國立交通大學

資訊管理研究所

碩士論文

差異化生產流程之批次排程

Batch Scheduling in Differentiation Flowshops

研究生：黃鼎智

指導教授：林妙聰　博士

中華民國　九十九　年　六　月

差異化生產流程之批次排程

Batch Scheduling in Differentiation Flowshops

研究生：黃鼎智
指導教授：林妙聰　博士

Student: Huang Ting-Chih
Advisor: Bertrand M.T. Lin, Ph.D.

國立交通大學
資訊管理研究所
碩士論文

A Thesis
Submitted to the Institute of Information Management
College of Management
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Master of Business Administration
in
Information Management
June 2010
Hsinchu, Taiwan, the Republic of China

中華民國　九十九　年　六　月

誌謝

交大如隙，我如白駒；白駒過隙，心猶未覺。

眞的很高興也很榮幸能有機會進入交大資管所就讀，並在此畫上我學生時代最終的句點。但要能有今日完美的句點，該感謝的人非常多，首爲我的父母親。從小，我的父母對於課業上的堅持以及每日的諄諄教誨，使得我對學習有良好的態度。這對我在往後的每一步學習路程上，都有非常大的幫助。 再者，能有幸進入最佳化理論與應用實驗室可說是我的榮幸。感謝我的指導教授　林妙聰老師。先是爲了我破例而收了三個學生。再者，雖然身心皆忙於所長之職，在面對我總是死纏爛打的追問之下，卻總能不辭勞苦的回答我，並導引我至正確的方向之上。而後，在論文晦澀艱深的數學理論之中，老師總能像那明亮的燈塔般的帶領我返航。若沒有老師的費心指導，這篇論文將不可能完成。

同時，尚要感謝研究室的各位夥伴與同學。感謝筱嵐學姐，妳那永保童稚的心與清晰的思維總能爲人帶來歡樂、鼓勵及新的想法。感謝怡菱學姐，妙不可言與爲人著想的妳，總能帶我從心煩中走出。感謝鋒樟學長，一起研究一起打球，爲我最後的學生生活添增不少趣味。感謝癸棠學長，有你對研究室的悉心付出，我們才能有這麼好的研究環境。感謝彥君學長，時常與我分享生活討論資訊技術，帶我認識許多新事物。感謝彥徵同學，讓我在最後，還能有充足的時間使我的論文與口試更臻完美。感謝昕逸學弟，總陪我吃飯聊天談心，使得研究生活不至於苦悶。

最後，感謝我身邊的所有人，有你們的陪伴與幫助，今天的我才能順利的繼續往人生的下一階段前進。我也期望你們在未來都過的很好，感謝！

Abstract

This thesis considers a three-machine two-stage differentiation flowshop to minimize the makespan. The flowshop comprises a common machine on stage-one and two independent dedicated machines on stage-two. All jobs consist of two operations, the first one is performed on the stage-one machine and the second is performed on the stage-two machine. All jobs share, and compete for, the common stage-one machine. A constant setup time is required whenever a batch is formed on the stage-one critical machine. Two different modes of batch composition, compatible and incompatible, are investigated. We also assume that the sequences of two types of jobs are fixed. The objective function to minimize is the makespan, or the maximum completion time. We proposed two polynomial-time dynamic programming algorithms for optimally solving the problems with compatible batching and incompatible batching, respectively. We then deploy the dynamic programming algorithms for computing lower bounds of the original problem without the assumption of given job sequences.

keywords: Delay differentiation; flowshop; batch; makespan; critical machine; lower bound; dynamic programming.

摘要

本篇論文探討一個三機台兩階段之差異化流線型機台組之生產模型，其包含一台主要機台與兩台專用機台。所有工作都包含二項作業，工作的第一項作業在第一階段的處理當中皆共享同一主要機台，而在第二階段當中，工作的第二項作業即由該工作所屬之專用機台處理之。在第一台機台上，所有的批次處理前皆需要一個固定的準備時間。本篇論文考慮二種批次處理的形式，分別爲可相容的與不可相容的批次處理。在某些情況中，我們假設二種類型工作的執行順序已經給定且不可更動。探討之目標函數是要最小化最大完工時間。針對兩類批次形式，我們分別提出動態規劃演算法求取最佳解。另外，我們也利用此兩個演算法求取未給定執行順序之原始問題最佳解之下界值。

關鍵字：差異化生產、流線型機組、批次排程、 最大完工時間、下界函數、動態規劃。

Contents

誌謝 **i**

Abstract **ii**

摘要 **iii**

1 Introduction **1**

2 Problem Statements and Literature Review **5**

2.1 Problem Definition and Notation 5

2.2 Literature Review 8

3 Batching with Compatible Items **11**

3.1 $F(1,2)|\text{comp, s-batch}|C_{\max}$ with regards to fixed sequences 11

3.2 $F(1,2)|\text{comp, s-batch}|C_{\max}$ 14

4 Batching with Incompatible Items **19**

4.1 Preliminaries 19

4.2 Recursions and Run Time Analysis 21

5 Concluding Remarks **29**

Bibliography **31**

List of Figures

1.1 Differentiation flowshop model 4

2.1 Gantt chart of batch sequence σ_1 8

2.2 Gantt chart of batch sequence σ_2 8

4.1 Gantt chart of one state of function f 21

Chapter 1

Introduction

Delayed differentiation or postponement is a concept in supply chain management where the manufacturing process starts by making a generic or family product that is later differentiated into a specific end-product. This is a widely used method, especially in industries with high demand uncertainty or the need of mass customization, and can be effectively used to address the final demand even if forecasts cannot be improved. In particular, the model, delayed differentiation flowshops, is one of the major approaches to achieving mass customization (Da Silveira, et al. 2001; Simchi-Levi, et al. 2000). In the global marketplace nowadays, the trend of customization is a major reason for demand uncertainty. In order to compete effectively in the marketplace, firms have to keep their operations flexible to overcome various problems, such as demand uncertainty. Operational flexibility has received considerable attention because it can give consideration to demand uncertainty and inventory cost. Since the flexibility of operations is crucial to the manufacturing firm's strategy, it is conceivable that the delayed differentiation can bring respectable revenue to the manufacturing firm if it can be adopted appropriately. Gupta and Benjaafar (2003) also mentioned that delayed differentiation carries several benefits. Maintaining stocks of semi-finished goods reduces the order-fulfillment delay relative to the pure MTO (Make-To-Order) system. Since many different end products have common parts, holding semi-finished goods inventory benefits from demand pooling, which is known to lower the amount of inventory needed to achieve a service-level performance equal to that of a comparable system with no pooling (Eppen 1979). Furthermore, investment in semi-finished inventories is smaller when compared with the option to maintain a similar amount of finished-goods inventory. There is also the benefit of learning, realized from having better demand information before committing generic semi-finished products to unique end products. Additional benefits from delaying differentiation include a significant streamlining of the MTS (Make-To-Stock) segment of the manufacturing process and simplification of production scheduling, sequencing and raw material purchasing.

In the following, we introduce five real applications where delayed differentiation is deployed.

1. Chair: The main body of the chair is manufactured on the first production stage. There are several different head-supports assembled on the second stage. Thus, each chair proceeds to a different machine for this second stage. Clearly, the second stage can consist of an assembly of a final part as described above, or of coloring (by one of several possible colors), or of packing (in one of several types of boxes), etc. (Cheng et al. 2009)

2. Pottery: The main glazing process of potteries may be performed on the first production stage. Several heating processes for distinct figures or dissimilar effects are applied on the second stage. That is, each pottery proceeds to a corresponding dedicated machine for baking process after being glazed. The second stage may consist of re-glazing, various thermal treatments, or packing. (Cheng et al. 2009)

3. Knitwear: Benetton is a world leader in knitwear. It resides in a large logistics network where many retailers are involved. The problem Benetton faced is that it is hard to predict what items, colors, etc. will sell. So Benetton redesigned their manufacturing process such that all knitwear are initially all white, and then dyed into different colors only when the season/customer color preference/demand is known. By postponing the step of dyeing, Benetton is able to successfully delay color selection until the season's fashion preferences become more established. (Bruce 1987)

4. Computer: Dell built a global business on selling and configuring personal com-

puters when orders are received, rather than stockpiling finished product on the basis of demand forecasts. Besides, it is almost impossible to use the MTS (Make-To-Stock) system, because the configuration of every client want is unpredictable. Dell postpones final assembly until an order arrives via its online retailing network. Clearly, it is also an application of differentiation flowshops if we treat the process of manufacturing the component of computers as a job on the stage-one and the process of assembling and configuring as a stage-two job. (Magretta 1998)

5. Printer: Delayed differentiation also is embedded in the manufacturing processes of Hewlett-Packard's printer. The company's Deskjet and Deskwriter printers are made in its Vancouver and Singapore plants and distributed to the U.S., Europe and Asia. Selling printers in Europe means following each country's requirements for printer configurations: different decals, a country-specific power plug and language-specific manuals. (Feitzinger and Lee 1997)

In this thesis, we link a two-stage supply chain with a two-stage flowshop. When we consider a supply chain in an abstract way, it can be viewed as a production line within an organization. A flowshop-type production consists of machine arranged in series such that all products need to be processed in the order of the machines are arranged. Kyparisis and Koulamas (2000) who studied this two-stage flowshop mention that "applications of the proposed flowshop model are encountered in manufacturing settings, where all jobs must first go through the same main process, and then they require a finishing operation special to the job".

