it is extremely computationally costly if not impossible to use exhaustive enumeration

Table 2.13: Performance indices comparison for four small problem instances.

No.	CTV^1_{Im}	CTV^2_{Im}	CTV^3_{Im}	CTV^4_{Im}	CTV^5_{Im}	$CB\text{-}CTV_{Im}$	CTV_S
1	99.31%	89.52%	73.1%	N/A	N/A	83.77%	35.1%
2	91%	94.74%	80.95%	N/A	N/A	89.86%	32.93%
3	87.54%	99.74%	79.75%	94.17%	98.51%	95.88%	34.16%
4	99.31%	87.24%	94.92%	99.75%	97.22%	96.37%	20.04%

to obtain optimal sequences. Hence, two recently developed algorithms, Verified Spiral (VS) and Balanced Spiral (BS), are used in this section to approximately solve the problem (Ye et al., 2007). These two algorithms show better performance than some existing algorithms such as FIFO (First-In-First-Out), SPT (Shortest Processing Time), and EC1.2 (Method 1.2 in (Eilon and Chowdhury, 1977)). Note that these two algorithms are developed for WTV minimization problems, but since optimal sequences of CTV and WTV minimization problems are antithetical, they can be modified and applied to CTV minimization problems. We simply describe these two modified algorithms as follows.

Assume a single machine needs to process a job set $p_1, p_2, \ldots, p_n$, where $p_1 \leq p_2 \leq \ldots \leq p_n$. VS method is as follows:

1. According to Schrage's conjecture, place the job p_n in the first position, the job p_{n-1} in the second position, and the job p_{n-2} in the last position. The shortest job p_1 is placed in the position between p_{n-1} and p_{n-2}.

2. Select the longest job from the unscheduled jobs. Place it either exactly before the job p_1 or exactly after the job p_1, depending on which way produces a smaller CTV of the job sequence so far.

3. Repeat Step 2 until all the jobs are scheduled.

BS method is as follows:

1. Place the job p_n in the first position, the job p_{n-1} in the second position, and the job p_{n-2} in the last position. Let sequence $Lt = \{p_{n-1}\}$ and sequence $Rt = \{p_{n-2}\}$. Denote by SUM_{Lt} and SUM_{Rt} respectively the sums of the processing times of the jobs in Lt and Rt.

2. If $SUM_{Lt} < SUM_{Rt}$, append the largest job from the unscheduled jobs to sequence Lt, and update SUM_{Lt}; If $SUM_{Lt} \geq SUM_{Rt}$, prepend the largest job from the unscheduled jobs to sequence Rt, and update SUM_{Rt}.

3. Repeat Step 2 until all the jobs are scheduled.

The computational output for the eight instances is shown in Table 2.14, where * denote the optimal value and `"VS"` or `"BS"` means the algorithm used for that corresponding instance.

Table 2.14 demonstrates that, for large problem instances, there is still a trade-off between CB-CTV and the overall CTV minimization. In addition, although the overall CTV becomes larger when pursuing the minimum CB-CTV, each class's CTV is reduced significantly. The performance improvement of each class's CTV aligns with users' needs and such class-based service stability will lead to user satisfaction with regard to service stability and consistency. For large instances, the rate of the overall CTV performance sacrifice is also much smaller than that of CB-CTV performance improvement, which indicates the desirability of our

Table 2.14: Performance comparison of CB-CTV and overall CTV for eight large problem instances.

No.	CTV^1_{Im}	CTV^2_{Im}	CTV^3_{Im}	CTV^4_{Im}	CTV^5_{Im}	$CB\text{-}CTV_{Im}$	CTV_S
1 (VS)	95.31%	92.39%	73.3%	N/A	N/A	81.7%	47.35%
2 (VS)	96.07%	89.1%	75.66%	N/A	N/A	85.05%	46.03%
3 (BS)	95.12%	90.64%	77.12%	N/A	N/A	85.2%	50.41%
4 (BS)	95.85%	90.64%	74.83%	N/A	N/A	85.61%	43.54%
5 (VS)	95.18%	96.67%	96.23%	94.09%	95.97%	95.64%	48.72%
6 (VS)	96.07%	95.71%	95.61%	95.63%	95.06%	95.69%	51.8%
7 (BS)	94.8%	96.58%	94.91%	95.38%	96.15%	95.53%	45.7%
8 (BS)	95.63%	94.72%	97.04%	95.51%	94.24%	95.51%	48.45%

objective. The relatively larger overall CTV performance sacrifice rates in large instances than in small ones are caused by the size of job set and the values of processing times.

2.5.5 Summary

In this section we consider CB-CTV minimization problems on a single machine, a generalized case of CTV minimization in which jobs are assumed to come from one class. CTV is a non-regular performance measure which penalizes both earliness and tardiness. It conforms to the Just-In-Time philosophy in manufacturing systems. CB-CTV minimization further takes into account the variability reduction from the customers' point of view to achieve service stability and consistency. CTV minimization problems have been studied extensively with many dominant properties in the literature, while little study has been done to CB-CTV minimization. We prove that a CB-CTV minimization problem can be transformed into a series of CTV minimization problems. Hence, we bridge $1||CB\text{-}CTV$ and $1||CTV$ problems, which allows us to apply well developed properties and methods of CTV problems to CB-CTV problems which are NP-hard.

Computational tests are conducted for both small and large problem instances. In the small-problem scenario, the optimal sequence for the overall CTV and CB-CTV minimization is obtained by exhaustive enumeration. In the large-problem scenario, we apply two recently developed algorithms (VS and BS, which have been shown in (Ye et al., 2007)) to calculate the overall CTV and CB-CTV. Note that for the CB-CTV minimization, since there are at least $(2^L \cdot L!)$ optimal sequences that have the minimum CB-CTV, the one with the smallest CTV is chosen. Both scenarios show that there is a trade-off between CB-CTV and the overall CTV. However, the reduction of individual class's CTV is more significant than the sacrifice of the overall CTV. From the perspective of customers, it is more desirable to achieve CB-CTV minimization for class-based service stability and consistency.

Chapter 3

Scheduling for Supply Chain Coordination

3.1 Introduction

A supply chain consists of a number of stages at which value is added to a product. These stages include the supply of raw materials, product manufacturing, packaging, transportation, distribution, and so on. Effectively managing a supply chain has advantages such as operating cost reduction. Successful examples of supply chain management include business giants like Wal-Mart supermarkets, Dell Computers, and Amazon.com.

Good management of a supply chain inevitably need the cooperation of all stages in the supply chain. This requires the coordination of decision making among the stages. In the review of supply chain management research, Thomas and Griffin (1996) demonstrate the necessity of studying coordination issues in supply chains. In another survey, Sarmiento and

Nagi (1999) further point out that for achieving reduced inventory levels, greater coordination among different stages of a supply chain is desired.

There are many aspects with respect to coordinated supply chain decision making. One of them is coordinate scheduling among different stages, which is called supply chain scheduling (Hall and Potts, 2003). It is concerned with the coordination of scheduling decisions among different decision makers in a supply chain. Due to different standpoints, decision makers usually have disagreement on the scheduling decision of a series of operations such as the manufacturing of products. This leads to scheduling conflict. Adopting a decision maker's optimal schedule will increase the cost of other decision makers and thus the cost of the whole supply chain. Therefore, in order to optimize the supply chain's cost, a coordinated scheduling decision is necessary.

