# SERVICE TIME OPTIMIZATION OF FLOW SHOP SYSTEMS

A DISSERTATION SUBMITTED TO
THE DEPARTMENT OF INDUSTRIAL ENGINEERING
AND THE INSTITUTE OF ENGINEERING AND SCIENCE
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

By
Ömer Selvi
November, 2008

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of doctor of philosophy.

Asst. Prof. Dr. Kağan Gökbayrak (Supervisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of doctor of philosophy.

Prof. Dr. M. Selim Aktürk

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of doctor of philosophy.

Prof. Dr. Erdal Erel

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of doctor of philosophy.

Prof. Dr. Ömer Kırca

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of doctor of philosophy.

Assoc. Prof. Dr. Emre Alper Yıldırım

Approved for the Institute of Engineering and Science:

Prof. Dr. Mehmet B. Baray
Director of the Institute

# ABSTRACT

## SERVICE TIME OPTIMIZATION OF FLOW SHOP SYSTEMS

Ömer Selvi
Ph.D. in Industrial Engineering
Supervisor: Asst. Prof. Dr. Kağan Gökbayrak
November, 2008

One of the key questions that engineers face in flow shop systems is the service time control, i.e., how long jobs should be processed at each machine. This is an important question because processing times can have great impacts on the cost efficiency of the flow shop systems. In order to meet job completion deadlines and to decrease inventory costs, one may set the service times as small as possible; however, this usually comes at the expense of reduced tool life increasing service costs. In this thesis, we study the flow shop systems under such trade-offs. We consider the service time optimization of deterministic flow shop systems processing identical jobs that arrive at the system at known times and are processed in the order they arrive within deadlines. The cost function to be minimized consists of service costs at machines and regular completion-time costs of jobs. The decision variables are the service times that are controllable within constraints.

We first consider the fixed service time flow shop systems formed of initially controllable machines, where the service times are set only once at the start up time and cannot be altered between processes, and uncontrollable machines, where the service times are fixed and known in advance. For such systems, we formulate a non-convex and non-differentiable optimization problem with a standard solution procedure based on the linearization of the constraints allowing for a convex optimization problem with high memory requirements. Regardless of the cost function, we present a set of waiting and completion time characteristics in such flow shop systems and employ them to derive a simpler equivalent convex optimization problem which improves solution times and alleviates the memory requirements enabling solutions for larger systems. However, the resulting simplified convex optimization problem still needs the use of a convex optimization solver which may not be available at some of the manufacturing companies. To

overcome such need, we introduce another equivalent convex optimization problem along with its subgradient algorithm yielding substantial improvements in solution times and solvable system sizes. We also consider a specific nonlinear decreasing service cost structure allowing us to introduce a new search algorithm much faster than the subgradient solution algorithm.

Building on the results for fixed service time flow shop systems, we also consider the mixed line flow shop systems formed of fully controllable machines, where the service times are adjustable for each process, initially controllable machines, and uncontrollable machines. Similarly, we formulate a non-convex and non-differentiable optimization problem for such systems and, as a standard way of solving the formulated problem, we apply the method of linearization on the constraints to present a convex optimization problem with high memory requirements. Then, we present a set of optimal waiting characteristics in such flow shop systems and employ them to derive simpler equivalent convex optimization problems. A "forward in time" algorithm is also proposed to decompose the resulting simplified equivalent convex optimization problem into smaller convex optimization problems for the flow shop systems formed of only fully controllable and uncontrollable machines. The computational results demonstrate that the simplifications and the decomposition not only improve the solution times considerably but also allow us to solve larger problems by alleviating memory constraints.

*Keywords:* Deterministic flow shop systems, Optimal control, Controllable service times, Controllable/Uncontrollable machines, Convex programming, Subgradient algorithm.

# ÖZET

# AKIŞ TİPİ İŞLİK SİSTEMLERDE İŞLEM SÜRELERİ ENİYİLEMESİ

Ömer Selvi
Endüstri Mühendisliği, Doktora
Tez Yöneticisi: Asst. Prof. Dr. Kağan Gökbayrak
Kasım, 2008

Mühendislerin akış tipi işlik sistemlerde cevaplaması gereken en kilit sorulardan birisi işlem sürelerinin nasıl denetleneceğidir yani işlerin her makinede ne kadar süre işlem görmesi gerektiğidir. Bu önemli bir sorudur çünkü işlem sürelerinin akış tipi işlik sistemlerin maliyet verimliliği üzerinde çok büyük etkileri olabilir. İşleri son bitim zamanına kadar tamamlamak ve envanter maliyetlerini düşürmek için işlem süreleri mümkün olduğunca küçük tutulabilir, fakat bu yaklaşım genellikle işlem maliyetlerini yükselten kısaltılmış takım ömürlerinden doğan masrafları beraberinde getirir. Biz bu tezde akış tipi işlik sistemlerde bu tip ilişkiler üzerine çalıştık. Bilinen zamanlarda gelen işleri geldikleri sırayla işleyen belirlenimci akış tipi işlik sistemlerde işlem süreleri eniyilemesi problemini ele aldık. Enküçültülecek maliyet fonksiyonunu makinelerdeki işlem maliyetlerinden ve kurallı iş bitim zamanı maliyetlerinden oluşturduk. Bir kısıt dahilinde denetlenebilir işlem sürelerini karar değişkenleri olarak belirledik.

Öncelikle, başlangıçta denetlenebilir, yani işlem süreleri sistemin çalışmaya başlama anında belirlenen ve işlemler arasında bir daha değiştirilemeyen, ve denetlenemez, yani işlem süreleri sabit olan ve önceden bilinen, makinelerden oluşan sabit işlem süreli akış tipi işlik sistemleri ele aldık. Bu tip sistemler için standart çözüm yöntemi yüksek bellek gereksinimli bir dışbükey eniyileme problemine olanak sağlayan kısıtların doğrusallaştırılması metoduna dayanan dışbükey olmayan ve türevlenemeyen bir eniyileme problemi oluşturduk. Maliyet fonksiyonundan bağımsız olarak, bu tip akış tipi işlik sistemler için bir dizi bekleme ve iş bitim zamanı özellikleri gösterdik ve bu özellikleri kullanarak çözüm sürelerini geliştiren ve daha büyük sistemlerin çözülmesine olanak sağlayacak şekilde bellek gereksinimini azaltan daha basit ve denk bir dışbükey eniyileme problemi çıkardık. Ne var ki sonuçta ortaya çıkan basitleştirilmiş dışbükey eniyileme problemi hala

bazı imalatçı şirketlerin tedarik edemeyeceği dışbükey eniyileme çözücüsü kullanımına ihtiyaç duymaktadır. Bu ihtiyacı gidermek için çözüm sürelerinde ve çözülebilir sistem boyutlarında oldukça ciddi iyileştirme sağlayan altgradyan algoritması eşliğinde bir başka denk dışbükey eniyileme problemi önerdik. Ayrıca, altgradyan algoritmasından çok daha hızlı çalışan yeni bir tarama algoritması geliştirmemize olanak sağlayan doğrusal olmayan ve azalan özel bir işlem maliyet yapısını da çözümledik.

Sabit işlem süreli akış tipi işlik sistemler için geçerli sonuçların üzerine inşa etmek suretiyle, bu tezde ayrıca tamamen denetlenebilir, yani işlem süreleri her işlem için ayrı ayrı ayarlanabilen, başlangıçta denetlenebilir ve denetlenemez makinelerden oluşan akış tipi işlik sistemleri de ele aldık. Benzer şekilde, bu tip sistemler için dışbükey olmayan ve türevlenemeyen bir eniyileme problemi oluşturduk ve oluşturduğumuz bu probleme standart çözüm yöntemi olarak, kısıt doğrusallaştırma metodu uygulamak suretiyle yüksek bellek gereksinimli bir dışbükey eniyileme problemi ortaya koyduk. Daha sonra, bu tip akış tipi işlik sistemler için bir dizi en iyi bekleme özellikleri gösterdik ve bu özellikleri kullanarak daha basit ve denk bir dışbükey eniyileme problemi çıkardık. Sadece tamamen denetlenebilir ve denetlenemez makinelerden oluşan akış tipi işlik sistemler için, sonuçta ortaya çıkan basitleştirilmiş dışbükey eniyileme problemini daha küçük dışbükey eniyileme problemlerine ayrıştıran "zamanda ilerleyen" bir algoritma da önerdik. Deneysel hesaplamalarımız gösterdi ki basitleştirmeler ve ayrıştırma sadece çözüm sürelerini geliştirmekle kalmadı aynı zamanda bellek gereksinimini azaltmak suretiyle daha büyük sistemleri çözmemize olanak sağladı.

*Anahtar sözcükler*: Belirlenimci akış tipi işlik sistemler, En iyi denetleme, Denetlenebilir işlem süreleri, Denetlenebilir/denetlenemez makineler, Dışbükey programlama, Altgradyan algoritması.

*Dedicated*

*to*

*my family*

# Acknowledgement

I would like to sincerely thank my advisor Asst. Prof. Dr. Kağan Gökbayrak for his valuable and perpetual guidance and encouragement throughout this study. His supervising with patience and interest made this thesis possible.

I gratefully acknowledge all the members of my committee who have given their time to read this manuscript and offered valuable advice.

My special thanks go to my family for their encouragement and sacrifice. This study is dedicated to them without whom it would not have been possible.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Flow shop systems have been an important area of research ever since 1950's. Since flow shop systems can often be found throughout many industries, i.e., manufacturing (e.g. automobile) industry, many researchers have recognized the importance of the subject and contributed to it. One of the key questions that engineers face in flow shop systems is the service time control, i.e., how long jobs should be processed at each machine. This is an important question because service times can have great impacts on the cost efficiency of flow shop systems.

Taylor's tool wear equation [24] states that tool life decreases rapidly with an increase in machining speed. Increased machining speeds may also result in more frequent tool breakages. Hence, increasing machining speeds rapidly increases the frequency of tool changing which implies increased tooling costs. Moreover, tooling has also direct cost implications. Industry data suggests that tooling, as one of the major component, accounts for 25% to 30% of production cost in an automated machining environment (see [26]). In flexible manufacturing systems (FMSs), the initial investment in cutting tools and fixtures may reach up to 25% of the total FMS investment (see [46]). Seven to ten times more money is spent on tools, jigs, fixtures, and consumables than on capital equipment during the useful life of the machines (see [3]). The impact of tooling problems on system management should also not be underestimated: 40% to 60% of a foreman's time is spent expediting tools and materials, 15% to 20% of scheduled production time

is missed due to unavailable tooling.

It is evident from the above discussion that, to decrease tooling costs, it is preferred to decrease machining speeds and, therefore, to increase service times as much as possible. However, longer service times mean longer flow times (time required to move a job through the system, from entry on the first machine to completion on the last machine) of jobs. Increased flow times require the anticipation of customers' needs in terms of product variations, options, and extra finished goods as well as work-in-process (WIP) inventories due to longer deliveries. Capital invested to inventories as long as they remain in the system provides no profit. Inventory quality also decreases as the unfinished items spend more time in the system because they are vulnerable to damages. Moreover, the ability to adapt the production structure according to the fast changing global market and to respond customer needs relies on shorter flow times and lower work-in-process inventories. For instance, a manufacturing system with shorter flow times eliminates the necessity of further finished products and WIP inventories, provides greater freedom of choice to the customer, and, therefore, becomes more competitive against any alternative system with longer delivery times. Many other advantages such as correcting quality problems and implementing engineering changes accompany to these due to faster responsiveness. Hence, to decrease the cost related to inventory held and customer satisfaction, which is hard to be quantified, service times are tried to be reduced as much as possible.

In this thesis, we study the flow shop systems under such trade-offs. We consider the service time optimization of deterministic flow shop systems processing identical jobs that arrive at the system at known times and are processed within their associated deadlines. The term "identical" used for jobs implies that the operational requirements of all jobs so as to change their physical characteristics according to certain specifications are the same at each machine so that each machine has its own specific service cost function applied to all jobs processed, i.e., homogeneous (same for each job but not for each machine) service cost functions are employed for jobs at machines. The flow shop system that we consider consists of *fully controllable* machines, where the service times are adjustable before each process, *initially controllable* machines, where the service times are set

only once at the start up time and could not be altered between processes, and *uncontrollable* machines, where the service times are fixed and known a priori. The existing CNC (Computer Numerical Control) machine technology allows us to change the service times very quickly by just changing few lines in the CNC programming code without incurring setup times and production errors. Hence, CNC machines are good examples of fully controllable machines. As opposed to CNC machines, some traditional (non-CNC) machines are manually controlled by human operators. During mass production, it may not be feasible to alter the service times of these machines because the setup times are idle times and the manual modifications are prone to errors. Therefore, the service times at these traditional machines are uncontrollable or initially controllable, i.e., they are set at the start-up time and are not altered afterwards. The cost objective we consider consists of service costs at fully and initially controllable machines, which are dependent on service times, and regular completion-time costs for jobs. Motivated by the above discussion, we assume that faster services increase service costs. Slower services, on the other hand, increase the regular completion-time costs and/or leading to the violation of constraints on job completion deadlines. This trade-off in setting the service times makes the problem nontrivial and we set our objective to determine the cost-minimizing service times.

The scheduling problems of the flow shops are known to be NP-hard even for fixed service times (see [39]). In these problems, the objective is to find the best sequence of jobs to be processed at machines. Except for two-machine systems with the objective of minimizing makespan, the scheduling literature for the flow shop systems is limited to heuristics and approximate solution methods. Introduction of controllable service times at machines further complicates the problem. A survey of results on the controllable service times in scheduling problems can be found in [36], [19], and [44]. In this thesis, searching for efficient solution methodologies yielding true optimal solutions, we assume that jobs are processed in the order they arrive at machines, i.e., the machines operate on a non-preemptive first-come first-served policy.

The related optimal control literature, on the other hand, assumed that jobs are processed in a given sequence, and concentrated on determining the optimal

control inputs which in turn determine the optimal service times. The idea of treating scheduling problems for deterministic queues as optimal control problems on discrete event dynamic systems first appeared in [12] where job arrival times to a single machine system were controlled to minimize the discrepancy between job completion times and desired due dates. Following this work, service time control problems for the systems, where the job arrival times are known in advance and the service times can be adjusted between processes, were considered. Pepyne and Cassandras, in [37], formulated a non-convex and non-differentiable optimal control problem for a single machine system with the objective of completing jobs as fast as possible with the least amount of control effort and used calculus of variations techniques to obtain structural properties of the optimal solution. In [38], Pepyne and Cassandras extended their results to jobs with non-regular completion-time costs penalizing earliness and tardiness with given due dates. The task of solving these problems was simplified by exploiting structural properties of the optimal sample path and it was shown that the optimal solution is unique by Cassandras et al. [7]. Further exploiting the structural properties of the optimal sample path for the single machine problem, "backward in time" and "forward in time" algorithms based on the decomposition of the original non-convex and non-differentiable optimization problem into a set of smaller convex optimization problems with linear constraints were presented by Wardi et al. [47] and Cho et al. [10], respectively. The "forward in time" algorithm presented by Cho et al. [10] was later improved by Zhang and Cassandras [48]. In a related work, Cassandras and Mookherjee [6] studied the case of uncertainty where only some future arrival information is available within a time window of length $T$ and introduced a receding horizon control scheme along with its several properties enabling the use of a controller based on rough estimates of unknown future arrivals with limited loss of optimality properties. Moon and Wardi [34] considered a single machine problem where the completed jobs wait in a finite size output buffer until their due dates. They presented an efficient solution algorithm for this system with blocking. Mao et al. [32] removed the completion-time costs and introduced deadline constraints. Some optimal solution properties of the resulting problem were identified leading to a highly efficient solution algorithm under the assumption that a feasible solution exists. In the absence of feasible solutions

due to deadline constraints, Mao and Cassandras [29] introduced an admission control scheme in which some jobs are removed with the objective of maximizing the number of remaining jobs which are all guaranteed feasibility and, through derivation of several optimality properties, developed a computationally efficient algorithm for solving the resulting admission control problem.

Flow shop systems are not the simple extensions of the single machine systems. It is much more difficult to solve the service time optimization problems in flow shop systems for two main reasons: *i*) there is an $M$-fold (where $M \geq 2$ is the number of machines in the flow shop) increase in the dimensionality of the decision variables, and *ii*) coupling among the machines' dynamics causes the failure of the structural properties exploited in single machine systems. Due to these difficulties, only a few works were conducted on the service time optimization problems on flow shop systems in the optimal control literature. The work on service time control problems for flow shop systems with identical jobs started out with Cassandras et al. [5], which derived some necessary conditions for optimality and introduced a solution technique using the Bezier approximation method for a two-machine flow shop system. Recently, building on the works in [32], Mao and Cassandras [30] considered two-machine flow shop systems with service costs that are decreasing on service times and derived some optimality properties that led to an iterative algorithm, which was shown to converge. The results in [30], were later extended to multi-machine flow shop systems with nonidentical jobs by Mao and Cassandras [31]. To the best of our knowledge, Mao and Cassandras [31] is the only study on service time optimization of multi-machine flow shop systems in the optimal control literature. Therefore, we can say that service time optimization of flow shops needs further attention and we aim to contribute to the literature in that sense.

In this thesis, we consider the flow shop system in Mao and Cassandras [31] with identical jobs and introduce initially controllable and uncontrollable machines, and job completion-time costs. Although it seems to be a simple extension, structural properties allowing for an efficient solution procedure for the system in [31], which focuses only on service costs at machines, no longer hold when we include job completion-time costs in the objective function even in the absence

of initially controllable and uncontrollable machines. Hence, the analysis changes completely. We first formulate an optimization problem minimizing a convex cost objective over a non-convex feasible region due to max-plus algebra used for the representation of departures of jobs from machines, which is, therefore, non-convex and non-differentiable. The standard way of solving this non-convex and non-differentiable problem is indeed to apply linearization on the equality constraints including *max* function to convexify the feasible region. However, the linearization process, which replaces each *max* equality constraint with two inequality constraints, doubles the number of constraints in the resulting convex formulation. Hence, the numeric solution of this convex optimization problem demands a large memory limiting the solvable system sizes due to its increased dimensionality. Hence, we search for more efficient solution methodologies in terms of both solvable system sizes and solution times in this thesis.

In Chapter 2, we consider the flow shop systems consisting only of initially controllable and uncontrollable machines termed *fixed service time flow shop systems*. In order to relieve the memory bottleneck for the convex formulation derived through aforementioned linearization process, we present a set of waiting and completion time characteristics of fixed service time flow shop systems regardless of the cost objective. Mainly, we show that no waiting is observed after the slowest machine, i.e., machine with the highest service time, of the system and jobs that do not wait at the slowest machine observe no waiting in the system. Based on these results, we introduce two alternative representations for the job completion times. Employing the first representation, we derive a simplified equivalent convex optimization problem, which improves the solution times and enables solutions for larger systems due to less memory requirements than the convex optimization problem obtained through linearization. However, the resulting simplified convex optimization problem still needs the use of a convex optimization solver which may not be available at some of the manufacturing companies. Hence, motivated by the need for a lower cost optimization tool, we employ the second representation of job completion times to introduce another equivalent convex optimization problem, which is non-differentiable, along with its subgradient descent algorithm. As demonstrated by a numerical study, the

subgradient descent algorithm not only eliminates the need for convex optimization solvers but also allows for the solution of larger systems due to its much less memory requirements and improves the solution times significantly. We also analyze a specific service cost structure inversely proportional to the services times. This cost structure allows us to sort the optimal service times of the machines and to introduce a new search algorithm much faster than the subgradient descent solution algorithm.

In Chapter 3, building on the results for fixed service time flow shop systems, we consider the flow shop systems formed of fully controllable, initially controllable and uncontrollable machines termed *mixed line flow shop systems*. Existence of fully controllable machines in the system brings new structural properties leading to new solution methodologies different from the ones developed for fixed service time flop shop systems. Hence, we continue with a new chapter for mixed line flow shops systems. To overcome the problem of limitation on solvable system sizes due to the huge memory requirements of the resulting convex optimization problem obtained through linearization on the *max* constraints, and to improve the solution times, we first present a set of optimal waiting characteristics of these systems. In particular, under the strict convexity assumption of service costs, we show that jobs do not wait on the optimal sample path after the first fully controllable machine. Employing the no-wait property, we then derive simplified equivalent convex optimization problems. For the flow shop systems formed of only fully controllable and uncontrollable machines, the associated simplified equivalent convex optimization problem is then decomposed by a "forward in time" algorithm into smaller convex optimization problems under an additional strict convexity assumption on the job completion-time costs. As shown by a computational study, the simplifications and the decomposition not only improve the solution times considerably but also allow us to solve larger problems by alleviating memory constraints.

The rest of the thesis is organized as follows: In Chapter 2, we formulate a non-convex and non-differentiable optimization problem for the flow shop systems formed of initially controllable and uncontrollable machines and obtain a convex programming formulation by the standard method of linearization. A set

of waiting and completion time characteristics of these systems regardless of the objective function is derived. These characteristics are then employed to derive a simpler equivalent convex optimization problem and another alternative equivalent convex optimization problem along with a subgradient descent algorithm. Finally, a new search algorithm is developed for a specific nonlinear decreasing service cost structure in this chapter. In Chapter 3, we formulate a non-convex and non-differentiable optimization problem for the flow shop systems formed of fully controllable, initially controllable, and uncontrollable machines and obtain a convex programming formulation by the standard method of linearization. We derive a set of optimal waiting characteristics of such systems and exploit them to derive equivalent simplified convex optimization problems. A forward decomposition algorithm is also presented in this chapter to decompose the associated simplified convex optimization problem into smaller convex optimization problems with linear constraints. Finally, we give concluding remarks, model extensions, and future research directions in Chapter 4.

# Chapter 2

# Service Time Optimization of Fixed Service Time Flow Shop Systems

In this chapter, we consider deterministic flow shop systems formed only of initially controllable and uncontrollable machines processing identical jobs with known arrival times and deadlines. Since the service times are fixed at the start-up time and are not changed during the whole process, we define these systems as fixed service time flow shops. The cost function we consider consists of service costs at initially controllable machines and regular completion-time costs of jobs. Motivated by the extended Taylor's tool-wear equation [24], we assume that faster services increase wear and tear on the tools due to increased temperatures, and may raise the need for extra supervision, increasing service costs. The losses of the product quality due to faster services are also lumped into these service costs. Slower services, on the other hand, may delay the completion times increasing the completion-time costs and/or leading to untimely job completions. We acknowledge this trade-off and set our objective as to determine the cost-minimizing service times.

In this chapter, we formulate a non-convex and non-differentiable optimization

problem and apply the standard method of linearization on the *max* constraints to get a convex formulation. Since, the resulting convex formulation provides solution only for small systems due to its high memory requirements, aiming to solve larger systems and to improve solution times, we first derive a set of waiting and completion time characteristics for such systems independent of the cost objective. Basically, we show that no waiting is observed at the downstream of the machine with the highest service time and if a job does not wait at this machine, it also observes no waiting in the system. Then, we exploit these waiting characteristics to derive an alternative representation of job completion times allowing us to present a simpler equivalent convex optimization problem. However, even though the resulting convex problem formulation improves the solution times and enables solutions for larger systems, it still needs the use of a solver which may not be available at some manufacturing companies. Hence, further exploiting the waiting and completion time characteristics, we come up with another alternative representation of job completion times and employ it to introduce another equivalent convex optimization problem, which is non-differentiable. A subgradient descent algorithm is also developed for solving this optimization problem. This algorithm eliminates the need for a solver and has considerably low memory requirements; therefore, it allows us to solve optimization problems of even larger systems in much shorter times. We also analyze a special case where the service costs at machines are nonlinear decreasing functions of service times. This cost structure allows us to sort the optimal service times of machines. First, we define differentiable subproblems that can be solved easily. Employing these subproblems, we then introduce a two-phase search algorithm that converges in a finite number of steps. Through improving the solution times drastically, this new search algorithm eliminates the need for the subgradient descent solution algorithm whose performance, as a drawback, is highly affected by the selection of its termination tolerance and step sizes at each iteration.

The rest of the chapter is organized as follows: We formulate a non-convex and non-differentiable optimization problem and apply the linearization method yielding a convex optimization problem in Section 2.1. In Section 2.2, regardless of the objective function, we derive a set of waiting and completion time characteristics

for fixed service time flow shop systems. Employing the waiting and completion time characteristics in fixed service time flow shop systems derived in Section 2.2, a simpler equivalent convex optimization problem is introduced in Section 2.3. In Section 2.4, an alternative equivalent convex optimization problem is presented along with a subgradient descent algorithm with projections. For a special type service cost structure inversely proportional to the service times, we define differentiable subproblems and present a two-phase search algorithm in Section 2.5. Section 2.6 presents a numerical example to illustrate the waiting and completion time characteristics derived in Section 2.2 under optimal service times. In this section, we also compare the performances of the proposed methodologies in terms of the solution times and the solvable system sizes through a computational study. Finally, Section 2.7 concludes the chapter.

## 2.1 Problem Formulation

The notation used throughout the chapter is as follows:

**Decision Variables:**

$x_{i,j}$ : departure time of job $i$ from machine $j$.

$s_j$ : service time at machine $j$.

**Parameters:**

$M$ : number of machines in the system.

$N$ : number of jobs that arrive at the system.

$a_i$ : arrival time of job $i$.

$d_i$ : deadline for the completion of job $i$.

$S_j$ : lower bound for the service time at machine $j$.

$\theta_j(s_j)$ : total service cost over all jobs at machine $j$.

$\phi_i(x_{i,M})$ : completion-time cost for job $i$.

We consider a sequence of $N$ identical jobs, denoted by $\{C_i\}_{i=1}^N$, arriving at an $M$-machine flow shop system at known times $0 \leq a_1 \leq a_2 \leq ... \leq a_N$. Machines process one job at a time on a first-come-first-served (FIFO) non-preemptive basis (i.e. a job in service can not be interrupted until its service completion). The buffers in front of the machines are assumed to be of infinite sizes.

Without loss of generality, uncontrollable machines can be treated as if they are initially controllable machines. Hence, to keep the notation simple and to make the rest of the chapter more readable, we acknowledge the reader that we study the flow shop systems consisting only of initially controllable machines whose results are applicable to flow shop systems including also uncontrollable machines.

We define a temporal state $x_{i,j}$ that keeps the departure time information of job $C_i$ from machine $j$. The relationships between the temporal states are given by the following max-plus equations (see [4]):

$$x_{i,j} = \max(x_{i,j-1}, x_{i-1,j}) + s_j, \tag{2.1}$$

$$x_{i,0} = a_i, \qquad x_{0,j} = -\infty \tag{2.2}$$

for $i = 1, ..., N$ and $j = 1, ..., M$, where the service time at machine $j \in \{1, ..., M\}$ is denoted by $s_j$. Note that the same service time $s_j$ is applied to all jobs at machine $j$. The deadlines $\{d_i\}_{i=1}^N$ are imposed to jobs $\{C_i\}_{i=1}^N$ so that

$$x_{i,M} \leq d_i. \tag{2.3}$$

The discrete-event optimal control problem, denoted by $P$, is the determination of the optimal service times:

$$P: \min_{\substack{s_j \geq S_j \\ j=1,...,M}} \left\{ J = \sum_{j=1}^{M} \theta_j(s_j) + \sum_{i=1}^{N} \phi_i(x_{i,M}) \right\} \tag{2.4}$$

subject to (2.1)-(2.3) for $i = 1, ..., N$ and $j = 1, ..., M$. In this formulation, $\theta_j$ denotes the total service cost over all jobs at machine $j$, and $\phi_i$ denotes the

completion-time cost for job $C_i$. The minimum service time required at machine $j$, a physical constraint, is denoted by $S_j$.

Due to the existence of lower bounds on the service times of machines, i.e., $s_j \geq S_j$ for all $j = 1, ..., M$, the system can not guarantee that all jobs meet their associated deadlines, that is, the optimization problem $P$ in (2.4) may be infeasible. In this thesis, we will study the feasible case. We can handle the infeasible case by introducing an admission control mechanism through which some jobs are selected and removed so that the system becomes feasible while minimizing the number of such jobs as described for the single machine case in [29]. However, we will not consider the job admission problem here.

The following assumptions are necessary to make the problem somewhat more tractable while preserving the originality of the problem.

**Assumption 2.1** *:* $\theta_j(\cdot)$*, for* $j = 1, ..., M$*, is monotonically decreasing and convex.*

**Assumption 2.2** *:* $\phi_i(\cdot)$*, for* $i = 1, ..., N$*, is monotonically increasing and convex.*

These assumptions indicate that longer services will decrease the service costs while increasing the departure times, hence, the completion-time costs.

Due to the *max* function in (2.1), the optimization problem $P$ in (2.4) is non-convex and non-differentiable. A standard method for solving the optimization problem $P$ is to replace (2.1) with two linear inequalities and to employ (2.2) for the first job. Since, by Assumptions 2.1 and 2.2, both costs are convex, we arrive at the following convex optimization problem:

$$\bar{P}: \min_{\substack{s_j \geq S_j \\ x_{i,j} \\ i=1,...,N \\ j=1,...,M}} \left\{ \bar{J} = \sum_{j=1}^{M} \theta_j(s_j) + \sum_{i=1}^{N} \phi_i(x_{i,M}) \right\} \tag{2.5}$$

subject to

$$
\begin{aligned}
x_{1,1} &= a_1 + s_1 && (2.6)\\
x_{1,j} &= a_1 + \sum_{k=1}^{j} s_k && (2.7)\\
x_{i,1} &\geq a_i + s_1 && (2.8)\\
x_{i,1} &\geq x_{i-1,1} + s_1 && (2.9)\\
x_{i,j} &\geq x_{i,j-1} + s_j && (2.10)\\
x_{i,j} &\geq x_{i-1,j} + s_j && (2.11)\\
x_{1,M} &\leq d_1 && (2.12)\\
x_{i,M} &\leq d_i && (2.13)
\end{aligned}
$$

for all $i = 2, ..., N$ and $j = 2, .., M$. There are $(N + 1)M$ variables, $M$ equality and $2(N - 1)M + N$ inequality constraints in this formulation excluding the $M$ boundary value constraints on the service times.

The optimization problem $\bar{P}$ is over a larger feasible set than the optimization problem $P$; therefore its optimal cost $\overline{J}^*$ is upper bounded by the optimal cost of $P$ denoted by $J^*$, i.e., $\overline{J}^* \leq J^*$. However, it can easily be verified that an optimal service time vector $\bar{s}^*$ for $\bar{P}$ is also an optimal solution for $P$ with $\overline{J}^* = J^*$. Hence, the optimal solution for the optimization problem $P$ can be determined by solving the convex optimization problem $\bar{P}$.

In the next section, independent of the cost structure, we derive a set of waiting and completion time characteristics of the flow shop systems with fixed service times. These waiting and completion time characteristics will then allow us to present alternative equivalent convex optimization problems.

## 2.2 Waiting Characteristics of Fixed Service Time Flow Shop Systems

In fixed service time flow shop systems, each machine $j$ performs some service of duration $s_j$. Based on these service times, we define the following:

**Definition 2.1** *Machine $u$ is a* local bottleneck *if its service time exceeds the service times of all upstream machines, i.e., $s_u > \max_{j=0,...,u-1} s_j$ where $s_0$ is defined to be zero.*

Since the first machine is a local bottleneck, there is at least one local bottleneck in each fixed service time flow shop system.

**Definition 2.2** *A contiguous set of machines $\{u, ..., v\}$ form a* flushing portion *if*

1. Machine $u$ is a local bottleneck, i.e., $s_u > \max_{j=0,...,u-1} s_j$;
2. There are no local bottlenecks in machines $\{u + 1, ..., v\}$, i.e., $s_u \geq \max_{j=u+1,...,v} s_j$;
3. If $v < M$, then machine $(v + 1)$ is a local bottleneck, i.e., $s_u < s_{v+1}$.

Each local bottleneck machine starts a flushing portion, and the last flushing portion is ended by machine $M$.

The following lemma establishes that jobs may wait at only the local bottleneck machines.