Scheduling refers to managerial decision making that allocates limited resources to activities so as to optimize, subject to functional constraints or assumptions, a certain set of performance measures. Scheduling is crucial to operations management of applications in manufacturing and service industries (Pinedo 2002, Pinedo and Chao 1999). Since the seminal work of Johnson (Johnson 1954), flowshop scheduling has been receiving considerable research attention (Dudek, et al. 1992; Reisman, et al. 1997). This broad topic contains many different settings and special cases, reflecting a wide range of applications. In this thesis, we consider a special three-machine two-stage flowshop called differentia-

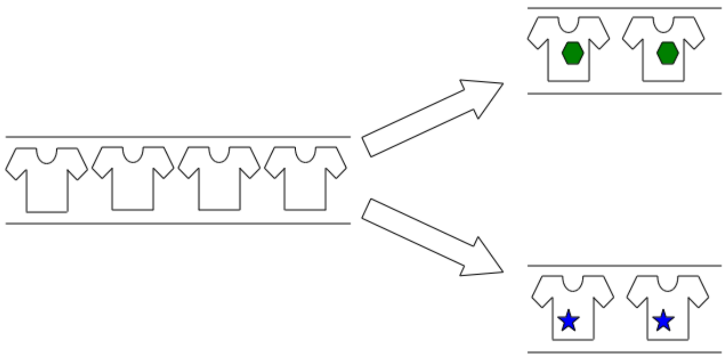

Figure 1.1: Differentiation flowshop model

tion flowshops, where all products (jobs) share a common critical machine on the primary (first) stage, and then each individual product proceeds to a dedicated machine on the successive (second) stage. Please refer to Figure 1.1 for the machine configuration. Many manufacturing environments which produce multiple final products are extensions of this basic model. In this thesis, we study the manufacturing environment shown in Figure 1.1. Under such an environment, the stage-one machine is common for all products. This means that the model can be used to achieve mass production of homogeneous products at the first stage, and the products proceed to the stage-two machines for further differentiation operations. As batch processing is common in mass production, we consider the production environment where batching is required on the stage-one machine.

The rest of this thesis is organized as follows. In Chapter 2, we will present formal statements of the problem definition, and the notation that is used throughout the paper. Review on related works follows. As the studied problem is known to be NP-hard, we will study the scenario that the production sequence of products on each stage-two machine is known and given. Chapter 3 presents an algorithm for the differentiation flowshop problem with compatible batching. We dedicate Chapter 4 to the development of a dynamic programming algorithm for the differentiation flowshop problem with incompatible batching. We give concluding remarks and suggest potential research directions in Chapter 5.

Chapter 2

Problem Statements and Literature Review

In this chapter, we first introduce the notation used in this thesis. A formal definition of the studied problem follows. A numerical example will be given for illustration. Related works will also be reviewed.

2.1 Problem Definition and Notation

The notation that will be used throughout this thesis is defined as follows:

Notation:

n_1: the number of type-1 jobs;

n_2: the number of type-2 jobs;

$n = n_1 + n_2$: the number of all jobs;

$\mathcal{I} = \{I_1, I_2, \ldots, I_{n_1}\}$: the set of type-1 jobs;

$\mathcal{J} = \{J_1, J_2, \ldots, J_{n_2}\}$: the set of type-2 jobs;

M_0: the stage-one common critical machine;

M_1: the stage-two dedicated machine for jobs in $\mathcal{I}$;

M_2: the stage-two dedicated machine for jobs in $\mathcal{J}$;

$I_{i,m}$: the operation of job $I_i \in \mathcal{I}$ on stage m, $m = 1, 2$;

$J_{j,m}$: the operation of job $J_j \in \mathcal{J}$ on stage m, $m = 1, 2$;

$p_{I_{i,m}}$: the processing time on stage m, $m = 1, 2$, of job $I_i \in \mathcal{I}$;

$p_{J_{j,m}}$: the processing time on stage m, $m = 1, 2$, of job $J_j \in \mathcal{J}$;

$p_{I_{(i),m}}$: the i-th smallest processing time on stage m, $m = 1, 2$, in $\{p_{I_{1,m}}, p_{I_{2,m}}, \ldots, p_{I_{n_1,m}}\}$;

$p_{J_{(j),m}}$: the j-th smallest processing time on stage m, $m = 1, 2$, in $\{p_{J_{1,m}}, p_{J_{2,m}}, \ldots, p_{J_{n_2,m}}\}$;

s: batch setup time on machine M_0;

B_l: the l-th batch on machine M_0;

$C^*_{\max}(\mathcal{I}' \cup \mathcal{J}')$: the optimal makespan of job set $\mathcal{I}' \cup \mathcal{J}'$ for $\mathcal{I}' \subseteq \mathcal{I}$ and $\mathcal{J}' \subseteq \mathcal{J}$;

$S(\mathcal{I}' \cup \mathcal{J}')$: a particular schedule of job set $\mathcal{I}' \cup \mathcal{J}'$ for $\mathcal{I}' \subseteq \mathcal{I}$ and $\mathcal{J}' \subseteq \mathcal{J}$;

$C_{\max}(S(\mathcal{I}' \cup \mathcal{J}'))$: the makespan of schedule $S(\mathcal{I}' \cup \mathcal{J}')$;

$S^*(\mathcal{I}' \cup \mathcal{J}')$: optimal schedule of job set $\mathcal{I}' \cup \mathcal{J}'$ for $\mathcal{I}' \subseteq \mathcal{I}$ and $\mathcal{J}' \subseteq \mathcal{J}$.

The problem is formally defined as follows. The manufacturing model is a three-machine two-stage differentiation flowshop consisting of a stage-one common critical machine and two independent dedicated machines in the second stage. The jobs belong to two different types: type 1, $\mathcal{I} = \{I_1, I_2, \ldots, I_{n_1}\}$ and type 2, $\mathcal{J} = \{J_1, J_2, \ldots, J_{n_2}\}$. There are in total $n = n_1 + n_2$ jobs to process in the differentiation flowshops. Each job in $\mathcal{I}$ comprises two operations, the first of which is performed on the stage-one common machine M_0, and the second is performed on the first dedicated machine M_1, as in the classical two-machine flowshop. Similarly, the jobs of $\mathcal{J}$ are processed first on the common critical machine and then on the second dedicated machine M_2. Jobs of both types are processed on the common critical machine in batches. A constant setup time s is required whenever a batch is formed on the stage-one critical machine. The batch scheduling model we adopt in this thesis is sum-batch or sequential-batch, under which the processing length of a batch is the setup time plus the processing lengths of the jobs contained in the batch. Batch availability is assumed, i.e., the first operation of a job is finished and the second operation is available for proceeding to the second stage when all operations in the batch to which the job belongs has been finished. We will investigate two modes of batch composition on the common machine: compatible and incompatible. In compatible composition, jobs from different types can reside in the same batch. On the other hand, incompatible composition requires that any batch must contain jobs of the same type. The objective function considered is the makespan, i.e., the maximum completion time of all jobs.

To the best of our knowledge, the production model was not investigated in the literature. Our study is to investigate the scheduling problem of minimizing the makespan, i.e., the maximum completion time of all jobs.

As the problem is known to be strongly NP-hard, in this thesis we consider a simplified situation where the sequences of jobs for each type on the stage-one machine are known and fixed. Subject to this assumption, the problem reduces to finding how to interleave two sequences of jobs and how to batching the jobs on stage-one machine. Besides, we consider only permutation schedule, that is, jobs of the same type have the same processing sequence on the critical machine and on their dedicated machine. Each machine can process at most one operation at any time, and no preemption is allowed. The objective is to batch and schedule the jobs so as to minimize the maximum completion time.

To illustrate the problem definition, a numerical instance is given as follows. There are four jobs in two types to be scheduled: type 1, $\mathcal{I} = \{I_1, I_2\}$ and type 2, $\mathcal{J} = \{J_1, J_2\}$. The batch setup time is 1. The processing times are shown below.

Jobs	I_1	I_2	J_1	J_2
stage-one	$p_{I_{1,1}} = 2$	$p_{I_{2,1}} = 5$	$p_{J_{1,1}} = 4$	$p_{J_{2,1}} = 3$
stage-two	$p_{I_{1,2}} = 4$	$p_{I_{2,2}} = 3$	$p_{J_{1,2}} = 6$	$p_{J_{2,2}} = 2$

Given two batch sequences $\sigma_1 = ((I_1, J_2), (J_1, I_2))$ and $\sigma_2 = ((J_2, J_1), (I_1, I_2))$, we have two corresponding Gantt charts as shown in Figure 2.1 and Figure 2.2. Batch sequence σ_1 has the makespan of 22 and batch sequence σ_2 has the makespan of 23. Note that batch sequence σ_1 is scheduled under the compatible batching mode, and batch sequence σ_2 is scheduled under the incompatible mode.