Chapter 2 studies scheduling problems simply from the viewpoint of a stage in a supply chain. The optimal schedules considered in Chapter 2 only apply to the stage itself. In this chapter, scheduling problems are addressed within the scope of a supply chain.

3.2 Literature Review

Hurter and Van Buer (1996) study a problem of coordination between printing and distribution of a medium-sized morning newspaper. The printing facility produces several different products (*e.g.*, different versions), and has an ideal sequence in which newspapers are produced so that the total production time is minimized. On the other side, the distribution department may desire a sequence in which newspapers for the most distant distribution center are produced first. The objective of Hurter and Van Buer (1996) is to reduce the number of vans required to deliver the newspapers to drop-off points, while satisfying the vehicle capacity and time constraints. They first solve several instances of the capacitated vehicle routing problem with time windows and then design a production schedule that supports the resulting routes. Assuming that the printing and distribution departments cooperate on this schedule, they compare their results with the prevailing practice using the data from a medium-sized newspaper company. The results show a significant decrease in distribution costs and time.

Hall and Potts (2003) evaluates the benefits of coordinated decision making in a supply chain that involves a supplier, multiple manufacturers, and multiple customers. The supplier makes deliveries to several manufacturers, who in turn supply several customers. They develop models that minimize the total scheduling and batch delivery costs. They demonstrate that coordination between a supplier and a manufacturer may reduce the total scheduling cost by values in a range of at least 20% up to arbitrarily close to 100%, depending upon the scheduling objective being considered.

Chen and Hall (2007) investigate the conflict and cooperation issues in an assembly systems, which includes a manufacturer and multiple suppliers. The suppliers provide parts to the manufacturer for the assembly of products. All suppliers are regarded as a unit, since they have consistent antagonistic interest with the manufacturer. Various models are analyzed based on the relative bargaining power between these two decision makers. They evaluate the conflicts between respective optimal schedules and show the benefits of cooperation of decision makers. Both conflict cost and cooperation saving are substantial in many scenarios.

Dawande et al. (2006) also study the conflict and cooperation issues. What is different is that they examine a supply chain that involves a manufacturer and a distributor. The manufacturer makes products that are shipped to customers by the distributor. Two specific problems are addressed, with the manufacture focusing on minimizing unproductive time and the distributor desiring to minimize customer cost measure in the first problem and to minimize inventory holding cost in the second problem. They evaluate the conflict of each party, which is the relative increase in cost that results from using the other party's optimal schedule. They consider several practical scenarios about the level of cooperation between the manufacturer and the distributor, and they demonstrate the significance of cost saving via cooperation.

Using limited buffer storage capacity, Agnetis et al. (2006) study models for re-sequencing jobs between a supplier and several manufacturers with different ideal sequences. They describe efficient solution procedures for a variety of objectives and identify the conditions under which the total cost is reduced via cooperation.

3.3 Scheduling Coordination in a Production and Distribution System

3.3.1 Problem Description

we are concerned with a two-echelon supply chain that involves a manufacturer and a distributor. The manufacturer produces products for retailers, and the distributor delivers finished products to retailers. Retailers may order multiple different products from the manufacturer. In such a case, these products are regarded as a single order. In other words, a retailer corresponds to an order. Assume that all retailers place their orders at time zero, and the products in each order are supposed to be produced consecutively. No setup time exists between the production of two adjacent orders. Moreover, the production of an order cannot be interrupted once its production starts. We also assume that, once an order is finished producing, the products are immediately delivered to the corresponding retailer. Additionally, the delivery is assumed to be instant, *i.e.*, the completion time of an order is the time the corresponding retailer receives the order.

Each retailer order has a due date, before which the products should be received. If the delivery of an order misses its due date, the distributor is subject to penalty for lateness. The least penalty is negative impression. Therefore, the distributor desires such a production schedule for the retailers' orders, under which the maximum lateness of the orders is minimized. On the other hand, in order to achieve high production efficiency, the manufacturer takes the minimization of the mean completion time of the orders as its objective. Since the manufacturer and the distributor have different objectives, a conflict arises with respect to the production schedule of the retailers' orders. This is because, an optimal production schedule for the manufacturer is usually not an optimal one for the distributor, and vice versa.

This section studies the resulting conflict, which is to be measured from both viewpoints of the manufacturer and the distributor respectively. In addition, we discuss the cooperation between the two parties and investigate cooperation savings. The following notation is to be used in the section:

n:	the number of retailers' orders
λ:	the optimal production schedule of orders for the manufacturer
γ:	the optimal production schedule of orders for the distributor
$p_i(S)$:	the processing time of the i-th order under a schedule S
$d_i(S)$:	the due date of the i-th order under a schedule S
$C_i(S)$:	the completion time of the i-th order under a schedule S
$L_i(S)$:	the lateness of the i-th order under a schedule S
$\bar{C}(S)$:	the mean completion time of the orders under a schedule S
$L_{max}(S)$:	the maximum lateness of the orders under a schedule S
$F(S)$:	the performance measure of the schedule S for the manufacturer
$G(S)$:	the performance measure of the schedule S for the distributor

$C_i(S),\ L_i(S),\ \bar{C}(S),\ L_{max}(S),\ F(S)$, and $G(S)$ are defined as below:

$$C_i(S) = \sum_{k=1}^{i} p_k(S)$$

$$L_i(S) = C_i(S) - d_i(S)$$

$$\bar{C}(S) = \frac{1}{n}\sum_{i=1}^{n} C_i(S)$$

$$L_{max}(S) = \max_{1\le i\le n} L_i(S)$$

$$F(S) = \bar{C}(S)$$

$$G(S) = L_{max}(S)$$

3.3.2 The Analysis of Conflict

As mentioned in Subsection 3.3.1, the manufacturer and the distributor have different opinions on the orders' production schedule. The manufacturer desires a schedule under which the mean completion time is minimized, whereas the distributor tends to have a schedule under which the maximum lateness is the minimum. According to scheduling literature such as (Pinedo, 2002), the Shortest-Processing-Time-first (SPT) rule and the Earliest-Due-Date-first (EDD) rule are optimal for the manufacturer and the distributor, respectively. In general, the schedule generated by the SPT rule is different from the one generated by the EDD rule. Therefore, a scheduling conflict arises between the manufacturer and the distributor. In order to have a quantitative notion of the conflict, we develop two mathematical measures for it, from the perspectives of the manufacturer and the distributor respectively.

These two measures, denoted by $Conf_m$ and $Conf_d$, are defined as follows:

$$Conf_m = \frac{\bar{C}(\gamma) - \bar{C}(\lambda)}{\bar{C}(\lambda)} * 100\% \; , \tag{3.1}$$

$$Conf_d = \frac{L_{max}(\lambda) - L_{max}(\gamma)}{L_{max}(\gamma)} * 100\% \; , \tag{3.2}$$

where λ and γ are the respective optimal schedules for the two parties (*i.e.*, the schedules generated by the SPT and EDD rules, respectively).