**Lemma 2.1** *No waiting is observed in a flushing portion after its local bottleneck machine.*

**Proof.** (By induction) Let us consider some flushing portion formed of machines $\{u, ..., v\}$. Since the first job does not wait at any machine, we have the basis for the induction. Now, let us assume that jobs $C_r$, $r = 1, ..., i-1$ do not wait at machines $\{u+1, ..., v\}$, i.e.,

$$x_{r,j} \geq x_{r-1,j+1} \tag{2.14}$$

holds for all $j = u, ..., v-1$. From (2.1), we have

$$\begin{aligned} x_{i,u} &= \max\left(x_{i,u-1}, x_{i-1,u}\right) + s_u \\ &\geq x_{i-1,u} + s_u \end{aligned} \tag{2.15}$$

and from the induction assumption (2.14), job $C_{i-1}$ does not wait at machine $(u+1)$; therefore

$$x_{i-1,u+1} = x_{i-1,u} + s_{u+1}. \tag{2.16}$$

Since machine $(u+1)$ resides in the flushing portion started by the local bottleneck machine $u$, $s_u \geq s_{u+1}$ by definition; hence, from (2.15) and (2.16), we get

$$x_{i,u} \geq x_{i-1,u+1},$$

i.e., job $C_i$ does not wait at machine $(u+1)$. Next, in addition to (2.14), let us assume that job $C_i$ does not wait at machines $\{u+1, ..., j\}$ where $j < v$, i.e.,

$$x_{i,k} \geq x_{i-1,k+1} \tag{2.17}$$

holds for all $k = u, ..., j-1$. From the induction assumptions (2.14) and (2.17), we can write

$$x_{i-1,j+1} = x_{i-1,u} + \sum_{l=u+1}^{j+1} s_l \tag{2.18}$$

and

$$\begin{aligned} x_{i,j} &= x_{i,u} + \sum_{l=u+1}^{j} s_l \\ &= \max\left(x_{i,u-1}, x_{i-1,u}\right) + s_u + \sum_{l=u+1}^{j} s_l \\ &\geq x_{i-1,u} + \sum_{l=u}^{j} s_l. \end{aligned} \tag{2.19}$$

Since $s_u \geq s_{j+1}$ by definition, from (2.18) and (2.19), we have

$$x_{i,j} \geq x_{i-1,j+1} \tag{2.20}$$

indicating that job $C_i$ does not wait at machine $(j+1)$, therefore, concluding the induction proof. ■

The next lemma suggests that, given the waiting status of a job at a local bottleneck machine, we may deduce its waiting status at a downstream or an upstream local bottleneck machine.

**Lemma 2.2** *If job $C_i$ waits for service at some local bottleneck, then it will wait for service at all downstream local bottlenecks.*

**Proof.** We consider two consecutive local bottleneck machines $u$ and $(v+1)$, and assume that job $C_i$ waits at machine $u$, so we have

$$x_{i,u-1} < x_{i-1,u}. \tag{2.21}$$

If these two local bottleneck machines are adjacent, i.e., if $v = u$ then, from (2.1) and (2.21), we have

$$x_{i,v} = x_{i-1,u} + s_u \tag{2.22}$$

and

$$x_{i-1,v+1} \geq x_{i-1,u} + s_{v+1}. \tag{2.23}$$

Since $s_u < s_{v+1}$ by definition, from (2.22) and (2.23), we get $x_{i,v} < x_{i-1,v+1}$, i.e., job $C_i$ waits at machine $(v+1)$.

If, on the other hand, these two local bottlenecks are not adjacent, i.e., $v > u$ then, from (2.1), (2.21), and by Lemma 2.1, we have

$$\begin{aligned} x_{i,v} &= x_{i,u} + \sum_{j=u+1}^{v} s_j \\ &= \max\left(x_{i,u-1}, x_{i-1,u}\right) + s_u + \sum_{j=u+1}^{v} s_j \\ &= x_{i-1,u} + s_u + \sum_{j=u+1}^{v} s_j \end{aligned} \tag{2.24}$$

and

$$\begin{aligned} x_{i-1,v+1} &= \max\left(x_{i-1,v}, x_{i-2,v+1}\right) + s_{v+1} \\ &\geq x_{i-1,v} + s_{v+1} \\ &\geq x_{i-1,u} + \sum_{j=u+1}^{v} s_j + s_{v+1}. \end{aligned} \tag{2.25}$$

Since $s_u < s_{v+1}$ by definition, from (2.24) and (2.25), we get $x_{i,v} < x_{i-1,v+1}$, i.e., job $C_i$ waits at machine $(v+1)$.

The result extends iteratively to all downstream local bottleneck machines concluding the proof. ■

As it turns out, waiting is observed only at the local bottleneck machines. Given the arrival times of the jobs and the service time of some local bottleneck machine $u$, we can determine which jobs wait at this machine. Let us define the average interarrival time between jobs $C_k$ and $C_l$, where $k > l$ as

$$\sigma_k^l = \frac{a_k - a_l}{k - l}. \tag{2.26}$$

The minimum of the average interarrival times for job $C_k$ is, then, defined as

$$\sigma_k = \begin{cases} \infty, & k = 1 \\ \min\limits_{l=1,\ldots,k-1} \sigma_k^l, & k > 1. \end{cases} \tag{2.27}$$

The following lemma allows us to determine whether a job waits or not at some local bottleneck machine $u$.

**Lemma 2.3** *A job $C_k$ waits for service at the local bottleneck machine $u$ if and only if $\sigma_k < s_u$.*

**Proof.** (Necessity) Let us assume that $C_k$ does not wait at the local bottleneck machine $u$. According to Lemmas 2.1 and 2.2, no waiting is observed by the job at the upstream machines; therefore we have

$$x_{k,u} = a_k + \sum_{j=1}^{u} s_j. \tag{2.28}$$

For previous jobs $\{C_i\}_{i=1}^{k-1}$, we can write

$$x_{i,u} \geq a_i + \sum_{j=1}^{u} s_j. \tag{2.29}$$

Hence, from (2.28) and (2.29), we get

$$x_{k,u} - x_{i,u} \leq a_k - a_i \tag{2.30}$$

for all $i = 1, ..., k-1$. Since the departure times (from machine $u$) of two consecutive jobs are at least $s_u$ apart, we can write

$$x_{k,u} - x_{i,u} \geq (k-i)s_u \tag{2.31}$$

for all $i = 1, ..., k-1$. From (2.26), (2.30), and (2.31), we have

$$\sigma_k^i \geq s_u$$

for all $i = 1, ..., k-1$, resulting in, from (2.27),

$$\sigma_k \geq s_u.$$

(Sufficiency) Let us assume that job $C_k$ waits at machine $u$. Then, we have

$$x_{k,u} > a_k + \sum_{j=1}^{u} s_j. \tag{2.32}$$

Let $C_i$ be the last job in $\{C_1, ..., C_{k-1}\}$ that does not wait at machine $u$ (since job $C_1$ does not wait at any machine, existence of such a job is guaranteed.) Then, according to Lemmas 2.1 and 2.2, $C_i$ does not wait at any upstream machine, so we can write

$$\begin{aligned} x_{k,u} &= x_{i,u} + (k-i)s_u \\ &= a_i + \sum_{j=1}^{u} s_j + (k-i)s_u. \end{aligned} \tag{2.33}$$

From (2.26), (2.32), and (2.33), we get

$$\sigma_k^i < s_u$$

resulting in, from (2.27),

$$\sigma_k < s_u.$$

■

We describe the waiting characteristics of jobs at local bottleneck machines by block structures.

**Definition 2.3** *A contiguous set of jobs $\{C_i\}_{i=k}^{n}$ is said to form a* block *at a local bottleneck machine $u$ if*

*1) Jobs $C_k$ and $C_{n+1}$ (if exists) do not wait at machine $u$, i.e., $x_{k,u-1} \geq x_{k-1,u}$ and $x_{n+1,u-1} \geq x_{n,u}$ for $n < N$;*

*2) Jobs* $\{C_i\}_{i=k+1}^{n}$ *wait at machine* $u$*, i.e.,* $x_{i,u-1} < x_{i-1,u}$ *for* $i = k+1, ..., n$*.*

For some local bottleneck machine $u$, each block starts with a non-waiting job $k$ and continues with waiting jobs $\{C_i\}_{i=k+1}^{n}$with departure times

$$x_{i,u} = x_{k,u} + (i-k)s_u. \tag{2.34}$$

**Definition 2.4** *A partition of jobs into blocks is called a* block structure*.*

For any given service time $s_u$, by modifying the arrival times, we can generate $2^N$ different block structures at a local bottleneck machine $u$. If the arrival times are given, however, by modifying the service time $s_u$, we can generate at most $N$ different block structures. The next lemma establishes this upper bound on the number of different block structures at a local bottleneck machine.

**Lemma 2.4** *There are at most* $N$ *different block structures at any local bottleneck machine* $u$*.*

**Proof.** From Lemma 2.3, a job $C_i$ starts a block at a local bottleneck machine $u$ iff $\sigma_i \geq s_u$. Reindexing $\sigma_i$'s as

$$\sigma_{(1)} \leq \sigma_{(2)} \leq ... \leq \sigma_{(N)},$$

each interval $(\sigma_{(k-1)}, \sigma_{(k)}]$, where $\sigma_{(0)} = 0$, defines a block structure: If $s_u \in (\sigma_{(k-1)}, \sigma_{(k)}]$, then all jobs in the set $\{C_i : \sigma_i \geq \sigma_{(k)}\}$ start blocks at machine $u$ while others do not. Since there are at most $N$ such intervals, there are at most $N$ different block structures. ■

According to Lemma 2.3, one could evaluate $\sigma_k$ values for all jobs $C_k$ and compare them to the service time of the local bottleneck machine to determine the block structure. The following lemma, however, presents a computationally simpler way to determine the block structure, which is implemented in the subgradient algorithm developed in Section 2.4.

**Lemma 2.5** *If jobs* $\{C_i\}_{i=k}^{n}$ *form a block at machine* $u$*, then,*

$$\sigma_i^k < s_u$$

*is satisfied for all* $i = k+1, ..., n$*.*

**Proof.** (By Induction) Since $C_k$ starts the block, we know by definition that it does not wait at machine $u$. Hence, by Lemma 2.3, we have $\sigma_k \geq s_u$, i.e., for all $l < k$, we can write

$$\sigma_k^l = \frac{a_k - a_l}{k-l} \geq s_u. \tag{2.35}$$

In order to show the basis step by a contradiction, we assume that

$$\sigma_{k+1}^k = a_{k+1} - a_k \geq s_u. \tag{2.36}$$

From (2.35) and (2.36), we get for all $l < k$

$$\begin{aligned} \sigma_{k+1}^l &= \frac{a_{k+1} - a_l}{k+1-l} \\ &= \frac{(a_{k+1} - a_k) + (a_k - a_l)}{k+1-l} \\ &\geq \frac{s_u + (k-l)s_u}{k+1-l} = s_u \end{aligned}$$

resulting in $\sigma_{k+1} \geq s_u$, which contradicts, by Lemma 2.3, that job $C_{k+1}$waits.

In order to show the induction step again by contradiction, we assume that

$$\sigma_i^k < s_u \tag{2.37}$$

for $i = k+1, ..., t-1$, where $t \leq n$ and

$$\sigma_t^k \geq s_u. \tag{2.38}$$

From (2.35) and (2.38), we have

$$\begin{aligned} \sigma_t^l &= \frac{a_t - a_l}{t - l} = \frac{(a_t - a_k) + (a_k - a_l)}{t - l} \\ &\geq \frac{(t-k)s_u + (k-l)s_u}{t-l} = s_u \end{aligned} \tag{2.39}$$

for all $l = 1, ..., k-1$. Moreover, from (2.37) and (2.38), we have

$$\begin{aligned} \sigma_t^i &= \frac{a_t - a_i}{t - i} = \frac{(a_t - a_k) - (a_i - a_k)}{t - i} \\ &\geq \frac{(t-k)s_u - (i-k)s_u}{t-i} = s_u \end{aligned} \tag{2.40}$$

for all $i = k+1, ..., t-1$. Hence, from (2.27), (2.39), and (2.40), we have $\sigma_t \geq s_u$, which contradicts, by Lemma 2.3, that job $C_t$ waits. ■

Starting with the first job $C_1$, which starts the first block, this lemma can be iteratively applied to determine the block structure at any local bottleneck machine. For this task, all we need are the arrival times of the jobs and the service time of the local bottleneck machine.

Next, we define the most downstream local bottleneck machine of the flow shop system as the global bottleneck, and derive the completion times of jobs.

**Definition 2.5** *The local bottleneck machine $m$ with the highest service time $s_m = \max_{j=1,...,M} s_j$ is the* global bottleneck.

There can be no local bottleneck machine downstream to a global bottleneck machine; therefore, by Lemma 2.1, no waiting is observed after the global bottleneck machine. Hence, the completion times can be determined as presented in the next lemma.

**Lemma 2.6** *The completion time of job $C_i$ is given by*

$$x_{i,M} = \max\left(a_i + \sum_{j=1}^{M} s_j, x_{i-1,M} + s_m\right), \tag{2.41}$$

*where* $x_{0,M} = -\infty$ *and* $s_m = \max_{j=1,\ldots,M} s_j$ *is the service time of the global bottleneck machine.*

**Proof.** From (2.1), the departure time of job $C_i$ from the global bottleneck machine $m$ is given as

$$x_{i,m} = \max\left(x_{i,m-1}, x_{i-1,m}\right) + s_m. \tag{2.42}$$

If job $C_i$ does not wait at the global bottleneck machine $m$, i.e., if $x_{i,m-1} \geq x_{i-1,m}$, by Lemma 2.2, it also does not wait at any upstream machine; therefore, we have

$$x_{i,m-1} = a_i + \sum_{j=1}^{m-1} s_j \geq x_{i-1,m}. \tag{2.43}$$

Hence, from (2.42) and (2.43), we get

$$x_{i,m} = \max\left(a_i + \sum_{j=1}^{m} s_j, x_{i-1,m} + s_m\right). \tag{2.44}$$

Since no waiting is observed after the global bottleneck machine $m$, from (2.44), we can write the completion time of the job $C_i$ as

$$\begin{aligned} x_{i,M} &= \begin{cases} x_{i,m}, & \text{if } m = M \\ x_{i,m} + \sum_{j=m+1}^{M} s_j, & \text{if } m < M \end{cases} \\ &= \max\left(a_i + \sum_{j=1}^{M} s_j, x_{i-1,M} + s_m\right). \end{aligned}$$

■

Alternatively, based on the block structure at the global bottleneck machine, the completion times can also be determined as presented in the next lemma.

**Lemma 2.7** *Let jobs* $\{C_i\}_{i=k}^{n}$ *form a block at the global bottleneck machine* $m$*.*

*Then, the completion times of these jobs are given as*

$$x_{i,M} = a_k + (i-k)s_m + \sum_{j=1}^{M} s_j \tag{2.45}$$

*for* $i = k, ..., n$.

**Proof.** Machines $\{m, ..., M\}$ form the last flushing portion of the system. By Lemma 2.1, jobs do not wait after the global bottleneck machine $m$,; hence the completion times of the jobs $\{C_i\}_{i=k}^{n}$ can be written as

$$x_{i,M} = x_{i,m} + \sum_{j=m+1}^{M} s_j \tag{2.46}$$

for $i = k, ..., n$. From Lemma 2.2, since $C_k$ does not wait at the global bottleneck machine $m$, it observes no waiting at the upstream machines. Hence, we can write

$$x_{k,m} = a_k + \sum_{j=1}^{m} s_j. \tag{2.47}$$

For jobs $\{C_i\}_{i=k+1}^{n}$ that wait at the global bottleneck machine $m$, we have

$$x_{i,m} = x_{k,m} + (i-k)s_m. \tag{2.48}$$

Hence, from (2.46), (2.47), and (2.48), the completion times of the jobs $\{C_i\}_{i=k}^{n}$ are given as

$$x_{i,M} = a_k + (i-k)s_m + \sum_{j=1}^{M} s_j.$$

■

In the next section, we employ the characteristics from this section to derive a simpler convex optimization problem formulation.

## 2.3 Simplified Convex Optimization Problem

The result of Lemma 2.6 allows us to replace (2.1) and (2.2) in the optimization problem $P$ by (2.41) resulting with the formulation

$$P: \min_{\substack{s_j \geq S_j \\ j=1,...,M}} \left\{ J = \sum_{j=1}^{M} \theta_j(s_j) + \sum_{i=1}^{N} \phi_i(x_{i,M}) \right\} \tag{2.49}$$

subject to

$$x_{1,M} = a_1 + \sum_{k=1}^{M} s_k \tag{2.50}$$

$$x_{i,M} = \max\left( x_{i-1,M} + \max_{j=1,...,M} s_j, a_i + \sum_{k=1}^{M} s_k \right) \tag{2.51}$$

$$x_{1,M} \leq d_1 \tag{2.52}$$

$$x_{i,M} \leq d_i \tag{2.53}$$

for $i = 2, ..., N$.

Similarly, by linearizing the *max* functions in (2.51), we get the convex optimization problem $Q$ given as

$$Q: \min_{\substack{s_j \geq S_j \\ x_{i,M} \\ i=1,...,N \\ j=1,...,M}} \left\{ J_Q = \sum_{j=1}^{M} \theta_j(s_j) + \sum_{i=1}^{N} \phi_i(x_{i,M}) \right\} \tag{2.54}$$

subject to

$$x_{1,M} = a_1 + \sum_{k=1}^{M} s_k \tag{2.55}$$

$$x_{i,M} \geq a_i + \sum_{k=1}^{M} s_k \tag{2.56}$$

$$x_{i,M} \geq x_{i-1,M} + s_j \tag{2.57}$$

$$x_{1,M} \leq d_1 \tag{2.58}$$

$$x_{i,M} \leq d_i \tag{2.59}$$

for all $i = 2, ..., N$ and $j = 1, ..., M$. In this formulation there are $(N + M)$ variables, one equality and $(N-1)(M+1)+N$ inequality constraints excluding the $M$ boundary value constraints on the service times. Therefore, compared to the convex optimization problem $\bar{P}$ given in (2.5), improvements in solution times and memory requirements are expected.

Similar to the problem $\bar{P}$, the convex optimization problem $Q$ has a larger feasible set compared to the original optimization problem $P$. However, it can easily be verified that the optimal solution for $Q$ is formed of the optimal service times for $P$ and the corresponding job completion times resulting from applying the optimal service times of $P$ evaluated through (2.50) and (2.51). Hence, solving the convex optimization problem $Q$ always yields an optimal solution for the optimization problem $P$.

Next, we further exploit the waiting and completion time characteristics derived in Section 2.2 to derive a *minmax* problem and present a subgradient descent algorithm with projections as its solution methodology.

## 2.4 Subgradient Descent Algorithm with Projections

In this section, we consider the flow shop systems with no job completion deadlines, i.e., we set $d_i = \infty$ for $i = 1, ..., N$. The relaxation of deadlines on job completion times will allow us to derive an alternative unconstrained equivalent convex optimization problem along with an efficient subgradient descent algorithm as its solution methodology.

Let us employ Lemma 2.7 to rewrite the optimization problem $P$ as

$$\hat{P}: \min_{\substack{s_j \geq S_j \\ j=1,\ldots,M}} \left\{ J(s) = \sum_{j=1}^{M} \theta_j(s_j) + \sum_{b=1}^{B(s)} \sum_{i=k^b(s)}^{n^b(s)} \phi_i \left( a_{k^b(s)} + s_{total} + (i - k^b(s)) s_{\max} \right) \right\}, \tag{2.60}$$

where, given the service times $s$, $s_{\max} = \max_{j=1,\ldots,M} s_j$ is the service time of the global bottleneck machine, $s_{total} = \sum_{j=1}^{M} s_j$ is the total service time, $B(s)$ is the number of blocks at the global bottleneck machine, $k^b(s)$ and $n^b(s)$ are the indices of the first and the last jobs of the $b$th block, respectively.

Let

$$J_l(s) = \sum_{j=1}^{M} \theta_j(s_j) + \sum_{b=1}^{B_l} \sum_{i=k_l^b}^{n_l^b} \phi_i \left( x_{i,M}^l \right) \tag{2.61}$$

be a cost function, where $B_l$ is the number of blocks, $k_l^b$ and $n_l^b$ are the indices of the first and the last jobs, respectively, of the $b$th block at some global bottleneck whose service time falls in the interval $(\sigma_{(l-1)}, \sigma_{(l)}]$, and $x_{i,M}^l$ is the completion time of job $C_i$ given, by Lemma 2.7, as

$$x_{i,M}^l = a_{k_l^b} + s_{total} + (i - k_l^b) s_{\max}.$$

Note that, by Assumptions 2.1 and 2.2, $J_l$ is continuous and convex in the service times. By Lemma 2.4, there are at most $N$ different block structures at the global bottleneck; hence we have at most $N$ different cost functions of this form.

If $s_{\max}$ falls in the interval $(\sigma_{(l-1)}, \sigma_{(l)}]$, then we have $J(s) = J_l(s)$. In other words, the formulation of $J(s)$ differs from interval to interval. The next lemma shows that $J(s)$ can be written as the maximum of all these functions, yielding a *minmax* optimization problem.

**Lemma 2.8** *The cost function $J_l(s)$ exceeds all other cost functions, i.e., $J_l(s) = \max_{t \in \{1,\ldots,N\}} J_t(s)$, when $s_{\max} \in (\sigma_{(l-1)}, \sigma_{(l)}]$.*

**Proof.** Let us take an arbitrary job $C_i$, where $i \in \{1, ..., N\}$, and let job $C_{k_l}$ start the block at the global bottleneck machine that job $C_i$ resides in when

the global bottleneck machine's service time falls in the interval $(\sigma_{(l-1)}, \sigma_{(l)}]$, i.e., when $s_{\max} \in (\sigma_{(l-1)}, \sigma_{(l)}]$. The completion time in this case is given by Lemma 2.7 as

$$x^l_{i,M} = a_{k_l} + s_{total} + (i - k_l)s_{\max}. \tag{2.62}$$

Let us also take an arbitrary block structure corresponding to some interval $(\sigma_{(t-1)}, \sigma_{(t)}]$, and let $C_{k_t}$ start the block at the global bottleneck machine that job $C_i$ resides in. Similarly, by Lemma 2.7, the completion time of job $C_i$ for this block structure is given as

$$x^t_{i,M} = a_{k_t} + s_{total} + (i - k_t)s_{\max}. \tag{2.63}$$

Now, assume that $s_{\max} \in (\sigma_{(l-1)}, \sigma_{(l)}]$. We would like to compare $J_l(s)$ and $J_t(s)$ under this assumption.

From (2.62) and (2.63), the completion times satisfy

$$x^l_{i,M} - x^t_{i,M} = (a_{k_l} - a_{k_t}) + (k_t - k_l)s_{\max}. \tag{2.64}$$

There are three cases to consider:

**Case 1:** For $t = l$, from (2.64), we have $x^l_{i,M} = x^t_{i,M}$.

**Case 2:** For $t < l$, i.e., for $\sigma_{(t)} < \sigma_{(l)}$, by Lemma 2.3, $k_t \geq k_l$ because decreasing the service time of the global bottleneck has the effect of separating blocks into smaller blocks. If $k_t = k_l$, then from (2.64), $x^l_{i,M} = x^t_{i,M}$. If, on the other hand, $k_t > k_l$, then job $C_{k_t}$ is in the block started by job $C_{k_l}$, which leads to $\sigma^{k_l}_{k_t} < s_{\max}$ by Lemma 2.5. Therefore, we have, from (2.26) and (2.64), that

$$\begin{aligned} x^l_{i,M} - x^t_{i,M} &= (k_t - k_l)\left[s_{\max} - \frac{(a_{k_t} - a_{k_l})}{(k_t - k_l)}\right] \\ &= (k_t - k_l)[s_{\max} - \sigma^{k_l}_{k_t}] \geq 0. \end{aligned}$$

**Case 3:** For $t > l$, i.e., for $\sigma_{(t)} > \sigma_{(l)}$, by Lemma 2.3, $k_t \leq k_l$ because increasing the service time of the global bottleneck has the effect of combining

blocks into larger blocks. If $k_t = k_l$, then from (2.64), $x^l_{i,M} = x^t_{i,M}$. If, on the other hand, $k_t < k_l$, then since $\sigma_{k_l} \geq s_{\max}$ by Lemma 2.3, we have, from (2.26), (2.27), and (2.64), that

$$\begin{aligned} x^l_{i,M} - x^t_{i,M} &= (k_t - k_l)\left[s_{\max} - \frac{(a_{k_l} - a_{k_t})}{(k_l - k_t)}\right] \\ &= (k_l - k_t)[\sigma^{k_t}_{k_l} - s_{\max}] \\ &\geq (k_l - k_t)[\sigma_{k_l} - s_{\max}] \geq 0. \end{aligned}$$

Hence, from all three cases, $x^l_{i,M} \geq x^t_{i,M}$, when $s_{\max} \in (\sigma_{(l-1)}, \sigma_{(l)}]$. By Assumption 2.2, $\phi_i$ is monotonically increasing; therefore, from (2.45) and (2.61),

$$J_l(s) - J_t(s) = \sum_{i=1}^{N} \left(\phi_i(x^l_{i,M}) - \phi_i(x^t_{i,M})\right) \geq 0.$$

Since $t \leq N$ is arbitrary, the result follows. ■

Hence, by Lemma 2.8, we can write the optimization problem as

$$R: \min_{\substack{s_j \geq S_j \\ j=1,...,M}} \left\{ J_R(s) = \max_l J_l(s) \right\}, \tag{2.65}$$

where $J_l$ is the convex and continuous cost function corresponding to the interval $(\sigma_{(l-1)}, \sigma_{(l)}]$. Being the maximum of convex and continuous functions, $J_R$ is a convex and continuous function of the service times.

According to Lemma 2.8, when the global bottleneck machine's service time $s_{\max}$ falls in an interval $(\sigma_{(l-1)}, \sigma_{(l)}]$ for some $l \leq N$, the cost is $J_R = J_l(s)$. Therefore, for this case, the sensitivities of $J_R$ to service times (at differentiable points) can be written as

$$\frac{\partial J_R}{\partial s_j} = \begin{cases} \theta'_j(s_j) + \sum_{b=1}^{B_l} \sum_{i=k^b_l}^{n^b_l} \phi'_i\left(x^l_{i,M}\right), & s_j < s_{\max} \\ \theta'_j(s_{\max}) + \sum_{b=1}^{B_l} \sum_{i=k^b_l}^{n^b_l} \left[\phi'_i\left(x^l_{i,M}\right)(1 + i - k^b_l)\right], & \sigma_{(l)} > s_j > \max_{i \neq j} s_i \end{cases} \tag{2.66}$$

for $j = 1, ..., M$. Note that when $s_j = \sigma_{(l)}$, i.e., when the block structure at the global bottleneck machine is about to change, or when $s_j = \max_{i \neq j} s_i$, i.e., when

there are other machines with the maximum service time, non-differentiability is observed. For these points, we define the left derivatives as

$$\left(\frac{\partial J_R}{\partial s_j}\right)^- = \begin{cases} \theta_j'(s_{\max}) + \sum_{b=1}^{B_l} \sum_{i=k_l^b}^{n_l^b} \phi_i'\left(x_{i,M}^l\right), & s_j = \max_{i \neq j} s_i \\ \theta_j'(\sigma_{(l)}) + \sum_{b=1}^{B_l} \sum_{i=k_l^b}^{n_l^b} \left[\phi_i'\left(x_{i,M}^l\right)(1+i-k_l^b)\right], & \sigma_{(l)} = s_j > \max_{i \neq j} s_i \end{cases} \tag{2.67}$$

for $j = 1, ..., M$.

Since $J_R$ is continuous and convex, yet not everywhere differentiable, we define the subgradients as the left derivative vector $\xi$ with components

$$\xi_j = \left(\frac{\partial J_R}{\partial s_j}\right)^-$$

for all $j = 1, ..., M$. The subgradient directions drive the following descent algorithm with projections, which runs until the stopping condition determined by an $\epsilon$ termination tolerance and a $d$ distance metric is satisfied:

**Algorithm 2.1** *Step 0: Start with an arbitrary initial solution* $s^0 = (s_1^0, ..., s_M^0)$.

*Repeat for* $k = 1, 2, ...$

*Step 1: Determine the global bottleneck machine* $m = \min\{v : s_v^{k-1} = \max_{j=1,...,M} s_j^{k-1}\}$.

*Step 2: Determine the block structure at the global bottleneck machine* $m$ *employing Lemma 2.5.*

*Step 3: Determine* $\xi_j^{k-1}$ *for all* $j = 1, ..., M$.

*Step 4: Update solution*

$$s^k = \Pi\left[s^{k-1} - \eta_k \xi^{k-1}\right]. \tag{2.68}$$

*Until* $d(s^k, s^{k-1}) < \epsilon$.

In (2.68), step sizes $\{\eta_k\}_{k=1}^{\infty}$ satisfy the standard conditions

$$\sum_{k=1}^{\infty} \eta_k^2 < \infty, \sum_{k=1}^{\infty} \eta_k = \infty$$

and $\Pi$ denotes the projection mapping onto the feasible solutions set $\{(s_1, ..., s_M) : s_1 \geq S_1, ..., s_M \geq S_M\}$. Subgradient descent algorithms with projections are known to converge to the optimal solution (see, e.g. in [2].) The computational complexity per iteration is given as $O(\max(M, N))$, i.e., the computational complexity per iteration is linear in both $M$ and $N$.

In the next section, we build on the results from this section and analyze the special case where the lower bounds for the services times are set to zero and the service costs are inversely proportional to the service times.

## 2.5 Two-Phase Search Algorithm

Most of the studies in the literature assumed the service costs to be decreasing linear functions of service times. This linearity assumption, however, fails to reflect the *law of diminishing marginal returns*: productivity increases at a decreasing rate with the amount of resource employed. Therefore, in this section, to assure the *law of diminishing marginal returns*, we employ the service cost function $\theta_j(\cdot)$ on machine $j$ defined as

$$\theta_j(s_j) = \frac{\beta_j}{s_j^{\kappa}}, \tag{2.69}$$

where $\beta_j$ is a positive parameter, $s_j$ is the service time at machine $j$, and $\kappa$ is a positive constant. This cost structure was shown to correspond to many industrial operations in [33].

Note that the service cost function given in (2.69) is strictly convex. Hence, to make the analyses more general, including this nice property, we modify the Assumption 2.1 in the following form:

**Assumption 2.3** : *$\theta_j(\cdot)$, for $j = 1, ..., M$, is continuously differentiable,*

*monotonically decreasing and strictly convex.*

In addition to the job completion deadline constraints, we relax the lower bounds on service times, i.e. , $s_j \geq 0$ ($S_j = 0$) for $j = 1, ..., M$, and recall the convex optimization formulation $R$ in (2.65) as

$$R: \min_{\substack{s_j \geq 0 \\ j=1,...,M}} \left\{ J_R(s) = \max_{k=1,...,N} \{J_k(s)\} \right\}, \tag{2.70}$$

where $J_k(s)$ is given in (2.61).

Let us define $r_i(k)$, the index of the job that starts the block in which job $C_i$ resides for the $k$th block structure, as

$$r_i(k) = \max \left\{ j : \sigma_j \geq \sigma_{(k)}, j \leq i \right\} \tag{2.71}$$

for all $i = 1, ..., N$ and $k = 1, ..., N$. Then, the completion time of job $C_i$ for the $k$th block structure $y_i^k(s)$ can be defined as

$$y_i^k(s) = a_{r_i(k)} + s_{total} + (i - r_i(k))\, s_{\max} \tag{2.72}$$

for all $i = 1, ..., N$ where $s_{\max} = \max_{j=1,...,M} s_j$ and $s_{total} = \sum_{j=1}^{M} s_j$. As an alternative representation in (2.61), we employ (2.72) and rewrite $J_k(s)$ as

$$J_k(s) = \sum_{j=1}^{M} \theta_j(s_j) + \sum_{i=1}^{N} \phi_i \left( y_i^k(s) \right). \tag{2.73}$$

Since $J_R(s)$ is continuous, it follows from (2.70) and Lemma 2.8 that $J_R(s) = J_k(s)$ when $s_{\max} \in [\sigma_{(k-1)}, \sigma_{(k)}]$.

Another important property of the cost functions $\{J_k\}_{k=1}^{N}$ is strict convexity, which is presented in the next theorem.

**Theorem 2.1** *The cost function $J_k(s)$, for $k = 1, ..., N$, is strictly convex on the service times vector $s = (s_1, ..., s_M)$.*

**Proof.** Let us define two distinct feasible solutions $s^1$ and $s^2$ such that

$$s^l = (s_1^l, ..., s_M^l)$$

for $l = 1, 2$. For some $\lambda \in (0, 1)$, let $s^3$ be

$$s^3 = \lambda s^1 + (1 - \lambda)s^2. \tag{2.74}$$

To have strict convexity, it suffices to show that the strict inequality

$$\lambda J_k(s^1) + (1 - \lambda)J_k(s^2) > J_k\left(\lambda s^1 + (1 - \lambda)s^2\right) = J_k(s^3) \tag{2.75}$$

holds.