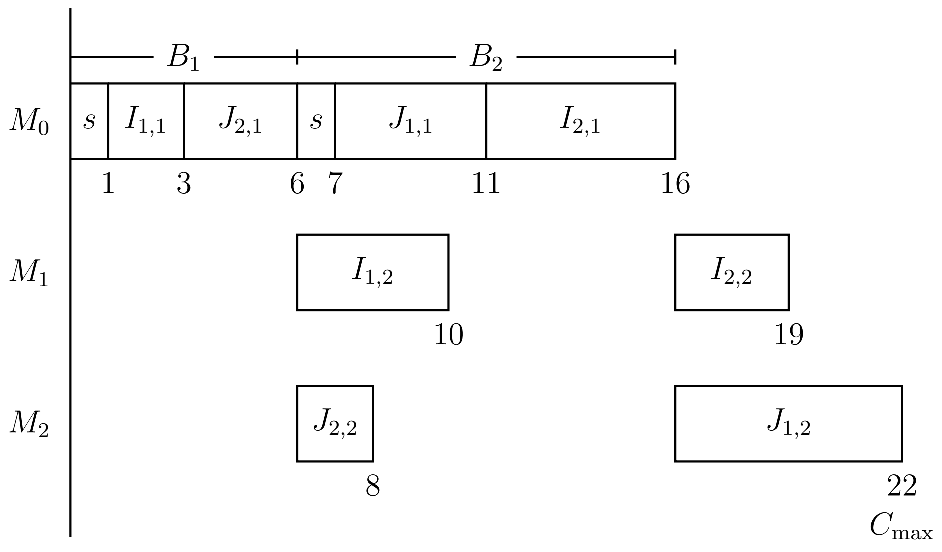

Figure 2.1: Gantt chart of batch sequence σ_1

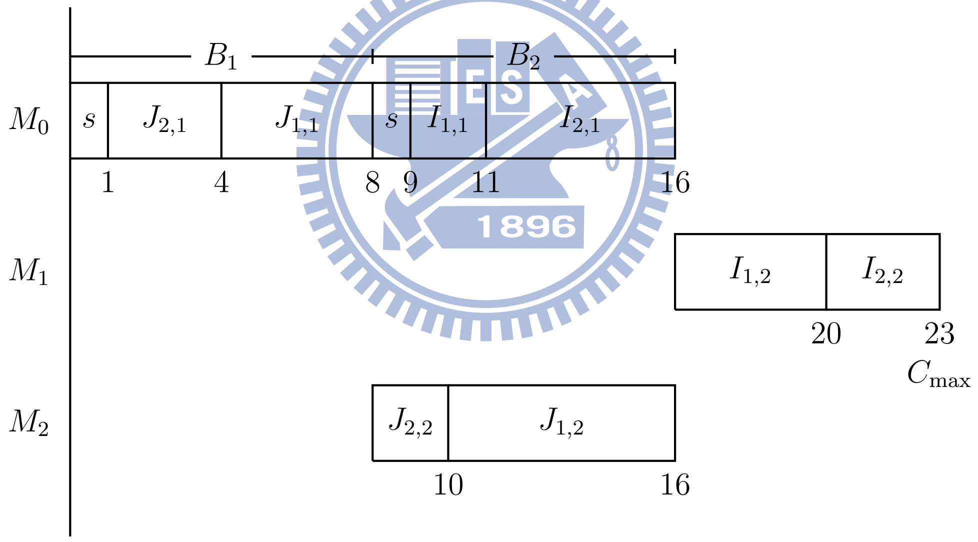

Figure 2.2: Gantt chart of batch sequence σ_2

2.2 Literature Review

It can be easily seen that when there is only one type of jobs and there is only one dedicated machine at stage two, the problem reduces to the classical Johnson's two-machine flowshop scheduling problem (Johnson 1954), which can be solved in $O(n \log n)$ time. The model of delayed differentiation studied in this thesis probably first investigated by Herrmann and Lee (1992), in which three objectives, makespan, number of tardy jobs and maximum tardiness, were investigated. In their research, they defined two types of dispatching rule, look-ahead and look-behind, to approach to scheduling job shops. Look-ahead and look-behind scheduling includes procedures that look around the shop for more information to use in making a scheduling decisions. Look-ahead models consider the machines where the jobs will be headed after this stage. Look-behind models on the other hand consider the job that will be arriving at this machine soon. They use the terms look-ahead and look-behind to designate scheduling procedures that do more than consider just the state of one machine.

Drobouchevitch and Strusevich (2000) studied the two-stage job shop scheduling problem with a bottleneck machine which can be thought as a general case of our problem. Given an arbitrary number of stage-two machines in a job shop, they designed a heuristic algorithm for makespan minimization with a performance ratio 3/2. Without knowing the existence of Herrmann and Lee (1992) and Drobouchevitch and Strusevich (2000), Kyparisis and Koulamas (2000) investigated the same model but with m types of jobs, and correspondingly m dedicated machines at the second stage in the case of flow shop and open shop. Their model is polynomially solvable under a strong assumption, called block assumption, that jobs of the same type must be processed contiguously on the stage-one machine. Under the such assumption, they developed a makespan minimizing algorithm in $O(m(n \log n + \log m))$, where n is the total number of jobs from all types. Mosheiov and Yovel (2004) improved Kyparisis and Koulamas's algorithm and reduced the time complexity to $O(n \log n)$ subject to the common constraint $m \leq n$. Cheng and Kovalyov (1998) incorporated the same model but with the job dependent setup time. They proposed an $O((n_1 + n_2)n_1^4 n_2^3)$ dynamic programming algorithm, where n_1 and n_2 are the number of jobs of type-1 and type-2, for makespan minimization. Particularly, the reverse model of delayed differentiation with dedicated machine installed on stage one was studied by Oğuz et al. (1997). They showed that minimizing makespan is ordinarily NP-hard by a reduction from Partition. A strongly NP-hard proof was given by Lin (1999) for the same problem. Neumytov and Sevastyanov (1993) in Russian studied the same problem.

While the three-machine two-stage differentiation flowshops model has been investigated in some research papers, to the best of our knowledge, no previous work has ever addressed the batch considerations. Batch scheduling has received considerable research attention in the past two decades. There is a large body of research works on this subject. Cheng et al. (1999) and Potts and Kovalyov (2000) are two excellent reviews provide a broad coverage and a comprehensive classification scheme. In the literature, batch scheduling adopting the max-batch model with batch-item availability options has been addressed in the single-machine setting (Aneja and Singh 1990; Baker 1988; Gerodimos et al. 2000; Lin 2002; Vickson et al. 1993) and two-machine flowshop settings (Cheng et al. 2000; Cheng and Wang 1999; Lin and Cheng 2002).

Chapter 3

Batching with Compatible Items

3.1 $F(1,2)|$comp, s-batch$|C_{\max}$ with regards to fixed sequences

In this section, we study the batch scheduling problem with compatible batches to minimize the makespan subject to the condition that the processing sequences of either types of jobs are known and fixed. For the studied problem, the term "comp" is present in the second field to specify that a batch is allowed to contain different types of jobs. Given a batch scheduling system and the set of jobs to be processed, the scheduling problem is to decide the composition of each batch, i.e., the assignment of jobs to batches, the sequence of the batches to be processed, and the sequence of the jobs in each batch. The assumption of fixed sequences refers to assumption that the job sequences of $\mathcal{I}$ and $\mathcal{J}$ are given and fixed. Since we have the assumption of fixed sequences, the decision remains here is how to interleave two given job sequences and also group the jobs into batches on the stage-one machine. It is easy to show that there exists at least one optimal solution which is a permutaton schedule. Therefore, we consider only permutaion schedule in this thesis, i.e., the processing sequences on both critical and dedicated machines of two types of jobs are the same.

The first issue concerns optimally interleaving two sub-sequences without batching. Herrmann and Lee(1992) proposed a polynomial time algorithm for resolving this issue.

Their approach first associates every job with a due data and transforms the problem into another scheduling problem that seeks to find a schedule with the minimum maximum lateness, $L_{\max}$. Denote the job sequence of $\mathcal{I}$ as $\sigma_{\mathcal{I}}$, $\sigma_{\mathcal{I}} = (I_1, I_2, \ldots, I_{n_1})$, and job sequence of $\mathcal{J}$ as $\sigma_{\mathcal{J}}$, $\sigma_{\mathcal{J}} = (J_1, J_2, \ldots, J_{n_2})$. The following algorithm can yield a schedule with the minimum $L_{\max}$:

Algorithm HL

Input: $\sigma_{\mathcal{I}}$ and $\sigma_{\mathcal{J}}$

Output: A sequence of jobs of $\mathcal{I} \cup \mathcal{J}$ with the minimum $L_{\max}$.

Step 1. For each job I_k of $\mathcal{I}$, define A_h as the set of jobs (not including I_k) that follow I_k in $\sigma_{\mathcal{I}}$. Then, $d_k = p_{I_{k,2}} + \sum_{A_h} p_{I_{k,2}}$.

Step 2. For each job J_k of $\mathcal{J}$, define A_h as the set of jobs (not including J_k) that follow J_k in $\sigma_{\mathcal{J}}$. Then, $d_k = p_{J_{k,2}} + \sum_{A_h} p_{J_{k,2}}$.

Step 3. Schedule the jobs on machine M_0 in non-increasing order of the d_k, starting at time zero, and start all second stage operations as soon as possible.

Step 1 and Step 2 of **Algorithm HL** respectively require $O(n_1)$ time and $O(n_2)$ time to compute the due-dates. Step 3 takes $O(n_1 + n_2) = O(n)$ time to interleave two job sequences, since jobs in each sequence are already sorted in non-increasing order of the d_k, and forming the one job sequence which will be scheduled on machine M_0 is only to combine the two sequences without changing the relative ordering in each sequence.