3.3.2.1 Numerical Examples

First we investigate the conflict from specific numerical examples. To this end, we simulate some small-sized problem instances. Table 3.1 presents four instances, with the first two instances including 10 orders each and the last two including 20 orders each. The processing times and the due dates of the orders are arbitrarily set to follow uniform distributions of Uniform(1, 50) and Uniform(1, 100) respectively. Furthermore, for reasonability we set $d_i > p_i$, $i = 1, \cdots, n$. It is easy to find the optimal two schedules λ and γ for each instance, by the SPT and EDD rules respectively. Then by the formulas (1) and (2), we calculate the corresponding conflict measures as in Table 3.2. We use processing times to represent orders in schedules λ and γ. Large values of $Conf_m$ and $Conf_d$ in Table 3.2 indicate that the scheduling conflict between the manufacturer and the distributor is very significant. Therefore, some actions must be taken to lessen the conflict. An effective action is to cooperate with each other.

Table 3.1: Problem Instances

1)	Job Number:	1, 2, 3, 4, 5, 6, 7, 8, 9, 10
	Processing Time:	13, 47, 7, 27, 45, 48, 17, 22, 24, 8
	Due Date:	14, 54, 73, 40, 87, 63, 25, 98, 65, 23
2)	Job Number:	1, 2, 3, 4, 5, 6, 7, 8, 9, 10
	Processing Time:	1, 1, 10, 30, 3, 19, 32, 36, 35, 5
	Due Date:	46, 45, 36, 68, 70, 73, 48, 56, 46, 72
3)	Job Number:	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20
	Processing Time:	7, 1, 33, 37, 2, 6, 45, 39, 30, 35, 49, 28, 39, 31, 50, 3, 50, 11, 46, 34
	Due Date:	47, 5, 47, 80, 29, 70, 72, 73, 74, 63, 99, 82, 54, 60, 51, 89, 62, 96, 83, 62
4)	Job Number:	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20
	Processing Time:	22, 30, 6, 16, 32, 45, 49, 2, 6, 21, 5, 15, 34, 49, 20, 42, 7, 43, 12, 4
	Due Date:	34, 58, 14, 87, 33, 49, 99, 97, 95, 46, 34, 17, 83, 65, 33, 75, 30, 87, 63, 6

Table 3.2: Numerical Results of the Conflict for Problem Instances

No.	Individual Optimal Schedules	$Conf_m$	$Conf_d$
1)	λ: 7, 8, 13, 17, 22, 24, 27, 45, 47, 48	27.54%	21.88%
	γ: 13, 8, 17, 27, 47, 48, 24, 7, 45, 22		
2)	λ: 1, 1, 3, 5, 10, 19, 30, 32, 35, 36	60.72%	17.17%
	γ: 10, 1, 1, 35, 32, 36, 30, 3, 5, 19		
3)	λ: 1, 2, 3, 6, 7, 11, 28, 30, 31, 33, 34, 35, 37, 39, 39, 45, 46, 49, 50, 50	34.71%	7.76%
	γ: 1, 2, 7, 33, 50, 39, 31, 50, 34, 35, 6, 45, 39, 30, 37, 28, 46, 3, 11, 49		
4)	λ: 2, 4, 5, 6, 6, 7, 12, 15, 16, 20, 21, 22, 30, 32, 34, 42, 43, 45, 49, 49	36.77%	9.42%
	γ: 4, 6, 15, 7, 32, 20, 22, 5, 21, 45, 30, 12, 49, 42, 34, 16, 43, 6, 2, 49		

3.3.2.2 Properties of the Conflict

Before presenting some properties of the conflict, we first prove that the Longest-Processing-Time-first (LPT) rule and the Latest-Due-Date-first (LDD) rule maximize the mean completion time and the maximum lateness, respectively.

Lemma 6. *For the single machine scheduling problem, the LPT rule maximizes the mean completion time.*

Proof. Without loss of generality, suppose that there are n jobs to be scheduled on a single machine. Let p_i be the processing time of the i-th job under a certain schedule, $i = 1, 2, \cdots, n$. Then the completion time of the i-th job

$$C_i = \sum_{k=1}^{i} p_k, \qquad i = 1, 2, \cdots, n$$

So the mean completion time of jobs under this schedule is:

$$\bar{C} = \frac{1}{n}\sum_{i=1}^{n} C_i = \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{i} p_k = \frac{1}{n}\sum_{i=1}^{n}(n+1-i)p_i$$

Therefore, in order to maximize the mean completion time, the n jobs should be scheduled in such an order that $p_1 \geq p_2 \geq \cdots \geq p_n$. That is, the LPT rule is optimal for maximizing the mean completion time. □

Lemma 7. *For the single machine scheduling problem, the LDD rule maximizes the maximum lateness.*

Proof. By contradiction. Let S be an optimal schedule that maximizes the maximum lateness. Assume that S does not follow the LDD rule, that is, there exists at least a pair of adjacent jobs i and j, where job i is processed before job j and $d_i < d_j$. We interchange jobs i and j and denote the new schedule by S'. See Figure 3.1.

Let C_k and L_k (C'_k and L'_k) be the completion time and the lateness of some job k under the schedule S (S'). On the one hand, by the definition of job completion time, it is clear that $C'_i > C_i$ (In fact, $C'_i = C_i + p_j$). Therefore, $C'_i - d_i > C'_i - d_i$, *i.e.*, $L'_i > L_i$. In addition, it is easy to derive that $C'_i = C_j$. Since $d_i < d_j$, we have $C'_i - d_i > C_j - d_j$, *i.e.*, $L'_i > L_j$. Then

$$\left.\begin{array}{l} L'_i > L_i \\ \\ L'_i > L_j \end{array}\right\} \Longrightarrow L'_i > \max\{L_i, L_j\}$$

On the other hand, for $k \neq i$ or j, $L'_k = L_k$ since $C'_k = C_k$. Therefore, $\max\limits_{1\leq i\leq n} L'_i \geq \max\limits_{1\leq i\leq n} L_i$. This violates the assumption that S is optimal for maximizing the maximum lateness. This completes the proof. $\square$

Theorem 2. *The conflict extent between the manufacturer and the distributor depends on the relationship between the processing times and the due dates of the orders. If it is the case that an order's processing time is longer, its due date is later, then there is no conflict*

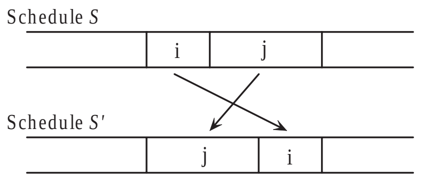


Figure 3.1: A pairwise interchange of jobs i and j

between the two parties. Contrarily, if it is true that an order's processing time is longer, its due date is earlier, then the conflict is the largest. In other cases, the conflict is in between.

Proof. Without loss of generality, assume that $p_1 \leq p_2 \leq \cdots \leq p_n$.

First consider the case where the longer an order's processing time, the later its due date. Obviously it follows that $d_1 \leq d_2 \leq \cdots \leq d_n$. Since the SPT and the EDD rules are the optimal production schedules of retailers' orders for the manufacturer and for the distributor respectively, in this concerned case the two parties have the same optimal schedules $\lambda = \gamma = \{1, 2, \cdots, n\}$ and therefore there is no conflict.