Due to strict convexity of $\theta_j(\cdot)$ by Assumption 2.3, we have

$$\sum_{j=1}^{M} \left[\lambda\theta_j(s_j^1) + (1 - \lambda)\theta_j(s_j^2)\right] > \sum_{j=1}^{M} \theta_j\left(\lambda s_j^1 + (1 - \lambda)s_j^2\right) = \sum_{j=1}^{M} \theta_j(s_j^3). \tag{2.76}$$

From (2.74), we have

$$\begin{aligned} s_{\max}^3 &\leq \lambda s_{\max}^1 + (1 - \lambda)s_{\max}^2, \\ s_{total}^3 &= \lambda s_{total}^1 + (1 - \lambda)s_{total}^2. \end{aligned}$$

Therefore, we obtain

$$\begin{aligned} \lambda y_i^k(s^1) + (1 - \lambda)y_i^k(s^2) &= a_{r_i(k)} + \left(\lambda s_{total}^1 + (1 - \lambda)s_{total}^2\right) \\ &\quad + (i - r_i(k))\left(\lambda s_{\max}^1 + (1 - \lambda)s_{\max}^2\right) \\ &\geq a_{r_i(k)} + s_{total}^3 + (i - r_i(k))\, s_{\max}^3 = y_i^k(s^3). \end{aligned} \tag{2.77}$$

Since $\phi_i$ is monotonically increasing and convex, we have from (2.77) that

$$\lambda\phi_i\left(y_i^k(s^1)\right) + (1 - \lambda)\phi_i\left(y_i^k(s^2)\right) \geq \phi_i\left(\lambda y_i^k(s^1) + (1 - \lambda)y_i^k(s^2)\right) \geq \phi_i\left(y_i^k(s^3)\right)$$

for all $i = 1, ..., N$. Hence, we have

$$\sum_{i=1}^{N} \left[\lambda\phi_i\left(y_i^k(s^1)\right) + (1-\lambda)\phi_i\left(y_i^k(s^2)\right)\right] \geq \sum_{i=1}^{N} \phi_i\left(y_i^k(s^3)\right). \tag{2.78}$$

The inequalities (2.76) and (2.78) imply (2.75) resulting with that $J_k$ is strictly convex. ■

Since $J_R$ is the maximum of $J_k$'s, the following corollary follows from Theorem 2.1:

**Corollary 2.1** *$J_R(s)$ is strictly convex function on the service times vector $s = (s_1, ..., s_M)$.*

Since $\{J_k\}_{k=1}^N$ and $J_R$ are strictly convex, they have unique minimizers. The search algorithm that we construct requires us to determine the minimizers of $\{J_k\}_{k=1}^N$. Hence, we first develop an efficient method to determine these unique minimizers.

### 2.5.1 Determining the Minimizers of $\{J_k\}_{k=1}^N$ Functions

Let $\{s^k\}_{k=1}^N$ be the unique minimizers of $\{J_k\}_{k=1}^N$. Note that, by Assumption 2.2, we have

$$\lim_{s_j \to \infty} \phi_i\left(y_i^k(s)\right) = \infty \tag{2.79}$$

for all $j = 1, ..., M$; therefore, the unique minimizers are finite. Moreover, since the cost structure in (2.69) satisfies

$$\lim_{s_j \to 0^+} \theta_j(s_j) = \lim_{s_j \to 0^+} \frac{\beta_j}{s_j^\kappa} = \infty \tag{2.80}$$

for all $j = 1, ..., M$, then $s_j^k > 0$.

In the next lemma, we state that there exists an ordering among the optimal service times $s_j^k$ determined by the $\beta_j$ values.

**Lemma 2.9** *For any two machines $u$ and $v$, if $\beta_u \geq \beta_v$ then $s_u^k \geq s_v^k$.*

**Proof.** For a contradiction, let us assume that while $\beta_u \geq \beta_v$, the optimal service times satisfy

$$s_u^k < s_v^k \tag{2.81}$$

and define the perturbed service times $\bar{s}_j$ for $j = 1, ..., M$ as

$$\bar{s}_j = \begin{cases} s_u^k + \Delta, & j = u \\ s_v^k - \Delta, & j = v \\ s_j^k, & \text{otherwise} \end{cases} \tag{2.82}$$

with $0 < \Delta \leq \frac{s_v^k - s_u^k}{2}$. Note that, from (2.81) and (2.82), we have $\max\left(s_u^k, s_v^k\right) > \max\left(\bar{s}_u, \bar{s}_v\right)$; therefore we can write

$$s_{\max}^k \geq \bar{s}_{\max} \tag{2.83}$$

and

$$s_{total}^k = \bar{s}_{total}. \tag{2.84}$$

Then, from (2.72), (2.83), and (2.84), we have

$$y_i^k(\bar{s}) \leq y_i^k(s^k) \tag{2.85}$$

for all $i = 1, ..., N$.

Moreover, since $\beta_u \geq \beta_v$, then the inequality

$$\theta_u'(s) \leq \theta_v'(s) \tag{2.86}$$

is satisfied for all $s$.

If we denote the cost of the perturbed solution as $\bar{J}_k$ and the cost of the unique minimizer $s^k$ as $J_k^*$, by Assumptions 2.3 and 2.2, and from (2.81), (2.82), (2.85),

and (2.86), we have

$$\bar{J}_k - J_k^* = \theta_u(s_u^k+\Delta) - \theta_u(s_u^k) - \theta_v(s_v^k) + \theta_v(s_v^k-\Delta) + \sum_{i=1}^{N} \left[\phi_i\left(y_i^k(\bar{s})\right) - \phi_i\left(y_i^k(s^k)\right)\right] < 0,$$

which contradicts the optimality assumption and concludes the proof. ■

For any service vector $s = (s_1, ..., s_M)$, the sensitivities for the cost function $J_k$ are given as

$$\frac{\partial J_k}{\partial s_j} = \begin{cases} \theta_j'(s_j) + \sum_{i=1}^{N} \phi_i'\left(y_i^k(s)\right), & s_j < s_{\max} \\ \theta_j'(s_j) + \sum_{i=1}^{N} \left[\phi_i'\left(y_i^k(s)\right)(1 + i - r_i(k))\right], & s_j > \max_{i\neq j} s_i \end{cases} \tag{2.87}$$

for $j = 1, ..., M$. Note that when $s_j = \max_{i\neq j} s_i$, i.e., when there are other machines with the maximum service time, non-differentiability is observed. In order to come up with a differentiable subproblem, we define a cost $J_k^\beta$ as

$$J_k^\beta(s) = \sum_{j\in I_\beta} \theta_j(s_m) + \sum_{j\notin I_\beta} \theta_j(s_j) + \sum_{i=1}^{N} \phi_i\left(y_i^k(s,\beta)\right), \tag{2.88}$$

where $I_\beta$ is the set $\{i : \beta_i \geq \beta\}$ with cardinality $K_\beta$, $s_m$ is the service time of machine $m$ with $\beta_m = \beta_{\max}$, and $y_i^k(s,\beta)$ is

$$y_i^k(s,\beta) = a_{r_i(k)} + (K_\beta + i - r_i(k))\, s_m + \sum_{j\notin I_\beta} s_j. \tag{2.89}$$

Then, we define a family of subproblems $Q_k^\beta$ as

$$Q_k^\beta : \min_s J_k^\beta(s)$$

subject to

$$\begin{aligned} s_j &= s_m \text{ for } j \in I_\beta \backslash \{m\} \\ s_j &\geq 0 \text{ for } j \in \{1, ..., M\} \end{aligned}$$

A specific member of this family, $Q_k^{\beta_k^*}$, will be of interest to us where $\beta_k^*$ is defined in the next lemma.

**Lemma 2.10** *There exists a* $\beta_k^* \in \{\beta_1, ..., \beta_M\}$ *for which* $I_{\beta_k^*} = \left\{i : s_i^k = \max_{j=1,...,M} s_j^k\right\}$

**Proof.** Let $I = \left\{i : s_i^k = \max_{j=1,...,M} s_j^k\right\}$ and machine $v \in I$ satisfy $\beta_v = \min_{i \in I} \beta_i$. It follows from Lemma 2.9 that if, for some machine $u$, $\beta_u \geq \beta_v$, then $s_u^k = \max_{j=1,...,M} s_j^k$, i.e., $u \in I$. Therefore $I_{\beta_v} = I$, i.e., $\beta_k^* = \beta_v$. ■

Note that in Lemma 2.10, we showed the existence of the $\beta_k^*$ but it is value was defined over $s^k$ which is not available. In fact, we will solve $Q_k^{\beta_k^*}$ to determine $s^k$ values as suggested by the next theorem. Determination of $\beta_k^*$, without knowing $s^k$, is covered in Subsection 2.5.1.1.

**Theorem 2.2** *The minimizer* $s^k$ *of* $J_k$ *is also the optimal solution for* $Q_k^{\beta_k^*}$.

**Proof.** Since $s^k$ is the minimizer of $J_k$, by Lemma 2.9, it is also the optimal solution for the problem

$$\min_s J_k$$

subject to

$$\begin{aligned} s_j &= s_m \text{ for } j \in I_{\beta_k^*} \backslash \{m\} \\ s_j &\geq 0 \text{ for } j \in \{1, ..., M\} \end{aligned}$$

where $\beta_m = \beta_{\max}$. From (2.73) and (2.88), $J_k(s) = J_k^{\beta_k^*}(s)$ for all the feasible points of this problem. Hence $s^k$ is the optimal solution for $Q_k^{\beta_k^*}$. ■

It follows from this theorem that we can work with the differentiable problem $Q_k^{\beta_k^*}$ to determine $s^k$. The $\beta_k^*$ value is to be determined, iteratively, along with $s^k$ as presented next.

#### 2.5.1.1 Determining $\beta_k^*$

The following theorem establishes that $\beta_k^*$ value can be determined by a one directional search.

**Theorem 2.3** *If* $\beta > \beta_k^*$, *then the optimal solution* $s^\beta$ *of* $Q_k^\beta$ *satisfies* $\max_{j \notin I_\beta} s_j^\beta \geq s_m^\beta$ *where* $\beta_m = \beta_{\max}$.

**Proof.** For a contradiction, assume that $s_j^\beta < s_m^\beta$ is satisfied for all $j \notin I_\beta$ while $\beta > \beta_k^*$. Since $s_{\max}^k = s_m^k$ and $s_{\max}^\beta = s_m^\beta$, then, from (2.73) and (2.88), we have

$$J_k(s^\beta) = J_k^\beta(s^\beta), \tag{2.90}$$

$$J_k(s^k) = J_k^\beta(s^k). \tag{2.91}$$

Note that, for some machine $u$ with $\beta > \beta_u \geq \beta_k^*$, i.e., $u \in I_{\beta_k^*} \backslash I_\beta$, we have $s_u^k = s_m^k = s_{\max}^k$ and $s_u^\beta < s_m^\beta = s_{\max}^\beta$, therefore, $s^k \neq s^\beta$. Since $s^k$ is the unique minimizer of $J_k$, then we have

$$J_k(s^\beta) > J_k(s^k). \tag{2.92}$$

It follows from (2.90), (2.91), and (2.92) that

$$J_k^\beta(s^\beta) > J_k^\beta(s^k),$$

which contradicts with the optimality of $s^\beta$. Hence the result follows. ■

In our search for the $\beta_k^*$ value, we start with $\beta = \beta_{\max}$ and solve $Q_k^\beta$ to check the condition in Theorem 2.3. If the optimal solution $s^\beta$ satisfies $\max_{j \notin I_\beta} s_j^\beta \geq s_m^\beta$ where $\beta_m = \beta_{\max}$, then we lower the $\beta$ value to $\max_{j \notin I_\beta} \beta_j$, the largest element of the set $\{\beta_1, ..., \beta_M\}$ smaller than $\beta$, and continue. At each step, in order to solve $Q_k^\beta$, we minimize the augmented cost defined as

$$\bar{J}_k^\beta(s) = J_k^\beta(s) + \sum_{j \in I_\beta \backslash \{m\}} \lambda_j (s_j - s_m). \tag{2.93}$$

Note that, by Assumption 2.2, we have

$$\lim_{s_j \to \infty} \phi_i \left( y_i^k(s, \beta) \right) = \infty \tag{2.94}$$

for all $j = 1, ..., M$, therefore, the optimal service times $s_j^\beta$ of $Q_k^\beta$ are finite. Moreover, from (2.80), $s_j^\beta > 0$ should be satisfied for all $j = 1, ..., M$. Hence, we have

$$\left.\frac{\partial \bar{J}_k^\beta}{\partial s_j}\right|_{s_j = s_j^\beta} = 0 \tag{2.95}$$

for all $j = 1, ..., M$. It follows from (2.88), (2.93), and (2.95) that

$$\begin{array}{cc} \theta_j'(s_j^\beta) + \sum_{i=1}^N \phi_i'\left(y_i^k(s^\beta, \beta)\right) = 0, & j \notin I_\beta \\ \sum_{j \in I_\beta} \theta_j'(s_m^\beta) + \sum_{i=1}^N \left[\phi_i'\left(y_i^k(s^\beta, \beta)\right)(K_\beta + i - r_i(k))\right] = 0, & j = m \\ \lambda_j = 0, & j \in I_\beta \backslash \{m\}. \end{array} \tag{2.96}$$

For the cost structure in (2.69), the first equality in (2.96) suggests that we can pick an arbitrary machine $u \notin I_\beta$ and write

$$s_v^\beta = c_{u,v} s_u^\beta \tag{2.97}$$

for all machines $v \notin I_\beta$ where $c_{u,v} = \left(\frac{\beta_v}{\beta_u}\right)^{1/(\kappa+1)}$. Employing (2.97) in (2.96), we end up with the following two nonlinear equations with two unknowns $s_m^\beta$ and $s_u^\beta$:

$$\theta_u'(s_u^\beta) + \sum_{i=1}^N \phi_i'\left(\bar{y}_i^k(s_m^\beta, s_u^\beta, \beta)\right) = 0, \tag{2.98}$$

$$\sum_{j \in I_\beta} \theta_j'(s_m^\beta) + \sum_{i=1}^N \left[\phi_i'\left(\bar{y}_i^k(s_m^\beta, s_u^\beta, \beta)\right)(K_\beta + i - r_i(k))\right] = 0, \tag{2.99}$$

where $\bar{y}_i^k(s_m^\beta, s_u^\beta, \beta)$ is defined as

$$\bar{y}_i^k(s_m^\beta, s_u^\beta, \beta) = a_{r_i(k)} + (K_\beta + i - r_i(k)) s_m^\beta + \sum_{j \notin I_\beta} c_{u,j} s_u^\beta \tag{2.100}$$

for all $i = 1, ..., N$.

Note that, independent of $M$, there are only two unknowns $s_m^\beta$ and $s_u^\beta$ in the equations (2.98) and (2.99). This system can be solved easily by well-known solution techniques (e.g. Trust-Region methods [11] can be utilized to solve the

system of nonlinear equations.)

The search algorithm results with a $\beta_k^*$ value along with the solution $s^k$ of $J_k$. We next describe how to determine $s^*$ of $R$ employing these solutions.

### 2.5.2 Locating the Optimal Solution of $J_R$

Now that we have established an efficient method to determine the unique minimizers $s^k$ of $\{J_k\}_{k=1}^N$, in the next theorem, we relate the unique optimal solution $s^*$ of $R$ to these minimizers.

**Theorem 2.4** *Let $s^k$ and $s^*$ be the unique minimizers of $J_k(s)$ and $J_R(s)$, respectively.*

*i*) If $s^k_{\max} \in [\sigma_{(k-1)}, \sigma_{(k)}]$, then $s^* = s^k$;

*ii*) if $s^k_{\max} > \sigma_{(k)}$, then $s^*_{\max} \geq \sigma_{(k)}$;

*iii*) if $s^k_{\max} < \sigma_{(k-1)}$, then $s^*_{\max} \leq \sigma_{(k-1)}$.

**Proof.** *i*) If $s^k_{\max} \in [\sigma_{(k-1)}, \sigma_{(k)}]$, then, by Lemma 2.8, $J_k(s^k) = J_R(s^k)$. Since $s^k$ minimizes $J_k$, we have

$$J_R(s^k) = J_k(s^k) \leq J_k(s) \leq \max_{t=1,\ldots,N} J_t(s) = J_R(s)$$

for all $s$, i.e., $s^k$ is the optimal solution of $R$.

*ii*) For a contradiction, let us assume that $s^k_{\max} > \sigma_{(k)}$ and $s^*_{\max} < \sigma_{(k)}$. Hence, we can define a nonempty set $I_1$ as

$$I_1 = \{i : s_i^k > \sigma_{(k)}\}.$$

Let us define an alternate solution $\bar{s}$ as

$$\bar{s} = s^* + \gamma_1(s^k - s^*),$$

where the constant $\gamma_1$ is defined as

$$\gamma_1 = \min_{j \in I_1} \left\{ \frac{\sigma_{(k)} - s_j^*}{s_j^k - s_j^*} \right\},$$

so that $\bar{s}_{\max} = \sigma_{(k)}$. Then, by Lemma 2.8 and Theorem 2.1, we have

$$J_k(s^k) < J_k(\bar{s}) = J_R(\bar{s}) < J_k(s^*) \leq \max_{t=1,\ldots,N} J_t(s^*) = J_R(s^*),$$

which contradicts the optimality of $s^*$ for $J_R$. Hence, the optimal solution for $R$ satisfies $s^*_{\max} \geq \sigma_{(k)}$.

*iii*) For a contradiction, let us assume that $s^k_{\max} < \sigma_{(k-1)}$ and $s^*_{\max} > \sigma_{(k-1)}$. Hence, we can define a nonempty set $I_2$ as

$$I_2 = \{i : s_i^* > \sigma_{(k-1)}\}.$$

Let us define an alternate solution $\hat{s}$ as

$$\hat{s} = s^k + \gamma_2(s^* - s^k),$$

where the constant $\gamma_2$ is defined as

$$\gamma_2 = \min_{j \in I_2} \left\{ \frac{\sigma_{(k-1)} - s_j^k}{s_j^* - s_j^k} \right\},$$

so that $\hat{s}_{\max} = \sigma_{(k-1)}$. Then, by Lemma 2.8 and Theorem 2.1, we have

$$J_k(s^k) < J_k(\hat{s}) = J_R(\hat{s}) < J_k(s^*) \leq \max_{t=1,\ldots,N} J_t(s^*) = J_R(s^*),$$

which contradicts the optimality of $s^*$ for $J_R$. Hence, the optimal solution for $R$ satisfies $s^*_{\max} \leq \sigma_{(k-1)}$. ■

The direct corollary of Theorem 2.4 is the following:

**Corollary 2.2** *Let $s^k$ be the unique minimizer of $J_k(s)$.*

*i) If* $s^k_{\max} > \sigma_{(k)}$, *then* $s^l_{\max} \notin [\sigma_{(l-1)}, \sigma_{(l)}]$ *for all* $l = 1, ..., k$;

*ii) if* $s^k_{\max} < \sigma_{(k-1)}$, *then* $s^l_{\max} \notin [\sigma_{(l-1)}, \sigma_{(l)}]$ *for all* $l = k, ..., N$.

Theorem 2.4 and Corollary 2.2 indicate that $s^k$, the minimizer of $J_k$, limits the search space for the optimal solution $s^*$. Motivated by this result, we develop a search algorithm that operates into two phases: In Phase 1, we search for a $J_k$ whose minimizer satisfies $s^k_{\max} \in [\sigma_{(k-1)}, \sigma_{(k)}]$. This phase can yield two different results: If $s^l_{\max} \in [\sigma_{(l-1)}, \sigma_{(l)}]$ for some $l = 1, ...., N$, then it will reach the unique optimal solution. If, on the other hand, $s^l_{\max} \notin [\sigma_{(l-1)}, \sigma_{(l)}]$ for all $l = 1, ...., N$, then it will yield a $k \in \{1, ..., N-1\}$ satisfying

$$s^k_{\max} > \sigma_{(k)} > s^{k+1}_{\max}.$$

In this case, from Theorem 2.4, we conclude that $s^*_{\max} = \sigma_{(k)}$ and proceed to Phase 2, which searches for the optimal solution at points that satisfy $s_{\max} = \sigma_{(k)}$.

Exploiting the result of Lemma 2.6 on completion times for the flow shop systems with fixed service times, we state that there exists an ordering among the optimal service times $s^*_j$ determined by the $\beta_j$ values in the next lemma.

**Lemma 2.11** *For any two machines* $u$ *and* $v$, *if* $\beta_u \geq \beta_v$ *then* $s^*_u \geq s^*_v$.

**Proof.** For a contradiction, let us assume that while $\beta_u \geq \beta_v$, the optimal service times satisfy

$$s^*_u < s^*_v. \tag{2.101}$$

We define perturbed service times $\bar{s}_j$ for $j = 1, ..., M$ as

$$\bar{s}_j = \begin{cases} s^*_u + \Delta, & j = u \\ s^*_v - \Delta, & j = v \\ s^*_j, & \text{otherwise} \end{cases} \tag{2.102}$$

with $0 < \Delta \leq \frac{s_v^* - s_u^*}{2}$. Note that, from (2.101) and (2.102), we have $\max(s_u^*, s_v^*) > \max(\bar{s}_u, \bar{s}_v)$; therefore we can write

$$s_{\max}^* \geq \bar{s}_{\max} \tag{2.103}$$

and

$$s_{total}^* = \bar{s}_{total}. \tag{2.104}$$

Next, we show by induction that the completion times of jobs for the perturbed solution do not exceed the completion times for the optimal solution, i.e., $\bar{x}_{i,M} \leq x_{i,M}^*$ for all $i = 1, ..., N$. As the basis step, we have from (2.2), (2.104), and by Lemma 2.6 that

$$\bar{x}_{1,M} = a_1 + \bar{s}_{total} = a_1 + s_{total}^* = x_{1,M}^*.$$

Now, assume that

$$\bar{x}_{i-1,M} \leq x_{i-1,M}^* \tag{2.105}$$

for some $i \in \{2, ..., M\}$. Then, from (2.103), (2.104), (2.105), and by Lemma 2.6, we have

$$\begin{aligned} \bar{x}_{i,M} &= \max(a_i + \bar{s}_{total}, \bar{x}_{i-1,M} + \bar{s}_{\max}) \\ &\leq \max\left(a_i + s_{total}^*, x_{i-1,M}^* + s_{\max}^*\right) = x_{i,M}^*. \end{aligned}$$

Hence, in the perturbed system, the completion times satisfy

$$\bar{x}_{i,M} \leq x_{i,M}^* \tag{2.106}$$

for all $i = 1, ..., N$.

Moreover, since $\beta_u \geq \beta_v$, then the inequality

$$\theta_u'(s) \leq \theta_v'(s) \tag{2.107}$$

is satisfied for all $s$.

If we denote the cost of the perturbed solution as $\bar{J}$ and the cost of the optimal solution as $J^*$, by Assumptions 2.3 and 2.2, and from (2.101), (2.102), (2.106),

(2.107), we have

$$\bar{J}-J^* = \theta_u(s_u^*+\Delta)-\theta_u(s_u^*)-\theta_v(s_v^*)+\theta_v(s_v^*-\Delta)+\sum_{i=1}^{N}\left[\phi_i\left(\bar{x}_{i,M}\right)-\phi_i\left(x_{i,M}^*\right)\right]<0,$$

which contradicts the optimality assumption and concludes the proof. ■

Since $J_R = J_k$ when $s_{\max} = \sigma_{(k)}$, in Phase 2, we solve a family of subproblems $\hat{Q}_k^{\beta}$ defined as

$$\hat{Q}_k^{\beta} : \min_s J_k^{\beta}(s)$$

subject to

$$\begin{aligned} s_j &= s_m \text{ for } j \in I_\beta \backslash \{m\} \\ s_m &= \sigma_{(k)} \\ s_j &\geq 0 \text{ for } j \in \{1,...,M\} \end{aligned}$$

to determine the optimal solution $s^*$.

A specific member of this family, $\hat{Q}_k^{\hat{\beta}_k^*}$, will be of interest to us where $\hat{\beta}_k^*$ is as presented in the next lemma.

**Lemma 2.12** *If $s_{\max}^* = \sigma_{(k)}$, then there exists a $\hat{\beta}_k^* \in \{\beta_1,...,\beta_M\}$ for which $I_{\hat{\beta}_k^*} = \left\{i : s_i^* = \sigma_{(k)}\right\}$*

**Proof.** Let $I = \left\{i : s_i^* = \sigma_{(k)}\right\}$ and machine $v \in I$ satisfy $\beta_v = \min_{i\in I}\beta_i$. It follows from Lemma 2.11 that if, for some machine $u$, $\beta_u \geq \beta_v$, then

$$s_{\max}^* = \sigma_{(k)} \geq s_u^* \geq s_v^* = \sigma_{(k)},$$

i.e., $u \in I$. Therefore $I_{\beta_v} = I$, i.e., $\hat{\beta}_k^* = \beta_v$. ■

We showed the existence of the $\hat{\beta}_k^*$ in Lemma 2.12, but its value was defined over $s^*$ which is not available. In fact, we will solve $\hat{Q}_k^{\hat{\beta}_k^*}$ to determine $s^*$ values as suggested by the next theorem. Determination of $\hat{\beta}_k^*$, without knowing $s^*$ is covered in Subsection 2.5.2.1.

**Theorem 2.5** *If $s^*_{\max} = \sigma_{(k)}$, then the minimizer $s^*$ of $J_R$ is also the optimal solution for $\hat{Q}_k^{\hat{\beta}^*_k}$.*

**Proof.** Since $s^*$ satisfies $s^*_i = \sigma_{(k)}$ for all $i \in I_{\hat{\beta}^*_k}$, it is also the optimal solution for the problem

$$\min_s J_R$$

subject to

$$\begin{aligned} s_j &= s_m \text{ for } j \in I_{\hat{\beta}^*_k} \backslash \{m\} \\ s_m &= \sigma_{(k)} \\ s_j &\geq 0 \text{ for } j \in \{1, ..., M\}, \end{aligned}$$

where $\beta_m = \beta_{\max}$. Since $J_R = J_k$ when $s_{\max} = \sigma_{(k)}$, from (2.73) and (2.88), $J_R(s) = J_k(s) = J_k^{\hat{\beta}^*_k}(s)$ for all the feasible points of this problem. Hence, the result follows. ■

It follows from this theorem that if $s^*_{\max} = \sigma_{(k)}$, we can solve the differentiable problem $\hat{Q}_k^{\hat{\beta}^*_k}$ to determine $s^*$. The determination of $\hat{\beta}^*_k$ is presented next.

#### 2.5.2.1 Determining $\hat{\beta}^*_k$

Similar to $\beta^*_k$ , the $\hat{\beta}^*_k$ value can be determined by a one directional search motivated by the following theorem.

**Theorem 2.6** *Given that $s^*_{\max} = \sigma_{(k)}$, if $\beta > \hat{\beta}^*_k$, then the optimal solution $\hat{s}^\beta$ of $\hat{Q}_k^\beta$ satisfies $\max_{j \notin I_\beta} \hat{s}^\beta_j \geq \hat{s}^\beta_m$ where $\beta_m = \beta_{\max}$.*

**Proof.** For a contradiction, assume that $\hat{s}^\beta_j < \hat{s}^\beta_m$ is satisfied for all $j \notin I_\beta$ while $\beta > \hat{\beta}^*_k$. Since $s^*_{\max} = s^*_m = \sigma_{(k)}$ and $\hat{s}^\beta_{\max} = \hat{s}^\beta_m = \sigma_{(k)}$, then $J_R = J_k$ and from (2.73) and (2.88), we have

$$J_R(\hat{s}^\beta) = J_k(\hat{s}^\beta) = J_k^\beta(\hat{s}^\beta), \tag{2.108}$$

$$J_R(s^*) = J_k(s^*) = J_k^\beta(s^*). \tag{2.109}$$

Note that, for some machine $u$ with $\beta > \beta_u \geq \hat{\beta}_k^*$, i.e., $u \in I_{\hat{\beta}_k^*} \backslash I_\beta$, we have $s_u^* = s_m^* = \sigma_{(k)}$ and $\hat{s}_u^\beta < \hat{s}_m^\beta = \sigma_{(k)}$; therefore $s^* \neq \hat{s}^\beta$. Since $s^*$ is the unique minimizer of $J_R$, then we have

$$J_R(\hat{s}^\beta) > J_R(s^*). \tag{2.110}$$

It follows from (2.108), (2.109), and (2.110) that

$$J_k^\beta(\hat{s}^\beta) > J_k^\beta(s^*),$$

which contradicts the optimality of $\hat{s}^\beta$. Hence the result follows. ■

In our search for the $\hat{\beta}_k^*$ value, we start with $\beta = \beta_{\max}$ and solve $\hat{Q}_k^\beta$ to check the condition in Theorem 2.6. If the optimal solution $\hat{s}^\beta$ satisfies $\max_{j \notin I_\beta} \hat{s}_j^\beta \geq \hat{s}_m^\beta$ where $\beta_m = \beta_{\max}$, then we lower the $\beta$ value to $\max_{j \notin I_\beta} \beta_j$, the largest element of the set $\{\beta_1, ..., \beta_M\}$ smaller than $\beta$, and continue. At each step, in order to solve $\hat{Q}_k^\beta$, we minimize the augmented cost function $\hat{J}_k^\beta$ defined as

$$\hat{J}_k^\beta(s) = J_k^\beta(s) + \sum_{j \in I_\beta \backslash \{m\}} \lambda_j(s_j - s_m) + \lambda_m(s_m - \sigma_{(k)}). \tag{2.111}$$

From (2.80) and (2.94), the optimal service times $\hat{s}_j^\beta$ of $\hat{Q}_k^\beta$ are finite and positive for all $j = 1, ..., M$. Hence, we have

$$\left. \frac{\partial \hat{J}_k^\beta}{\partial s_j} \right|_{s_j = \hat{s}_j^\beta} = 0 \tag{2.112}$$

for all $j = 1, ..., M$. It follows from (2.88), (2.111), and (2.112) that

$$\begin{array}{ll} \theta_j'(\hat{s}_j^\beta) + \sum_{i=1}^N \phi_i' \left( y_i^k(\hat{s}^\beta, \beta) \right) = 0, & j \notin I_\beta \\ \sum_{j \in I_\beta} \theta_j'(\hat{s}_m^\beta) + \sum_{i=1}^N \left[ \phi_i' \left( y_i^k(\hat{s}^\beta, \beta) \right) (K_\beta + i - r_i(k)) \right] + \lambda_m = 0, & j = m \\ \lambda_j = 0, & j \in I_\beta \backslash \{m\}. \end{array} \tag{2.113}$$

For the cost structure in (2.69), the first equality (2.113) suggests that we can pick an arbitrary machine $u \notin I_\beta$ and write

$$\hat{s}_v^\beta = c_{u,v}\hat{s}_u^\beta \tag{2.114}$$

for all machines $v \notin I_\beta$ where $c_{u,v} = \left(\frac{\beta_v}{\beta_u}\right)^{1/(\kappa+1)}$. Setting $s_m = \sigma_{(k)}$ and employing (2.114) in (2.113), we end up with the following nonlinear equation with only one unknown $\hat{s}_u^\beta$:

$$\theta_u'(\hat{s}_u^\beta) + \sum_{i=1}^{N} \phi_i'\left(\bar{y}_i^k(\hat{s}_u^\beta, \beta, \sigma_{(k)})\right) = 0, \tag{2.115}$$

where $\bar{y}_i^k(\hat{s}_u^\beta, \beta, \sigma_{(k)})$ is defined as

$$\bar{y}_i^k(\hat{s}_u^\beta, \beta, \sigma_{(k)}) = a_{r_i(k)} + (K_\beta + i - r_i(k))\sigma_{(k)} + \sum_{j \notin I_\beta} c_{u,j}\hat{s}_u^\beta \tag{2.116}$$

for all $i = 1, ..., N$.

Note that, independent of $M$, there is only one unknown $\hat{s}_u^\beta$ in the equation (2.115) that can be solved easily by the utilization of Trust-Region methods.

### 2.5.3 The Algorithm

In light of above discussions, we develop a two-phase search algorithm as presented in Figure 2.1.

In the *Initialization* step, we first determine a machine $m$ with $\beta_m = \beta_{\max}$. Then, we determine $\sigma_i$ values for $i = 1, ..., N$ by employing (2.27) and sort them in ascending order. Finally, in order to start at the middle interval, we set $k = \lceil (N+1)/2 \rceil$.