While the interleaving issue is resolved, we proceed to the batching issue. On a single machine, problem $1|\text{s-batch}|L_{\max}$ is to sequence as well as batch the jobs so as to minimize the maximum lateness.

Lemma 3.1. *(Webster and Baker, 1995) There is an optimal schedule for the* $1|\textit{s-batch}|L_{\max}$ *problem in which the jobs are sequenced by the EDD rule.* $\square$

Based upon Lemma 3.1, Webster and Baker (1995) developed a backward dynamic programming algorithm for the $1|\text{s-batch}|L_{\max}$ problem. Assume that the jobs are indexed in non-decreasing order of their due dates. Let $B(i)$ denote the optimal $L_{\max}$ of the single machine problem for jobs $J_i, J_{i+1}, ..., J_n$. The dynamic program defines a recursive formula to derive $B(i)$ from $B(j), 1 \leq i < j \leq n$, by inserting a batch of jobs $\{J_i, ..., J_{j-1}\}$ in front of the schedule associated with $B(j)$. The algorithm is given as follows:

Algorithm WB

Initialization:

Set $B(n+1) = -\infty$.

Recursion:

$$B(i) = \min_{i+1 \leq j \leq n+1} \left\{ \left(s_1 + \sum_{r=i}^{j-1} p_{1r} \right) + \max\{-d_i, B(j)\} \right\}.$$

The optimal $L_{\max}$ is given by $B(1)$. The detail grouping decision can be obtained by backtracking. As for the running time, we note that there are $O(n)$ states, each of which requires $O(n)$ time to enumerate different job indices j. The overall computation effort for calculating $B(1)$ is thus $O(n^2)$.

Based upon their algorithm, we can solve the $F(1,2)|\text{comp, fixed\_seq, s-batch}|C_{\max}$ problem optimally in polynomial time. We first treat the operations of jobs of $\mathcal{I} \cup \mathcal{J}$ on machine M_0 as jobs and associate them with due dates as defined in **Algorithm HL**. Then, Step 3 of **Algorithm HL** produces an EDD job sequence on machine M_0. Then, we apply **Algorithm WB** to optimally group the jobs in the sequence. The following theorem thus follows.

Theorem 3.1. *Problem $F(1,2)|$comp, fixed_seq, s-batch$|C_{\max}$ can be solved optimally in $O(n^2)$ time.*

Proof. As explained above, solving $F(1,2)|\text{comp, fixed\_seq, s-batch}|C_{\max}$ consists of invoking two algorithms, **Algorithm HL** and **Algorithm WB**. The **Algorithm HL** takes $O(n)$ time to interleave two job sequences. And, **Algorithm WB** requires $O(n^2)$ to batch the jobs on machine M_0. Therefore, the overall complexity is $O(n^2)$ and the proof is complete. $\square$

3.2 $F(1,2)|$comp, s-batch$|C_{\max}$

In this section, we will return to the $F(1,2)|$comp, s-batch$|C_{\max}$ problem without the assumption of two fixed sequences. Without the assumption of fixed sequences here, the general problem of differentiation flowshop to minimize makespan, $F(1,2)||C_{\max}$, was proven to be strongly NP-hard by a reduction from 3-partition by Herrmann and Lee (1992). From the existing result, it is obvious that $F(1,2)|$comp, s-batch$|C_{\max}$ is strongly NP-hard even without any batching concern.

The NP-hardness indicates that it is very unlikely to design a polynomial time algorithm for producing optimal solutions. Branch and bound algorithms are one of the exact methods which is widely adopted for tackling hard optimization problems. Effective lower bounds, used to pruned off non-promising solutions, are crucial to the efficiency of branch and bound algorithms. In the following, we will develop a lower bound based upon the result of Theorem 3.1.

For any job set $\mathcal{I} \cup \mathcal{J}$ to be processed, we can derive another job set $\mathcal{I}' \cup \mathcal{J}'$ by rearranging the operations of the jobs to create a data set called *ideal data set*. In makespan minimization of a two-machine flowshop problem, a job is preferred to be processed first if its processing time on stage-one is shorter and its processing time on stage-two is longer. A data set is ideal if and only if the data set contains jobs where a job with a shorter processing time on stage-one has a longer processing time on stage-two. In the following, we create such an ideal data set $\mathcal{I}' \cup \mathcal{J}'$ from the given job set $\mathcal{I} \cup \mathcal{J}$. For all $k = 1, 2, \ldots, n_1$, job I'_k in $\mathcal{I}'$ is defined by two parameters:

1. $p_{I'_k,1} = p_{I_{(k)},1}$, i.e., the k-th smallest element among $p_{I_{1,1}}, p_{I_{2,1}}, \ldots, p_{I_{n_1,1}}$.
2. $p_{I'_k,2} = p_{I_{(k)},2}$, i.e., the k-th largest element among $p_{I_{1,2}}, p_{I_{2,2}}, \ldots, p_{I_{n_1,2}}$.

In a similar way, job J'_k in $\mathcal{J}'$ is defined by $p_{J_{(k)},1}$ and $p_{J_{(n_2-k+1)},2}$ for all $k = 1, 2, \ldots, n_2$.

We index the jobs in $\mathcal{I}'$ and $\mathcal{J}'$ in non-decreasing order of their processing time on machine M_0 and denote the sequence of $\mathcal{I}'$ (respectively, $\mathcal{J}'$) as $\sigma_{\mathcal{I}'}$ (respectively, $\sigma_{\mathcal{J}'}$).

Lemma 3.2. *There is an optimal schedule of the instance $\mathcal{I}' \cup \mathcal{J}'$ where jobs of $\mathcal{I}'$ are sequenced by $\sigma_{\mathcal{I}'}$ and jobs of $\mathcal{J}'$ are sequenced by $\sigma_{\mathcal{J}'}$.*

Proof. Without loss of generality, we assume that the processing times of all operations are distinct. Given a schedule of the instance $S(\mathcal{I}' \cup \mathcal{J}')$, for each pair of jobs $\{(I'_k, I'_{k+1})\,|k = 1, 2, \ldots, n_1 - 1\}$ in $S(\mathcal{I}' \cup \mathcal{J}')$, we swap their position if $p_{I'_{k,1}} > p_{I'_{k+1,1}}$. Note that if the condition is met, it implies $p_{I'_{k,2}} < p_{I'_{k+1,2}}$. The similar job-interchange technique is applied to job set $\mathcal{J}'$. Since the total idle time on either dedicated machine will not increase after the job interchange, it is clear that the makespan will not increase in the derived schedule. Repeating the job interchange, if necessary, will finally lead to a schedule in which all jobs of $\mathcal{I}'$ are sequenced by $\sigma_{\mathcal{I}'}$ and jobs of $\mathcal{J}'$ are sequenced by $\sigma_{\mathcal{J}'}$. We complete the proof. □

Lemma 3.3. *An optimal schedule of the job set $\mathcal{I}' \cup \mathcal{J}'$ can be found in $O(n^2)$ time.*

Proof. To decide how to group jobs into batches, it can be easily done with the **Algorithm WB** which has the time complexity $O(n^2)$. Therefore, an optimal schedule of the job set $\mathcal{I}' \cup \mathcal{J}'$ can be found in $O(n^2)$ time and the lemma follows. □

Lemma 3.4. *The optimal makespan of the job set $\mathcal{I}' \cup \mathcal{J}'$ will be no greater than the optimal makespan of the original job set $\mathcal{I} \cup \mathcal{J}$, i.e., $C^*_{\max}(\mathcal{I}' \cup \mathcal{J}') \leq C^*_{\max}(\mathcal{I} \cup \mathcal{J})$.*

Proof. Consider the schedule $S^*(\mathcal{I} \cup \mathcal{J})$, i.e., the schedule of job set $\mathcal{I} \cup \mathcal{J}$ with optimal makespan or $C^*_{\max}(\mathcal{I} \cup \mathcal{J})$. For each pair of the stage-one (respectively, stage-two) operations of jobs $\{(I_k, I_{k+1})\,|k = 1, 2, \ldots, n_1 - 1\}$, we swap their positions if $p_{I_{k,1}} > p_{I_{k+1,1}}$ (respectively, $p_{I_{k,2}} < p_{I_{k+1,2}}$) and leave their stage-two (respectively, stage-one) operations unaltered in their original positions. In the similar way, we apply the operation-interchange technique to the operations of $\mathcal{J}$ on both stage-one and stage-two operations. Clearly, the makespan will not increase after we change the positions of the operations under the specific condition. Therefore, the derived schedule contains jobs of the job set $\mathcal{I}' \cup \mathcal{J}'$ and has a makespan that is no worse than the previous one in the same batch composition, i.e., $C_{\max}(S(\mathcal{I}' \cup \mathcal{J}')) \leq C^*_{\max}(\mathcal{I} \cup \mathcal{J})$. The optimal makespan of job set $\mathcal{I}' \cup \mathcal{J}'$ has the minimum makespan among all possible schedules of job set $\mathcal{I}' \cup \mathcal{J}'$ or $C^*_{\max}(\mathcal{I}' \cup \mathcal{J}') \leq C_{\max}(S(\mathcal{I}' \cup \mathcal{J}'))$. By transitivity, we can see that $C^*_{\max}(\mathcal{I}' \cup \mathcal{J}') \leq C^*_{\max}(\mathcal{I} \cup \mathcal{J})$ and complete the proof. □

In summary, by rearranging the operations in the two given job sets, we will get two data sets, $\mathcal{I}'$ and $\mathcal{J}'$, which constitute an ideal data set. With the two fixed sequences of jobs, the scheduling problem is now equivalent to $F(1,2)|\text{comp, fixed\_seq, s-batch}|C_{\max}$ which can be solved optimally by the algorithm in the previous section (by Theorem 3.1). Thus we get the minimum makespan of the derived problem, we also obtain a lower bound for the original problem.