In the completely opposite case where the longer an order's processing time, the earlier its due date, it holds that $d_1 \geq d_2 \geq \cdots \geq d_n$. For the manufacturer, the optimal schedule is $\lambda = \{1, 2, \cdots, n\}$, which is according to the SPT rule. While for the distributor, the optimal schedule is $\gamma = \{n, n-1, \cdots, 2, 1\}$, which is based on the EDD rule. By Lemma 6, γ maximizes the desired performance measure of the manufacturer, *i.e.*, the mean completion time. This results in the largest $Conf_m$ by the formula (3.1). Similarly by Lemma 7, λ maximizes the adopted performance measure of the distributor, *i.e.*, the maximum lateness. This leads to the largest $Conf_d$ by the formula (3.2).

In other cases than the above two, since the derived optimal schedule of one party does not maximize the performance measure of the other party, the conflict is between zero and the largest. □

Theorem 3. *Both measures of the conflict between the manufacturer and the distributor are confined in their own ranges. Assume that $p_1 \leq p_2 \leq \cdots \leq p_n$, the ranges are:*

$$0 \leq Conf_m \leq \frac{\sum_{k=1}^{n} kp_k}{\sum_{k=1}^{n}(n+1-k)p_k} - 1 \ , \tag{3.3}$$

$$0 \leq Conf_d \leq \frac{\underset{1\leq i\leq n}{max}\{\sum_{k=i}^{n} p_{s_k} - d_{s_i}\}}{\underset{1\leq i\leq n}{max}\{\sum_{k=1}^{i} p_{s_k} - d_{s_i}\}} - 1 \ , \tag{3.4}$$

where $\{s_1, s_2, \cdots, s_n\}$ is a permutation of $\{1, 2, \cdots, n\}$ and satisfies $d_{s_1} \leq d_{s_2} \leq \cdots \leq d_{s_n}$.

Proof. According to their definitions, it is clear that $Conf_m \geq 0$ and $Conf_d \geq 0$. Now let us focus on the upper bounds, which is denoted by $Conf_m^*$ and $Conf_d^*$, respectively.

First consider $Conf_m$. By the SPT rule, the optimal schedule for the manufacturer is $\lambda = \{1, 2, \cdots, n\}$. Contrarily, by Lemma 6 the worst schedule is $\{n, n-1, \cdots, 2, 1\}$. We denote this schedule by α. Based on Theorem 2, when $s_i = n+1-i(i = 1, 2, \cdots, n)$, *i.e.*, $d_n \leq d_{n-1} \leq \cdots \leq d_1$, the optimal schedule for the distributor is $\{n, n-1, \cdots, 2, 1\} = \alpha$, and the largest conflict arises. Hence,

$$\bar{C}(\lambda) = \frac{1}{n}\sum_{i=1}^{n} C_i(\lambda) = \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{i} p_k = \frac{1}{n}\sum_{k=1}^{n}(n+1-k)p_k \ ,$$

$$\bar{C}(\alpha) = \frac{1}{n}\sum_{i=1}^{n} C_i(\alpha) = \frac{1}{n}\sum_{i=1}^{n}\sum_{k=n+1-i}^{n} p_k = \frac{1}{n}\sum_{k=1}^{n} kp_k \ ,$$

$$Conf_m^* = \frac{\bar{C}(\alpha) - \bar{C}(\lambda)}{\bar{C}(\lambda)} = \frac{\sum_{k=1}^{n} kp_k}{\sum_{k=1}^{n}(n+1-k)p_k} - 1 \ .$$

Then consider $Conf_d$. By the EDD rule, the optimal schedule for the distributor is $\gamma = \{s_1, s_2, \cdots, s_n\}$. Contrarily, by Lemma 7 the worst schedule (denoted by β) is $\beta =$

$\{s_n, s_{n-1}, \cdots, s_2, s_1\}$. From another perspective, β is the optimal schedule for the manufacturer when it holds that $p_{s_n} \leq p_{s_{n-1}} \leq \cdots \leq p_{s_2} \leq p_{s_1}$, *i.e.*, $s_i = n+1-i (i = 1, 2, \cdots, n)$. Under this scenario, there exists the largest conflict. Then we have

$$C_{s_i}(\gamma) = \sum_{k=1}^{i} p_{s_k}, \qquad i = 1, 2, \cdots, n \ ,$$

$$L_{max}(\gamma) = \max_{1 \leq i \leq n} \{L_i(\gamma)\} = \max_{1 \leq i \leq n} \{C_{s_i}(\gamma) - d_{s_i}\} = \max_{1 \leq i \leq n} \{\sum_{k=1}^{i} p_{s_k} - d_{s_i}\} \ ,$$

$$C_{s_i}(\beta) = \sum_{k=i}^{n} p_{s_k}, \qquad i = 1, 2, \cdots, n \ ,$$

$$L_{max}(\beta) = \max_{1 \leq i \leq n} \{L_i(\beta)\} = \max_{1 \leq i \leq n} \{C_{s_i}(\beta) - d_{s_i}\} = \max_{1 \leq i \leq n} \{\sum_{k=i}^{n} p_{s_k} - d_{s_i}\} \ ,$$

$$Conf_d^* = \frac{L_{max}(\beta) - L_{max}(\gamma)}{L_{max}(\gamma)} = \frac{\max_{1 \leq i \leq n} \{\sum_{k=i}^{n} p_{s_k} - d_{s_i}\}}{\max_{1 \leq i \leq n} \{\sum_{k=1}^{i} p_{s_k} - d_{s_i}\}} - 1 \ .$$

This completes the proof. $\square$

3.3.3 Cooperation Mechanisms and Savings

From Subsection 3.3.2, it is evident that the conflict is significant. The cooperation is therefore needed between the manufacturer and the distributor. As is known, the performance measures $F(S)$ and $G(S)$ are desired to minimize for the two parties, respectively. To some degree $F(S)$ and $G(S)$ can be regarded as the costs the two parties incur when the schedule S is adopted for the production of the retailers' orders. This perspective is used in (Chen and Hall, 2007) as well.

3.3.3.1 Cooperation Mechanisms

Based on the relative bargaining power between the manufacturer and the distributor, below we propose cooperation mechanisms and analyze the resulting savings.

A) Manufacturer Dominates, Distributor Negotiates

In this scenario, the manufacturer is dominant and it will adopt the schedule that benefits it, *i.e.*, it will use its own optimal schedule λ. For the distributor, however, this schedule is usually not good. So the distributor wishes to persuade the manufacturer to use some alternative schedule θ through providing the manufacturer some incentive μ. According to the bargaining theory of Nash (1950, 1953), the incentive must be enough for the manufacturer to accept the alternative schedule. In mathematical words, this incentive μ must satisfy $F(\theta) - \mu \leq F(\lambda)$ and $G(\theta) + \mu \leq G(\lambda)$. Simultaneously, the distributor certainly desires its cost (*i.e.*, $G(\theta) + \mu$) to be the minimum. Therefore, the distributor needs to find a set of (θ, μ) that solves the following optimization problem:

$$\begin{aligned} \min \quad & G(\theta) + \mu && (3.5) \\ \text{s.t.} \quad & F(\theta) - \mu \leq F(\lambda) && (3.6) \\ & G(\theta) + \mu \leq G(\lambda) && (3.7) \\ & \mu \geq 0 && (3.8) \end{aligned}$$

Theorem 4. *Let θ^* be the schedule that minimizes $F(\theta) + G(\theta)$, and $\mu^* = F(\theta^*) - F(\lambda)$. Then (θ^*, μ^*) solves the above optimization problem (3.5-3.8).*

Proof. First prove the feasibility of (θ^*, μ^*). By the definition of μ^*, the constraint (3.6) follows. By the definition of θ^*, $F(\theta^*) + G(\theta^*) \leq F(\lambda) + G(\lambda)$. So $G(\theta^*) + (F(\theta^*) - F(\lambda)) \leq G(\lambda)$, *i.e.*, the constraint (3.7) is satisfied. In addition, the constraint (3.8) naturally follows due to the definitions of μ^* and λ.