In Phase 1, we obtain the service times minimizing the cost function $J_k$ by solving a series of differentiable subproblems $\left\{Q_k^\beta\right\}_{\beta=\beta_k^*}^{\beta=\beta_{\max}}$ with the solution methodology described in Subsection 2.5.1.1. If we obtain a solution $s_{\max}^{\beta_k^*} \in [\sigma_{(k-1)}, \sigma_{(k)}]$ for some $k = 1, ..., N$, by Theorem 2.4, we conclude that it is the optimal solution

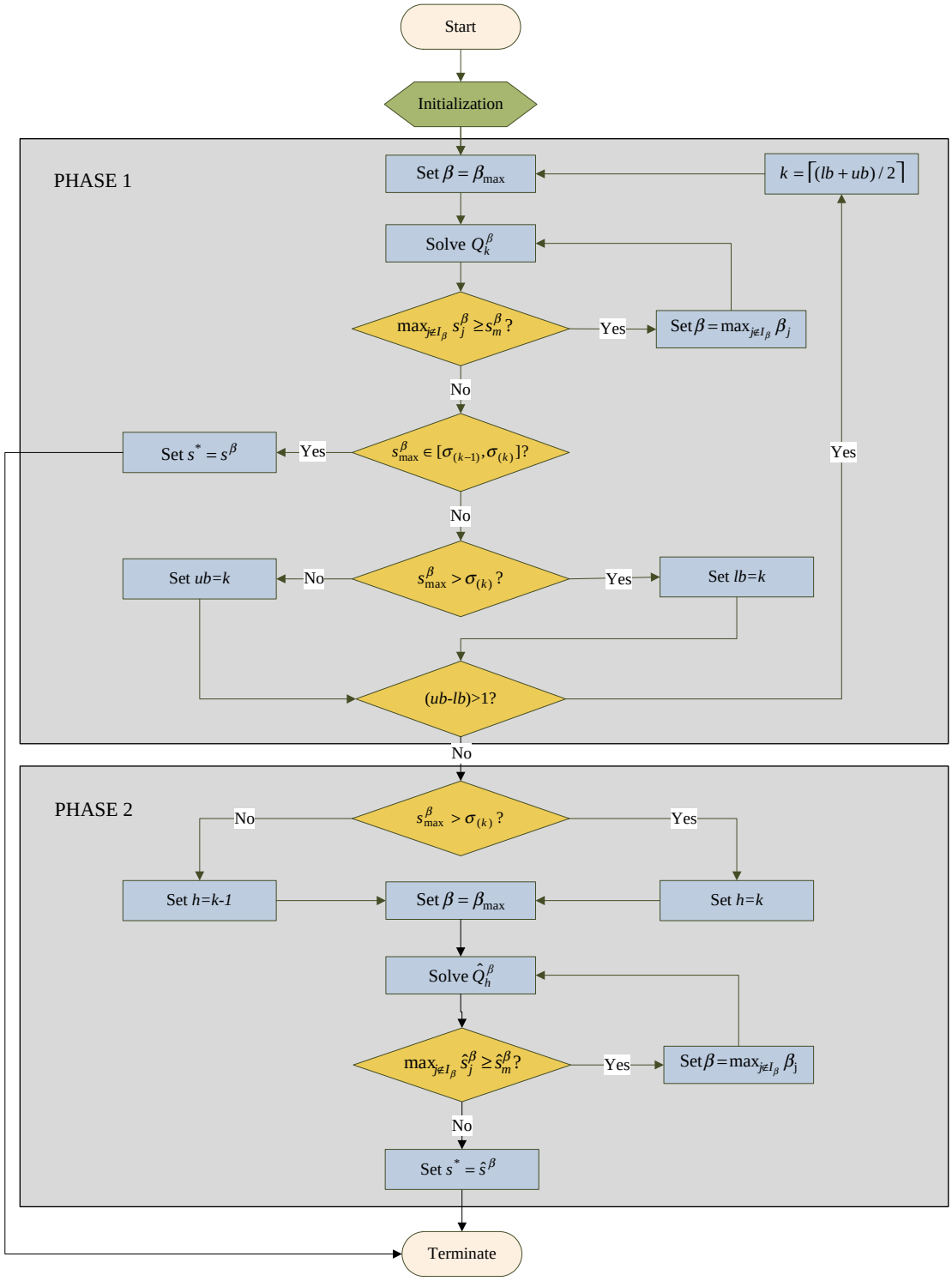


Figure 2.1: Flowchart of the Two-Phase Search Algorithm

of $J_R$. Otherwise, we proceed with Phase 2 and solve a series of differentiable subproblems $\left\{\hat{Q}_k^{\beta}\right\}_{\beta=\hat{\beta}_k^*}^{\beta=\beta_{\max}}$ with the solution methodology described in Subsection 2.5.2.1.

In the worst case, this algorithm solves $M\lceil \log_2 N \rceil$ of $Q_k^{\beta}$ problems involving two nonlinear equations and two unknowns, and $M$ of $\hat{Q}_k^{\beta}$ problems involving one nonlinear equation and one unknown. Hence, it determines the optimal solution in a finite number of steps.

## 2.6 Numerical Study

Let us consider an $M$-machine flow shop system processing an identical set of $N$ jobs. The service cost $\theta_j(s_j)$ at machine $j$ is given as

$$\theta_j(s_j) = \frac{\beta_j}{s_j} \tag{2.117}$$

for some constant $\beta_j$ where $\kappa$ in (2.69) is set to 1. The completion-time cost for job $C_i$, on the other hand, is given by a cost defined as

$$\phi_i(x_{i,M}) = 10(x_{i,M} - a_i)^2. \tag{2.118}$$

Note that the costs given by (2.117) and (2.118) satisfy Assumptions 2.1 (as well as 2.3) and 2.2 respectively.

### 2.6.1 Verification of the Waiting Characteristics

To illustrate the waiting characteristics of fixed service time flow systems derived in Section 2.2, we first consider the system, where ten jobs are to be processed in a flow shop of four machines, i.e., $N = 10$ and $M = 4$, with the costs given by (2.117) and (2.118). The arrival times of the jobs are given as $a = [0.0, 2.3, 2.4, 4.9, 5.0, 5.5, 9.0, 9.5, 11.0, 13.0]$, while the $\beta$ parameter vector in

(2.117) is given as $\beta = [100, 50, 200, 100]$. The service times on these four machines are bounded below by $S = [0.20, 0.20, 0.30, 0.35]$, while the due dates for job completions are set to infinity, i.e. , $d_i = \infty$ for all $i = 1, ..., 10$.

The optimal service times are found to be $s^* = [0.4942, 0.3495, 0.5593, 0.4942]$ yielding the optimal cost of 1329.01 units. Hence, for the optimal solution, the first machine, as in all fixed service time flow shop systems, turns out to be a local bottleneck, while the third machine is the global bottleneck, and there are no other local bottleneck machines. Therefore, the system can be divided into two flushing portions: one is formed of the first and the second machines, and the other is formed of the third and the fourth machines. From Lemma 2.1, we expect to see no waiting in front of the second and the fourth machines. Given the arrival times $a$, the minimums of the average interarrival times are evaluated as $\sigma = [\infty, 2.3, 0.1, 1.3, 0.1, 0.3, 1.34, 0.5, 1, 1.\bar{3}]$. From Lemma 2.3, we expect jobs $\{C_3, C_5, C_6\}$ to wait in front of the first machine, because $\sigma_3 = \sigma_5 \leq \sigma_6 < s_1^*$. They are also expected to wait in front of the third machine, the downstream local (and global) bottleneck machine, as stated in Lemma 2.2. Since $s_1^* \leq \sigma_8 < s_3^*$, job $C_8$ is expected to wait only in front of the third machine. Finally, since jobs $\{C_1, C_2, C_4, C_7, C_9, C_{10}\}$ are not expected to wait in front of the third (global bottleneck) machine because $s_3^* \leq \sigma_9 \leq \sigma_4 \leq \sigma_{10} \leq \sigma_7 \leq \sigma_2 \leq \sigma_1$, they are expected to observe no waiting in the system. The optimal departure times resulting from operating with optimal service times $s^*$ are given in Table 2.1 verifying the realization of expected waiting characteristics. (Note that boldface numbers denote waiting after departure.)

To observe its convergence behavior, the optimization problem $R$ is also solved for the above system with the subgradient descent algorithm. The service times are initially set to their lower bounds $s_j^0 = S_j$ for all $j = 1, ..., 4$. The termination tolerance $\epsilon$ is given to be $10^{-8}$ and the step size at iteration $k$ is given as $\eta_k = \frac{0.002}{k}$. The implemented distance metric is given as

$$d\left(s^k, s^{k-1}\right) = \max_{j=1,...,M} \left| s_j^k - s_j^{k-1} \right|. \tag{2.119}$$

The service times in the first twenty iterations are shown in Figure 2.2.

| | **Arrival** | **Machine 1** | **Machine 2** | **Machine 3** | **Machine 4** |
|---|---|---|---|---|---|
| **Job 1** | 0.0000 | 0.4942 | 0.8437 | 1.4029 | 1.8972 |
| **Job 2** | 2.3000 | 2.7942 | 3.1437 | 3.7029 | 4.1972 |
| **Job 3** | **2.4000** | 3.3127 | **3.6838** | 4.2622 | 4.7564 |
| **Job 4** | 4.9000 | 5.3942 | 5.7437 | 6.3029 | 6.7972 |
| **Job 5** | **5.0000** | 5.9054 | **6.2796** | 6.8622 | 7.3564 |
| **Job 6** | **5.5000** | 6.4444 | **6.8311** | 7.4214 | 7.9156 |
| **Job 7** | 9.0000 | 9.4942 | 9.8437 | 10.4029 | 10.8972 |
| **Job 8** | 9.5000 | 10.0234 | **10.3885** | 10.9622 | 11.4564 |
| **Job 9** | 11.0000 | 11.4942 | 11.8437 | 12.4029 | 12.8972 |
| **Job 10** | 13.0000 | 13.4942 | 13.8437 | 14.4029 | 14.8972 |

Table 2.1: Optimal Departure Times

In Figure 2.2, we observe oscillations during the initial iterations, and afterwards the algorithm enters the "convergence mode". This is very typical behavior for steepest descent methods with decreasing step sizes. Selecting a very small initial step size may eliminate oscillations; however, this selection may also end up with slower convergence. Another factor that affects the performance of the algorithm is the termination tolerance $\epsilon$. Selecting a large $\epsilon$ value may result with a "premature termination", i.e., the algorithm may stop far from the optimal. In short, the selection of $\epsilon$ and $\eta_k$ affects the performance.

### 2.6.2 Comparison of Different Solution Methodologies

In order to compare the solution performances of different solution methodologies, we study problems with different choices of $M$ and $N$. The $\beta_j$ values are randomly selected from the set $\beta = \{5i : i = 1, ..., 20\}$ and the job interarrival times are realized from an exponential distribution with a mean of 2 units. For convenience, we set $S_j = 0$ for $j = 1, ..., M$ and $d_i = \infty$ for $i = 1, ..., N$.

The computation times are based on averages over ten optimization problems (obtained by varying arrival sequences $\{a_i\}_{i=1}^N$ and the cost parameters $\{\beta_j\}_{j=1}^M$) for each $M$ and $N$ combination. The times are reported from a computer with 2.0GHz Intel Core2Duo T7200 processor and 2GB of RAM running Matlab.

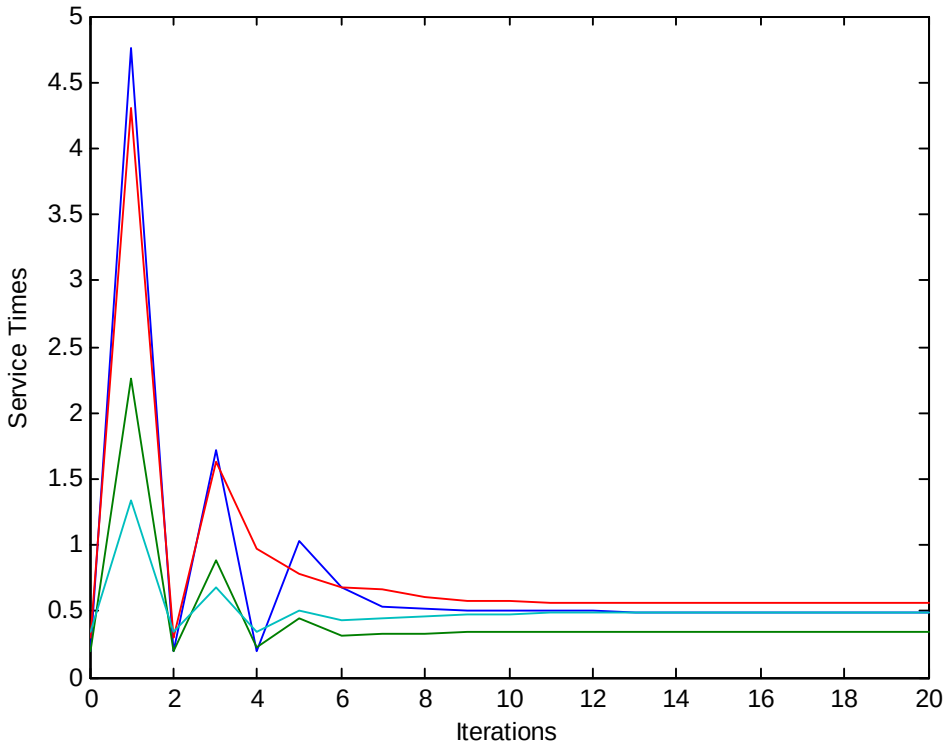


Figure 2.2: Evolutions of Service Times for Subgradient Descent Algorithm

#### 2.6.2.1 $\bar{P}$ Formulation vs $Q$ Formulation

In order to demonstrate the benefits due to simplifications in Section 2.3, we solve both $\bar{P}$ and $Q$ formulations by *cvx* (see [17]), a modeling system for convex programming developed in Stanford University. The computation times for different $M$ and $N$ settings are given in Table 2.2. Note that a dash sign represents an "out of memory" crash.

The results in Table 2.2 suggest that it is definitely faster to solve the $Q$ formulation as expected. One may argue, however, that since these computations are to be performed offline before the manufacturing operation starts, the improvement in the computation times are not that important. After all, thanks to today's fast CPUs, both problems are solved within minutes. (It takes about

| | **M = 20** | | **M = 40** | | **M = 60** | |
|---|---|---|---|---|---|---|
| **N** | $\bar{P}$ | $Q$ | $\bar{P}$ | $Q$ | $\bar{P}$ | $Q$ |
| **100** | 3.32 | 1.66 | 7.32 | 2.44 | 10.30 | 4.53 |
| **200** | 6.06 | 2.43 | 13.93 | 4.14 | - | 10.06 |
| **300** | 9.76 | 3.41 | - | 6.31 | - | 19.86 |
| **400** | 14.43 | 4.46 | - | 8.30 | - | 34.11 |
| **500** | - | 5.43 | - | 10.69 | - | 55.52 |

Table 2.2: Computation Times for $\bar{P}$ and $Q$ Formulations (in seconds)

55.5 seconds to solve the $Q$ formulation when $M = 60$ and $N = 500$.) In fact, the results suggest that the bottleneck in these calculations is not the CPU speed but the memory size. (Numerical solution of $\bar{P}$ is feasible only for small systems.) Hence, we relieve this bottleneck and allow the optimization of larger systems by the proposed simplifications.

#### 2.6.2.2 $Q$ Formulation vs Subgradient Descent Algorithm

Since the $Q$ formulation outperforms the $\bar{P}$ formulation, we directly compare the performance of the subgradient descent algorithm solving $R$ against *cvx* solver solving $Q$ under different $M$ and $N$ settings. The distance measure in (2.119) is employed with an $\epsilon$ value of $10^{-5}$, and the step sizes are given by $\eta_k = \frac{10^{-5}}{k}$.

For all $M$ and $N$ combinations, the subgradient descent algorithm solving $R$ produced the same solutions (according to our precision determined by the $\epsilon$ value) as the *cvx* solver solving $Q$. Moreover, our subgradient descent algorithm not only improved the solution times but also increased the solvable system sizes as can be observed in Table 2.3. (Note again that a dash sign indicates an "out of memory" crash.)

#### 2.6.2.3 Subgradient Descent Algorithm vs Two-Phase Search Algorithm

Finally, since the subgradient descent algorithm solving $R$ outperforms *cvx* solving both $Q$ and $\bar{P}$ formulations, we directly compare the performance of the

| | **M = 20** | | **M = 40** | | **M = 60** | |
|---|---|---|---|---|---|---|
| **N** | ***Q*** | ***R*** | ***Q*** | ***R*** | ***Q*** | ***R*** |
| **500** | 5.43 | 0.76 | 10.69 | 1.13 | 55.52 | 1.74 |
| **1000** | 26.54 | 1.73 | 48.33 | 2.98 | 355.05 | 4.38 |
| **1500** | 51.37 | 3.28 | 99.50 | 5.46 | 2251.50 | 8.17 |
| **2000** | 82.51 | 5.40 | 169.32 | 9.07 | - | 12.90 |
| **2500** | 130.39 | 7.79 | - | 12.64 | - | 18.47 |
| **3000** | - | 10.80 | - | 17.81 | - | 25.68 |

Table 2.3: Computation Times for $Q$ Formulation and Subgradient Descent Algorithm (in seconds)

two-phase search algorithm against the subgradient descent algorithm under different $M$ and $N$ settings. Similarly, the subgradient descent algorithm (SD) employs the precision measure with an $\epsilon$ value of $10^{-5}$ and the step sizes given by $\eta_k = \frac{10^{-5}}{k}$. The two-phase search algorithm (2PS) uses *fsolve* function employing a variant of the Powell dogleg method described in [40] to solve the $Q_k^\beta$ and $\hat{Q}_k^\beta$ problems. The computation times for both algorithms are presented in Table 2.4.

| | **M = 20** | | **M = 40** | | **M = 60** | |
|---|---|---|---|---|---|---|
| **N** | **SD** | **2PS** | **SD** | **2PS** | **SD** | **2PS** |
| **500** | 0.83 | 0.32 | 1.28 | 0.34 | 1.70 | 0.35 |
| **1000** | 1.72 | 0.33 | 2.97 | 0.35 | 4.22 | 0.37 |
| **1500** | 3.24 | 0.35 | 5.34 | 0.37 | 7.71 | 0.39 |
| **2000** | 6.23 | 0.37 | 9.45 | 0.39 | 12.60 | 0.41 |
| **2500** | 9.75 | 0.39 | 13.77 | 0.42 | 18.53 | 0.44 |
| **3000** | 13.27 | 0.41 | 19.55 | 0.45 | 25.83 | 0.48 |
| **4000** | 20.63 | 0.50 | 31.16 | 0.54 | 41.30 | 0.59 |
| **5000** | 29.52 | 0.66 | 46.75 | 0.71 | 63.47 | 0.77 |
| **6000** | 41.57 | 0.83 | 65.82 | 0.89 | 89.39 | 0.96 |
| **10000** | 106.87 | 1.72 | 177.68 | 1.79 | 235.99 | 1.87 |

Table 2.4: Computation Times for Subgradient Descent and Two-Phase Search Algorithms (in seconds)

The two-phase search algorithm improved the solution times compared to the subgradient descent algorithm as seen in Table 2.4. The improvement in solution times gets drastic as the problem size increases. For instance, for 100 machines and 50000 jobs, i.e., $M = 100$ and $N = 50000$, the average computation time over ten sample problems is 11352.26 seconds for the subgradient descent algorithm,

while this value is only 33.06 seconds for the two-phase search algorithm.

## 2.7 Conclusion

In this chapter, we considered the deterministic flow shop systems, where the service times of the machines are fixed known values or set only once and can not be altered between processes. The arrival times of the jobs and their associated deadlines were assumed to be known a priori. A standard solution method based on the linearization of the *max* constraints yielded a convex optimization problem with high memory requirements. In order to derive an equivalent convex optimization problem with lower memory requirements, a set of waiting and completion time characteristics for fixed service time flow shop systems was derived and exploited. Basically, it was shown that jobs do not wait at the downstream of the machine with highest service time and a certain job always waits at this machine if it observes waiting in the system. Exploiting these results, job completion times were shown to be represented as a function of the service times of the machines in the system. This representation was then incorporated into the problem to derive a simplified equivalent convex optimization problem through eliminating many of its constraints. As shown by the numerical study, the resulting simplified convex optimization problem had significantly lower memory requirements and somewhat lower solution times. However, since this formulation still required a convex programming problem solver, which may not be available in small manufacturing companies, and needed several GB's of memory for large problems, we also proposed an alternative convex formulation which is not everywhere differentiable, and a subgradient descent solution method employing subgradients for directions. While deriving this alternative convex optimization problem along with its subgradient solution method, we employed an alternative representation of job completion times, again as the function of the service times of all machines in the system, derived through further exploitation of the waiting and completion time characteristics of fixed service time flow shop systems. As demonstrated by the numerical study, substantial improvements in solution times and solvable system sizes were achieved by the proposed subgradient descent solution method.

Finally, we focused on a specific service cost structure which allowed us to sort the optimal service times and to define differentiable subproblems leading to a two-phase search algorithm. This new search algorithm, which is guaranteed to find the optimal service times in a finite number of iterations, was shown to improve the solution times drastically compared to the subgradient descent solution method by the numerical study.

# Chapter 3

# Service Time Optimization of Mixed Line Flow Shop Systems

In this chapter, we consider deterministic flow shop systems formed of fully controllable, initially controllable, and uncontrollable machines processing identical jobs with known arrival times and deadlines. We note that mixed line flow shop systems with no fully controllable machines turn out to be the fixed service time flow shop systems that were analyzed comprehensively in Chapter 2. Hence, throughout the chapter, we assume that there exists at least one fully controllable machine in the system. In fact, analysis of mixed line flow shops extensively exploits the structural properties of fixed service time flow shop systems presented in Chapter 2. However, including fully controllable machines in the system allows for some more optimal waiting characteristics which lead to new solution methodologies different from the ones developed for fixed service time flop shop systems. Hence, we devote a new chapter to mixed line flow shops systems. The cost function we consider consists of service costs at fully and initially controllable machines, which are dependent on service times, and regular completion-time costs for jobs. Based on the same reasoning in Chapter 2, we assume that faster services increase service costs, while slower services increase the regular completion-time costs and/or leading to failures in accomplishing jobs within their deadlines. This trade-off is what makes the problem challenging

and our objective is to determine the cost-minimizing service times at fully and initially controllable machines.

In this chapter, we first formulate a non-convex and non-differentiable optimization problem which can be solved by the convex formulation derived through the standard method of linearization on the *max* constraints. However, high memory requirement of the resulting convex formulation limits its solution only to small systems. Motivated by the need for solving larger systems within reasonable solution times, we present a set of optimal waiting characteristics for such flow shop systems and employ them to develop efficient solution methodologies. Mainly, under some cost assumptions, we show that jobs do not wait on the optimal sample path at the downstream of the first fully controllable machine. Utilizing this property, we derive simplified equivalent convex optimization problems. For the flow shop systems formed of only fully controllable and uncontrollable machines, the associated simplified equivalent convex optimization problem is then decomposed by a "forward in time" algorithm into smaller convex optimization problems under an additional strict convexity assumption on the completion-time costs.

The rest of this chapter is organized as follows: We formulate a non-convex and non-differentiable optimization problem and present a convex programming formulation derived through linearization in Section 3.1. In Section 3.2, we derive a set of waiting characteristics of such systems and show that, on the optimal sample path, no waiting is observed after the first fully controllable machine of the system. In Section 3.3, we develop simplified convex programming formulations by employing the no-wait property derived in Section 3.2. For the flow shop systems of fully controllable and uncontrollable machines, a forward decomposition algorithm is presented in Section 3.4 to decompose the associated simplified convex optimization problem derived in Section 3.3 into smaller convex optimization problems with linear constraints. In Section 3.5, a numerical study is held to visualize the optimal waiting characteristics derived in Section 3.2, to demonstrate the benefits of the simplification and the decomposition in terms of solution performances, and to compare the forward decomposition algorithm against another algorithm (applicable to a special case of our model) from the

literature through a set of example systems. Analysis of the effects of replacing an initially controllable machine with a fully controllable machine and the locations of fully controllable machines on the optimal cost, and service and completion-time costs on the optimal service times are also included in Section 3.5. Finally, Section 3.6 concludes the chapter.

## 3.1 Problem Formulation

The notation used throughout the chapter is as follows:

**Decision Variables:**

$x_{i,j}$ : departure time of job $i$ from machine $j$.
$s_{i,j}$ : service time of job $i$ at fully controllable machine $j$.
$s_j$ : service time at initially controllable machine $j$.

**Parameters:**

$M$ : number of machines in the system.
$N$ : number of jobs that arrive at the system.
$a_i$ : arrival time of job $i$.
$d_i$ : deadline for the completion of job $i$.
$\breve{s}_j$ : service time at uncontrollable machine $j$.
$S_j$ : lower bound for the service time at machine $j$.
$S_F$ : index set of fully controllable machines in the system.
$S_I$ : index set of initially controllable machines in the system.
$S_U$ : index set of uncontrollable machines in the system.
$\theta_j(s_j)$ : total service cost over all jobs at initially controllable machine $j$.
$\theta_j(s_{i,j})$ : service cost of job $i$ at fully controllable machine $j$.
$\phi_i(x_{i,M})$ : completion-time cost for job $i$.

Let us consider an $M$-machine flow shop system. A sequence of $N$ identical jobs arrive at the system at known times $0 \leq a_1 \leq a_2 \leq ... \leq a_N$ and are processed in all machines sequentially. We denote these jobs by $\{C_i\}_{i=1}^N$ and assign them job completion deadlines as $\{d_i\}_{i=1}^N$. Machines, with buffers of infinite sizes, process one job at a time on a first-come first-served non-preemptive basis (i.e. a job in service cannot be interrupted until its service completion). The system consists of *fully controllable* machines, where the service times can be adjusted before each process, *initially controllable* machines, where the service time are set only once before arrival of the first job, and *uncontrollable* machines, where the service times are fixed and foreknown. We define the sets $S_F$, $S_I$, and $S_U$ as the index sets of the fully controllable, initially controllable, and uncontrollable machines, respectively. Then, the service times at each machine $j \in S_F$ are denoted by $\{s_{i,j}\}_{i=1}^N$, while the service times are denoted by $s_j$ and $\breve{s}_j$ at each machine $j \in S_I$ and $j \in S_U$, respectively.

To keep the notation simple and to make the rest of the chapter more readable, without loss of generality, we treat the uncontrollable machines as if they are initially controllable machines as done in Chapter 2 and we study the flow shop systems consisting only of fully and initially controllable machines. We acknowledge that the results of such system are applicable to flow shop systems including also uncontrollable machines.

We define $x_{i,j}$ as the departure time of job $C_i$ from machine $j$ and consider the discrete-event optimal control problem, denoted by $P_M$, which has the following form:

$$P_M : \min_{\substack{s_{i,j}, s_j \geq S_j \\ i=1,...,N \\ j=1,...,M}} \left\{ J_M = \sum_{i=1}^{N} \sum_{j \in S_F} \theta_j(s_{i,j}) + \sum_{j \in S_I} \theta_j(s_j) + \sum_{i=1}^{N} \phi_i(x_{i,M}) \right\} \tag{3.1}$$

subject to

$$x_{i,j} = \max(x_{i,j-1}, x_{i-1,j}) + s_{i,j}, \qquad j \in S_F \tag{3.2}$$

$$x_{i,j} = \max(x_{i,j-1}, x_{i-1,j}) + s_j, \qquad j \in S_I \tag{3.3}$$

$$x_{i,M} \leq d_i \tag{3.4}$$

$$x_{i,0} = a_i, \quad x_{0,j} = -\infty, \qquad \forall j \tag{3.5}$$

for $i = 1, ..., N$. In this formulation, $\theta_j$ denotes the service cost at machine $j$, and $\phi_i$ denotes the completion-time cost for job $C_i$.

Similarly as in Chapter 2, the lower bounds on service times at machines may prevent some jobs from meeting their associated deadlines, therefore cause infeasibility. Here, we will consider the case where a feasible solution exists for the optimization problem $P_M$. If that is not the case, then a separate job admission control problem has to precede our analysis so as to reject some jobs and lead to a feasible problem as in [29], which is not considered here.

The following assumptions are necessary to make the problem somewhat more tractable while preserving the originality of the problem.

**Assumption 3.1** *:* $\theta_j(\cdot)$, *for* $j = 1, ..., M$, *is continuously differentiable, monotonically decreasing, and strictly convex.*

**Assumption 3.2** *:* $\phi_i(\cdot)$, *for* $i = 1, ..., N$, *is continuously differentiable, monotonically increasing, and convex.*

Note that for the costs satisfying these assumptions, longer services will decrease the service costs, while possibly increasing the completion times, hence, the completion-time costs.

By replacing the constraints $x_{i,j} = \max(x_{i,j-1}, x_{i-1,j}) + s_{i,j}$ in (3.2) by the constraints

$$x_{i,j} \geq x_{i,j-1} + s_{i,j},$$
$$x_{i,j} \geq x_{i-1,j} + s_{i,j},$$

and the constraints $x_{i,j} = \max(x_{i,j-1}, x_{i-1,j}) + s_j$ in (3.3) by the constraints

$$x_{i,j} \geq x_{i,j-1} + s_j,$$
$$x_{i,j} \geq x_{i-1,j} + s_j,$$

resulting with the convex optimization problem $\bar{P}_M$ defined as

$$\bar{P}_M : \min_{\substack{s_{i,j}, s_j \geq S_j \\ x_{i,j} \\ i=1,...,N \\ j=1,...,M}} \left\{ \bar{J}_M = \sum_{i=1}^{N} \sum_{j \in S_F} \theta_j(s_{i,j}) + \sum_{j \in S_I} \theta_j(s_j) + \sum_{i=1}^{N} \phi_i(x_{i,M}) \right\}$$

subject to

$$x_{i,j} \geq x_{i,j-1} + s_{i,j}, \qquad j \in S_F \tag{3.6}$$
$$x_{i,j} \geq x_{i-1,j} + s_{i,j}, \qquad j \in S_F \tag{3.7}$$
$$x_{i,j} \geq x_{i,j-1} + s_j, \qquad j \in S_I \tag{3.8}$$
$$x_{i,j} \geq x_{i-1,j} + s_j, \qquad j \in S_I \tag{3.9}$$
$$x_{i,M} \leq d_i \tag{3.10}$$
$$x_{i,0} = a_i, \quad x_{0,j} = -\infty, \quad \forall j \tag{3.11}$$

for all $i = 1, ..., N$.

Since the feasible set of the convex optimization problem $\bar{P}_M$ contains the feasible set of the optimization problem $P_M$, its optimal cost $\bar{J}^*_M$ is upper bounded by the optimal cost of $P_M$ denoted by $J^*_M$, i.e., $\bar{J}^*_M \leq J^*_M$. It can be easily verified that an optimal service vector $\bar{s}^*$ for the convex optimization problem $\bar{P}_M$ is also optimal for the problem $P_M$. Hence, by solving $\bar{P}_M$ we can obtain an optimal solution for $P_M$. Moreover, any optimal solution for $P_M$ along with its corresponding departure times constitute an optimal solution for $\bar{P}_M$.

In the next section, we show that no waiting is observed after the first fully controllable machine, a property that is needed to simplify the optimization problem $P_M$.

## 3.2 Waiting Characteristics of the Optimal Sample Path

The machines in the flow shop can be decomposed into portions formed of sequentially located same type (fully controllable or initially uncontrollable) of machines.

**Definition 3.1** *A contiguous set of machines* $\{u, ..., v\}$ *form an initially controllable (or a fully controllable) portion if*

1. Machine $(u-1)$, if exists, is a fully controllable (or an initially controllable) machine;
2. Machines $\{u, ..., v\}$ are initially controllable (or fully controllable) machines;
3. Machine $(v+1)$, if exists, is a fully controllable (or an initially controllable) machine.

We analyze the initially controllable and fully controllable portions separately in the following subsections.

### 3.2.1 Initially Controllable Portions

Initially controllable portions can be treated as fixed service time flow shop systems previously studied in Chapter 2; therefore, the results therein are applicable. Hence, skipping the proofs, we borrow the following three lemmas from Chapter 2 and adapt them to initially controllable portions. The counterpart of the global bottleneck machine in fixed service time flow shop systems is called as the *retarding* machine of the initially controllable portion defined as follows:

**Definition 3.2** *Let machines* $\{u, ..., v\}$ *form an initially controllable portion. The machine* $m$ *with the highest service time (and the most upstream in case of a tie) satisfying* $m = \min\{k : s_k = \max_{j=u,...,v} s_j\}$ *is called as the retarding machine of this portion.*

The arrival time of job $C_k$ at the initially controllable portion formed of machines $\{u, ..., v\}$ is given as $x_{k,u-1}$, and the average interarrival time between jobs $C_k$ and $C_l$ at this portion is given as

$$\sigma_k^l = \frac{x_{k,u-1} - x_{l,u-1}}{k - l}, \tag{3.12}$$

where $l < k$.