Theorem 3.2. *A lower bound of $F(1,2)|comp,\ s\text{-}batch|C_{\max}$ can be found in $O(n^2)$.*

Proof. The process of constructing $\mathcal{I}'$ and $\mathcal{J}'$ takes $O(n \log n)$ time because of the sorting operations. Interleaving two sequences takes $O(n)$ time, and **Algorithm WB** takes $O(n^2)$ time to group jobs. Therefore, the overall time complexity for obtaining a lower bound of $F(1,2)|\text{comp, s-batch}|C_{\max}$ is $O(n^2)$. □

An example is given in the following for illustrating the lower bound calculation for the problem $F(1,2)|\text{comp, s-batch}|C_{\max}$. There are four jobs in two types to be scheduled: type 1, $\mathcal{I} = \{I_1, I_2\}$ and type 2, $\mathcal{J} = \{J_2, J_2\}$. The batch setup time is 1. The processing times of jobs are shown below.

Jobs	I_1	I_2	J_1	J_2
stage-one	$p_{I_{1,1}} = 2$	$p_{I_{2,1}} = 5$	$p_{J_{1,1}} = 3$	$p_{J_{2,1}} = 4$
stage-two	$p_{I_{1,2}} = 3$	$p_{I_{2,2}} = 4$	$p_{J_{1,2}} = 2$	$p_{J_{2,2}} = 6$

An ideal job set $\mathcal{I}' \cup \mathcal{J}'$ can be derived from the given job set $\mathcal{I} \cup \mathcal{J}$ by the operation interchange technique. The processing times of jobs in $\mathcal{I}' \cup \mathcal{J}'$ are shown below.

Jobs	I'_1	I'_2	J'_1	J'_2
stage-one	$p_{I'_{1,1}} = 2$	$p_{I'_{2,1}} = 5$	$p_{J'_{1,1}} = 3$	$p_{J'_{2,1}} = 4$
stage-two	$p_{I'_{1,2}} = 4$	$p_{I'_{2,2}} = 3$	$p_{J'_{1,2}} = 6$	$p_{J'_{2,2}} = 2$

Applying **Algorithm HL**, we can determine the sequence of jobs on machine M_0. Let $D = 0$, we can get the due dates of all jobs as shown below.

Jobs	J'_1	I'_1	I'_2	J'_2
stage-one	$p_{J'_{1,1}} = 3$	$p_{I'_{1,1}} = 2$	$p_{I'_{2,1}} = 5$	$p_{J'_{2,1}} = 4$
stage-two	$p_{J'_{1,2}} = 6$	$p_{I'_{1,2}} = 4$	$p_{I'_{2,2}} = 3$	$p_{J'_{2,2}} = 2$
d_k	$d_{J'_1} = 8$	$d_{I'_1} = 7$	$d_{I'_2} = 3$	$d_{J'_2} = 2$
D_k	$D_{J'_1} = -8$	$D_{I'_1} = -7$	$D_{I'_2} = -3$	$D_{J'_2} = -2$

The following is the recursive steps of **Algorithm WB**.

$$B(1) = \min \begin{cases} 1 + \sum_{k=1}^{1} p_k + \max\{8, B(2)\} \\ 1 + \sum_{k=1}^{2} p_k + \max\{8, B(3)\} \\ 1 + \sum_{k=1}^{3} p_k + \max\{8, B(4)\} \\ 1 + \sum_{k=1}^{4} p_k + \max\{8, B(5)\} \end{cases}$$

$$B(2) = \min \begin{cases} 1 + \sum_{k=2}^{2} p_k + \max\{7, B(3)\} \\ 1 + \sum_{k=2}^{3} p_k + \max\{7, B(4)\} \\ 1 + \sum_{k=2}^{4} p_k + \max\{7, B(5)\} \end{cases}$$

$$B(3) = \min \begin{cases} 1 + \sum_{k=3}^{3} p_k + \max\{3, B(4)\} \\ 1 + \sum_{k=3}^{4} p_k + \max\{3, B(5)\} \end{cases}$$

$$B(4) = \min \left\{ 1 + \sum_{k=4}^{4} p_k + \max\{2, B(5)\} \right.$$

$$B(5) = -\infty$$

$$B(4) = \min \left\{ 1 + 4 + \max\{2, -\infty\} \right\} = 7$$

$$B(3) = \min \left\{ \begin{array}{l} 1 + 5 + \max\{3, 7\} \\ 1 + 9 + \max\{3, -\infty\} \end{array} \right\} = 13$$

$$B(2) = \min \left\{ \begin{array}{l} 1 + 2 + \max\{7, 13\} \\ 1 + 7 + \max\{7, 7\} \\ 1 + 11 + \max\{7, -\infty\} \end{array} \right\} = 15$$

$$B(1) = \min \left\{ \begin{array}{l} 1 + 3 + \max\{8, 15\} \\ 1 + 5 + \max\{8, 13\} \\ 1 + 10 + \max\{8, 7\} \\ 1 + 14 + \max\{8, -\infty\} \end{array} \right\} = 19$$

Therefore, the lower bound of the optimal makespan of job set $\mathcal{I} \cup \mathcal{J}$ is 19, the corresponding schedule is $((J'_1), (I'_1, I'_2), (J'_2))$.

Chapter 4

Batching with Incompatible Items

While differentiation flowshop scheduling with compatible batches subject to given two job sequences was solved in the previous chapter, we investigate another batching policy in this chapter. The batching policy we concern here is incompatible batching which specify that a batch is allowed to contain only a single type of jobs. We will develop a dynamic programming algorithm to solve this problem recursively. We adopt the three-field notation $F(1,2)|$incp, fixed_seq, s-batch$|C_{\max}$ to denote the studied problem.

4.1 Preliminaries

To deal with the $F(1,2)|$incp, fixed_seq, s-batch$|C_{\max}$ problem, we also adopt the dynamic programming approach. A recursive program for constructing an optimal solution to an instance usually requires sufficient information on the optimal solutions in order to decompose the instance into smaller ones. An optimal solution to the incompatible case clearly specifies the makespan. However, the exact completion times on the two dedicated machines remain unknown. Such a difficulty may hinder the development of dynamic programming algorithms. In the following, we introduce an approach for retaining the information on the completion times on the two dedicated machines.

Define binary function $f(i, j, \alpha_1, \beta_1, \alpha_2, \beta_2, \gamma_1, \delta_1, \gamma_2, \delta_2, \varepsilon, \varepsilon') = 1$ if the last job on machine M_0 belongs to type-1, and there exists at least one schedule which satisfies the following conditions; otherwise, $f(\bullet) = 0$.

1. Job I_i is the last type-1 job scheduled on machine M_0.

2. Job J_j is the last type-2 job scheduled on machine M_0.

3. Batch formed from jobs $\{I_{\alpha_1}, I_{\alpha_1+1}, \ldots, I_{\beta_1}\}$ is the last type-1 batch that has no idle time prior to its dedicated operations on machine M_1.

4. Batch formed from jobs $\{J_{\alpha_2}, J_{\alpha_2+1}, \ldots, J_{\beta_2}\}$ is the last type-2 batch that has no idle time prior to its dedicated operations on machine M_2.

5. Batch formed from jobs $\{I_{\gamma_1}, I_{\gamma_1+1}, \ldots, I_{\delta_1}\}$ is the last type-1 batch on machine M_0.

6. Batch formed from jobs $\{J_{\gamma_2}, J_{\gamma_2+1}, \ldots, J_{\delta_2}\}$ is the last type-2 batch on machine M_0.

7. There are ε_1 (respectively, ε_2) batches before the critical batch of type-1 job (respectively, type-2), or ε_k batches before job I_k (or, J_k if it is a type-2 job).

8. There are ε' batches on machine M_0.

A binary function $g(i, j, \alpha_1, \beta_1, \alpha_2, \beta_2, \gamma_1, \delta_1, \gamma_2, \delta_2, \varepsilon, \varepsilon') = 1$ is similarly defined for type-2 batch formed from jobs $\{J_{\gamma_2}, J_{\gamma_2+1}, \ldots, J_{\delta_2}\}$ scheduled last on machine M_0.

If $f(i, j, \alpha_1, \beta_1, \alpha_2, \beta_2, \gamma_1, \delta_1, \gamma_2, \delta_2, \varepsilon, \varepsilon') = 1$ we can calculate the completion times of two stage-two machines in the actual corresponding schedule as follows:

Completion time on machine M_1 is

$$s \times \varepsilon_{\alpha_1} + \sum_{k=1}^{\beta_1} p_{I_{k,1}} + \sum_{k=\alpha_1}^{\delta_1} p_{I_{k,2}}, \text{ and}$$

$$s \times \varepsilon_{\alpha_2} + \sum_{k=1}^{\beta_2} p_{J_{k,1}} + \sum_{k=\alpha_2}^{\delta_2} p_{J_{k,2}}$$

on machine M_2. Therefore, the makespan of the corresponding schedule is

$$\max\left\{ s \times \varepsilon_1 + \sum_{k=1}^{\beta_1} p_{I_{k,1}} + \sum_{k=\alpha_1}^{\delta_1} p_{I_{k,2}}, s \times \varepsilon_2 + \sum_{k=1}^{\beta_2} p_{J_{k,1}} + \sum_{k=\alpha_2}^{\delta_2} p_{J_{k,2}} \right\}.$$

Figure 4.1 shows the configuration of state $f(i, j, \alpha_1, \beta_1, \alpha_2, \beta_2, \gamma_1, \delta_1, \gamma_2, \delta_2, \varepsilon, \varepsilon')$.