Now prove the optimality of (θ^*, μ^*). Consider an arbitrary feasible solution (θ, μ). By the constraint (3.6), we have $\mu \geq F(\theta) - F(\lambda)$. Hence, $G(\theta) + \mu \geq G(\theta) + [F(\theta) - F(\lambda)] \geq F(\theta^*) + G(\theta^*) - F(\lambda)$ (by the definition of θ^*). This lower bound can be achieved by using the feasible solution (θ^*, μ^*). This completes the proof. □

Theorem 5. *The cost saving of the distributor from the cooperation with the manufacturer is* $[F(\lambda) + G(\lambda)] - [F(\theta^*) + G(\theta^*)]$.

Proof. Without the cooperation, the cost of the distributor is $G(\lambda)$, since the manufacturer is the dominant party and the schedule λ is adopted. While with the cooperation, based on Theorem 4 the cost becomes $G(\theta^*) + \mu^* = F(\theta^*) + G(\theta^*) - F(\lambda)$. Therefore, the cost saving is $G(\lambda) - [G(\theta^*) + \mu^*] = [F(\lambda) + G(\lambda)] - [F(\theta^*) + G(\theta^*)]$. □

Remark 1. Since θ^* is the schedule that minimizes $F(\theta) + G(\theta)$, the cost saving is nonnegative. On the other hand, the cost of the manufacturer is constant regardless of there is a cooperation or not, since the constraint (3.6) is binding at (θ^*, μ^*). In spite of so, the manufacturer still needs cooperation, because at least one of the advantages from the cooperation lies in that there is a favorable relationship between it and the distributor.

B) Distributor Dominates, Manufacturer Negotiates

In this scenario, the distributor has dominant bargaining power over the manufacturer. So generally the distributor will use its own optimal schedule γ. However, this schedule γ is usually not a favorable schedule for the manufacturer. Therefore, the manufacturer needs to provide an incentive ν that is attractive enough to persuade the distributor to adopt an alternative schedule η (Nash, 1950, 1953). This incentive should satisfy $G(\eta) - \nu \leq G(\gamma)$ and $F(\eta) + \nu \leq F(\gamma)$. In addition, the manufacturer desires its cost after cooperation as minimum as possible. Therefore, it is confronted with the following optimization problem:

$$\min \quad F(\eta) + \nu \tag{3.9}$$

$$\text{s.t.} \quad G(\eta) - \nu \leq G(\gamma) \tag{3.10}$$

$$F(\eta) + \nu \leq F(\gamma)) \tag{3.11}$$

$$\nu \geq 0 \tag{3.12}$$

Theorem 6. *Let η^* be the schedule that minimizes $F(\eta) + G(\eta)$, and $\nu^* = G(\eta^*) - G(\gamma)$. Then (η^*, ν^*) solves the above optimization problem (3.9-3.12).*

Proof. First prove the feasibility of (η^*, ν^*). The constraint (3.10) follows by the definition of ν^*. The left-hand side of the constraint (3.11), $F(\eta^*) + \nu^* = F(\eta^*) + G(\eta^*) - G(\gamma)$, is less than or equal to $F(\gamma)$ by the definition of η^*. In addition, the definitions of ν^* and γ guarantee the satisfaction of the constraint (3.12).

Now prove the optimality of (η^*, ν^*). Given an arbitrary feasible solution (η, ν), by the constraint (3.10), we have $\nu \geq G(\eta) - G(\gamma)$. Hence, $F(\eta) + \nu \geq F(\eta) + [G(\eta) - G(\gamma)] \geq F(\eta^*) + G(\eta^*) - G(\gamma)$ (by the definition of η^*). This lower bound is achieved by the solution (η^*, ν^*). This completes the proof. $\square$

Theorem 7. *The cost saving of the manufacturer from the cooperation with the distributor is* $[F(\gamma) + G(\gamma)] - [F(\eta^*) + G(\eta^*)]$.

Proof. Since the distributor has dominant bargaining power, its own optimal schedule γ is used. So the cost of the manufacturer without the cooperation is $F(\gamma)$. While with the cooperation with the distributor, based on Theorem 6 the cost of the manufacturer is $F(\eta^*) + \nu^* = F(\eta^*) + G(\eta^*) - G(\gamma)$. Therefore, the cost saving from the cooperation is $F(\gamma) - [F(\eta^*) + G(\eta^*) - G(\gamma)] = [F(\gamma) + G(\gamma)] - [F(\eta^*) + G(\eta^*)]$. $\square$

Remark 2. The cost saving is nonnegative since η^* minimizes $F(\eta) + G(\eta)$. However, the cooperation does not reduce the cost of the distributor, because the constraint (3.10) is binding at (η^*, ν^*). Although so, the distributor still needs to accept the incentive and co-operate with the manufacturer, in view of the maintaining and development of the favorable relationship between the two parties.

Remark 3. θ^* and η^* actually represent the same schedule that minimizes $1||\{\bar{C} + L_{max}\}$. A property for such a schedule is developed as follows.

Property 11. *If processing time and due date is agreeable, the SPT (or EDD) rule is optimal for the* $1||\{\bar{C}+L_{max}\}$ *problem; If processing time and due date is not agreeable, two adjacent jobs are scheduled in the SPT rule for the* $1||\{\bar{C}+L_{max}\}$ *problem if the following condition holds:*

$$min\{p_i, p_j, |d_i - d_j|\} \leq \frac{|p_i - p_j|}{n} \tag{3.13}$$

where p_i, p_j *are two adjacent jobs,* n *is the number of jobs, and* $|\cdot|$ *represents the absolute value.*

Proof. First we prove the first part. If processing time and due date is agreeable, *i.e.*, $p_s \leq p_t$ implies $d_s \leq d_t$, $s, t \in \{1, 2, \cdots, n\}$, then the sequence generated by the SPT rule is the same as that generated by the EDD rule. In addition, it is well known that the SPT and EDD rules are optimal for the $1||\bar{C}$ and $1||L_{max}$ problems, respectively. Therefore, the SPT (or EDD) rule is optimal for the $1||\{\bar{C}+L_{max}\}$ problem.

Now we prove the second part (by contradiction). There are two cases for this part. First consider the case where $p_i > p_j$ and $d_i < d_j$. Assume that in an optimal schedule S, job i precedes job j. Interchange jobs i and j as shown in Figure 3.2. Denote the new schedule by S'. Let $C_i(C_i')$ and $L_i(L_i')$ be the completion time and lateness of the *i-th* job under the schedule $S(S')$, and let $\bar{C}(\bar{C}')$ and $L_{max}(L_{max}')$ be the mean completion time and maximum lateness of jobs under the schedule $S(S')$.