The minimum of the average interarrival times for job $C_k$ at this initially controllable portion is, then, defined as

$$\sigma_k = \begin{cases} \infty, & k = 1 \\ \min\limits_{l=1,...,k-1} \sigma_k^l, & k > 1 \end{cases} \tag{3.13}$$

and, according to the following lemma, it allows us to determine whether a job waits or not at the retarding machine $m$.

**Lemma 3.1** *A job $C_k$ waits for service at the retarding machine $m$ if and only if $\sigma_k < s_m$.*

At this point, we borrow the following two definitions from Chapter 2 and decompose the jobs into blocks according to their waiting characteristics at some machine $j$.

**Definition 3.3** *A contiguous set of jobs $\{C_i\}_{i=k}^n$ is said to form a* block *at machine $j$ if*

*1) Jobs $C_k$ and $C_{n+1}$ (if exists) do not wait at machine $j$, i.e., $x_{k,j-1} \geq x_{k-1,j}$ and $x_{n+1,j-1} \geq x_{n,j}$;*

*2) Jobs $\{C_i\}_{i=k+1}^n$ wait at machine $j$, i.e., $x_{i,j-1} < x_{i-1,j}$ for $i = k+1, ..., n$.*

**Definition 3.4** *A partition of jobs into blocks is called a* block structure.

According to Lemma 3.1, one could evaluate $\sigma_k$ values for all jobs $C_k$ and compare them to the service time of the retarding machine to determine the block structure. The following lemma, however, presents a simpler way.

**Lemma 3.2** *If jobs* $\{C_i\}_{i=k}^{n}$ *form a block at the retarding machine* $m$*, then,*

$$\sigma_i^k < s_m$$

*is satisfied for all* $i = k+1, ..., n$*.*

Being the counterpart of global bottleneck machine in fixed service time flow shop systems, no waiting is observed after the retarding machine. Hence, the departure times from initially controllable portions can be determined as presented in the next lemma.

**Lemma 3.3** *For the initially controllable portion* $\{u, ..., v\}$*, the departure time of job* $C_i$ *is given by*

$$x_{i,v} = \max\left(x_{i,u-1} + \sum_{k=u}^{v} s_k, x_{i-1,v} + s_m\right),$$

*where* $x_{0,u-1} = -\infty$ *and* $s_m = \max_{k=u,...,v} s_k$ *is the service time of the retarding machine.*

Before proceeding with the next lemma, which states that no waiting is observed at initially controllable portions downstream to a fully controllable machine, let us define $x_{N+1,j} = \infty$ for all $j = 1, ..., M$.

**Lemma 3.4** *Let machines* $\{u, ..., v\}$ *form an initially controllable portion. If* $u > 1$*, i.e., if there is a fully controllable machine upstream to machine* $u$*, then, on the optimal sample path, jobs do not wait at these machines, i.e.,* $x^*_{i,j-1} \geq x^*_{i-1,j}$ *for all* $i = 1, ..., N$ *and* $j = u, ..., v$*.*

**Proof.** Let machine $m$ be the retarding machine of the initially controllable portion $\{u, ..., v\}$. Since $s_m = \max_{j=u,...,v} s_j$, we may deduce from Lemma 3.1 that if a job waits for service at some machine in the initially controllable portion $\{u, ..., v\}$, then it also waits for service at the retarding machine $m$. Hence, to prove the statement of lemma, it suffices to show that jobs do not wait at the retarding machine $m$.

(By Induction) Since the first job does not wait at any machine, we have the basis for the induction. For a contradiction in the inductive step, let us assume that, on the optimal sample path, jobs $\{C_i\}_{i=1}^{k}$ do not wait while the job $C_{k+1}$ waits for service at the retarding machine $m$.

Let jobs $\{C_i\}_{i=k}^{n}$ form the block in which job $C_{k+1}$ resides at the retarding machine $m$. Then, by Lemma 3.3, we have

$$x^*_{i,u-1} + \sum_{j=u}^{v} s^*_j < x^*_{i-1,v} + s^*_m \tag{3.14}$$

for $i = k+1, ..., n$ and

$$x^*_{n+1,u-1} + \sum_{j=u}^{v} s^*_j \geq x^*_{n,v} + s^*_m. \tag{3.15}$$

By Lemma 3.2 and from (3.12), since job $C_{k+1}$ resides in the block started by job $C_k$, we also have

$$\left(\sigma^k_{k+1}\right)^* = x^*_{k+1,u-1} - x^*_{k,u-1} < s^*_m. \tag{3.16}$$

Moreover, from (3.2), we have

$$x^*_{k+1,u-1} \geq x^*_{k,u-1} + s^*_{k+1,u-1}. \tag{3.17}$$

It follows from (3.16) and (3.17) that

$$s^*_{k+1,u-1} < s^*_m. \tag{3.18}$$

If $x^*_{n+1,u-2} \leq x^*_{n,u-1}$, we have from (3.2) that

$$x^*_{n+1,u-1} = x^*_{n,u-1} + s^*_{n+1,u-1}. \tag{3.19}$$

Since job $C_{n+1}$ does not wait at machine $m$, from (3.12) and (3.13), and by Lemma 3.1, we have

$$\left(\sigma^n_{n+1}\right)^* = x^*_{n+1,u-1} - x^*_{n,u-1} \geq s^*_m. \tag{3.20}$$

It follows from (3.19) and (3.20) that

$$s^*_{n+1,u-1} \geq s^*_m. \tag{3.21}$$

Hence, it follows from (3.18) and (3.21) that if $x^*_{n+1,u-2} \leq x^*_{n,u-1}$, then

$$s^*_{n+1,u-1} > s^*_{k+1,u-1}. \tag{3.22}$$

Next, let us define the perturbed solution as

$$\hat{s}_j = s^*_j \tag{3.23}$$

for all $j \in S_I$ and

$$\hat{s}_{i,j} = \begin{cases} s^*_{i,j} + \Delta_1, & i = k+1, j = u-1 \\ s^*_{i,j} - \Delta_2, & i = n+1, j = u-1 \\ s^*_{i,j}, & \text{otherwise} \end{cases} \tag{3.24}$$

for all $i = 1, ..., N$ and $j \in S_F$ where the positive perturbation $\Delta_1$ and the non-negative perturbation $\Delta_2$ are defined as

$$\Delta_1 = \begin{cases} \min\{\frac{s^*_{n+1,u-1} - s^*_{k+1,u-1}}{2}, \Delta_a\}, & x^*_{n+1,u-2} \leq x^*_{n,u-1} \\ \min\{x^*_{n+1,u-2} - x^*_{n,u-1}, \Delta_a\}, & x^*_{n+1,u-2} > x^*_{n,u-1} \end{cases} \tag{3.25}$$

$$\Delta_2 = \begin{cases} \min\{\frac{s^*_{n+1,u-1} - s^*_{k+1,u-1}}{2}, \Delta_a\}, & x^*_{n+1,u-2} \leq x^*_{n,u-1} \\ 0, & x^*_{n+1,u-2} > x^*_{n,u-1}. \end{cases} \tag{3.26}$$

Note that in (3.25) and (3.26), $\Delta_a$ is defined as

$$\Delta_a = \min_{i=k+1,...,n} \left\{ (x^*_{i-1,v} + s^*_m) - (x^*_{i,u-1} + \sum_{j=u}^{v} s^*_j) \right\}.$$

Since $\Delta_2 \leq \Delta_1$ from (3.25) and (3.26), then we have from (3.22) and (3.24) that

$$\hat{s}_{n+1,u-1} = s^*_{n+1,u-1} - \Delta_2 \geq s^*_{k+1,u-1} + \Delta_1 = \hat{s}_{k+1,u-1} \geq S_{u-1}; \tag{3.27}$$

therefore, our perturbed solution is a feasible solution for the optimization problem $P_M$.

On this alternative sample path, the perturbation does not affect the jobs before $C_{k+1}$ and the machines before $(u-1)$. Hence, the perturbed departures, denoted by $\hat{x}_{i,j}$'s, satisfy

$$\hat{x}_{i,j} = x^*_{i,j} \tag{3.28}$$

for all $i = 1, ..., k$ and $j = 1, ..., M$, or $i = k+1, ..., N$ and $j = 1, ..., u-2$. We, next, determine the effects of this perturbation on $\hat{x}_{i,j}$'s for all $i = k+1, ..., N$ and $j = u-1, ..., M$. First, we show, by induction, that

$$\hat{x}_{i,u-1} \leq x^*_{i,u-1} + \Delta_1$$

for all $i = k+1, ..., n$. From (3.2), (3.24), and (3.28),

$$\begin{aligned} \hat{x}_{k+1,u-1} &= \max\left(\hat{x}_{k+1,u-2}, \hat{x}_{k,u-1}\right) + \hat{s}_{k+1,u-1} \\ &= \max\left(x^*_{k+1,u-2}, x^*_{k,u-1}\right) + s^*_{k+1,u-1} + \Delta_1 \\ &= x^*_{k+1,u-1} + \Delta_1. \end{aligned} \tag{3.29}$$

Next, let us assume that

$$\hat{x}_{r,u-1} \leq x^*_{r,u-1} + \Delta_1 \tag{3.30}$$

for some $r \in \{k+1, ..., n-1\}$. Then, from (3.2), (3.24), (3.28), and (3.30),

$$\begin{aligned} \hat{x}_{r+1,u-1} &= \max\left(\hat{x}_{r+1,u-2}, \hat{x}_{r,u-1}\right) + \hat{s}_{r+1,u-1} \\ &\leq \max\left(x^*_{r+1,u-2}, x^*_{r,u-1} + \Delta_1\right) + s^*_{r+1,u-1} \\ &\leq x^*_{r+1,u-1} + \Delta_1. \end{aligned}$$

Hence, we have

$$\hat{x}_{i,u-1} \leq x^*_{i,u-1} + \Delta_1 \tag{3.31}$$

for all $i = k+1, ..., n$.

From (3.14), (3.23), (3.25), (3.28), (3.29), and by Lemma 3.3, the departure time of the job $C_{k+1}$ from the initially controllable portion formed of machines $\{u, ..., v\}$ for the perturbed solution is given as

$$\begin{aligned} \hat{x}_{k+1,v} &= \max\left(\hat{x}_{k+1,u-1} + \sum_{j=u}^{v} \hat{s}_j, \hat{x}_{k,v} + \hat{s}_m\right) \\ &= \max\left(x^*_{k+1,u-1} + \Delta_1 + \sum_{j=u}^{v} s^*_j, x^*_{k,v} + s^*_m\right) \\ &= x^*_{k+1,v} \end{aligned} \tag{3.32}$$

and starting with (3.32), we recursively obtain from (3.31) that

$$\begin{aligned} \hat{x}_{i,v} &= \max\left(\hat{x}_{i,u-1} + \sum_{j=u}^{v} \hat{s}_j, \hat{x}_{i-1,v} + \hat{s}_m\right) \\ &= \max\left(x^*_{i,u-1} + \Delta_1 + \sum_{j=u}^{v} s^*_j - \varepsilon_i, x^*_{i-1,v} + s^*_m\right) \\ &= x^*_{i,v} \end{aligned} \tag{3.33}$$

for all $i = k+2, ..., n$ where $\varepsilon_i \geq 0$. Hence, the perturbation does not affect the departure times of the jobs $\{C_i\}_{i=k+1}^{n}$ at machines $\{v, ..., M\}$ resulting in

$$\hat{x}_{i,M} = x^*_{i,M} \tag{3.34}$$

for all $i = 1, ..., n$.

Further, from (3.2), (3.24), (3.25), (3.26), (3.28), and (3.31), the departure time of the job $C_{n+1}$ from machine $(u-1)$ for the perturbed solution is written as

$$\begin{aligned} \hat{x}_{n+1,u-1} &= \max\left(\hat{x}_{n+1,u-2}, \hat{x}_{n,u-1}\right) + \hat{s}_{n+1,u-1} \\ &\leq \max\left(x^*_{n+1,u-2}, x^*_{n,u-1} + \Delta_1\right) + s^*_{n+1,u-1} - \Delta_2 \\ &\leq x^*_{n+1,u-1}; \end{aligned} \tag{3.35}$$

therefore, from (3.23), (3.33), (3.35), and by Lemma 3.3, the departure time of this job from initially controllable portion $\{u, ..., v\}$ is given by

$$\begin{aligned} \hat{x}_{n+1,v} &= \max\left(\hat{x}_{n+1,u-1} + \sum_{j=u}^{v} \hat{s}_j, \hat{x}_{n,v} + \hat{s}_m\right) \\ &\leq \max\left(x^*_{n+1,u-1} + \sum_{j=u}^{v} s^*_j, x^*_{n,v} + s^*_m\right) \\ &\leq x^*_{n+1,v}. \end{aligned} \tag{3.36}$$

Hence, from (3.2), (3.3), (3.23), (3.24), and (3.36)

$$\hat{x}_{i,j} \leq x^*_{i,j} \tag{3.37}$$

for all $i = n+1, ..., N$, $j = v, ..., M$ indicating the possibility of decrease in the departure times of the jobs $\{C_i\}_{i=n+1}^{N}$ at machines $\{v, ..., M\}$ due to the perturbation.

Let us denote the cost of this alternative solution as $\hat{J}$. Then, from (3.23), (3.24), (3.34), and (3.37), the cost variation, denoted by $\Delta\hat{J} = \hat{J} - J^*_M$, is given by

$$\begin{aligned} \Delta\hat{J} &= \left[\theta_{u-1}(s^*_{k+1,u-1} + \Delta_1) - \theta_{u-1}(s^*_{k+1,u-1})\right] \\ &\quad - \left[\theta_{u-1}(s^*_{n+1,u-1}) - \theta_{u-1}(s^*_{n+1,u-1} - \Delta_2)\right] \\ &\quad + \sum_{i=n+1}^{N} \left[\phi_i(\hat{x}_{i,M}) - \phi_i(x^*_{i,M})\right]. \end{aligned}$$

Since, by Assumption 3.1, the service cost $\theta_{u-1}$ is monotonically decreasing and strictly convex, from (3.22), (3.25), and (3.26),

$$\big[\theta_{u-1}(s^*_{k+1,u-1}+\Delta_1)-\theta_{u-1}(s^*_{k+1,u-1})\big]-\big[\theta_{u-1}(s^*_{n+1,u-1})-\theta_{u-1}(s^*_{n+1,u-1}-\Delta_2)\big]<0. \tag{3.38}$$

Moreover, since, by Assumption 3.2, the completion-time cost $\phi_i$ is monotonically increasing, from (3.37),

$$\sum_{i=n+1}^{N}\left[\phi_i(\hat{x}_{i,M})-\phi_i(x^*_{i,M})\right]\leq 0. \tag{3.39}$$

From (3.38) and (3.39), we have $\Delta\hat{J}<0$ which contradicts the optimality assumption. Hence, the result follows. ■

In this subsection, we showed that no waiting is observed at the initially controllable portions downstream to a fully controllable machine. In the following subsection, we show that the first fully controllable machine is the only fully controllable machine that allows for waiting, i.e., no waiting is observed in the flow shop system after the first fully controllable machine.

### 3.2.2 Fully Controllable Portions

Applying calculus of variations techniques (see in [25]) on the convex optimization problem $\bar{P}_M$, we obtain a set of necessary conditions for optimality. Let us start with introducing the Lagrangian multipliers $\lambda_{i,j}$, $\upsilon_{i,j}$, $\mu_{i,j}$, $\mu_j$, and $\gamma_i$ to form the

augmented cost

$$\begin{aligned} J_A = & \sum_{i=1}^{N} \sum_{j \in S_F} \theta_j(s_{i,j}) + \sum_{j \in S_I} \theta_j(s_j) + \sum_{i=1}^{N} \phi_i(x_{i,M}) \\ & + \sum_{i=1}^{N} \sum_{j \in S_F} \left( \lambda_{i,j} \left[ x_{i,j-1} + s_{i,j} - x_{i,j} \right] + \upsilon_{i,j} \left[ x_{i-1,j} + s_{i,j} - x_{i,j} \right] \right) \\ & + \sum_{i=1}^{N} \sum_{j \in S_I} \left( \lambda_{i,j} \left[ x_{i,j-1} + s_j - x_{i,j} \right] + \upsilon_{i,j} \left[ x_{i-1,j} + s_j - x_{i,j} \right] \right) \\ & + \sum_{i=1}^{N} \sum_{j \in S_F} \mu_{i,j} \left( S_j - s_{i,j} \right) + \sum_{j \in S_I} \mu_j \left( S_j - s_j \right) + \sum_{i=1}^{N} \gamma_i \left( x_{i,M} - d_i \right). \end{aligned}$$

As stated earlier, the optimal solution vector $s^*$ of $P_M$, together with its corresponding departure time vector $x^*$ form an optimal solution for $\bar{P}_M$. Hence, from the augmented cost formulation, $\{s^*, x^*\}$ satisfy the following necessary conditions for optimality:

**1)** For $i = 1, ..., N$ and $j \in S_F$

$$\left. \frac{\partial J_A}{\partial s_{i,j}} \right|_{s^*_{i,j}} = \theta'_j(s^*_{i,j}) + \lambda^*_{i,j} + \upsilon^*_{i,j} - \mu^*_{i,j} = 0, \tag{3.40}$$

$$\lambda^*_{i,j} \geq 0, \quad \lambda^*_{i,j} \left[ x^*_{i,j-1} + s^*_{i,j} - x^*_{i,j} \right] = 0, \tag{3.41}$$

$$\upsilon^*_{i,j} \geq 0, \quad \upsilon^*_{i,j} \left[ x^*_{i-1,j} + s^*_{i,j} - x^*_{i,j} \right] = 0, \tag{3.42}$$

$$\mu^*_{i,j} \geq 0, \quad \mu^*_{i,j} \left( S_j - s^*_{i,j} \right) = 0, \tag{3.43}$$

$$s^*_{i,j} \geq S_j, \tag{3.44}$$

and since $\theta'_j(\cdot) < 0$ by Assumption 3.1, from (3.40) and (3.43),

$$\lambda^*_{i,j} + \upsilon^*_{i,j} > 0, \tag{3.45}$$

and finally from (3.41), (3.42), and (3.45),

$$x^*_{i,j} = \max\left(x^*_{i,j-1}, x^*_{i-1,j}\right) + s^*_{i,j}. \tag{3.46}$$

**2)** For $i = 1, ..., N$ and $j \in S_I$

$$\left.\frac{\partial J_A}{\partial s_j}\right|_{s^*_j} = \theta'_j(s^*_j) + \sum_{i=1}^{N}\left(\lambda^*_{i,j} + \upsilon^*_{i,j}\right) - \mu^*_j = 0, \tag{3.47}$$

$$\lambda^*_{i,j} \geq 0, \quad \lambda^*_{i,j}\left[x^*_{i,j-1} + s^*_j - x^*_{i,j}\right] = 0, \tag{3.48}$$

$$\upsilon^*_{i,j} \geq 0, \quad \upsilon^*_{i,j}\left[x^*_{i-1,j} + s^*_j - x^*_{i,j}\right] = 0, \tag{3.49}$$

$$\mu^*_j \geq 0, \quad \mu^*_j\left(S_j - s^*_j\right) = 0, \tag{3.50}$$

$$s^*_j \geq S_j, \tag{3.51}$$

and since by Assumption 3.1, $\theta'_j(\cdot) < 0$, from (3.47) and (3.50),

$$\sum_{i=1}^{N}\left(\lambda^*_{i,j} + \upsilon^*_{i,j}\right) > 0. \tag{3.52}$$

**3)** For $i = 1, ..., N$ and $j = 1, ..., M$

$$\gamma^*_i \geq 0, \quad \gamma^*_i\left(x^*_{i,M} - d_i\right) = 0, \tag{3.53}$$

$$\left.\frac{\partial J_A}{\partial x_{i,j}}\right|_{x^*_{i,j}} = 0, \tag{3.54}$$

leading to

**3a)** For $i = 1, ..., N-1$ and $j = 1, ..., M-1$

$$\lambda^*_{i,j} + \upsilon^*_{i,j} = \lambda^*_{i,j+1} + \upsilon^*_{i+1,j}. \tag{3.55}$$

**3b)** For $i = 1, ..., N-1$

$$\lambda^*_{i,M} + \upsilon^*_{i,M} = \phi'_i(x^*_{i,M}) + \upsilon^*_{i+1,M} + \gamma_i. \tag{3.56}$$

**3c)** For $j = 1, ..., M-1$

$$\lambda^*_{N,j} + \upsilon^*_{N,j} = \lambda^*_{N,j+1}. \tag{3.57}$$

**3d)**

$$\lambda^*_{N,M} + \upsilon^*_{N,M} = \phi'_N(x^*_{N,M}) + \gamma_M. \tag{3.58}$$

The characteristics (3.40)-(3.58) are needed for the proofs that follow.

The following lemma, which establishes the monotonicity property of the optimal service times at fully controllable machines, is needed for the contradiction argument in the proof of Lemma 3.6.

**Lemma 3.5** *(Monotonicity Property) Let machine $j$ be a fully controllable machine, i.e., $j \in S_F$. Then, for some $i \in \{2, ..., N\}$, if jobs $C_{i-1}$ and $C_i$ are in the same block of the $j$th machine on the optimal sample path then the optimal service times satisfy $s^*_{i,j} \geq s^*_{i-1,j}$.*

**Proof.** Let us assume that $s^*_{i,j} < s^*_{i-1,j}$. From (3.44), there are two possible cases:

*Case 1:* $s^*_{i-1,j} > s^*_{i,j} = S_j$: From (3.43), $\mu^*_{i-1,j} = 0$ and $\mu^*_{i,j} \geq 0$.

*Case 2:* $s^*_{i-1,j} > s^*_{i,j} > S_j$: From (3.43), $\mu^*_{i-1,j} = \mu^*_{i,j} = 0$.

From both cases, for the assumption $s^*_{i-1,j} > s^*_{i,j}$, we get

$$\mu^*_{i,j} - \mu^*_{i-1,j} \geq 0. \tag{3.59}$$

Since jobs $C_{i-1}$ and $C_i$ are in the same block of the $j$th machine on the optimal

sample path, we have

$$x^*_{i,j-1} < x^*_{i-1,j}. \tag{3.60}$$

Further, from (3.41), (3.46), and (3.60), we also have

$$\lambda^*_{i,j} = 0. \tag{3.61}$$

From (3.41), (3.48), (3.53), (3.55), (3.56), (3.61), and by Assumption 3.2, we have

$$\lambda^*_{i-1,j} + \upsilon^*_{i-1,j} - \upsilon^*_{i,j} - \lambda^*_{i,j} = \begin{cases} \lambda^*_{i-1,j+1}, & j < M \\ \phi'_{i-1}(x^*_{i-1,M}) + \gamma^*_{i-1}, & j = M \end{cases} \geq 0. \tag{3.62}$$

It follows from (3.40), (3.59), and (3.62) that

$$\theta'_j(s^*_{i,j}) - \theta'_j(s^*_{i-1,j}) \geq 0.$$

By Assumption 3.1, $\theta'_j$ is monotonically increasing; therefore $s^*_{i,j} \geq s^*_{i-1,j}$, which contradicts the initial assumption $s^*_{i,j} < s^*_{i-1,j}$. Hence, within a block of a machine $j$, the optimal service times are non-decreasing in the job index. ■

In the next lemma, we show that a fully controllable machine prevents buffering in its downstream fully controllable machines.

**Lemma 3.6** *Let machines $h$ and $t$, where $h < t$, be two consecutive fully controllable machines, possibly separated by the initially controllable portion $\{h+1, ..., t-1\}$. On the optimal sample path, no waiting is observed at the fully controllable machine $t$.*

**Proof.** (By Induction) Since the first job does not wait at any machine, we have the basis for the induction. For a contradiction in the inductive step, let us assume that, on the optimal sample path, jobs $\{C_i\}_{i=1}^{k}$ do not wait while the job $C_{k+1}$ waits for service at the fully controllable machine $t$. Let jobs $\{C_i\}_{i=k}^{n}$ form the block in which job $C_{k+1}$ resides at the fully controllable machine $t$. Then, by Lemma 3.5,

$$s^*_{n,t} \geq s^*_{k,t} \tag{3.63}$$

and from the block definition,

$$x^*_{k,t-1} \geq x^*_{k-1,t}, \tag{3.64}$$

$$x^*_{n+1,t-1} \geq x^*_{n,t}, \tag{3.65}$$

$$x^*_{i,t-1} < x^*_{i-1,t} \tag{3.66}$$

for $i = k+1, ..., n$.

Since, by Lemma 3.4, no waiting is observed at initially controllable portions followed by a fully controllable machine, we have

$$x^*_{i,t-1} = \begin{cases} x^*_{i,h} + \sum_{j=h+1}^{t-1} s^*_j, & t > h+1 \\ x^*_{i,h}, & t = h+1 \end{cases} \tag{3.67}$$

for all $i = 1, ..., N$. From (3.2), (3.64), (3.66), (3.67), and since job $C_k$ does not wait at machine $t$, we have

$$s^*_{k+1,h} \leq x^*_{k+1,h} - x^*_{k,h} < s^*_{k,t} \tag{3.68}$$

and

$$x^*_{n,h} < x^*_{k,h} + \sum_{i=k}^{n-1} s^*_{i,t}. \tag{3.69}$$

Similarly, since jobs $\{C_i\}_{i=k}^n$ form a block at machine $t$, from (3.2), (3.64), (3.65), (3.66), and (3.67), we have

$$x^*_{n+1,h} \geq x^*_{k,h} + \sum_{i=k}^{n} s^*_{i,t}. \tag{3.70}$$

If $x^*_{n+1,h-1} \leq x^*_{n,h}$, from (3.2), (3.63), (3.68), (3.69), and (3.70),

$$s^*_{k+1,h} < s^*_{k,t} \leq s^*_{n,t} < s^*_{n+1,h}. \tag{3.71}$$

If $t > h+1$, i.e., if there is an initially controllable portion between machines $h$ and $t$, then let us denote the retarding machine of this portion by machine $m$.

As established in Lemma 3.4, jobs do not wait at the initially controllable portion $\{h+1, ..., t-1\}$; hence, by Lemma 3.1, and from (3.12), (3.13), (3.63), and (3.68),

$$s^*_{n,t} \geq s^*_{k,t} > x^*_{k+1,h} - x^*_{k,h} \geq s^*_m. \tag{3.72}$$

Further, from (3.2), (3.65), (3.66), (3.67), and (3.72),

$$x^*_{n+1,t-1} = x^*_{n+1,h} + \sum_{j=h+1}^{t-1} s^*_j > x^*_{n,t-1} + s^*_m. \tag{3.73}$$

Next, let us define the perturbed solution as

$$\hat{s}_j = s^*_j \tag{3.74}$$

for $j \in S_I$ and

$$\hat{s}_{i,j} = \begin{cases} s^*_{i,j} + \Delta_1, & i = k+1, j = h \\ s^*_{i,j} - \Delta_2, & i = n+1, j = h \\ s^*_{i,j}, & \text{otherwise} \end{cases} \tag{3.75}$$

for all $i = 1, ..., N$ and $j \in S_F$ where the perturbations $\Delta_1$ and $\Delta_2$ are given as

$$\Delta_1 = \begin{cases} \min\{\frac{s^*_{n+1,h} - s^*_{k+1,h}}{2}, \Delta_a\}, & x^*_{n+1,h-1} \leq x^*_{n,h}, t = h+1 \\ \min\{\frac{s^*_{n+1,h} - s^*_{k+1,h}}{2}, \Delta_a, \Delta_b\}, & x^*_{n+1,h-1} \leq x^*_{n,h}, t > h+1 \\ \min\{x^*_{n+1,h-1} - x^*_{n,h}, \Delta_a\}, & x^*_{n+1,h-1} > x^*_{n,h}, t = h+1 \\ \min\{x^*_{n+1,h-1} - x^*_{n,h}, \Delta_a, \Delta_b\}, & x^*_{n+1,h-1} > x^*_{n,h}, t > h+1 \end{cases} \tag{3.76}$$

and

$$\Delta_2 = \begin{cases} \min\{\frac{s^*_{n+1,h} - s^*_{k+1,h}}{2}, \Delta_a\}, & x^*_{n+1,h-1} \leq x^*_{n,h}, t = h+1 \\ \min\{\frac{s^*_{n+1,h} - s^*_{k+1,h}}{2}, \Delta_a, \Delta_b\}, & x^*_{n+1,h-1} \leq x^*_{n,h}, t > h+1 \\ 0, & x^*_{n+1,h-1} > x^*_{n,h}, t = h+1 \\ 0, & x^*_{n+1,h-1} > x^*_{n,h}, t > h+1. \end{cases} \tag{3.77}$$

Note that in (3.76) and (3.77), $\Delta_a$ and $\Delta_b$ are defined as

$$\begin{aligned} \Delta_a &= \min_{i=k+1,\ldots,n} \left\{ x^*_{i-1,t} - x^*_{i,t-1} \right\}, \\ \Delta_b &= x^*_{n+1,t-1} - x^*_{n,t-1} - s^*_m, \end{aligned}$$

and from (3.71), (3.73), and the block definition we have $\Delta_1 > 0$ while $\Delta_2 \geq 0$.

Since $\Delta_2 \leq \Delta_1$ from (3.76) and (3.77), then we have from (3.71) and (3.75) that

$$\hat{s}_{n+1,h} = s^*_{n+1,h} - \Delta_2 \geq s^*_{k+1,h} + \Delta_1 = \hat{s}_{k+1,h} \geq S_h; \tag{3.78}$$

therefore, our perturbed solution is a feasible solution for the optimization problem $P_M$.

On this alternative sample path, the perturbation does not affect the jobs before $C_{k+1}$ and the machines before $h$. Therefore, the perturbed departure times, denoted by $\hat{x}_{i,j}$'s, satisfy

$$\hat{x}_{i,j} = x^*_{i,j} \tag{3.79}$$

for all $i = 1, ..., k$ and $j = 1, ..., M$ or $i = k+1, ..., N$ and $j = 1, ..., (h-1)$. From (3.2), (3.75), and (3.79), we write

$$\begin{aligned} \hat{x}_{k+1,h} &= \max\left(\hat{x}_{k+1,h-1}, \hat{x}_{k,h}\right) + \hat{s}_{k+1,h} \\ &= \max\left(x^*_{k+1,h-1}, x^*_{k,h}\right) + s^*_{k+1,h} + \Delta_1 \\ &= x^*_{k+1,h} + \Delta_1 \end{aligned} \tag{3.80}$$

and applying (3.2), (3.75), and (3.79) recursively, we obtain

$$\begin{aligned} \hat{x}_{i,h} &= \max\left(\hat{x}_{i,h-1}, \hat{x}_{i-1,h}\right) + \hat{s}_{i,h} \\ &\leq \max\left(x^*_{i,h-1}, x^*_{i-1,h} + \Delta_1\right) + s^*_{i,h} \\ &\leq x^*_{i,h} + \Delta_1 \end{aligned} \tag{3.81}$$

for all $i = k+1, ..., n$. Hence, from (3.74), (3.79), (3.80), and by Lemmas 3.3 and 3.4, the departure times from machine $(t-1)$ satisfy

$$\begin{aligned} \hat{x}_{k+1,t-1} &= \max\left(\hat{x}_{k+1,h} + \sum_{j=h+1}^{t-1} \hat{s}_j, \hat{x}_{k,t-1} + \hat{s}_m\right) \\ &= \max\left(x^*_{k+1,h} + \Delta_1 + \sum_{j=h+1}^{t-1} s^*_j, x^*_{k,t-1} + s^*_m\right) \\ &= x^*_{k+1,t-1} + \Delta_1 \end{aligned} \tag{3.82}$$

and starting with (3.82), we recursively obtain from (3.81) that

$$\begin{aligned} \hat{x}_{i,t-1} &= \max\left(\hat{x}_{i,h} + \sum_{j=h+1}^{t-1} \hat{s}_j, \hat{x}_{i-1,t-1} + \hat{s}_m\right) \\ &\leq \max\left(x^*_{i,h} + \Delta_1 + \sum_{j=h+1}^{t-1} s^*_j, x^*_{i-1,t-1} + \Delta_1 + s^*_m\right) \\ &\leq x^*_{i,t-1} + \Delta_1 \end{aligned} \tag{3.83}$$

for all $i = k+2, ..., n$.