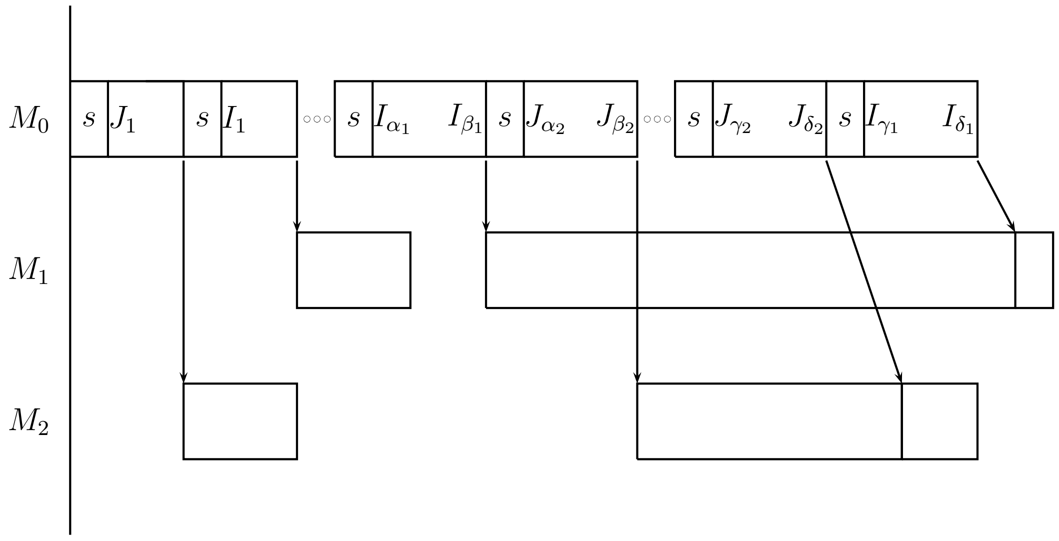

Figure 4.1: Gantt chart of one state of function f

The goal is to find some $f(n_1, n_2, *, *, *, *, *, *, *, *, *, *) = 1$ or $g(n_1, n_2, *, *, *, *, *, *, *, *, *, *) = 1$ whose schedule has the minimum makespan.

4.2 Recursions and Run Time Analysis

To determine whether entry $f(i, j, \alpha_1, \beta_1, \alpha_2, \beta_2, \gamma_1, \delta_1, \gamma_2, \delta_2, \varepsilon, \varepsilon') = 1$ or 0, we consider the following four cases.

1. $\alpha_1 \neq \gamma_1, \beta_1 \neq \delta_1, \gamma_1 \neq \delta_1$

   This case implies that the critical batch and the last batch on machine M_0 are the different batches, and the last batch contains more than one jobs.

   $f(i, j, \alpha_1, \beta_1, \alpha_2, \beta_2, \gamma_1, \delta_1, \gamma_2, \delta_2, \varepsilon, \varepsilon') = 1$ if $f(i-1, j, \alpha_1, \beta_1, \alpha_2, \beta_2, \gamma_1, \delta_1-1, \gamma_2, \delta_2, \varepsilon, \varepsilon')$ $= 1$.

2. $\alpha_1 = \gamma_1$,$\beta_1 = \delta_1$,$\gamma_1 = \delta_1$

   This case specifies that the critical batch is also the last batch on machine M_0 and contains a single job I_{δ_1}.

   Case 2.1: The job before I_{δ_1} is a type-1 job.

   $f(i, j, \alpha_1, \beta_1, \alpha_2, \beta_2, \gamma_1, \delta_1, \gamma_2, \delta_2, \varepsilon, \varepsilon') = 1$ if $f(i-1, j, m_1, n_1, \alpha_2, \beta_2, q_1, \delta_1-1, \gamma_2, \delta_2, \varepsilon_{m_1},$

$\varepsilon' - 1) = 1$ and the following inequality

$$s \times \varepsilon_{m_1} + \sum_{k=1}^{n_1} p_{I_{k,1}} + \sum_{k=1}^{a} p_{J_{k,1}} + \sum_{k=m_1}^{\delta_1 - 1} p_{I_{k,2}} \leq s \times \varepsilon' + \sum_{k=1}^{\delta_1} p_{I_{k,1}} + \sum_{k=1}^{b} p_{J_{k,1}}$$

holds for some a and b.

a: the number of type-2 jobs before job I_{n_1}.

b: the number of type-2 jobs before job I_{q_1}.

In this situation, we have to find the critical batch in the schedule resulted from deleting the last batch, which is the current critical batch. Since the detail information on critical batch is crucial to recursion formulation and makespan computation, it is necessary to find the critical batch before any further computation. The information like makespan can be calculated only when the critical batch is known. We give a recurrence function to examine if one batch which is formed from jobs $\{I_{m_1}, I_{m_1+1}, \ldots, I_{n_1}\}$, $\{m_1, n_1\} \in D_P$, is a candidate to be the critical batch or not. A batch is a candidate means that it is critical for the successor jobs, but it can be an idle or non-idle batch for the preceding jobs. A critical batch must be a candidate and cannot induce any idle time. We define the domain D_P as the feasible range of m_1 and n_1, so
$D_P = \{(m_1, n_1)\,|1 \leq m_1 \leq n_1, m_1 \leq n_1 \leq q_1 - 1, n_1 < q_1 \leq \delta_1 - 1\}$. The recurrence function is given as follows.

Recursion:

$$P_f(x,y,v,t)=\begin{cases} P_f(y+1,u,v+1,t+\sum_{k=x}^{y}p_{I_{k,2}}), & \text{if } u\neq q_1-1 \\ \quad y+1\leq u\leq q_1-1, & \text{and } \sum_{k=u}^{m_1}p_{I_{k,1}}+s\times(v-1) \\ & \leq t+\sum_{k=x}^{y}P_{I_{k,2}} \\ 1, & \text{if } u=q_1-1 \\ & \text{and } \sum_{k=u}^{m_1}p_{I_{k,1}}+s\times(v-1) \\ & \leq t+\sum_{k=x}^{y}p_{I_{k,2}} \\ 0, & \text{otherwise} \end{cases}$$

Goal: Find an instance of m_1 and n_1 in D_P such that $P_f(m_1,n_1,1,s+\sum_{k=n_1}^{m_1}p_{I_{k,1}})=1$.

A recurrence relation P_g is similarly defined for the scenario in which the last job on machine M_0 is a type-2 job.

If a candidate of critical batch which is formed from jobs $\{I_{m_1},I_{m_1+1},\ldots,I_{n_1}\}$ can be obtained from the recurrence function, then the next step is to determine if this candidate is a real critical batch or not. The inspection process can be done through a function similar to P_f. The inspection function will examine if the candidate is an idle or non-idle batch. If a candidate is a non-idle batch, then this candidate is a critical batch.

If such a critical batch exists, we can get it from function P_f. Then, we can go to the next step to examine if function f is 1 or 0 in this case. We denote the feasible range of a and b as the domain D_1, so $D_1=\{(a,b)\,|0\leq a\leq j, a\leq b\leq j\}$.

$$H_1^f(a,b)=\begin{cases} 1, & \text{if } s\times\varepsilon_{m_1}+\sum_{k=1}^{n_1}p_{I_{k,1}}+\sum_{k=1}^{a}p_{J_{k,1}}+\sum_{k=m_1}^{\delta_1-1}p_{I_{k,2}} \\ & -s\times\varepsilon'+\sum_{k=1}^{\delta_1}p_{I_{k,1}}+\sum_{k=1}^{b}p_{J_{k,1}}\leq 0 \\ 0, & \text{otherwise} \end{cases}$$

$$f(i,j,\alpha_1,\beta_1,\alpha_2,\beta_2,\gamma_1,\delta_1,\gamma_2,\delta_2,\varepsilon,\varepsilon') = \begin{cases} 1, & \text{if } f(i-1,j,m_1,n_1,\alpha_2,\beta_2,q_1, \\ & \delta_1-1,\gamma_2,\delta_2,\varepsilon_{m_1},\varepsilon'-1) = 1 \\ & \text{and } \bigvee_{D_1} H_1^f(a,b) = 1 \\ 0, & \text{otherwise} \end{cases}$$

Case 2.2: the job before I_{δ_1} is a type-2 job.

$f(i,j,\alpha_1,\beta_1,\alpha_2,\beta_2,\gamma_1,\delta_1,\gamma_2,\delta_2,\varepsilon,\varepsilon') = 1$ if $g(i-1,j,m_1,n_1,\alpha_2,\beta_2,q_1,\delta_1-1,q_2,r_2,\varepsilon_{m_1},\varepsilon'-1) = 1$ and the following inequality

$$s \times \varepsilon_{m_1} + \sum_{k=1}^{n_1} p_{I_{k,1}} + \sum_{k=1}^{a} p_{J_{k,1}} + \sum_{k=m_1}^{\delta_1-1} p_{I_{k,2}} \leq s \times \varepsilon' + \sum_{k=1}^{\delta_1} p_{I_{k,1}} + \sum_{k=1}^{b} p_{J_{k,1}}$$

holds for some a and b.

a: the number of type-2 jobs before job I_{n_1}.

b: the number of type-2 jobs before job I_{q_1}.