It is easy to know that, $\bar{C} - \bar{C}' = \frac{p_i - p_j}{n}$. Since $C_i' = C_j$ and $d_i < d_j$, we have $C_i' - d_i > C_j - d_j$, *i.e.*, $L_i' > L_j$. Further, $L_i' - L_j = d_j - d_i$. Also, it is obvious that $C_i' > C_i$. So $L_i' > L_i$. Further, $L_i' - L_i = p_j$. Additionally, it is evident that $C_i' > C_j'$. Since $d_i < d_j$, we

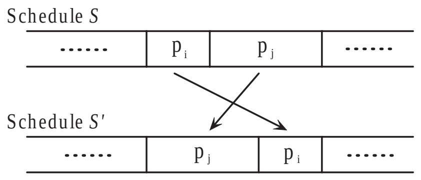


Figure 3.2: The interchange of job i and job j

have $L'_i > L'_j$. Besides, $L_k(k \neq i$ or $j)$ is constant whether the interchange occurs or not. Therefore, from $L'_i > L_i$, $L'_i > L_j$, and $L'_i > L'_j$, we have that $L'_{max} \geq L_{max}$. Specifically, $L'_{max} - L_{max} \leq \min\{p_j, d_j - d_i\}$.

Under the considered situation, Equation 3.13 is equivalent to $\min\{p_j, d_j - d_i\} \leq \frac{p_i - p_j}{n}$. Hence, if Equation 3.13 holds, then $L'_{max} - L_{max} \leq \bar{C} - \bar{C}'$, *i.e.*, $\bar{C}' + L'_{max} \leq \bar{C} + L_{max}$. This violates that S is an optimal schedule for the $1||\{\bar{C} + L_{max}\}$ problem. Therefore, job i should be scheduled after job j. That is, the adjacent jobs i and j should be scheduled in the SPT rule.

Consider the other case where $p_i < p_j$ and $d_i > d_j$. Consider the schedules S and S' as represented in Figure 3.2, with the same notation. In a similar way, we can derive that $\bar{C}' - \bar{C} = \frac{p_j - p_i}{n}$ and $L_{max} - L'_{max} \leq \min\{p_i, d_i - d_j\}$. If Equation 3.13 holds, then $L_{max} - L'_{max} \leq \bar{C}' - \bar{C}$, *i.e.*, $\bar{C} + L_{max} \leq \bar{C}' + L'_{max}$. Therefore, in this case job i should be scheduled preceding job j. In other words, jobs i and j should be scheduled in the SPT rule.

Combining these two cases, the second part of the property holds. $\square$

3.3.3.2 Lower Bounds of Relative Cost Savings

Let Sav_m and Sav_d be the relative cost savings via cooperation for the manufacturer and the distributor, respectively. Based on Theorems 5 and 7, we define them as follows:

$$Sav_m = \frac{[F(\gamma) + G(\gamma)] - [F(\theta^*) + G(\theta^*)]}{F(\gamma)} * 100\% \; , \tag{3.14}$$

$$Sav_d = \frac{[F(\lambda) + G(\lambda)] - [F(\theta^*) + G(\theta^*)]}{G(\lambda)} * 100\% \; , \tag{3.15}$$

where λ, γ, and θ^* are defined as above.

Below we develop a heuristic algorithm for obtaining a schedule that reduces $1||\{\bar{C} + L_{max}\}$. Denote the derived schedule by δ. Obviously, $F(\delta) + G(\delta) \geq F(\theta^*) + G(\theta^*)$ since θ^* minimizes $F(\cdot) + G(\cdot)$. Therefore, lower bounds of the relative cost savings for the manufacturer and the distributor are:

$$Sav_m \geq \frac{[F(\gamma) + G(\gamma)] - [F(\delta) + G(\delta)]}{F(\gamma)} * 100\% \; , \tag{3.16}$$

$$Sav_d \geq \frac{[F(\lambda) + G(\lambda)] - [F(\delta) + G(\delta)]}{G(\lambda)} * 100\% \; . \tag{3.17}$$

Here is the proposed heuristic, which is inspired by both the SPT and EDD rules. First calculate the product of processing time and due date for each job, and then schedule jobs in increasing order of the products. In other words, the job with the smallest product of processing time and due date is scheduled first, the job with the second smallest product is scheduled second, the job with the k smallest product is scheduled k-th, and the job with the largest product is scheduled last.

Using the problem instances in Table 3.1, we derive the job sequences by the heuristic and compute the lower bounds according to the expressions (3.16) and (3.17). The output is demonstrated in Table 3.3, where $LbSav_m$ and $LbSav_d$ stand for the lower bounds of relative cost savings for the manufacturer and the distributor, respectively. The results show the significant reduction of savings for both the manufacturer and the distributor.

3.3.4 Summary

In this section, we study scheduling coordination in a production and distribution system that consists of a manufacturer and a distributor. The manufacturer produces products and the distributor delivers the finished products to the downstream retailers. Each retailer order has a due date associated with it. The distributor is penalized for late deliveries. So in order to reduce lateness fees, the distributor desires a production schedule that minimizes the maximum lateness of the orders. However, the manufacturer tends to minimize the mean completion time of the orders in the pursuit of high production efficiency. Therefore, a conflict regarding the orders' production scheduling arises between the manufacturer and the distributor, due to different scheduling objectives. In this section, the resulting conflict

Table 3.3: Examples of Lower Bounds of Relative Cost Savings

No.	The Schedules by the Heuristic	$LbSav_m$	$LbSav_d$
1)	δ: 13, 8, 17, 7, 27, 24, 22, 47, 48, 45	10.09%	10.41%
2)	δ: 1, 1, 3, 10, 5, 19, 32, 35, 36, 30	30.16%	8.79%
3)	δ: 1, 2, 3, 7, 6, 11, 33, 31, 39, 34, 35, 30, 28, 50, 39, 37, 50, 45, 46, 49	23.70%	6.07%
4)	δ: 4, 6, 5, 2, 7, 15, 6, 20, 22, 12, 21, 32, 16, 30, 45, 34, 42, 49, 43, 49	24.54%	7.38%

is measured and studied. Moreover, based on different scenarios of the relative bargaining power between the two parties, cooperation mechanisms are proposed and the resulting cooperation savings are investigated. In addition, we reveal a property of the optimal cooperation schedule. Furthermore, a heuristic algorithm for deriving a cooperation schedule is proposed and lower bounds of relative cost savings are developed.