From (3.76), (3.82), and (3.83), we have

$$\hat{x}_{i,t-1} \leq x^*_{i,t-1} + \Delta_1 \leq x^*_{i-1,t} \tag{3.84}$$

for $i = k+1, ..., n$. Applying (3.2), (3.75), and (3.84) recursively, we get

$$\hat{x}_{i,t} = x^*_{i,t} \tag{3.85}$$

for $i = k+1, ..., n$. Hence, the perturbation does not affect the departure times of the jobs $\{C_i\}_{i=k+1}^n$ at machines $\{t, ..., M\}$ resulting in

$$\hat{x}_{i,M} = x^*_{i,M} \tag{3.86}$$

for all $i = 1, ..., n$.

Further, from (3.2), (3.75), (3.76), (3.77), (3.79), and (3.81), the departure

time of the job $C_{n+1}$ from machine $h$ is given by

$$\begin{aligned} \hat{x}_{n+1,h} &= \max\left(\hat{x}_{n+1,h-1}, \hat{x}_{n,h}\right) + \hat{s}_{n+1,h} \\ &\leq \max\left(x^*_{n+1,h-1}, x^*_{n,h} + \Delta_1\right) + s^*_{n+1,h} - \Delta_2 \\ &\leq x^*_{n+1,h}. \end{aligned} \tag{3.87}$$

Hence, from (3.74), (3.76), (3.83), (3.87), and by Lemma 3.3, the departure time of the job $C_{n+1}$ from machine $(t-1)$ for the perturbed solution is given by

$$\begin{aligned} \hat{x}_{n+1,t-1} &= \max\left(\hat{x}_{n+1,h} + \sum_{j=h+1}^{t-1} \hat{s}_j, \hat{x}_{n,t-1} + \hat{s}_m\right) \\ &\leq \max\left(x^*_{n+1,h} + \sum_{j=h+1}^{t-1} s^*_j, x^*_{n,t-1} + \Delta_1 + s^*_m\right) \\ &\leq x^*_{n+1,t-1}; \end{aligned} \tag{3.88}$$

therefore, from (3.2), (3.75), (3.85), and (3.88), the departure time of the job $C_{n+1}$ from machine $t$ becomes

$$\begin{aligned} \hat{x}_{n+1,t} &= \max\left(\hat{x}_{n+1,t-1}, \hat{x}_{n,t}\right) + \hat{s}_{n+1,t} \\ &\leq \max\left(x^*_{n+1,t-1}, x^*_{n,t}\right) + s^*_{n+1,t} \\ &\leq x^*_{n+1,t} \end{aligned} \tag{3.89}$$

and from (3.2), (3.3), (3.74), (3.75), and (3.89), we obtain

$$\hat{x}_{i,j} \leq x^*_{i,j} \tag{3.90}$$

for all $i = n+1, ..., N$ and $j = t, ..., M$ indicating the possibility of decrease in the departure times of the jobs $\{C_i\}_{i=n+1}^{N}$ at machines $\{t, ..., M\}$ due to the perturbation.

Let us denote the cost of this alternative solution as $\hat{J}$. Then, from (3.75),

(3.86), and (3.90), the cost variation, denoted by $\Delta\hat{J} = \hat{J} - J_M^*$, is given by

$$\begin{aligned}\Delta\hat{J} &= \left[\theta_h(s^*_{k+1,h} + \Delta_1) - \theta_h(s^*_{k+1,h})\right] \\ &\quad - \left[\theta_h(s^*_{n+1,h}) - \theta_h(s^*_{n+1,h} - \Delta_2)\right] \\ &\quad + \sum_{i=n+1}^{N} \left[\phi_i(\hat{x}_{i,M}) - \phi_i(x^*_{i,M})\right].\end{aligned}$$

Since, by Assumption 3.1, the service cost $\theta_h$ is monotonically decreasing and strictly convex, from (3.71), (3.76), and (3.77),

$$\left[\theta_h(s^*_{k+1,h} + \Delta_1) - \theta_h(s^*_{k+1,h})\right] - \left[\theta_h(s^*_{n+1,h}) - \theta_h(s^*_{n+1,h} - \Delta_2)\right] < 0. \tag{3.91}$$

Moreover, since the completion-time cost $\phi_i$ is monotonically increasing by Assumption 3.2, from (3.90),

$$\sum_{i=n+1}^{N} \left[\phi_i(\hat{x}_{i,M}) - \phi_i(x^*_{i,M})\right] \leq 0. \tag{3.92}$$

From (3.91) and (3.92), we have $\Delta\hat{J} < 0$. Hence, the assumption that the job $C_{k+1}$ waits for service at machine $t$ on the optimal sample path contradicts the optimality assumption, concluding the proof. ■

Applying Lemma 3.4 and Lemma 3.6 recursively, we arrive at the following theorem:

**Theorem 3.1** *On the optimal sample path, no waiting is observed after the first fully controllable machine.*

**Proof.** The result directly follows from Lemma 3.4 and Lemma 3.6. ■

Next, we employ this result to arrive at simpler convex optimization problem formulations compared to $\bar{P}_M$.

## 3.3 Simplified Convex Optimization Problems

For each machine $j$, let us define a binary parameter $\omega_j$ indicating whether the machine is fully controllable or not as

$$\omega_j = \begin{cases} 1, & j \in S_F \\ 0, & \text{otherwise} \end{cases} \tag{3.93}$$

and let machine $f$ be the first fully controllable machine in the system. Employing Theorem 3.1 and (3.93), we can write the departure times of the jobs from the flow shop system as

$$x_{i,M} = x_{i,f} + \sum_{j>f} \left(\omega_j s_{i,j} + (1-\omega_j)s_j\right). \tag{3.94}$$

### 3.3.1 Flow Shop Systems Starting with Fully Controllable Portions

If the flow shop system starts out with a fully controllable machine, i.e., if $f = 1$, then the optimization problem $P_M$ can simply be reformulated as the optimization problem $\tilde{P}_M$:

$$\tilde{P}_M : \min_{\substack{s_{i,j}, s_j \geq S_j \\ i=1,\ldots,N \\ j=1,\ldots,M}} \left\{ \tilde{J}_M = \begin{pmatrix} \sum_{i=1}^{N} \phi_i \left( x_{i,1} + \sum_{j=2}^{M} \left(\omega_j s_{i,j} + (1-\omega_j)s_j\right)\right) \\ + \sum_{i=1}^{N} \sum_{j\in S_F} \theta_j(s_{i,j}) + \sum_{j\in S_I} \theta_j(s_j) \end{pmatrix} \right\} \tag{3.95}$$

subject to

$$x_{1,1} = a_1 + s_{1,1} \tag{3.96}$$

$$x_{i,1} = \max\left(a_i, x_{i-1,1}\right) + s_{i,1} \tag{3.97}$$

$$x_{i,1} \geq x_{i-1,1} + \omega_2 s_{i-1,2} + (1-\omega_2)s_2 \tag{3.98}$$

$$x_{i,1} \geq x_{i-1,1} + \sum_{k=2}^{j-1} \omega_k \left(s_{i-1,k} - s_{i,k}\right) + \omega_j s_{i-1,j} + (1-\omega_j)s_j \tag{3.99}$$

$$x_{1,1} \leq d_1 - \sum_{j=2}^{M} (\omega_j s_{1,j} + (1-\omega_j) s_j) \tag{3.100}$$

$$x_{i,1} \leq d_i - \sum_{j=2}^{M} (\omega_j s_{i,j} + (1-\omega_j) s_j) \tag{3.101}$$

for $i = 2, ..., N$ and $j = 3, ..., M$. (Note that the constraint set in (3.99) exists only if $M > 2$.)

By linearizing the *max* function in (3.97), we get the convex optimization problem $\tilde{Q}$ given as

$$\tilde{Q}: \min_{\substack{s_{i,j}, s_j \geq S_j \\ x_{i,1} \\ i=1,...,N \\ j=1,...,M}} \left\{ J_{\tilde{Q}} = \left( \begin{array}{c} \sum_{i=1}^{N} \phi_i \left( x_{i,1} + \sum_{j=2}^{M} (\omega_j s_{i,j} + (1-\omega_j) s_j) \right) \\ + \sum_{i=1}^{N} \sum_{j \in S_F} \theta_j(s_{i,j}) + \sum_{j \in S_I} \theta_j(s_j) \end{array} \right) \right\} \tag{3.102}$$

subject to

$$x_{1,1} = a_1 + s_{1,1} \tag{3.103}$$

$$x_{i,1} \geq a_i + s_{i,1} \tag{3.104}$$

$$x_{i,1} \geq x_{i-1,1} + s_{i,1} \tag{3.105}$$

$$x_{i,1} \geq x_{i-1,1} + \omega_2 s_{i-1,2} + (1-\omega_2) s_2 \tag{3.106}$$

$$x_{i,1} \geq x_{i-1,1} + \sum_{k=2}^{j-1} \omega_k (s_{i-1,k} - s_{i,k}) + \omega_j s_{i-1,j} + (1-\omega_j) s_j \tag{3.107}$$

$$x_{1,1} \leq d_1 - \sum_{j=2}^{M} (\omega_j s_{1,j} + (1-\omega_j) s_j) \tag{3.108}$$

$$x_{i,1} \leq d_i - \sum_{j=2}^{M} (\omega_j s_{i,j} + (1-\omega_j) s_j) \tag{3.109}$$

for all $i = 2, ..., N$ and $j = 3, ..., M$. In this formulation, there are $N(1 + \sum_{j=1}^{M} \omega_j) + (M - \sum_{j=1}^{M} \omega_j)$ variables, one equality and $(N-1)(M+1) + N$ inequality constraints excluding the boundary value constraints on the service times. Therefore, compared to the $\bar{P}_M$ problem (incorporating the fact that the first job observes no waiting) with $N(M + \sum_{j=1}^{M} \omega_j) + (M - \sum_{j=1}^{M} \omega_j)$ decision

variables, $M$ equality, and $2M(N-1)+N$ inequality constraints (excluding the boundary value constraints), improvements in solution times and memory requirements are expected. (Note that the constraint set in (3.107) exists only if $M > 2$.)

Note that the convex optimization problem $\tilde{Q}$ has a larger feasible set compared to the optimization problem $\tilde{P}_M$; hence its optimal cost $J^*_{\tilde{Q}}$ is upper bounded by the optimal cost of $\tilde{P}_M$ denoted by $\tilde{J}^*_M$, i.e., $J^*_{\tilde{Q}} \leq \tilde{J}^*_M$. However, we can easily show that any optimal solution $\left\{s^{\tilde{Q}}, x_1^{\tilde{Q}}\right\}$ for the convex optimization problem $\tilde{Q}$ satisfies

$$x^{\tilde{Q}}_{i,1} = \max\left(a_i, x^{\tilde{Q}}_{i-1,1}\right) + s^{\tilde{Q}}_{i,1}$$

for all $i = 2, ..., N$; therefore it is in the feasible set of $\tilde{P}_M$. Since the optimal solution of $\tilde{Q}$ is feasible for $\tilde{P}_M$, the optimal costs $J^*_{\tilde{Q}}$ and $\tilde{J}^*_M$ are equal, i.e., $J^*_{\tilde{Q}} = \tilde{J}^*_M$. Hence, the optimal solution for $\tilde{Q}$ is formed of the optimal service times for $\tilde{P}_M$ and the corresponding job departure times resulting from applying the optimal service times of $\tilde{P}_M$ evaluated through (3.96) and (3.97).

### 3.3.2 Flow Shop Systems Starting with Initially Controllable Portions

If the flow shop system starts out with an initially controllable portion, i.e., if $f > 1$, this initially controllable portion can be treated as a flow shop system with fixed service times. Hence, employing Lemma 3.3, we can write the departure times of the jobs from machine $(f-1)$ as

$$x_{i,f-1} = \max\left(a_i + \sum_{k=1}^{f-1} s_k, x_{i-1,f-1} + \max_{k=1,...,f-1} s_k\right). \tag{3.110}$$

In this case, we employ (3.94), (3.110), and Theorem 3.1 to reformulate the optimization problem formulation $P_M$ as the optimization problem $\hat{P}_M$:

$$\hat{P}_M : \min_{\substack{s_{i,j},s_j \geq S_j \\ i=1,...,N \\ j=1,...,M}} \left\{ \hat{J}_M = \left( \begin{array}{c} \sum_{i=1}^{N} \phi_i \left( x_{i,f} + \sum_{j>f} \left( \omega_j s_{i,j} + (1-\omega_j) s_j \right) \right) \\ + \sum_{i=1}^{N} \sum_{j \in S_F} \theta_j(s_{i,j}) + \sum_{j \in S_I} \theta_j(s_j) \end{array} \right) \right\} \tag{3.111}$$

subject to

$$x_{1,f-1} = a_1 + \sum_{k=1}^{f-1} s_k \tag{3.112}$$

$$x_{1,f} = a_1 + \sum_{k=1}^{f-1} s_k + s_{1,f} \tag{3.113}$$

$$x_{i,f-1} = \max \left( a_i + \sum_{k=1}^{f-1} s_k, x_{i-1,f-1} + \max_{k=1,...,f-1} s_k \right) \tag{3.114}$$

$$x_{i,f} = \max \left( x_{i,f-1}, x_{i-1,f} \right) + s_{i,f} \tag{3.115}$$

$$x_{i,f} \geq x_{i-1,f} + \omega_{f+1} s_{i-1,f+1} + (1-\omega_{f+1}) s_{f+1} \tag{3.116}$$

$$x_{i,f} \geq x_{i-1,f} + \sum_{k=f+1}^{j-1} \omega_k \left( s_{i-1,k} - s_{i,k} \right) + \omega_j s_{i-1,j} + (1-\omega_j) s_j \tag{3.117}$$

$$x_{1,f} \leq d_1 - \sum_{j>f} \left( \omega_j s_{1,j} + (1-\omega_j) s_j \right) \tag{3.118}$$

$$x_{i,f} \leq d_i - \sum_{j>f} \left( \omega_j s_{i,j} + (1-\omega_j) s_j \right) \tag{3.119}$$

for $i = 2, ..., N$ and $j = f+2, ..., M$. (Note that the constraint set in (3.116) exists only if $M > f$, while the constraint set in (3.117) exists only if $M > f+1$.)

By linearizing the *max* functions in (3.114) and (3.115), we get the convex optimization problem $\hat{Q}$ given as

$$\hat{Q} : \min_{\substack{s_{i,j},s_j \geq S_j \\ x_{i,f-1},x_{i,f} \\ i=1,...,N \\ j=1,...,M}} \left\{ J_{\hat{Q}} = \left( \begin{array}{c} \sum_{i=1}^{N} \phi_i \left( x_{i,f} + \sum_{j>f} \left( \omega_j s_{i,j} + (1-\omega_j) s_j \right) \right) \\ + \sum_{i=1}^{N} \sum_{j \in S_F} \theta_j(s_{i,j}) + \sum_{j \in S_I} \theta_j(s_j) \end{array} \right) \right\} \tag{3.120}$$

subject to

$$x_{1,f-1} = a_1 + \sum_{k=1}^{f-1} s_k \tag{3.121}$$

$$x_{1,f} = a_1 + \sum_{k=1}^{f-1} s_k + s_{1,f} \tag{3.122}$$

$$x_{i,f-1} \geq a_i + \sum_{k=1}^{f-1} s_k \tag{3.123}$$

$$x_{i,f-1} \geq x_{i-1,f-1} + s_l \tag{3.124}$$

$$x_{i,f} \geq x_{i,f-1} + s_{i,f} \tag{3.125}$$

$$x_{i,f} \geq x_{i-1,f} + s_{i,f} \tag{3.126}$$

$$x_{i,f} \geq x_{i-1,f} + \omega_{f+1} s_{i-1,f+1} + (1-\omega_{f+1}) s_{f+1} \tag{3.127}$$

$$x_{i,f} \geq x_{i-1,f} + \sum_{k=f+1}^{j-1} \omega_k \left(s_{i-1,k} - s_{i,k}\right) + \omega_j s_{i-1,j} + (1-\omega_j) s_j \tag{3.128}$$

$$x_{1,f} \leq d_1 - \sum_{j>f} \left(\omega_j s_{1,j} + (1-\omega_j) s_j\right) \tag{3.129}$$

$$x_{i,f} \leq d_i - \sum_{j>f} \left(\omega_j s_{i,j} + (1-\omega_j) s_j\right) \tag{3.130}$$

for all $i = 2, ..., N$, $l = 1, ..., f-1$ and $j = f+2, ..., M$. In this formulation there are $N(2+\sum_{j=1}^{M} \omega_j) + (M - \sum_{j=1}^{M} \omega_j)$ variables, two equality and $(N-1)(M+2)+N$ inequality constraints excluding the boundary value constraints on the service times. Therefore, compared to the $\bar{P}_M$ problem (incorporating the fact that the first job observes no waiting) with $N(M + \sum_{j=1}^{M} \omega_j) + (M - \sum_{j=1}^{M} \omega_j)$ decision variables, $M$ equality, and $2M(N-1) + N$ inequality constraints (excluding the boundary value constraints), improvements in solution times and memory requirements are expected. (Note that the constraint set in (3.127) exists only if $M > f$, while the constraint set in (3.128) exists only if $M > f+1$.)

Similarly, the convex optimization problem $\hat{Q}$ has a larger feasible set compared to the optimization problem $\hat{P}_M$; hence its optimal cost $J^*_{\hat{Q}}$ is upper bounded by the optimal cost of $\hat{P}_M$ denoted by $\hat{J}^*_M$, i.e., $J^*_{\hat{Q}} \leq \hat{J}^*_M$. Again, we can easily show that the optimal service times for $\hat{Q}$ is also the optimal solution for $\hat{P}_M$, as it is feasible for $\hat{P}_M$. Hence, it suffices to solve the convex optimization problem

$\hat{Q}$ to get the optimal service times for the optimization problem $\hat{P}_M$.

In the next section, we replace all initially controllable machines with uncontrollable machines and study the resulting flow shop systems formed of fully controllable and uncontrollable machines. Absence of initially controllable machines in the system enables us to present a decoupling property which allows for the decomposition of the resulting simplified convex optimization problem into smaller convex optimization problems. For systems with rare arrivals, the decomposition method improves the solutions times further, when it is faster to solve many smaller problems instead of solving one large problem.

## 3.4 Forward Decomposition Algorithm

In this section, we focus on a special case of flow shop systems consisting only of fully controllable and uncontrollable machines, i.e., the flow shop system with no initially controllable machines. Removing the initially controllable machines from the system, allows us to decouple the jobs into special structures that can be treated independently and so to decompose the simplified convex optimization problems derived in the previous section into smaller convex optimization problems, one for each of these independent structures. Since it solves a series of smaller convex optimization problems, the resulting decomposition method is expected to alleviate the memory requirements resulting in further improvements in solvable system sizes. This decomposition method also improves the solutions times when many smaller problems can be solved faster than one large problem.

We assume, for convenience, that the flow shop starts out with a fully controllable machine, i.e., $\{1\} \subset S_C$. The flow shop systems that start out with a sequence of uncontrollable machines can easily be reduced to our setting by adjusting the arrival times via (3.3) and (3.5) and removing the uncontrollable machines upstream to the first fully controllable machine. Moreover, the cost incurred by uncontrollable machines does not affect the optimization. Hence, recalling that the service time at some machine $j \in S_U$ is denoted by $\breve{s}_j$, we modify

the optimization problem $\tilde{P}_M$ in (3.95) as

$$\breve{P}_M : \min_{\substack{s_{i,j} \geq S_j \\ i=1,...,N \\ j \in S_F}} \left\{ \breve{J}_M = \sum_{i=1}^{N} \sum_{j \in S_F} \theta_j(s_{i,j}) + \sum_{i=1}^{N} \phi_i \left( x_{i,1} + \sum_{j=2}^{M} (\omega_j s_{i,j} + (1-\omega_j)\breve{s}_j) \right) \right\} \tag{3.131}$$

subject to

$$\begin{aligned}
x_{1,1} &= a_1 + s_{1,1} && (3.132)\\
x_{i,1} &= \max(a_i, x_{i-1,1}) + s_{i,1} && (3.133)\\
x_{i,1} &\geq x_{i-1,1} + \omega_2 s_{i-1,2} + (1-\omega_2)\breve{s}_2 && (3.134)\\
x_{i,1} &\geq x_{i-1,1} + \sum_{k=2}^{j-1} \omega_k (s_{i-1,k} - s_{i,k}) + \omega_j s_{i-1,j} + (1-\omega_j)\breve{s}_j && (3.135)\\
x_{1,1} &\leq d_1 - \sum_{j=2}^{M} (\omega_j s_{1,j} + (1-\omega_j)\breve{s}_j) && (3.136)\\
x_{i,1} &\leq d_i - \sum_{j=2}^{M} (\omega_j s_{i,j} + (1-\omega_j)\breve{s}_j) && (3.137)
\end{aligned}$$

for $i = 2, ..., N$ and $j = 3, ..., M$. Note that the constraint set in (3.135) exists only if $M > 2$.

Let us denote the cost incurred by the jobs $\{C_i\}_{i=k}^{n}$ as

$$J(k,n) = \sum_{i=k}^{n} \sum_{j \in S_F} \theta_j(s_{i,j}) + \sum_{i=k}^{n} \phi_i \left( x_{i,1} + \sum_{j=2}^{M} (\omega_j s_{i,j} + (1-\omega_j)\breve{s}_j) \right) \tag{3.138}$$

and define the convex optimization problem

$$Q(k,n) : \min_{\substack{s_{i,j} \geq S_j \\ x_{i,1} \\ i=k,...,n \\ j \in S_F}} J(k,n) \tag{3.139}$$

subject to

$$x_{k,1} = a_k + s_{k,1} \tag{3.140}$$

$$x_{i,1} \geq a_i + s_{i,1} \tag{3.141}$$

$$x_{i,1} \geq x_{i-1,1} + s_{i,1} \tag{3.142}$$

$$x_{i,1} \geq x_{i-1,1} + \omega_2 s_{i-1,2} + (1-\omega_2)\breve{s}_2 \tag{3.143}$$

$$x_{i,1} \geq x_{i-1,1} + \sum_{k=2}^{j-1} \omega_k \left(s_{i-1,k} - s_{i,k}\right) + \omega_j s_{i-1,j} + (1-\omega_j)\breve{s}_j \tag{3.144}$$

$$x_{k,1} \leq d_k - \sum_{j=2}^{M} \left(\omega_j s_{k,j} + (1-\omega_j)\breve{s}_j\right) \tag{3.145}$$

$$x_{i,1} \leq d_i - \sum_{j=2}^{M} \left(\omega_j s_{i,j} + (1-\omega_j)\breve{s}_j\right) \tag{3.146}$$

for $i = k+1, ..., n$ and $j = 3, ..., M$. Again, note that the constraint set in (3.144) exists only if $M > 2$. We can easily show that the convex optimization problem $Q(1, N)$ yields the optimal solution for $\breve{P}_M$. The simplified convex optimization problem $Q(1, N)$ has $N(1 + \sum_{j=1}^{M} \omega_j)$ decision variables, one equality and $(N-1)(M+1)+N$ inequality constraints (excluding the boundary value constraints). Compared to $\bar{P}_M$ (incorporating the fact that the first job observes no waiting) with $N(M+\sum_{j=1}^{M} \omega_j)$ decision variables, $M$ equality, and $2M(N-1)+N$ inequality constraints (excluding the boundary value constraints), $Q(1, N)$ is expected to be solved faster.

The decomposition algorithm that follows will require $Q(k, n)$ to have a unique optimal solution. A sufficient condition for this to be satisfied is given in the next assumption replacing Assumption 3.2.

**Assumption 3.3 :** *$\phi_i(\cdot)$, for $i = 1, ..., N$, is continuously differentiable, monotonically increasing, and strictly convex.*

**Lemma 3.7** *The convex optimization problem $Q(k, n)$ has a unique solution.*

**Proof.** The feasible set defined by the constraints (3.140)-(3.146) is convex.

Then a sufficient condition for $Q(k,n)$ to have a unique optimal solution is that the cost

$$J = \sum_{i=k}^{n} \sum_{j \in S_F} \theta_j(s_{i,j}) + \sum_{i=k}^{n} \phi_i(x_{i,M})$$

is strictly convex, where $x_{i,M}$ is given by

$$x_{i,M} = x_{i,1} + \sum_{j=2}^{M} \left(\omega_j s_{i,j} + (1-\omega_j)\breve{s}_j\right). \tag{3.147}$$

Let us define two distinct feasible solutions $y^1$ and $y^2$ such that

$$y^l = \left(x^l_{k,1}, ..., x^l_{n,1}, s^l_{k,1}, ..., s^l_{n,\hat{M}}\right)$$

for $l = 1, 2$, where $\hat{M}$ is the last fully controllable machine of the flow shop. Due to convexity of $\theta_j(\cdot)$ and $\phi_i(\cdot)$ (as in Assumptions 3.1 and 3.3), we can write for $0 < \delta < 1$ that

$$\delta J\left(y^1\right) + (1-\delta) J\left(y^2\right) \geq J\left(\delta y^1 + (1-\delta) y^2\right).$$

For strict inequality (and strict convexity) it suffices to show that for some $i$ and $j$, $s^1_{i,j} \neq s^2_{i,j}$ or $x^1_{i,M} \neq x^2_{i,M}$.

Since $y^1$ and $y^2$ are distinct, they should differ in at least one component. If $s^1_{i,j} \neq s^2_{i,j}$ for some $i$ and $j$, since $\theta_j(\cdot)$ is strictly convex by Assumption 3.1, strict inequality is obtained. If, on the other hand, $s^1_{i,j} = s^2_{i,j}$ for all $i$ and $j$, then for some $i$ we should have $x^1_{i,1} \neq x^2_{i,1}$. From (3.147), it follows that $x^1_{i,M} \neq x^2_{i,M}$. Since $\phi_i(\cdot)$ is strictly convex by Assumption 3.3, strict inequality is obtained. Hence, for distinct feasible solutions $y^1$ and $y^2$, we have

$$\delta J\left(y^1\right) + (1-\delta) J\left(y^2\right) > J\left(\delta y^1 + (1-\delta) y^2\right),$$

i.e., $J(\cdot)$ is strictly convex and, therefore, $Q(k,n)$ has a unique optimal solution. ■

In what follows we determine the unique optimal solution for $Q(1, N)$. Independent period structures, defined next, simplify our task.

**Definition 3.5** *A contiguous set of jobs* $\{C_i\}_{i=k}^{n}$ *is said to form an* independent period *for the system if*

*1)* $x_{k-1,1} < a_k$ *and* $x_{k-1,j} < a_k + \sum_{l=1}^{j-1} \omega_l S_l + (1-\omega_l)\breve{s}_l$ *for all* $j = 2, ..., M$*;*

*2)* $x_{n,1} < a_{n+1}$ *and* $x_{n,j} < a_{n+1} + \sum_{l=1}^{j-1} \omega_l S_l + (1-\omega_l)\breve{s}_l$ *for all* $j = 2, ..., M$*;*

*3) For all* $i \in \{k, ..., n-1\}$, $x_{i,1} \geq a_{i+1}$ *or* $x_{i,j} \geq a_{i+1} + \sum_{l=1}^{j-1} \omega_l S_l + (1-\omega_l)\breve{s}_l$ *for some* $j = 2, ..., M$*.*

**Definition 3.6** *An* independent period structure *for the system is a partition of jobs* $\{C_i\}_{i=1}^{N}$ *into independent periods.*

The next lemma presents the decoupling property between independent periods.

**Lemma 3.8** *Consider a contiguous job sequence* $\{C_i\}_{i=k}^{n}$ *forming an independent period on the optimal sample path. The optimal service times for these jobs do not depend on the arrival times* $\{a_1, ..., a_{k-1}, a_{n+1}, ..., a_N\}$ *and the job completion deadlines* $\{d_1, ..., d_{k-1}, d_{n+1}, ..., d_N\}$ *of the other jobs.*

**Proof.** From (3.44) and the independent period definition, we have for all $j = 1, ..., M-1$ that

$$x^*_{k-1,j+1} < a_k + \sum_{l=1}^{j} \omega_l S_l + (1-\omega_l)\breve{s}_l \leq x^*_{k,j}, \tag{3.148}$$

$$x^*_{k-1,1} < a_k \tag{3.149}$$

for $k > 1$ and

$$x^*_{n,j+1} < a_{n+1} + \sum_{l=1}^{j} \omega_l S_l + (1-\omega_l)\breve{s}_l \leq x^*_{n+1,j}, \tag{3.150}$$

$$x^*_{n,1} < a_{n+1} \tag{3.151}$$

for $n < N$. Note that the optimal departures satisfies (3.2)-(3.5); therefore from (3.42), (3.49), (3.55), (3.150), and (3.151), we have for $j = 1, ..., M-1$

$$\lambda^*_{n,j} + \upsilon^*_{n,j} = \upsilon^*_{n+1,j} + \lambda^*_{n,j+1} = \lambda^*_{n,j+1}$$

and

$$\lambda^*_{n,M} + \upsilon^*_{n,M} = \phi'_n(x^*_{n,M}) + \upsilon^*_{n+1,M} + \gamma^*_n = \phi'_n(x^*_{n,M}) + \gamma^*_n.$$

Hence, there is no dependence of $\lambda^*_{n,j}$ for $j = 1, ..., M$ to jobs $\{C_{n+1}, ..., C_N\}$, i.e., the information is not propagated in the backward direction between independent periods. Note that if $n = N$ then there is no need for checking the backward information propagation.

Similarly, let us employ inequalities (3.148) and (3.149) in (3.2) and (3.3) to observe that

$$\begin{aligned} x^*_{k,j} &= \max\left(x^*_{k,j-1}, x^*_{k-1,j}\right) + \omega_j s^*_{k,j} + (1-\omega_j)\breve{s}_j \\ &= x^*_{k,j-1} + \omega_j s^*_{k,j} + (1-\omega_j)\breve{s}_j. \end{aligned}$$

Hence, there is no dependence of $x^*_{k,j}$ for $j = 1, ..., M$ to jobs $\{C_1, ..., C_{k-1}\}$, i.e., the information is not propagated in the forward direction between independent periods. Note that if $k = 1$ then there is no need for checking the forward information propagation.

Hence, independent periods are decoupled from each other. ■

Let us assume that the optimal independent period structure is given. Let $B$ be the number of independent periods in this structure, and let $C_{k(b)}$ and $C_{n(b)}$ denote the first and the last jobs of the $b$th independent period where $b = 1, ..., B$.

We can rewrite $\breve{P}_M$ as

$$\breve{P}_M : \min_{\substack{s_{i,j} \geq S_j \\ i=1,...,N \\ j \in S_F}} \left\{ J = \sum_{b=1}^{B} J\left(k(b), n(b)\right) \right\}$$

subject to (3.132)-(3.137). By the independent period definition, the constraints (3.133)-(3.135) for $i = k(b)$ can be reduced to $x_{k(b),1} = a_{k(b)} + s_{k(b),1}$ which is satisfied by the optimal solution. Along with the decoupling property shown in Lemma 3.8, this allows for the decomposition of $\breve{P}_M$ into $B$ smaller optimization problems $P(k(b), n(b))$ defined as

$$P(k(b), n(b)) : \min_{\substack{s_{i,j} \geq S_j \\ i=k(b),...,n(b) \\ j \in S_F}} J\left(k(b), n(b)\right)$$

subject to

$$\begin{aligned}
x_{k(b),1} &= a_{k(b)} + s_{k(b),1} \\
x_{i,1} &= \max\left(a_i, x_{i-1,1}\right) + s_{i,1} \\
x_{i,1} &\geq x_{i-1,1} + \omega_2 s_{i-1,2} + (1-\omega_2)\breve{s}_2 \\
x_{i,1} &\geq x_{i-1,1} + \sum_{k=2}^{j-1} \omega_k \left(s_{i-1,k} - s_{i,k}\right) + \omega_j s_{i-1,j} + (1-\omega_j)\breve{s}_j \\
x_{k(b),1} &\leq d_{k(b)} - \sum_{j=2}^{M} \left(\omega_j s_{k(b),j} + (1-\omega_j)\breve{s}_j\right) \\
x_{i,1} &\leq d_i - \sum_{j=2}^{M} \left(\omega_j s_{i,j} + (1-\omega_j)\breve{s}_j\right)
\end{aligned}$$

for $i = k(b)+1, ..., n(b)$ and $j = 3, ..., M$. Note that, similarly, $Q(k(b), n(b))$ can easily be shown to have the same optimal solution as that of $P(k(b), n(b))$.