Domain D_2 is defined as the feasible range of a and b, so $D_2 = \{(a,b)\,|0 \leq a \leq b, a \leq b \leq j\}$.

$$H_2^f(a,b) = \begin{cases} 1, & \text{if } s \times \varepsilon_{m_1} + \sum_{k=1}^{n_1} p_{I_{k,1}} + \sum_{k=1}^{a} p_{J_{k,1}} + \sum_{k=m_1}^{\delta_1-1} p_{I_{k,2}} \\ & -s \times \varepsilon' + \sum_{k=1}^{\delta_1} p_{I_{k,1}} + \sum_{k=1}^{b} p_{J_{k,1}} \leq 0 \\ 0, & \text{otherwise} \end{cases}$$

$$f(i,j,\alpha_1,\beta_1,\alpha_2,\beta_2,\gamma_1,\delta_1,\gamma_2,\delta_2,\varepsilon,\varepsilon') = \begin{cases} 1, & \text{if } g(i-1,j,m_1,n_1,\alpha_2,\beta_2,q_1, \\ & \delta_1-1,q_2,r_2,\varepsilon_{m_1},\varepsilon'-1) = 1 \\ & \text{and } \bigvee_{D_2} H_2^f(a,b) = 1 \\ 0, & \text{otherwise} \end{cases}$$

3. $\alpha_1 = \gamma_1, \beta_1 = \delta_1, \gamma_1 \neq \delta_1$

   This case specifies that the critical batch is also the last batch on machine M_0 and contains more than one job.

   $f(i, j, \alpha_1, \beta_1, \alpha_2, \beta_2, \gamma_1, \delta_1, \gamma_2, \delta_2, \varepsilon, \varepsilon') = 1$ if $f(i-1, j, m_1, n_1, \alpha_2, \beta_2, \gamma_1, \delta_1-1, \gamma_2, \delta_2, \varepsilon, \varepsilon')$ $= 1$ and the inequality

$$s \times \varepsilon_{m_1} + \sum_{k=1}^{n_1} p_{I_{k,1}} + \sum_{k=1}^{a} p_{J_{k,1}} + \sum_{k=m_1}^{\delta_1-1} p_{I_{k,2}} \leq s \times \varepsilon' + \sum_{k=1}^{\delta_1} p_{I_{k,1}} + \sum_{k=1}^{b} p_{J_{k,1}}$$

   holds for some a and b.

   a: the number of type-2 jobs before job I_{n_1}.

   b: the number of type-2 jobs before job I_{δ_1}.

   Domain D_3 is defined as the feasible range of a and b, so
   $D_3 = \{a, b | 0 \leq a \leq j, a \leq b \leq j\}$.

$$H_3^f(a,b) = \begin{cases} 1, & \text{if } s \times \varepsilon_{m_1} + \sum_{k=1}^{n_1} p_{I_{k,1}} + \sum_{k=1}^{a} p_{J_{k,1}} + \sum_{k=m_1}^{\delta_1-1} p_{I_{k,2}} \\ & -s \times \varepsilon' + \sum_{k=1}^{\delta_1} p_{I_{k,1}} + \sum_{k=1}^{b} p_{J_{k,1}} \leq 0 \\ 0, & \text{otherwise} \end{cases}$$

$$f(i, j, \alpha_1, \beta_1, \alpha_2, \beta_2, \gamma_1, \delta_1, \gamma_2, \delta_2, \varepsilon, \varepsilon') = \begin{cases} 1, & \text{if } f(i-1, j, m_1, n_1, \alpha_2, \beta_2, \\ & \gamma_1, \delta_1 - 1, \gamma_2, \delta_2, \varepsilon, \varepsilon') = 1 \\ & \text{and } \bigvee_{D_3} H_3^f(a,b) = 1 \\ 0, & \text{otherwise} \end{cases}$$

4. $\alpha_1 \neq \gamma_1, \beta_1 \neq \delta_1, \gamma_1 = \delta_1$

   This case implies that the critical batch and the last batch on machine M_0 are different, and the last batch contains only one job.

   Case 4.1: the job before I_{δ_1} is a type-1 job.

$f(i,j,\alpha_1,\beta_1,\alpha_2,\beta_2,\gamma_1,\delta_1,\gamma_2,\delta_2,\varepsilon,\varepsilon') = 1$ if $f(i-1,j,\alpha_1,\beta_1,\alpha_2,\beta_2,m_1,\delta_1-1,\gamma_2,\delta_2,\varepsilon,$ $\varepsilon'-1) = 1$ and the inequality

$$s \times \varepsilon_{\alpha_1} + \sum_{k=1}^{\beta_1} p_{I_{k,1}} + \sum_{k=1}^{a} p_{J_{k,1}} + \sum_{k=\alpha_1}^{\delta_1-1} p_{I_{k,2}} > s \times \varepsilon' + \sum_{k=1}^{\delta_1} p_{I_{k,1}} + \sum_{k=1}^{b} p_{J_{k,1}}$$

holds for some a and b.

a: the number of type-2 jobs before job I_{α_1}.

b: the number of type-2 jobs before job I_{δ_1}.

Domain D_4 is defined as the feasible range of a and b, so

$D_4 = \{(a,b)\,|0 \le a \le j, a \le b \le j\}$.

$$H_4^f(a,b) = \begin{cases} 1, & \text{if } s \times \varepsilon_{\alpha_1} + \sum_{k=1}^{\beta_1} p_{I_{k,1}} + \sum_{k=1}^{a} p_{J_{k,1}} + \sum_{k=\alpha_1}^{\delta_1-1} p_{I_{k,2}} \\ & -s \times \varepsilon' + \sum_{k=1}^{\delta_1} p_{I_{k,1}} + \sum_{k=1}^{b} p_{J_{k,1}} > 0 \\ 0, & \text{otherwise} \end{cases}$$

$$f(i,j,\alpha_1,\beta_1,\alpha_2,\beta_2,\gamma_1,\delta_1,\gamma_2,\delta_2,\varepsilon,\varepsilon') = \begin{cases} 1, & \text{if } f(i-1,j,\alpha_1,\beta_1,\alpha_2,\beta_2, \\ & m_1,\delta_1-1,\gamma_2,\delta_2,\varepsilon,\varepsilon'-1) = 1 \\ & \text{and } \bigvee_{D_4} H_4^f(a,b) = 1 \\ 0, & \text{otherwise} \end{cases}$$

Case 4.2: the job before I_{δ_1} is a type-2 job.

$f(i,j,\alpha_1,\beta_1,\alpha_2,\beta_2,\gamma_1,\delta_1,\gamma_2,\delta_2,\varepsilon,\varepsilon') = 1$ and $g(i-1,j,\alpha_1,\beta_1,\alpha_2,\beta_2,m_1,\delta_1-1,\gamma_2,\delta_2,\varepsilon,$ $\varepsilon'-1) = 1$ and the inequality

$$s \times \varepsilon_{\alpha_1} + \sum_{k=1}^{\beta_1} p_{I_{k,1}} + \sum_{k=1}^{a} p_{J_{k,1}} + \sum_{k=\alpha_1}^{\delta_1-1} p_{I_{k,2}} > s \times \varepsilon' + \sum_{k=1}^{\delta_1} p_{I_{k,1}} + \sum_{k=1}^{b} p_{J_{k,1}}$$

holds for some a and b.

a: the number of type-2 jobs before job I_{α_1}.

b: the number of type-2 jobs before job I_{δ_1}.

Domain D_5 is defined as the feasible range of a and b, so

$D_5 = \{a, b | 0 \le a \le h, a \le b \le h\}$.

$$H_5^f(a,b) = \begin{cases} 1, & \text{if } s \times \varepsilon_{\alpha_1} + \sum_{k=1}^{\beta_1} p_{I_{k,1}} + \sum_{k=1}^{a} p_{J_{k,1}} + \sum_{k=\alpha_1}^{\delta_1 - 1} p_{I_{k,2}} \\ & -s \times \varepsilon' + \sum_{k=1}^{\delta_1} p_{I_{k,1}} + \sum_{k=1}^{b} p_{J_{k,1}} > 0 \\ 0, & \text{otherwise} \end{cases}$$

$$f(i,j,\alpha_1,\beta_1,\alpha_2,\beta_2,\gamma_1,\delta_1,\gamma_2,\delta_2,\varepsilon,\varepsilon') = \begin{cases} 1, & \text{if } g(i-1,j,\alpha_1,\beta_1,\alpha_2,\beta_2, \\ & m_1, \delta_1 - 1, \gamma_2, \delta_2, \varepsilon, \varepsilon' - 1) = 1 \\ & \text{and } \bigvee_{D_5} H_5^f(a,b) = 1 \\ 0, & \text{otherwise} \end{cases}$$

Next we analyze the time complexity of the recursive program. There are $O(n_1^5 n_2^4 n^2)$ entries in function f and $O(n_1^4 n_2^5 n^2)$ entries in function g. We first note that case 2 has a dominant time complexity to other cases. In case 2.1, processing function P_f requires a time complexity of $O(\max\{n_1^2 n, n_2^2 n\})$ and function H_1^f is $O(n^2)$. Case 2.2 has the same time complexity with case 2.1. Therefore, the time complexity of case 2 is $O(\max\{\max\{n_1^2 n, n_2^2 n\}, n^2\})$. In summary, the overall time complexity is $O(n_1^4 n_2^4 n^3 (n_1 + n_2) \times (\max\{\max\{n_1^2, n_2^2\}, n\}))$. Our discussion is concluded in the following theorem.