Chapter 4

Conclusions

4.1 Summarized Work

This dissertation studies scheduling problems for service stability and for supply chain coordination as well. They are illustrated in Chapters 2 and 3, respectively. In Chapter 2, service stability is represented by the performance measure of completion time variance (CTV), and several job scheduling problems with CTV minimization are studied. CTV is a non-regular performance measure since it is not nondecreasing in completion times. CTV minimization means that jobs are desired to be completed within a relative concentrated period of time. Thus, both earliness and tardiness are undesirable. This point conforms to the popular Just-in-time (JIT) philosophy, which desires that jobs are completed neither early nor tardy. This is because that early completion increases inventory cost while tardy completion results in customer dissatisfaction. In Sections 2.3 and 2.4, job scheduling problems with CTV minimization are addressed on multiple identical parallel machines, with the unrestricted case and the restricted case, respectively. Restriction here means that machines do not have idle times before they begin to process jobs. In other words, at time zero, all machines must begin processing jobs in the restricted case. However, in the unrestricted case, machines can have any length of idle times before they begin processing jobs. In these two sections, several dominant properties of each problem are discovered, and for each problem an efficient heuristic algorithm is developed based on the discovered properties. The performance of each heuristic algorithm is compared with optimal schedules when problem instances are small and compared to existing scheduling algorithms when problem instances are large. Numerical examples demonstrate that each proposed heuristic algorithm is near-optimal when considering small-sized instances and greatly outperforms existing algorithms when considering large-sized instances. In Section 2.5, job scheduling problem is taken into account from different perspective: users' perspective. The problem is investigated in the single-machine environment. The concerned performance measure is class-based completion time variance. Jobs are assumed to come from several users. The study of this problem is motivated by the phenomenon that the CTV of jobs from a user may be very large even if the overall CTV of the system is minimized. However, for the standpoint of the users, they do not care the overall CTV. It will cause dissatisfaction of the users if the overall CTV is taken as the only scheduling objective. So in this section the class-based completion time variance (CB-CTV) is regarded as the dominant objective while the overall CTV is adopted as a second objective. The CB-CTV problem is proven to be able to be transformed into multiple CTV problems. This discovery significantly simplifies the CB-CTV problem, since many favorable properties of the CTV problem have been developed. Furthermore in this section, the tradeoff between CB-CTV minimization and the overall CTV minimization is examined.

Chapter 3 studies scheduling problems for supply chain coordination, which is also denoted by supply chain scheduling. Different from Chapter 2, this chapter considers scheduling problems from the perspective of a supply chain. In recent years, supply chain coordination has been gaining more and more attention from both the academia and the industry. Fierce competition makes the firms in a supply chain pay more attention to the overall performance of the supply chain rather than of itself. The coordination in a supply chain is crucial. On the other hand, scheduling is widely needed in practice. Therefore, coordinated scheduling among decision makers in a supply chain are important. Because of different standpoints, decision makers of different stages in a supply chain usually have different opinions with respect to the scheduling operation decision at a certain stage. Accordingly, conflicts arise. The coordination is therefore needed to resolve the conflicts. Section 3.3 investigates a production and distribution system consisting of a manufacturer and a distributor. The manufacturer produces products and the distributor delivers products to retailers. Each retailer order is associated with a due date. The distributor is punished if a late delivery occurs. If the manufacture focuses on its own operation efficiency and the distributor is only concerned with lateness penalty, then a disagreement exists on the production scheduling policy of the retailers' orders. In this section, from both viewpoints of the manufacturer and the distributor, scheduling conflicts are measured and analyzed. According to different scenarios of the relative bargaining power between the manufacturer and the distributor, cooperation mechanisms are proposed and cooperation savings are investigated. In addition, a heuristic algorithm is proposed for obtaining a coordinated schedule, and lower bounds of relative cost savings are provided.

4.2 Potential Applications

The study results of this dissertation can be widely applied to the real world. Firstly, the developed heuristic algorithms for job scheduling problems on identical parallel machines with CTV minimization can be employed in the situations where service stability is highly desired, such as Internet package dispatching and other web services. It also can be used in the circumstances where JIT is favored, for example, Toyota Production System (TPS). Secondly, the result of the CB-CTV scheduling problem can be used in the scenarios where user preference is very important. There are so many such examples since customer satisfaction is increasingly important for a company. Thirdly, coordinated scheduling in a supply chain can be applied to almost all supply chains, since all supply chains involve scheduling problems and require the coordination among the decision makers in supply chains.

4.3 Future Work and Directions

In Sections 2.3 and 2.4, some assumptions are made to simplify the problems. Later on, we may release several assumptions for future research. For instance, we may consider the different release times of jobs, and we may allow preemption during job processing. Moreover, these two sections only consider a single performance measure of CTV. Combining CTV with other performance measures is another feasible research direction. For example, we can study the scheduling problem of n jobs on m identical and parallel machines with the objective of minimizing both CTV and the mean completion time. Additionally, both the sections only consider the deterministic cases of the problems where the processing times of jobs are known in advance. However, in the real world the processing times are often unknown and random, which leads to the research of the stochastic versions of $Pm||CTV$ problems.

Section 2.5 deals with CB-CTV minimization problems on a single machine. Future research can be conducted under the parallel-machine environment, denoted by $P_m||CB\text{-}CTV$. Also, this section only addresses the deterministic CB-CTV minimization problems. The stochastic case of CB-CTV minimization problems is open for future study. Furthermore, it is of interest to investigate some variations of the CB-CTV minimization problem in which bi-criteria is considered to minimize a combination of regular and non-regular performance measures.

As for Section 3.3, we consider the simple performance measures of scheduling, the mean completion time and the maximum lateness. Future research can take into account more complicated performance measures such as maximum tardiness, completion time variance, and so forth. Additionally, instead of single machine environment, we also can consider the scheduling problem on multiple machines. The system Section 3.3 considers is a production and distribution system. The scheduling problem in other systems is still open for exploration. Moreover, the scheduling decision coordination issue among multiple (more than two) decision makers is another possible research direction.

Bibliography

Agnetis, A., Hall, N. G., and Pacciarelli, D. (2006). Supply chain scheduling: Sequence coordination. *Discrete Applied Mathematics*, 154:2044–2063.

Al-Turki, U., Fediki, C., and Andijani, A. (2001). Tabu search for a class of single-machine scheduling problems. *Computers and Operations Research*, 28:1223–1230.

Anderson, E. J. and Potts, C. N. (2002). On-line scheduling of a single machine to minimize total weighted completion time. In *Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 548–557.

Anupindi, R. and Akella, R. (1993). Diversification under supply uncertainty. *Management Science*, 39:944–963.

Axsäter, S. (2006). *Inventory Control*. New York: Springer, 2nd edition.

Baker, K. and Scudder, G. (1990). Sequencing with earliness and tardiness penalties: a review. *Operations Research*, 38(1):22–36.

Banker, R. and Khosla, I. (1995). Economics and operations management: a research perspective. *Journal of Operations Management*, 12:423–425.

Brucker, P. (2004). *Scheduling Algorithms*. Springer, 4th edition.

Cai, X. and Cheng, T. (1998). Multi-machine scheduling with variance minimization. *Discrete Applied Mathematics*, 84:55–70.

Chandra, P. and Fisher, M. (1994). Coordination of production and distribution planning. *European Journal of Operational Research*, 72:503–517.

Chen, B., Potts, C., and Woeginger, G. (1998). *A review of machine scheduling: complexity, algorithms, and approximation*. Kluwer, Dordrecht.

Chen, Z.-L. and Hall, N. G. (2007). Supply chain scheduling: Conflict and cooperation in assembly systems. *Operations Research*, 55(6):1072–1089.

Cheng, T. and Sin, C. (1990). A state-of-the-art review of parallel-machine scheduling research. *European Journal of Operational Research*, 47:271–292.