We denote the optimal solution of $Q(k, n)$ as $s^*_{i,j}(k, n)$ and $x^*_{i,1}(k, n)$, and the corresponding departure times as $x^*_{i,j}(k, n)$ for $i = k, ..., n$ and $j = 1, ..., M$. The following corollary follows from Lemma 3.8 and relates the optimal solution of $Q(k, n)$ to the optimal solution of $P_M$.

**Corollary 3.1** *If the job sequence* $\{C_i\}_{i=k}^{n}$ *forms an independent period on the optimal sample path, then the optimal solution of* $Q(k,n)$ *satisfies*

$$s_{i,j}^{*}(k,n) = s_{i,j}^{*},$$
$$x_{i,1}^{*}(k,n) = x_{i,1}^{*}$$

*for* $i = k, ..., n$ *and* $j = 1, ..., M$.

This corollary forms the basis for our decomposition algorithm.

So far we have shown that if the optimal independent period structure can be identified, $Q(1,N)$ can be decomposed into a set of smaller problems. The following lemma provides necessary and sufficient conditions for identifying the independent periods on the optimal sample path based on the solution of $Q(k,n)$.

**Lemma 3.9** *Let* $C_k$ *initiate an independent period on the optimal sample path. The job* $C_n$ $(k \leq n)$ *ends the independent period if and only if the following conditions are satisfied:*

*1.* $x_{n,1}^{*}(k,n) < a_{n+1}$ *and* $x_{n,j}^{*}(k,n) < a_{n+1} + \sum_{l=1}^{j-1} \omega_l S_l + (1-\omega_l)\breve{s}_l$ *for all* $j = 2, ..., M$*;*

*2. For all* $i \in \{k, ..., n-1\}$, $x_{i,1}^{*}(k,n) \geq a_{i+1}$ *or* $x_{i,j}^{*}(k,n) \geq a_{i+1} + \sum_{l=1}^{j-1} \omega_l S_l + (1-\omega_l)\breve{s}_l$ *for some* $j = 2, ..., M$.

**Proof.** (Necessity) Since jobs $\{C_i\}_{i=k}^{n}$ form an independent period on the optimal sample path, and since $x_{i,j}^{*}(k,n) = x_{i,j}^{*}$ by Corollary 3.1, the result follows from the independent period definition.

(Sufficiency) Assume that even though both conditions are satisfied, $C_n$ does not end the independent period, i.e., for some $n' > n$, jobs $\{C_i\}_{i=k}^{n'}$ form an independent period on the optimal sample path.

Let us define a solution for $Q(k, n')$ as

$$\hat{s}_{i,j} = \begin{cases} s^*_{i,j}(k,n), & i \in \{k, ..., n\} \text{ and } j \in S_F \\ s^*_{i,j}(k,n') + \tau, & i = n+1,\ j = 1 \\ s^*_{i,j}(k,n'), & \text{otherwise} \end{cases} \tag{3.152}$$

and

$$\hat{x}_{i,1} = \begin{cases} x^*_{i,1}(k,n), & i \in \{k, ..., n\} \\ x^*_{i,1}(k,n'), & i \in \{n+1, ..., n'\}, \end{cases} \tag{3.153}$$

where $\tau \geq 0$ is defined as

$$\tau = \max\left(x^*_{n,1}(k,n') - a_{n+1}, 0\right).$$

By the independent period definition, since $C_n$ is not the last job of the independent period, $x^*_{n,1}(k,n') \geq a_{n+1}$ or $x^*_{n,j}(k,n') \geq a_{n+1} + \sum_{l=1}^{j-1} \omega_l S_l + (1-\omega_l)\breve{s}_l$ for some $j = 2, ..., M$. It follows from the first condition that

$$s^*_{i,j}(k,n) \neq s^*_{i,j}(k,n') \tag{3.154}$$

or

$$x^*_{i,1}(k,n) \neq x^*_{i,1}(k,n') \tag{3.155}$$

for some $i = k, ..., n$ and $j = 1, ..., M$. Therefore, by Lemma 3.7, the solution given in (3.152) and (3.153) is not optimal for $Q(k, n')$.

Let us check the feasibility of this non-optimal solution: Since $\hat{s}_{i,j}$ and $\hat{x}_{i,1}$ for $i = k, ..., n$ and $j \in S_F$ are the solutions for the problem $Q(k, n)$, they satisfy the constraints (3.140)-(3.146) for $i = k+1, ..., n$ and $j = 3, ..., M$. For $i = n+1$, we

have

$$\begin{aligned}\hat{x}_{n+1,1} &= x^*_{n+1,1}(k,n') \\ &= \max\left(x^*_{n,1}(k,n'), a_{n+1}\right) + s^*_{n+1,1}(k,n') \\ &= a_{n+1} + \tau + s^*_{n+1,1}(k,n') \\ &= a_{n+1} + \hat{s}_{n+1,1} \\ &= \max\left(a_{n+1}, \hat{x}_{n,1}\right) + \hat{s}_{n+1,1}\end{aligned}$$

satisfying the constraints (3.141), (3.142), and (3.146). It follows from the first condition and from $\hat{s}_{n+1,j} \geq S_j$ for all $j \in S_F$ that the constraints (3.143) and (3.144) are also satisfied for $i = n+1$. Finally, since $\hat{s}_{i,j}$ and $\hat{x}_{i,1}$ for $i = n+2, ..., n'$ and $j \in S_F$ are the solutions for $Q(k,n')$, the constraints (3.141)-(3.144), and (3.146) are satisfied for $i = n+2, ..., n'$ and $j = 3, ..., M$. Hence, the non-optimal solution is feasible for the $Q(k,n')$ problem.

Let us recall from Corollary 3.1 that

$$\begin{aligned}s^*_{i,j} &= s^*_{i,j}(k,n'), \\ x^*_{i,1} &= x^*_{i,1}(k,n')\end{aligned}$$

for $i = k, ..., n'$ and $j \in S_F$, and consider the problem $Q(k,n')$. Its optimal cost can be written as

$$J^*(k,n') = J^*_A(k,n') + J^*_B(k,n'), \tag{3.156}$$

where

$$J^*_A(k,n') = \sum_{i=k}^{n} \sum_{j \in S_F} \theta_j(s^*_{i,j}) + \sum_{i=k}^{n} \phi_i \left( x^*_{i,1} + \sum_{j=2}^{M} (\omega_j s^*_{i,j} + (1-\omega_j)\breve{s}_j) \right)$$

and

$$J^*_B(k,n') = \sum_{i=n+1}^{n'} \sum_{j \in S_F} \theta_j(s^*_{i,j}) + \sum_{i=n+1}^{n'} \phi_i \left( x^*_{i,1} + \sum_{j=2}^{M} (\omega_j s^*_{i,j} + (1-\omega_j)\breve{s}_j) \right).$$

The cost $\hat{J}(k,n')$ due to applying the non-optimal solution in (3.152) and (3.153)

can be written as

$$\hat{J}(k,n') = J^*(k,n) + \hat{J}_C, \tag{3.157}$$

where

$$\hat{J}_C = J^*_B(k,n') + \theta_1\left(s^*_{n+1,1}(k,n') + \tau\right) - \theta_1\left(s^*_{n+1,1}(k,n')\right).$$

Since the optimal solution of $Q(k,n)$ is unique by Lemma 3.7, from (3.154) and (3.155),

$$J^*_A(k,n') > J^*(k,n). \tag{3.158}$$

Moreover, due to Assumption 3.1,

$$J^*_B(k,n') \geq \hat{J}_C, \tag{3.159}$$

which also accounts for the case that $\tau = 0$. From (3.156), (3.157), (3.158), and (3.159),

$$J^*(k,n') > \hat{J}(k,n'),$$

in other words, the cost of the non-optimal solution is lower than the optimal cost, which is a contradiction. Hence, the result follows. ■

Lemma 3.9 allows us to identify independent periods on the optimal sample path. If $C_k$ is the first job of an independent period, we can identify this independent period by sequentially solving $Q(k,n)$ and checking for all jobs $\{C_i\}_{i=k}^n$ to see if both conditions in Lemma 3.9 are satisfied.

The next lemma indicates that checking only for job $C_n$ to see if the first condition in Lemma 3.9 is satisfied suffices to identify the independent period.

**Lemma 3.10** *Let $C_k$ initiate an independent period on the optimal sample path. For all $i \geq k$, if $x^*_{i,1}(k,i) \geq a_{i+1}$ or $x^*_{i,j}(k,i) \geq a_{i+1} + \sum_{l=1}^{j-1}\omega_l S_l + (1-\omega_l)\breve{s}_l$ for some $j = 2,...,M$, then for all $n \geq i$, $x^*_{i,1}(k,n) \geq a_{i+1}$ or $x^*_{i,j}(k,n) \geq a_{i+1} + \sum_{l=1}^{j-1}\omega_l S_l + (1-\omega_l)\breve{s}_l$ for some $j = 2,...,M$.*

**Proof.** (By contradiction) Let us assume that there exists $n > i$ such that $x^*_{i,1}(k,n) < a_{i+1}$ and $x^*_{i,j}(k,n) < a_{i+1} + \sum_{l=1}^{j-1}\omega_l S_l + (1-\omega_l)\breve{s}_l$ for all $j = 2,...,M$.

In that case job $C_i$ ends an independent period (not necessarily the one that was started by job $C_k$). From the decoupling property in Lemma 3.8 and Corollary 3.1, we should have

$$x^*_{i,j}(k,i) = x^*_{i,j}(k,n).$$

However, since $x^*_{i,1}(k,i) \geq a_{i+1}$ or $x^*_{i,j}(k,i) \geq a_{i+1} + \sum_{l=1}^{j-1} \omega_l S_l + (1-\omega_l)\check{s}_l$ for some $j = 2, ..., M$, a contradiction is observed. Hence the result follows. ■

Lemma 3.10 asserts that an independent period formed by jobs $\{C_i\}_{i=k}^n$ is identified as soon as the problem $Q(k,n)$ is solved. This result is formalized in the following theorem.

**Theorem 3.2** *Jobs $\{C_i\}_{i=k}^n$ form an independent period on the optimal sample path if and only if the following conditions are satisfied:*

*1) $x^*_{k-1,1} < a_k$ and $x^*_{k-1,j} < a_k + \sum_{l=1}^{j-1} \omega_l S_l + (1-\omega_l)\check{s}_l$ for all $j = 2, ..., M$;*

*2) For all $i \in \{k, ..., n-1\}$, $x^*_{i,1}(k,i) \geq a_{i+1}$ or $x^*_{i,j}(k,i) \geq a_{i+1} + \sum_{l=1}^{j-1} \omega_l S_l + (1-\omega_l)\check{s}_l$ for some $j = 2, ..., M$;*

*3) $x^*_{n,1}(k,n) < a_{n+1}$ and $x^*_{n,j}(k,n) < a_{n+1} + \sum_{l=1}^{j-1} \omega_l S_l + (1-\omega_l)\check{s}_l$ for all $j = 2, ..., M$.*

**Proof.** It follows from Lemma 3.9, Lemma 3.10, and the independent period definition. ■

Our forward decomposition algorithm is based on Theorem 3.2. While identifying the optimal independent period structure, this algorithm also determines the optimal solution.

**Algorithm 3.1** *Step 1: (initialization) $k = 1, n = 1, a_{N+1} = \infty$.*

*while $n \leq N$ do*

*Step 2: Solve subproblem $Q(k,n)$.*

*Step 3: (identify independent periods)*

*If* $x^*_{n,1}(k,n) < a_{n+1}$ *and* $x^*_{n,j}(k,n) < a_{n+1} + \sum_{l=1}^{j-1} \omega_l S_l + (1-\omega_l)\breve{s}_l$ *for all* $j = 2$ *to* $M$*, then*

$s^*_{i,j} = s^*_{i,j}(k,n)$ *for* $i = k, ..., n$ *and* $j \in S_F$;

$k = n + 1$;

*endif.*

*Step 4: (increment index* $n$*)*

$n = n + 1$.

Note that this decomposition algorithm requires only $N$ iterations. However, these iterations are not identical in complexity and depend on the arrival sequence along with the cost parameters. The best case for this algorithm would be an optimal sample path where each job forms an independent period of its own. In this case, $Q(i,i)$ for all $i = 1, ..., N$ are solved. The worst case for this algorithm, on the other hand, would be an optimal sample path where all jobs reside in the same independent period and no decomposition is observed. In the worst case, we solve $Q(1,i)$ for all $i = 1, ..., N$. If the number of independent periods expected is low, e.g. for the bulk arrivals case we have only one independent period, we may choose to solve $Q(1,N)$ directly.

## 3.5 Numerical Study

Let us consider an $M$-machine flow shop system processing an identical set of $N$ jobs. The service cost $\theta_j(s)$ for job $C_i$ at the fully controllable machine $j$ is given as

$$\theta_j(s_{i,j}) = \frac{\beta_j}{s_{i,j}}, \tag{3.160}$$

while the service cost $\theta_j(s_j)$ at initially controllable machine $j$ is given as

$$\theta_j(s_j) = \frac{\beta_j}{s_j} \tag{3.161}$$

for some constant $\beta_j$. The completion-time cost for job $C_i$, on the other hand, is given by a cost defined as

$$\phi_i(x_{i,M}) = \alpha(x_{i,M} - a_i)^2 \tag{3.162}$$

for a positive parameter $\alpha$. Note that the service costs given by (3.160) and (3.161) are continuously differentiable, monotonically decreasing, and strictly convex satisfying Assumption 3.1. Similarly, the completion-time cost given by (3.162) is continuously differentiable, monotonically increasing, and strictly convex for $x_{i,M} \geq a_i$, hence, satisfies Assumption 3.2 (as well as 3.3).

### 3.5.1 Verification of the Optimal Waiting Characteristics

To illustrate the optimal waiting characteristics of mixed line flow shop systems, derived throughout the chapter, we consider the system in Chapter 2, where ten jobs are to be processed in a flow shop of four machines, i.e., $M = 4$ and $N = 10$, with the costs given by (3.160), (3.161), and (3.162) where $\alpha = 10$. The arrival times of the jobs are given as $a = [0.0, 2.3, 2.4, 4.9, 5.0, 5.5, 9.0, 9.5, 11.0, 13.0]$ while the binary parameter vector $\omega$ defined in (3.93) is given as $\omega = [1, 0, 0, 1]$, where the second and the third machines are initially controllable machines, i.e., $\{2, 3\} \in S_I$. The service times on these four machines are bounded below by $S = [0.20, 0.20, 0.30, 0.35]$. Since ten jobs are to be processed, the $\beta$ parameter vector in Chapter 2 is adjusted to become $\beta = [10, 50, 200, 10]$.

The system is solved to yield the optimal service times and the optimal departure times, given in Tables 3.1 and 3.2, respectively.

| Service Times | Machine 1 | Machine 2 | Machine 3 | Machine 4 |
|---|---|---|---|---|
| **Job 1** | 0.5032 | 0.3502 | 0.6179 | 0.5032 |
| **Job 2** | 0.3476 | 0.3502 | 0.6179 | 0.5217 |
| **Job 3** | 0.6179 | 0.3502 | 0.6179 | 0.4663 |
| **Job 4** | 0.2803 | 0.3502 | 0.6179 | 0.5302 |
| **Job 5** | 0.6179 | 0.3502 | 0.6179 | 0.4726 |
| **Job 6** | 0.6179 | 0.3502 | 0.6179 | 0.4617 |
| **Job 7** | 0.4533 | 0.3502 | 0.6179 | 0.5089 |
| **Job 8** | 0.5712 | 0.3502 | 0.6179 | 0.4957 |
| **Job 9** | 0.5032 | 0.3502 | 0.6179 | 0.5032 |
| **Job 10** | 0.5032 | 0.3502 | 0.6179 | 0.5032 |

Table 3.1: Optimal Service Times

| | Arrival | Machine 1 | Machine 2 | Machine 3 | Machine 4 |
|---|---|---|---|---|---|
| **Job 1** | 0.0000 | 0.5032 | 0.8534 | 1.4713 | 1.9745 |
| **Job 2** | 2.3000 | 2.6476 | 2.9978 | 3.6157 | 4.1374 |
| **Job 3** | **2.4000** | 3.2655 | 3.6157 | 4.2336 | 4.6998 |
| **Job 4** | 4.9000 | 5.1803 | 5.5305 | 6.1484 | 6.6786 |
| **Job 5** | **5.0000** | 5.7982 | 6.1483 | 6.7662 | 7.2388 |
| **Job 6** | **5.5000** | 6.4161 | 6.7662 | 7.3841 | 7.8458 |
| **Job 7** | 9.0000 | 9.4533 | 9.8035 | 10.4214 | 10.9303 |
| **Job 8** | 9.5000 | 10.0712 | 10.4214 | 11.0393 | 11.5350 |
| **Job 9** | 11.0000 | 11.5032 | 11.8534 | 12.4713 | 12.9745 |
| **Job 10** | 13.0000 | 13.5032 | 13.8534 | 14.4713 | 14.9745 |

Table 3.2: Optimal Departure Times

Since, the system starts out with a fully controllable machine, from Theorem 3.1, we expect to see no waiting after the first machine. The optimal departure times, given in Table 3.2, meet this expectation. (Note that boldface numbers denote waiting after departure.)

The optimal departure times, given in Table 3.2, also reveal that there are only two blocks formed of more than one job at the first machine on the optimal sample path: $\{C_2, C_3\}$ and $\{C_4, C_5, C_6\}$. Hence, since the first machine is fully controllable, from Lemma 3.5, we expect to see $s^*_{2,1} \leq s^*_{3,1}$ and $s^*_{4,1} \leq s^*_{5,1} \leq s^*_{6,1}$. Note that these expectations are verified by the optimal service times given in Table 3.1.

The optimal cost for this system is given as 1299.45 compared to 1329.01 given in Chapter 2. The cost difference is due to having the flexibility to adjust the service times between processes for fully controllable machines in the system. It may be viewed as the benefit to be gained by replacing the non-CNC machines by CNC machines.

### 3.5.2 Analysis of the Replacement of Initially Controllable Machines with Fully Controllable Machines

To further analyze the effects of replacing an initially controllable (non-CNC) machine with a fully controllable (CNC) machine, we consider a set of 4-machine flow shop systems processing 100 jobs. The job interarrival times are realized from an exponential distribution with a mean of 1 unit. We employ the cost structures given by (3.160)-(3.162) with $\alpha = 10$ and different choices of $\beta$. For simplicity, we set $S_j = 0$ for $j = 1, ..., 4$ and $d_i = \infty$ for $i = 1, ..., 100$. Table 3.3 shows the corresponding optimal costs for different machine replacement configurations. Note that a bold face number in Table 3.3 is the minimum of its column so that it stands for the best replacement action for the corresponding system in terms of cost reductions.

| | $\boldsymbol{\beta}$ | | | |
|---|---|---|---|---|
| $\boldsymbol{\omega}$ | $[\mathbf{10, 5, 20, 10}]$ | $[\mathbf{5, 10, 40, 30}]$ | $[\mathbf{5, 10, 40, 10}]$ | $[\mathbf{5, 10, 10, 30}]$ |
| $[\mathbf{0, 0, 0, 0}]$ | 13776.03 | 20856.56 | 17438.70 | 15697.69 |
| $[\mathbf{0, 0, 0, 1}]$ | 13768.10 | 20740.50 | 17421.76 | **15199.67** |
| $[\mathbf{0, 0, 1, 0}]$ | 13519.69 | 20379.00 | **16751.32** | 15323.33 |
| $[\mathbf{0, 1, 0, 0}]$ | 13573.82 | **20145.88** | 16848.75 | 15239.53 |
| $[\mathbf{1, 0, 0, 0}]$ | **13454.62** | 20184.63 | 16856.33 | 15209.23 |

Table 3.3: Optimal Costs of Different Replacement Actions for Alternative Systems

We note that the $\beta$ vectors in Table 3.3 are given as if all machines in the system are fully controllable. Hence, if some machine $j$ is initially controllable, i.e., $\omega_j = 0$, since 100 jobs are to be processed, the $\beta_j$ value in Table 3.3 is adjusted accordingly so that it becomes $100 * \beta_j$.

Due having flexibility to adjust the service times between processes for a fully controllable machine, we expect cost reductions by replacing an initially controllable machine with its fully controllable counterpart. Table 3.3 meets this expectation: independent of its location and $\beta_j$ value, replacement action of any initially controllable machine $j$ always reduces the cost. The second question for which we look an answer is the choice of the initially controllable machine to be replaced for the highest cost reduction. Table 3.3 reveals that the location of the initially controllable machine to be replaced for the highest cost reduction relies on the machine configurations in the system: replacement of initially controllable machines with four different locations turned out to be the best policy in terms of cost reductions for four different system configurations. Hence, there is no simple rule for the best replacement policy in terms of cost reductions.

### 3.5.3 Analysis of the Effects of the Locations of Fully Controllable Machines

To analyze the effects of the locations of fully controllable machines in the system, we consider a 3-machine flow shop system processing 100 jobs. The job interarrival times are realized from an exponential distribution with a mean of 1 unit and the cost functions in (3.160)-(3.162) with $\alpha = 10$ and $\beta = [5, 10, 20]$ are used. However, we should note that, for some initially controllable machine $j$, we adjust its $\beta_j$ value accordingly so that it becomes $100 * \beta_j$. Finally, we set $S_j = 0$ for $j = 1, ..., 4$ and $d_i = \infty$ for $i = 1, ..., 100$. Tables 3.4 and 3.5 represent the optimal costs for each possible layout configuration with at least one fully controllable and one initially controllable machine for this system. We note that $\beta_j^*$ indicates that machine $j$ is fully controllable and the optimal cost of the cost minimizing layout configuration for the corresponding system is represented by a bold face character in these tables.

When we examine the optimization problems $\tilde{P}_M$ in (3.95) and $\hat{P}_M$ in (3.111) presented in Section 3.3, a change in the order of the initially controllable machines within an initially controllable portion does not change the overall problem.

| $\boldsymbol{\beta}$ | **Optimal Cost** | $\boldsymbol{\beta}$ | **Optimal Cost** | $\boldsymbol{\beta}$ | **Optimal Cost** |
|---|---|---|---|---|---|
| [5,10,20*] | 9649.43 | [5,20,10*] | 9914.59 | [10,20,5*] | 9916.53 |
| [10,5,20*] | 9649.43 | [20,5,10*] | 9914.59 | [20,10,5*] | 9916.53 |
| [5,20*,10] | 9612.74 | [5,10*,20] | 9659.47 | [10,5*,20] | 9744.13 |
| [10,20*,5] | 9649.43 | [20,10*,5] | 9914.59 | [20,5*,10] | 9921.15 |
| [20*,5,10] | **9600.09** | [10*,5,20] | **9629.94** | [5*,10,20] | **9565.30** |
| [20*,10,5] | **9600.09** | [10*,20,5] | **9629.94** | [5*,20,10] | **9565.30** |

Table 3.4: Optimal Costs for Different Layout Configurations with only One Fully Controllable Machine

| $\boldsymbol{\beta}$ | **Optimal Cost** | $\boldsymbol{\beta}$ | **Optimal Cost** | $\boldsymbol{\beta}$ | **Optimal Cost** |
|---|---|---|---|---|---|
| [5,10*,20*] | 9416.74 | [10,5*,20*] | 9536.59 | [20,5*,10*] | 9907.00 |
| [5,20*,10*] | 9552.68 | [10,20*,5*] | 9642.04 | [20,10*,5*] | 9905.48 |
| [10*,5,20*] | **9386.96** | [5*,10,20*] | 9348.90 | [5*,20,10*] | 9554.48 |
| [20*,5,10*] | 9548.66 | [20*,10,5*] | 9548.66 | [10*,20,5*] | 9619.71 |
| [10*,20*,5] | **9386.96** | [5*,20*,10] | **9320.37** | [5*,10*,20] | **9381.13** |
| [20*,10*,5] | 9548.66 | [20*,5*,10] | 9557.98 | [10*,5*,20] | 9490.37 |

Table 3.5: Optimal Costs for Different Layout Configurations with Two Fully Controllable Machines

Hence, the solution of the system will not be affected by the change of the layout inside an initially controllable portion. The results in Table 3.4 justify this property. Actually, the same property applies to the fixed service time flow shop systems studied in Chapter 2: even though we change the order of the machines in the system, the optimization problem $P$ in (2.49) remains the same. However, this property does not extend to fully controllable portions as seen in Table 3.5.

We can deduce from Tables 3.4 and 3.5 that the best layout scheme is the one where a fully controllable portion formed of the fully controllable machines of the system located in the increasing order of their $\beta$ values is followed by an initially controllable portion formed of the initially controllable machines of the system with an arbitrary order. This result is expected for flow shops with the cost structures in (3.160)-(3.162). Smaller $\beta$ values mean smaller service times and a fully controllable machine prevents buffering at its downstream. Hence, locating fully controllable machines before initially controllable machines in the increasing order of $\beta$ values will give us a chance to adjust the service times so

that waiting times for jobs are utilized as processing times as much as possible leading to decreased service costs.

### 3.5.4 Analysis of the Relative Effects of Service and Completion-Time Costs on the Optimal Solution

To analyze the relative effects of service and completion-time costs on the optimal service times and the optimal cost, we consider the same flow shop system in Section 3.5.1 with all machines being fully controllable, i.e., $\omega = [1, 1, 1, 1]$. In order to visualize the tail behaviors of optimal service times, we set lower bounds for service times at each machine to zero, i.e., $S_j = 0$ for $j = 1, ..., 4$. We use the same costs given by (3.160) and (3.162) with $\alpha = 10$, $\beta = [10, 5, 20, 10]$, and $a = [0.0, 2.3, 2.4, 4.9, 5.0, 5.5, 9.0, 9.5, 11.0, 13.0]$. Assigning relative weights to total service and completion-time costs which add up to 1, we optimize the following weighted cost objectives:

$$J_{\lambda_i}(s) = \lambda_i \sum_{i=1}^{N} \sum_{j=1}^{M} \theta_j(s_{i,j}) + (1 - \lambda_i) \sum_{i=1}^{N} \phi_i(x_{i,M})$$

where $\lambda_i = 0.01i$ for $i = 1, ..., 99$. Figure 3.1 represents the optimal service times (averaged over ten jobs) at machines with respect to the weight of total service cost.

It can be inferred from Figure 3.1 that, not surprisingly, increase in the weight of the total completion-time cost leads to a decrease in the optimal service times up to their lower bounds. On the other hand, as the total service cost becomes more dominant, optimal service times are forced to increase as much as possible so that job completion times become as close as possible to their deadlines.

In order to analyze the individual effects of the service costs at machines on the optimal service times, we consider the same flow shop system above for $\beta_2 = 5 + 10i$ for all $i = 1, ..., 100$. Figure 3.2 represents the optimal service times (averaged over ten jobs) at machines versus increasing $\beta_2$ values.

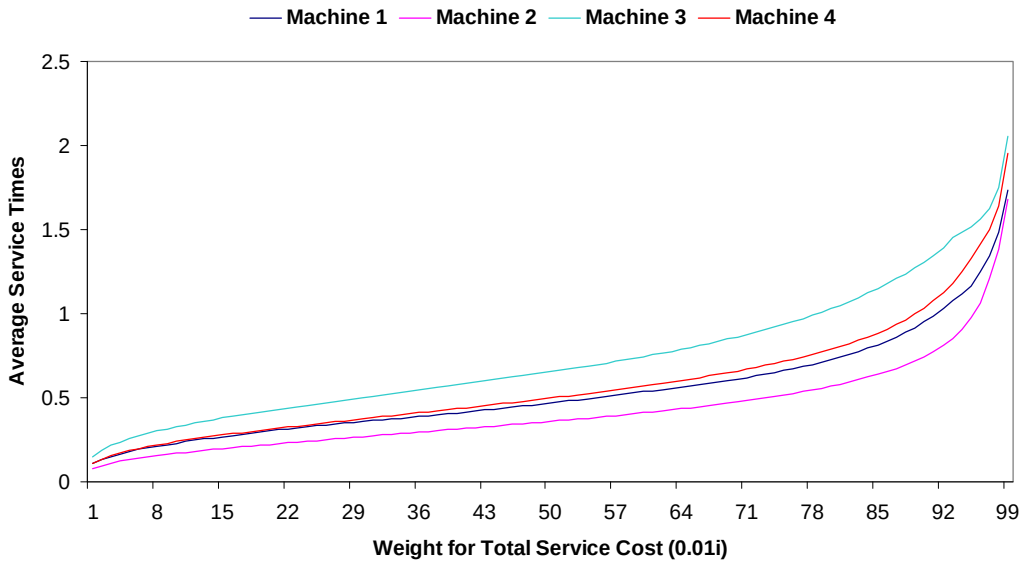


Figure 3.1: Optimal Service Times Averaged over All Jobs at Machines versus Weight of Total Service Cost

For the service costs in (3.160), if we increase the $\beta$ value of a machine, while keeping all other parameters the same, we expect increased service times for this machine. To eliminate possibly increased waiting times at this machine, increased service times at all upstream machines are also expected. On the other hand, in order to reduce job completion-time costs and to satisfy job completion deadlines, we expect decreased service times for the downstream machines. These expectations are met by Figure 3.2: increase in $\beta_2$ led the optimal service times to increase at machines 1 and 2, and to decrease at machines 3 and 4.

### 3.5.5 Comparison of Different Solution Methodologies

In order to compare the solution performances of different solution methodologies, we study problems with different choices of $M$ and $N$. The $\beta_j$ values are randomly selected from the set $\beta = \{5i : i = 1, ..., 10\}$ and the job interarrival times are realized from an exponential distribution with a mean of 2 units.

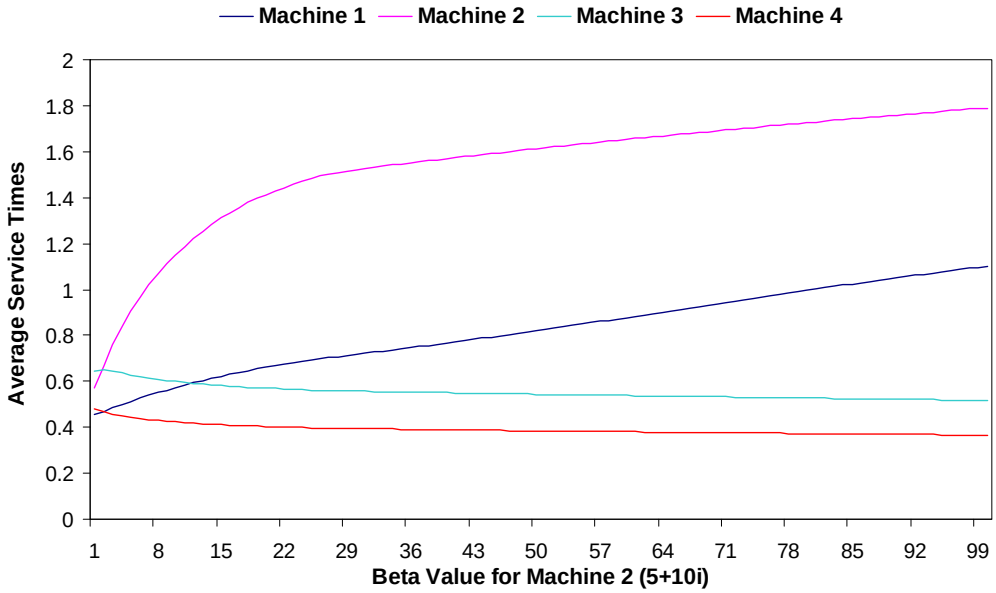


Figure 3.2: Optimal Service Times Averaged over All Jobs at Machines versus $\beta_2$ value at Machine 2

The computation times are reported from a computer with 2.0GHz Intel Core2Duo T7200 processor and 2GB of RAM running Matlab.

#### 3.5.5.1 $\bar{P}_M$ Formulation vs $Q(1, N)$ Formulation and Forward Decomposition Algorithm

In order to understand the effects of the simplifications and the decomposition on solution times, we study the flow shop systems having equal number of fully controllable and uncontrollable machines, i.e., each system is formed of $M/2$ fully controllable and $M/2$ uncontrollable machines, with randomized locations. The $S_j$ and $\breve{s}_k$ values, for $j \in S_F$ and $k \in S_I$, are randomly selected from the sets $S = \{0.05 + 0.05i : i = 1, ..., 5\}$ and $\breve{s} = \{0.15 + 0.05i : i = 1, ..., 5\}$, respectively. To guarantee feasibility, the deadlines are determined by the equation

$$d_i = a_i + \gamma_i \max_{j \in S_F, k \in S_U} \{S_j, \breve{s}_k\},$$

where the coefficients $\gamma_i$, $i = 1, ..., N$, are uniformly realized from the interval $[1.25, 1.50]$.