Theorem 4.1. *The $F(1,2)|$incp, fixed_seq, s-batch$|C_{\max}$ problem can be solved in $O(n_1^4 n_2^4 n^3 (n_1 + n_2) \times (\max\{\max\{n_1^2, n_2^2\}, n\}))$ time, which is polynomial in terms of the input length.* □

Before closing this chapter, it could be noticed that the development of lower bound by the dynamic programming algorithm addressed in the previous chapter suggests a possible adoption of the current dynamic programming algorithm to derive a lower bound for the general problem subject to incompatible batching without the assumption of fixed job sequences. More precisely, by adopting the same concept of ideal data set to problem $F(1,2)|$incp, s-batch$|C_{\max}$, we can get an ideal data set which is derived from the data set of the original problem. When the ideal data set is available, two fixed job sequences, representing optimal sequencing of the job of each type, readily follow. Since we have two fixed job sequences, the problem is now equivalent to the $F(1,2)|\text{incp, fixed\_seq, s-batch}|C_{\max}$ problem. Therefore, both an optimal makespan of the problem with fixed sequences or the derivation of a lower bound of the original problem can be computed by the above dynamic programming algorithm in polynomial time. The following corollary directly follows from Theorem 4.1.

Corollary 4.1. *A lower bound of $F(1,2)|incp, s\text{-}batch|C_{\max}$ can be computed in $O(n_1^4 n_2^4 n^3 (n_1+n_2) \times (\max\{\max\{n_1^2, n_2^2\}, n\}))$ time.* $\square$

Chapter 5

Concluding Remarks

In this thesis, we investigated a makespan minimization scheduling problem in a three-machine two-stage flowshop, known as differentiation flowshop which has a critical machine at stage one and two independent dedicated machines at stage two. The first-stage machine process the jobs in batches subject to the assumption of continuous process and batch availability.

For the batch scheduling problem with compatible items, it is known that the problem is NP-hard in the strong sense even if the batch setup time is 0. While the problem is computationally intractable, we proposed an algorithm to derive a lower bound of the problem in $O(n^2)$ time. The lower bound can be used for the design of branch and bound algorithms or the evaluation of heuristic approaches. Moreover, we investigated one special case that is polynomially solvable. With two given fixed sequences of two types of jobs, a polynomial time algorithm is devised to produce the optimal solutions.

On the other hand, for the batch problem with incompatible items under the assumption of fixed sequences, we developed an $O(n_1^4 n_2^4 n^3 (n_1 + n_2) \times (\max\{\max\{n_1^2, n_2^2\}, n\}))$ dynamic programming algorithm for solving the problem in a recursive way. Although the recursive program can solve this problem in polynomial time, its time complexity may be unaffordable for larger instances. The development of faster algorithms is clearly required for practical significance.

For further research, it may be interesting to generalize the studied problem to the setting with a variable number of parallel dedicated machines, i.e., the number m of parallel dedicated machines is part of the input. Even though this problem is clearly NP-hard in the strong sense, it may be interesting to determine the complexity status of $F(1, m)$ model in such special cases as (1) all jobs have the same processing time on the critical machine, or (2) sequences of different types of jobs are fixed and given. Developing lower bounds and dominance properties for designing branch-and-bound algorithms can be another worthy direction.

Bibliography

[1] Y. Aneja and N. Singh, "Scheduling production of common components at a single facility," *IIE Transactions*, vol. 22, pp. 234–237, 1990.

[2] K. Baker, "Scheduling the production of components at a common facility," *IIE Transactions*, vol. 20, pp. 32–35, 1988.

[3] T. Cheng, J. Gupta, and G. Wang, "A review of flowshop scheduling research with setup times, production and operations management," *Production and Operations Management*, vol. 9, pp. 262–282, 2000.

[4] T. Cheng, B. Lin, and Y. Tian, "Minimizing the weighted sum of machine completion times in a two-stage flow shop with a critical machine," *Computers & Operations Research*, vol. 36, pp. 3031–3040, 2009.

[5] T. Cheng, B. Lin, and A. Toker, "Flowshop batching and scheduling to minimize the makespan," *Naval Research Logistics*, vol. 47, no. 2, pp. 128–144, 2000.

[6] T. Cheng and Q. Wang, "Scheduling the fabrication and assembly of components in a two-machine flowshop," *IIE Transactions*, vol. 31, pp. 135–148, 1999.

[7] R. Dudek, S. Panwalkar, and M. Smith, "The lessons of flowshop scheduling research," *Operations Research*, vol. 40, pp. 7–13, 1992.

[8] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco, CA: Freedman, 1979.

[9] A. Gerodimos, C. Glass, and C. Potts, "Scheduling the production of two-component jobs on a single machine," *European Journal of Operational Research*, vol. 120, pp. 250–259, 2000.

[10] J. Herrmann, *An Investigation of Production Scheduling Problems Motivated by Semiconductor Manufacturing*. PhD thesis, University of Florida, Gainesville, Florida, 1993.

[11] G. Kyparisis and C. Koulamas, "Flow shop and open shop scheduling with a critical machine and two operations per job," *European Journal of Operational Research*, vol. 127, no. 1, pp. 120–125, 2000.

[12] B. Lin, "The strong np-hardness of two-stage flowshop scheduling problem with a common second-stage machine," *Computers & Operations Research*, vol. 27, no. 6, pp. 695–698, 1999.

[13] B. Lin, "Fabrication scheduling in a two-machine flowshop with due date constraints," *European Journal of Operational Research*, vol. 136, no. 1, pp. 95–105, 2002.

[14] B. Lin and T. Cheng, "Batch scheduling in a no-wait two-machine flowshop," *Computers & Operations Research*, vol. 28, no. 7, pp. 613–624, 2001.

[15] B. Lin and T. Cheng, "Two-machine flowshop batching and scheduling," *Annals of Operations Research*, vol. 133, pp. 149–161, 2005.

[16] B. Lin, T. Cheng, and A. Chou, "Scheduling in an assembly-type production chain with batch transfer," *Scheduling in an assembly-type production chain with batch transfer*, vol. 35, pp. 143–151, 2007.

[17] G. Mosheiov and U. Yovel, "Comments on flow shop and open shop scheduling with a critical machine and two operations per job," *European Journal of Operational Research*, vol. 157, no. 1, pp. 257–261, 2004.

[18] Y. Neumytov and S. Sevastyanov, "Approximation algorithm with tight bound for three-machine counter-routes problem," *Upravlyaemye Sistemy*, vol. 31, pp. 53–65, 1993.

[19] C. Oğuz, B. Lin, and T. Cheng, "Two-stage flowshop scheduling problem with a common second-stage machine," *Computers & Operations Research*, vol. 24, no. 12, pp. 1169–1174, 1997.

[20] C. Potts and M. Kovalyov, "Scheduling with batching: A review," *European Journal of Operational Research*, vol. 102, pp. 228–249, 2000.

[21] A. Reisman, A. Kumar, and J. Motwani, "Flowshop scheduling/sequencing research: A statistical review of the literature, 1952-1994," *IEEE Transactions on Engineering Management*, vol. 44, pp. 316–329, 1997.

[22] R. Vickson, M. Magazine, and C. Santos, "Batching and sequencing of components at a single facility," *IIE Transactions*, vol. 25, pp. 65–70, 1993.

[23] M. Pinedo, *Scheduling: Theory, Algorithms and Systems.* Englewood Cliffs, NJ: Prentice Hall, 2002.

[24] M. Pinedo and X. Chao, *Operations Scheduling with Applications in Manufacturing and Services.* Boston, MA: McGraw-Hill, 1999.

[25] D. Gupta and S. Benjaafar, "Make-to-order, make-to-stock, or delay product differentiation? a common framework for modeling and analysis," *IIE Transactions*, vol. 36, pp. 529–546, 2004.

[26] G. Eppen, "Effects of centralization on expected costs in multi-location newsboy problems," *Management Science*, vol. 25, pp. 498–501, 1979.

[27] E. Feitzinger and H. Lee, "Mass customization at hewlett packard: the power of postponement," *Harvard Business Review*, vol. 75, pp. 116–121, 1997.

[28] L. Bruce, "The bright new worlds of benetton," *Bruce, L. (1987International Management*, vol. 42, pp. 24–35, 1987.

[29] J. Magretta, “The power of virtual integration: An interview with dell computer’s michael dell,” *Harvard Business Review*, vol. 76, pp. 72–84, 1998.

[30] S. Johnson, “Optimal two and three-stage production schedules with setup times included,” *Naval Research Logistics Quarterly*, vol. 1, pp. 61–68, 1954.

[31] D. IG and S. VA, “Heuristics for the two-stage job shop scheduling problem with a bottleneck machine.,” *European Journal of Operational Research*, vol. 123, pp. 229–240, 2000.