Chopra, S. and Sodhi, M. S. (2004). Managing risk to avoid supply-chain breakdown. *MIT Sloan Management Review*, 46(1):52–61.

Dawande, M., Geismar, H. N., Hall, N. G., and Sriskandarajah, C. (2006). Supply chain scheduling: Distribution systems. *Production and Operations Management*, 15(2):243–261.

De, P., Ghosh, J. B., and Wells, C. E. (1992). On the minimization of completion time variance with a bicriteriaextension. *Operations Research*, 40(6):1148–1155.

Eilon, S. and Chowdhury, I. G. (1977). Minimizing waiting time variance in the single machine problem. *Management Science*, 23(6):567–575.

Gajpal, Y. and Rajendran, C. (2006). An ant-colony optimization algorithm for minimizing the completion-timevariance of jobs in flowshops. *Int. J. Production Economics*, 101:259–272.

Graham, R. L., Lawler, E. L., Lenstra, J. K., and Kan, A. H. R. (1979). Optimization and approximation in deterministic sequencing and scheduling: A survey. *Annals of Discrete Mathematics*, 5:287–326.

Gupta, M. C., Gupta, Y. P., and Bector, C. R. (1990). Minimizing the flow-time variance in single-machine systems. *Journal of the Operational Research Society*, 41:767–779.

Gupta, M. C., Gupta, Y. P., and Kumar, A. (1993). Minimizing flow time variance in a single machine system using geneticalgorithm. *European Journal of Operational Research*, 70:289–303.

Hall, N. and Potts, C. (2003). Supply chain scheduling: batching and delivery. *Operations Research*, 51(4):566–584.

Hall, N. G. and Kubiak, W. (1991). Proof of a conjecture of schrage about the completion time varianceproblem. *Operations Research Letters*, 10:467–472.

Hoogeveen, H. and Vestjens, A. P. A. (1996). Optimal on-line algorithms for single-machine scheduling. In Cunningham, W., McCormick, S., and Queyranne, M., editors, *Proceedings of the Fifth Conference on Integer Programming and Combinatorial Optimization*, volume 1084, pages 404–414. Springer. Lecture Notes in Computer Science.

Hurter, A. P. and Van Buer, M. G. (1996). The newspaper production/distribution problem. *Journal of Business Logistics*, 17(1):85–107.

Kanet, J. J. (1981). Minimizing variation of flow time in single machine systems. *Management Science*, 27:1453–1459.

Kubiak, W. (1993). Completion time variance on a single machine is difficult. *Operations Research Letter*, 12:49–59.

Kubiak, W., Cheng, J., and Kovalyov, M. Y. (2002). Fast fully polynomial approximation schemes for minimizing completiontime variance. *European Journal of Operational Research*, 137:303–309.

Lee, H. L., Padmanabhan, V., and Whang, S. (1997). Information distortion in a supply chain: The bullwhip effect. *Management Science*, 43(4):546–558.

Lee, H. L., So, K. C., and Tang, C. S. (2000). The value of information sharing in a two-level supply chain. *Management Science*, 46(5):626–643.

Lee, H. L. and Whang, S. (2000). Information sharing in a supply chain. *International Journal of Manufacturing Technology and Management*, 1(1):79–93.

Leung, J. Y.-T., editor (2004). *Handbook of Scheduling: Algorithms, Models and Performance Analysis*. Chapman & Hall/CRC, 1st edition.

Li, X., Chen, Y., Sun, Y., and Sawhney, R. (2009). On the minimization of job completion time variance on identical parallel machines. (Submitted).

Manna, D. K. and Prasad, V. R. (1997). Pseudopolynomial algorithms for ctv minimization in single machinescheduling. *Computers and Operations Research*, 24(12):1119–1128.

Manna, D. K. and Prasad, V. R. (1999). Bounds for the position of the smallest job in completion time varianceminimization. *European Journal of Operational Research*, 114(2):411–419.

Megow, N. and Schulz, A. (2004). On-lline scheduling to minimize average completion time revisited. *Operations Reaesrch Letters*, 32(5):485–490.

Merten, A. and Muller, M. (1972). Variance minimization in single machine sequencing problems. *Management Science*, 18(9):518–528.

Mittenthal, J. and Raghavachari, M. (1993). Stochastic single machine scheduling with quadratic early-tardy penalties. *Operations Research*, 41:786–796.

Mittenthal, J., Raghavachari, M., and Rana, A. (1993). A hybrid simulated annealing approach for single machine schedulingproblems with non-regular penalty functions. *Computers and Operations Research*, 20(2):103–111.

Nash, J. F. (1950). The bargaining problem. *Econometrica*, 18:155–162.

Nash, J. F. (1953). Two-person cooperative games. *Econometrica*, 21:128–140.

Pinedo, M. (2002). *Scheduling Theory, Algorithms, and Systems*. Prentice-Hall, Inc., 2nd edition.

Prasad, V. R. and Manna, D. K. (1997). Minimization of expected variance of completion times on single machine for stochastic jobs. *Naval Research Logistics*, 44:97–108.

Sarmiento, A. and Nagi, R. (1999). A review of integrated analysis of production-distribution systems. *IIE Transactions*, 31:1061–1074.

Schrage, L. (1975). Minimizing the time-in-system variance for a finite jobset. *Management Science*, 21(5):540–543.

Silver, E., D.F., P., and P., R. (1998). *Inventory Management and Production Planning and Scheduling*. New York: Wiley, 3rd edition.

Simchi-Levi, D., Kaminsky, P., and Simchi-Levi, E. (2003). *Designing & Managing the Supply Chain: Concepts, Strategies & Case studies*. McGraw-Hill/Irwin.

Thomas, D. and Griffin, P. (1996). Coordinated supply chain management. *European Journal of Operational Research*, 94:1–15.

Tomlin, B. (2006). On the value of mitigation and contingency strategies for managing supply chain disruption risks. *MANAGEMENT SCIENCE*, 52(5):639–657.

Vani, V. and Raghavachari, M. (1987). Deterministic and random single machine sequencing with varianceminimization. *Operations Reserach*, 35(1):111–120.

Viswanathkumar, G. and Srinivasan, G. (2003). A branch and bound algorithm to minimize completion time varianceon a single processor. *Computers and Operations Research*, 30(8):1135–1150.

Xu, X. and Ye, N. (2007). Minimization of job waiting time variance on identical parallel machines. *IEEE Transactions on Systems, Man, and Cybernetics, Part C.*, 37(5):917–927.

Ye, N., Li, X., Farley, T., and Xu, X. (2007). Job scheduling methods for reducing waiting time variance. *Computers and Operations Research*, 34:3069–3083.

Vita

Yuerong Chen received both B.S. degree in Information Sciences and M.S. degree in Probability and Statistics from College of Mathematical Sciences, Nankai University, Tianjin, China. In the spring of 2006, she joined as a graduate research assistant the Intelligent Information Engineering and Systems Laboratory (IIESL) at the Department of Industrial and Information Engineering, The University of Tennessee at Knoxville. She is expected to complete her Doctor of Philosophy degree in 2009. She has published several peer-review journal and conference papers. She received Best Master's Thesis Award from Nankai University in 2005 and Best Paper Award from the 2007 Industrial Engineering Research Conference. She is an IIE member.