The problems are solved by *cvx* (see [17]), a modeling system for convex programming developed in Stanford University, implemented in Matlab. The computation times, based on the averages over ten optimization problems (obtained by varying parameter sets $\{a_i\}_{i=1}^N$, $\{d_i\}_{i=1}^N$, $\{\beta_j\}_{j\in S_F}$, $\{S_j\}_{j\in S_F}$, and $\{\breve{s}_k\}_{k\in S_U}$), of the alternative methodologies for different $M$ and $N$ settings are presented in Table 3.6, where a dash sign indicates an "out of memory" crash. Due to space limitations, the resulting optimal service and departure times are not reported here. However, as expected, no waiting is observed after the first (fully controllable) machine of the system.

| | **M = 20** | | | **M = 30** | | | **M = 40** | | |
|---|---|---|---|---|---|---|---|---|---|
| **N** | $\bar{\mathbf{P}}_{\mathbf{M}}$ | $\mathbf{Q(1,N)}$ | **FDA** | $\bar{\mathbf{P}}_{\mathbf{M}}$ | $\mathbf{Q(1,N)}$ | **FDA** | $\bar{\mathbf{P}}_{\mathbf{M}}$ | $\mathbf{Q(1,N)}$ | **FDA** |
| **50** | 26.8 | 25.2 | 49.0 | 51.3 | 45.3 | 68.1 | 83.7 | 72.8 | 88.4 |
| **100** | 82.8 | 70.8 | 104.3 | 175.1 | 151.0 | 145.1 | - | 257.6 | 188.3 |
| **150** | 173.3 | 144.9 | 160.7 | - | 328.9 | 224.1 | - | 582.0 | 289.2 |
| **200** | - | 246.0 | 214.4 | - | 540.1 | 296.4 | - | - | 381.5 |
| **250** | - | 376.5 | 267.7 | - | - | 370.3 | - | - | 478.2 |
| **300** | - | 536.6 | 319.3 | - | - | 438.9 | - | - | 570.2 |
| **350** | - | - | 370.3 | - | - | 506.7 | - | - | 658.3 |

Table 3.6: Computation Times for $\bar{P}_M$ and $Q(1, N)$ Formulations, and Forward Decomposition Algorithm (in seconds)

As expected, the methodology solving $Q(1, N)$ outperforms the methodology solving $\bar{P}_M$; the former is not only faster but also allows us to solve larger problems.

Forward Decomposition Algorithm (FDA) outperforms the $Q(1, N)$ solution methodology in terms of solution times for large and uncongested systems where several independent periods are observed. Moreover, due to solving several smaller problems, memory is no longer an active constraint. For congested or smaller systems, though, solving $Q(1, N)$ should be preferred. For the bulk arrival case, for example, instead of solving just the $Q(1, N)$ problem, FDA solves $Q(1, i)$ for $i = 1, ..., N$; hence, not only it takes longer to obtain the result, but also no memory benefit is observed.

#### 3.5.5.2 Forward Decomposition Algorithm vs Virtual Deadline Algorithm

We also compare the solution performance of Forward Decomposition Algorithm (FDA) with recently developed Virtual Deadline Algorithm (VDA) by Mao and Cassandras [31]. The main idea of this algorithm is to introduce *virtual* deadlines at each machine except the last one so that the $M$-machine problem is replaced by $M$ single machine problems which can be solved very effectively by existing solution methodologies such as *Critical Task Decomposition Algorithm* (CTDA) developed in [32]. However, determination of the appropriate virtual deadlines for each machine requires the solution of additional $N+M-1$ simple $M$-dimensional convex optimization problems. The sequence of solutions to simple single machine problems with virtual deadlines updated through solving these $M$-dimensional convex optimization problems at each step was shown to converge to the global optimal solution of the original problem.

In particular, VDA is a solution procedure for flow shop systems consisting only of fully controllable machines where the job completion-time costs are not included. To notice that, in our model, we may also have uncontrollable machines in the system and we include job completion-time costs in the objective function. However, VDA is also applicable for non-homogeneous service cost structures, i.e., the service cost functions may be different both for each machine and each job, while we assign the same service cost functions to all jobs at fully controllable machines, i.e., homogeneous service costs for all jobs at fully controllable machines.

We study the flow shop systems consisting only of fully controllable machines by removing the job completion-time costs from the objective function so that VDA becomes applicable. The deadlines are determined by the equation

$$d_i = a_i + M * \gamma_i,$$

where the coefficients $\gamma_i$, $i = 1, ..., N$, are uniformly realized from the interval $[0.25, 0.30]$. We note that feasibility is guaranteed by these deadlines. Since the

original problem is feasible, the corresponding $M$ single machine problems solved by VDA are also feasible. It was shown in [32] that if a single machine problem is feasible, then the lower-bound constraints on service times can be replaced by simple non-negativity constraints. Hence, we set $S_j = 0$ for $j = 1, ..., M$.

The computation times, based on the averages over ten optimization problems (obtained by varying arrival $\{a_i\}_{i=1}^N$, and deadline $\{d_i\}_{i=1}^N$ sequences, and the cost parameters $\{\beta_j\}_{j=1}^M$) for each $M$ and $N$ combination, for both algorithms are presented in Table 3.7.

| | **M = 10** | | **M = 20** | | **M = 30** | | **M = 40** | |
|---|---|---|---|---|---|---|---|---|
| **N** | **VDA** | **FDA** | **VDA** | **FDA** | **VDA** | **FDA** | **VDA** | **FDA** |
| **50** | 6.31 | 27.15 | 93.71 | 41.73 | 440.54 | 57.54 | 1363.47 | 74.53 |
| **100** | 12.32 | 54.07 | 189.55 | 83.39 | 862.48 | 115.31 | 2774.17 | 148.02 |
| **150** | 18.68 | 82.79 | 287.98 | 127.91 | 1356.34 | 177.40 | 4134.87 | 228.60 |
| **200** | 24.85 | 109.44 | 379.03 | 168.72 | 1806.78 | 238.15 | 5528.90 | 301.27 |
| **250** | 31.14 | 136.14 | 476.16 | 209.03 | 2279.76 | 294.52 | 6914.59 | 375.09 |
| **300** | 37.31 | 163.43 | 581.60 | 249.37 | 2866.37 | 352.80 | 8273.62 | 451.50 |
| **350** | 43.56 | 191.93 | 677.04 | 291.58 | 3285.19 | 412.36 | 9665.99 | 529.29 |
| **400** | 49.75 | 218.73 | 769.48 | 334.44 | 3766.09 | 474.47 | 11038.39 | 607.33 |

Table 3.7: Computation Times for Virtual Deadline Algorithm and Forward Decomposition Algorithm (in seconds)

As seen in Table 3.7, for small values of $M$, VDA is much faster than FDA in terms of solution times. However, as $M$ increases, VDA takes a lot longer to converge compared to FDA, limiting its usage to small flow shops.

## 3.6 Conclusion

This chapter studied the service time optimization of flow shop systems consisting of fully controllable machines, where the service times are adjustable before each process, initially controllable machines, where the service times are set only once at the start up time and applied to all jobs processed, and uncontrollable machines, where the service times are fixed and known a priori. For

these systems with given arrival times and deadlines, a non-convex and non-differentiable optimization problem was formulated. The convex formulation derived through linearizing the *max* constraints solves the formulated non-convex and non-differentiable optimization problem only for small systems due its huge memory requirements. To solve larger systems with improved solution times, a set of optimal waiting characteristics of the system was first derived and it was shown that no waiting is observed on the optimal sample path after the first fully controllable machine. Employing this result, simplified convex optimization formulations were then introduced through eliminating $N$ variables and $N$ constraints from the convex formulation (obtained through linearization) at each machine where no waiting is observed. A "forward in time" decomposition algorithm was also developed to decompose the associated simplified convex optimization problem into smaller convex optimization problems for the flow shop systems with no initially controllable machines. As shown by a numerical example, the simplifications and the decomposition not only improved the solution times considerably but also allowed us to solve larger problems by alleviating memory constraints. A numerical example was also presented to compare the forward decomposition algorithm against a competing virtual deadline algorithm for flow shop systems where both methods are applicable. The decomposition algorithm turned out to be superior for flow shop systems with large number of machines, because the convergence speed of the virtual deadline algorithm decreased considerably as the number of machines in the system increased. Finally, the effects of replacing an initially controllable machine with a fully controllable machine and the locations of fully controllable machines on the optimal cost, and service and completion-time costs on the optimal service times were analyzed.

# Chapter 4

# Conclusion

In this chapter, we first summarize the work we did in this thesis in Section 4.1 and present some extensions in Section 4.2. Then, we conclude the thesis by stating possible future research directions in Section 4.3.

## 4.1 Concluding Remarks

In this thesis, we have considered the service time optimization of deterministic flow shop systems processing identical jobs in the order they arrive. The reason why we use the term "identical" for jobs is that homogeneous service cost functions are employed for jobs at machines, i.e., the service cost functions may be different for each machine but, since they have the same operational requirements, the same service cost function is applied to all jobs at any machine. The arrival times of these jobs and their associated deadlines were assumed to be known in advance. We have assumed that service times can be reduced by increasing additional resources such as facilities, funds, manpower, energy, etc. increasing the service costs. However, these costs will be offset by savings incurred due to the early job completions. Due to this trade-off in setting service times, our objective was to determine the service times minimizing the sum of service costs at machines and regular completion-time costs for jobs.

The flow shop systems that we have considered consist of *fully controllable* machines, where the service times can be adjusted between processes, *initially controllable* machines, where the service times are initially set to a certain value and applied to all jobs processed, and *uncontrollable* machines, where the service times are pre-specified fixed values. The ability to change the service times at Computer Numerical Control (CNC) machines without setups allows us to classify these machines as fully controllable machines. Unlike computer controlled (CNC) machines, traditional (non-CNC) machines require a human operator to turn several knobs for service time modifications. The mode of operation during mass production is to set the service times initially to a good value so as not to have the production line stop for frequent setups and to eliminate human errors. Hence, these traditional machines are the examples of initially controllable machines.

Flow shops systems with controllable service times have been studied in scheduling literature, where determination of the sequence of the jobs was the primary concern. However, the job sequencing problems of flow shops are still known to be NP-hard even for fixed service times. Thus, the scheduling literature on flow shops systems with controllable service times could not go beyond heuristics and approximate solution procedures. Since we have searched for efficient exact solution procedures, we have not dealt with the job sequencing problem. Instead, we have assumed that the job sequence is given as done in the optimal control literature

Efficient solution methodologies for single machine systems were presented in the optimal control literature. However, coupling among the machines' dynamics prevents the extension of single machine results to flow shop systems complicating any solution methodologies significantly. Moreover, the dimensions of the problem increase drastically as the number of machines in the system increases. Hence, optimization of the service times in flow shop systems becomes significantly more difficult. Basically, only one recent study [31] is known to be conducted on service time optimization of multi-machine flow shops in the optimal control literature. Different from the system that we have addressed in this thesis, job completion-time costs were removed from the objective function for the flow shops formed of only fully controllable machines processing nonidentical

jobs in this study. Unfortunately, the structural properties yielding an efficient solution procedure for the system in [31], fail to hold when we integrate job completion-time costs into the objective function, even in the absence of initially controllable and uncontrollable machines, therefore completely different analysis is required. For the systems that we have considered in this thesis, we have first formulated an optimization problem which is non-convex and non-differentiable due to max-plus algebra used for the representation of departures of jobs from machines. The convex formulation, derived through the replacement of equality constraints including *max* function with two inequality constraints, is the standard way of solving this non-convex and non-differentiable optimization problem. However, large memory requirements due to increased dimensionality limit the solution of derived convex formulation to small systems. Hence, motivated by need for more efficient solution methodologies yielding solutions for larger systems in shorter times, we have further dwelled on such flow shops systems in this thesis.

In Chapter 2, fixed service time flow shop systems formed of initially controllable and uncontrollable machines have been studied. To overcome the limitations on the solvable system sizes due to high memory requirements of the convex formulation obtained by linearizing *max* equations and to improve the solution times, independent of the objective function, a set of waiting and completion time characteristics in fixed service time flow shop systems were derived. Mainly, it was shown that jobs do not wait after the machine with the highest service time and jobs not waiting at this machine do not wait anywhere else in the system. These results allowed for an alternative representation for the job completion times which was then employed to introduce a simpler equivalent convex optimization problem with lower memory requirements. The resulting simplified convex optimization problem was shown to solve larger problems in shorter times due to its less memory requirements. However, since that formulation required a convex programming problem solver, which may be costly for small manufacturing companies, and still run out of memory for large problems, an alternative convex formulation along with a subgradient descent solution method was presented. For this new formulation, further waiting and completion time characteristics of fixed

service time flow shop systems enabling another representation for job completion times were exploited. The computational results showed that significant improvements in solution times and solvable system sizes were achieved by the subgradient descent solution method. A specific nonlinear decreasing service cost structure was also considered for such systems. This specific cost structure allowed us to sort the optimal service times and to define differentiable subproblems leading to a two-phase search algorithm. This search algorithm, which is guaranteed to find the optimal service times in a finite number of iterations, improved the solution times drastically compared to the subgradient descent solution method.

In Chapter 3, building on the results for fixed service time flow shop systems, mixed line flow shop systems formed of fully controllable, initially controllable and uncontrollable machines have been considered. Introducing fully controllable machines leaded to some optimality properties allowing for new solution methodologies different from the ones developed for fixed service time flop shop systems. Hence, we studied the mixed line flow shops systems within a new chapter. To relieve the memory bottleneck for the resulting convex formulation obtained through linearizing *max* equations, some characteristics of the optimal service times were derived and it was shown that, on the optimal sample path, jobs do not observe waiting after the most upstream fully controllable machine. The no-wait property eliminated $N$ variables and $N$ constraints from the resulting convex optimization problem at each machine it is observed and allowed us to introduce simpler equivalent convex optimization formulations. In the absence of initially controllable machines in such flow shop systems, strictly convex completion-time costs resulted in unique optimal solution and enabled us to decompose the associated simplified equivalent convex optimization problem into smaller convex optimization problems. Through a numerical study, the simplifications and the decomposition were shown to result in considerable improvements in solution times and solvable system sizes by alleviating the memory requirements.

## 4.2 Model Extensions

Our results can be extended to flow shop systems with fixed setup and transfer times known in advance. Let $t_j$ be the fixed transfer time from machine $j-1$ to machine $j$ for $j = 1, ..., M$, where $t_1$ is assumed to be zero. Let us also denote by $z_j$ the fixed setup time at machine $j$ for $j = 1, ..., M$. Note that we have assumed that initially controllable machines are set up for once at the start-up time where fixed service times of these machines are determined. Hence, setups for each job at initially controllable and uncontrollable machines can simply be thought of as compulsory load/unload operations of jobs at these machines. The departure time $x_{i,j}$ of job $C_i$ from machine $j$ can then be given by the max-plus equations

$$x_{i,j} = \max\left(x_{i,j-1} + t_j, x_{i-1,j}\right) + z_j + \hat{s}_{i,j},$$

where $x_{i,0} = a_i$ for $i = 1, ..., N$, $x_{0,j} = -\infty$ for $j = 1, ..., M$, and $\hat{s}_{i,j}$ is

$$\hat{s}_{i,j} = \begin{cases} s_{i,j}, & j \in S_F \\ s_j, & j \in S_I \\ \breve{s}_j, & j \in S_U \end{cases} \tag{4.1}$$

for $i = 1, ..., N$ and $j = 1, ..., M$. Then, by defining $\bar{s}_{i,j} = z_j + \hat{s}_{i,j}$ and by modifying the service costs as $\theta_j(\hat{s}_{i,j}) = \theta_j(\bar{s}_{i,j} - z_j)$ for $i = 1, ..., N$ and $j = 1, ..., M$, we adapt our model to these systems. Going along the same line of arguments, we can easily show that all the results that we have derived for both fixed service time and mixed line flow shop systems in Chapters 2 and 3 are valid for such systems, e.g. no-wait property still holds for mixed line flow shop systems. However, we should note that fixed transfer times $t_j$'s are carefully included in the model as constant values wherever necessary. Even though inclusion of non-zero setup and transfer times does not affect the waiting characteristics of the system and the derived solution methodologies as well, resulting optimal solutions may considerably deviate from the one for the original model, where these values are set to zero.

There may be flow shop systems where, in addition to controllable service

times on machines, the arrival (release) times can also be controlled. As a second extension of our results in this thesis, such systems can be modeled by adding a dummy fully controllable machine to the system from the starting point, i.e., we attach a new fully controllable dummy machine as the first machine of the new flow shop. We set the service cost of this dummy machine to zero and we assume that all jobs are ready in front of this dummy machine at time zero. Then, this new flow shop system, which starts out with a fully controllable machine, can be solved by the solution methodologies presented in Chapter 3 and optimal arrival (release) times can be recovered from the optimal service times of the dummy machine recursively, i.e., $a_i^* = \sum_{k=1}^{i} s_{k,1}^*$.

## 4.3 Future Research Directions

We have already shown that controllable arrival (release) times can be integrated to our model if the service times at machines are also controllable. However, if the service times of the jobs at machines are known in advance, i.e., the service times are no more decision variables, there seems no direct extension of our results except the flow shop systems formed of only uncontrollable machines with fixed service times. To our knowledge, existing research on controllable arrival (release) times with known service times is limited to single machine systems. The related scheduling problems on single machine systems were studied in [20], [21], [23], [9], [27], [28], [22], and [45]. On the other hand, there are only two related studies in the optimal control literature. Gazarik and Wardi [12], identified a necessary and sufficient optimality condition and employed it to present a simple algorithm to obtain optimal job arrival (release) times minimizing the discrepancy between job completion times and desired due dates for a single machine system with known job service times not necessarily equal. Moon and Wardi [35], considered the system in [12] where the completed jobs wait in a finite size output buffer until their due dates, and presented an efficient solution algorithm for this system with blocking. Hence, optimization of flow shop systems with known service times and controllable arrival (release) times is an open question for future research.

In this thesis, we have assumed that the arrival times of the jobs are known a priori. Instead of having a fully known arrival schedule, we can focus on uncertainty on the arrival sequence. For instance, we can deal with the case of uncertainty where only some future arrival information is available within a time window of length $T$. Under such uncertainties, studies (see e.g. [6]) on single machine systems have introduced a receding horizon control scheme with several properties enabling the use of a controller based on rough estimates of unknown future arrivals with limited loss of optimality properties. However, to the best of our knowledge, there is no research on flow shop systems where the arrival sequence is partially known. Hence, such systems deserve to be investigated further. Through deriving and exploiting their structural properties, one can try to develop efficient solution procedures for these systems. Another future research step can also be to investigate the possibility of the extension of the solution procedures such as "Forward Decomposition Algorithm" derived in this thesis to these flow shop systems.

Throughout our study, we have assumed the same service cost functions at machines for all jobs, i.e., jobs are identical in terms of service costs at machines, and built most of our results on this assumption. However, operational requirements of jobs, due to their physical properties and/or quality specifications etc., may differ at machines leading to different service cost structures. It's worth noticing that, throughout a numerical study, we have observed that no-wait property no longer holds for the nonidentical jobs case; therefore, with no doubt, existence of nonidentical jobs further complicates the problem. Recently, Mao and Cassandras [31] considered the flow shops with differing service cost functions for jobs at machines. However, in contrast to our model, Mao and Cassandras did not include the job completion-time costs in the objective function. Removal of job completion-time costs allowed them to develop an efficient iterative algorithm for flow shop systems consisting only of fully controllable machines. Unfortunately, due to job completion-time costs in the objective function, and initially controllable and uncontrollable machines in the system, the results and the solution procedure derived in [31] fail to extend to our model. Hence, the assumption of nonidentical jobs can be adapted to the flow shop systems considered in this

thesis leading to new challenging problems.

In this study, we have also assumed that a feasible solution exists under associated job completion deadlines. Since this may not always be the case, we may introduce an admission control scheme in which some jobs are removed with the objective of maximizing the number of remaining jobs to guarantee feasibility. Such an admission control scheme was recently introduced for a single machine system and, through derivation of several optimality properties, a computationally efficient algorithm was developed for solving the resulting admission control problem in [29]. However, there is no known study on admission control for flow shop systems. Hence, the admission control issues for flow shop systems are open to future research. The results in [29] for single machine environment may be combined with the ones we have derived in this thesis to extend to flow shop systems. Moreover, we may still apply an admission control even in the existence of a feasible solution. Admitting a job to the system will probably decrease the service times incurring, therefore, extra service costs. Hence, we may associate an award for admission of a job and try to maximize the objective of total awards minus total cost. As a real life example of such model, a manufacturing company may have an order that can not be accomplished within time limitations set by the customer or even if it is possible to accomplish an order within its deadline, the extra cost incurred for this order may offset the revenue gained. Hence, sometimes it is better to reject such orders and try to maximize the revenue for the manufacturing company.

Note that we have required strictly convex objective function for the solution uniqueness which formed the basis for the development of "Forward Decomposition Algorithm" for mixed line flow shops consisting only of fully controllable and uncontrollable machines. In fact, the strict convexity of both the service and completion-time costs does not suffice to obtain a strictly convex objective function in service times. In particular, we have formed our objective function from the service costs monotonically decreasing in service times and the completion-time costs monotonically increasing in completion times. Since the sum of strictly convex functions is again strictly convex, the sum of the service costs is strictly convex. Thus, the strict convexity of the objective function in our model depends

on whether the completion-time costs are convex in service times; this would be ensured if the completion-time costs are non-decreasing. Since the job completion times are monotonically non-decreasing in service times, this condition is satisfied for the completion-time costs in our model, which were assumed to be monotonically increasing in job completion times. Hence, the completion-time costs are convex in service times. Since the sum of a strictly convex function and a convex function is strictly convex, our objective function is also strictly convex. In fact, strict convexity of the objective function is a sufficient condition for solution uniqueness, but it is not a necessary condition. For instance, Cassandras et al. [7], different from us, considered strictly convex but not monotonically non-decreasing completion-time costs in a single machine environment and established the uniqueness of the optimal solution despite the fact that the resulting cost function involved was non-convex and non-differentiable. Hence, the identification of the cost structures satisfying unique solution property to eliminate the need for strict convexity assumption can also be discussed as a future research direction.

In this thesis, buffers in front of the machines have been assumed to be of infinite sizes. In practice, there are many manufacturing environments, in which the buffer capacities have to be taken into account, and there may be limits on the capacities of buffers between two consecutive machines. Flow shops with finite buffers have also received attention in the scheduling literature (see [18] for a survey). However, research on flow shop scheduling problems with finite buffers is limited to uncontrollable service times. The only study combining controllable service times with finite buffers on flow shops was carried out by Shabtay et al. [43]. In this study, a two-machine zero-buffer flow shop with the objective of minimizing makespan was considered. Service times at machines were assumed to be convex decreasing functions of the amount of limited resources allocated. By reducing it to a special case of the travelling salesman problem, the problem was shown to be strongly NP-hard. Shabtay et al. [43] also developed two heuristic algorithms producing close-to-optimal solutions for the reduced problem. The reason why we have included the study of Shabtay et al. [43] as an example of finite buffers case is that the finite buffer problems can be modeled as the zero

buffer problems. This is because each unit of storage buffer may be modeled as a machine at which each job has a service time of zero, i.e., zero service time uncontrollable machine in our model. Thus, one may directly focus on the zero buffer case and utilize the resulting solution methods to the finite buffers case. On the other hand, the max-plus algebra used to represent the departure times of jobs from machines changes: the departure time $x_{i,j}$ of job $C_i$ from machine $j$ is given by the max-plus equation

$$x_{i,j} = \max(x_{i,j-1}, x_{i-1,j}, x_{i-b_{j+1}-1,j+1}) + \hat{s}_{i,j}, \tag{4.2}$$

where $x_{i,0} = a_i$ for $i = 1, ..., N$, $x_{0,j} = -\infty$ for $j = 1, ..., M$, $\hat{s}_{i,j}$ is given by (4.1), and $b_j$ is the capacity of the buffer in front of machine $j$. We note that the last term $x_{i-b_{j+1}-1,j+1}$ of the *max* function in (4.2) stands for the blocking after service completion at machine $j$ and we have $x_{i-b_{j+1}-1,j+1} = -\infty$ for $i = 1, ..., b_j + 1$ and $j = 1, ..., M$. In fact, no waiting means no buffering for the machines downstream to the slowest machine of fixed service time flow shops and the first fully controllable machine of mixed line flow shops. Hence, we expect that the results, i.e., no-wait property, derived for infinite buffers case (both for fixed service time and mixed line flow shops) extend to finite buffers case. It's worth noticing that the results of the numerical study that we held support our expectations: we have observed that no-wait property still holds for the flow shops with finite buffers. Hence, another future research direction is to extend the results for infinite buffers case to finite buffers case which seems not to be a simple task due to blocking issues. Moreover, in addition to service times, buffer sizes can also be treated as decision variables in stead of being system parameters by future researchers.

The optimal control literature together with the scheduling literature on controllable service times assume setups as ignorable to simplify the overall analysis (Literature on scheduling problems involving setup considerations completely relies on uncontrollable service times even for single machine environments. A comprehensive survey on scheduling problems with setup times for flow shop systems can be found in [1] and [8]). This assumption may be justified for CNC machines which greatly reduce the influence of setup times. However, there are

still many environments, i.e., traditional (manually controlled) machines, where setup times are significant. Therefore negligence of setups adversely affects the solution quality for these systems which require explicit treatment of setup times. In fact, we have extended our model to flow shop systems with fixed setup times in the previous section with the assumption that machines need to be set up before each process without a setup cost. However, in reality, a setup cost may be incurred when a machine is set up. In addition, affecting the job completion times and therefore the associated job completion-time costs, setup times have direct cost implications. On the other hand, we may lower the service costs by infrequent setups due to increased flexibility on service time control. Hence, to reflect the reality, infrequent setups should also be included in the analysis. To model the flow shop systems with infrequent setups, we first associate a setup cost $Z_j$ at machine $j$ for each setup and define binary variable $e_{i,j}$ representing whether machine $j$ is set up for job $C_i$ or not as

$$e_{i,j} = \begin{cases} 1, & \text{Machine } j \text{ is set up for job } C_i \\ 0, & \text{otherwise} \end{cases} \tag{4.3}$$

for $i = 1, ..., N$ and $j \in (S_F \cup S_I)$. Then, by integrating the total setup cost $\sum_{j \in S_F \cup S_I} \sum_{i=1}^{N} e_{i,j} Z_j$ into the objective function and rewriting the departure time $x_{i,j}$ of job $C_i$ from machine $j$ as

$$x_{i,j} = \max(x_{i,j-1}, x_{i-1,j}) + e_{i,j} z_j + \hat{s}_{i,j}$$

with $z_j$ being the fixed setup time at machine $j$ and $\hat{s}_{i,j}$ given by (4.1) for $i = 1, ..., N$ and $j \in (S_F \cup S_I)$ in our model, we come up with a mixed integer programming formulation where our decision variables are the service times at fully and initially controllable machines and binary variables given by (4.3). It's obvious that introducing setups with associated costs complicates the analysis significantly and therefore flow shop systems with infrequent setups need to be further investigated.

In this thesis, we have studied the flow shop systems in the context of optimal control literature, where jobs are assumed to be processed in a given sequence at

all machines. Flow shops systems with controllable service times have been studied in scheduling literature, where determination of the sequence of the jobs was the primary concern. Due to the difficulties in reaching exact solutions caused by nonlinear costs, most of the researchers have focused on linear or simple special form of nonlinear convex costs. However, the job sequencing problems of flow shops are still known to be NP-hard even for fixed service times and the simplest scheduling performance measures. Thus, the scheduling literature on flow shops systems with controllable service times could not go beyond heuristics and approximate solution procedures. A survey of results on the controllable service times in scheduling problems can be found in [36], [19], and [44]. Different from the current scheduling literature, we work with more general form of nonlinear convex costs and non-zero job arrival times. In fact, by fixing the sequence of jobs at machines, the problem clearly becomes much simpler than the one where sequencing is a part of the control variable. However, we have obtained very efficient solution methodologies yielding true optimal solutions for a given sequence. Motivated by the fact that an efficient algorithm for determining the optimal service times for a given sequence can be useful in an iterative technique for determining the optimal schedule, as another future research direction, we can further dwell on the sequencing problem for the flow shops with more general form of nonlinear convex costs and non-zero job arrival times studied in this thesis.

# Bibliography


[1] A. Allahverdi, J. N. D. Gupta, T. Aldowaisan. A review of scheduling research involving setup considerations. *Omega, The International Journal of Management Science*, 27(2):219–239, April 1999.

[2] D. P. Bertsekas. *Nonlinear Programming.* Athena Scientific, Belmont, Massachusetts, 1995.

[3] C. Boyle. Using group technology for tool inventory control. *Proceedings of the 3rd Biennial International Machine Tool Technical Conference*, strony 67–71, September 1986.

[4] C. G. Cassandras, S. Lafortune. *Introduction to Discrete Event Systems.* Kluwer Academic Publishers, 1999.

[5] C. G. Cassandras, Q. Liu, D. L. Pepyne, K. Gokbayrak. Optimal control of a two-stage hybrid manufacturing system model. *Proceedings of 38th IEEE Conference on Decision and Control*, strony 450–455, 1999.

[6] C. G. Cassandras, R. Mookherjee. Properties of receding horizon controllers for some hybrid systems with event uncertainties. *Proceedings of 2003 IFAC Conference on Analysis and Design of Hybrid Systems*, strony 413–418, 2003.

[7] C. G. Cassandras, D. L. Pepyne, Y. Wardi. Optimal control of a class of hybrid systems. *IEEE Transactions on Automatic Control*, 46(3):398–415, 2001.

[8] T. C. E. Cheng, J. N. D. Gupta, G. Wang. A review of flowshop scheduling research with setup times. *Production and Operations Management*, 9(3):262–282, 2000.

[9] T. C. E. Cheng, A. Janiak. Resource optimal control in some single-machine scheduling problems. *IEEE Transactions on Automatic Control*, 39(6):1243–1246, 1994.

[10] Y. C. Cho, C. G. Cassandras, D. L. Pepyne. Forward decomposition algorithms for optimal control of a class of hybrid systems. *International Journal of Robust and Nonlinear Control*, 11:497–513, 2001.

[11] A. R. Conn, N. I. M. Gould, P. L. Toint. *Trust-Region Methods*. Society for Industrial Mathematics, 1987.

[12] M. Gazarik, Y. Wardi. Optimal release times in a single server: An optimal control perspective. *IEEE Transactions on Automatic Control*, 43(7):998–1002, 1998.

[13] K. Gökbayrak, Ö. Selvi. Optimal hybrid control of a two-stage manufacturing system. *Proceedings of 2006 American Control Conference*, strony 3364–3369, 2006.

[14] K. Gökbayrak, Ö. Selvi. Constrained optimal hybrid control of a flow shop system. *IEEE Transactions on Automatic Control*, 52(12):2270–2281, 2007.

[15] K. Gökbayrak, Ö. Selvi. A subgradient descent algorithm for optimization of initially controllable flow shop systems. *Discrete Event Dynamic Systems: Theory and Applications*, submitted.

[16] K. Gökbayrak, Ö. Selvi. Optimization of a flow shop system of initially controllable machines. *IEEE Transactions on Automatic Control*, to appear.

[17] M. Grant, S. Boyd. CVX: Matlab software for disciplined convex programming. *http://stanford.edu/ boyd/cvx*, 2007.

[18] N.G. Hall, C. Sriskandarajah. A survey of machine scheduling problems with blocking and no-wait in process. *Operations Research*, 44:510–525, 1996.

[19] H. Hoogeveen. Multicriteria scheduling. *European Journal of Operational Research*, 167:592–623, 2005.

[20] A. Janiak. Scheduling independent one-processor tasks with linear models of release dates under a given maximum schedule length to minimize resource consumption. *Journal of System Analysis Modelling Simulation*, 7(11-12):885–890, 1990.

[21] A. Janiak. Single machine scheduling problem with a common deadline and resource dependent release dates. *European Journal of Operational Research*, 53:317–325, 1991.

[22] A. Janiak. Single machine sequencing with linear models of release dates. *Naval Research Logistics*, 45:99–113, 1998.

[23] A. Janiak, C. L. Li. Scheduling to minimize the total weighted completion time with a constraint on the release time resource consumption. *Mathematical and Computer Modelling*, 20:53–58, 1994.

[24] S. Kalpakjian, S. R. Schmid. *Manufacturing Engineering and Technology.* Pearson Prentice Hall, 2006.

[25] D. E. Kirk. *Optimal Control Theory.* Prentice-Hall, 1970.

[26] A. E. Kuchinic, A. Seidman. Tool management in automated manufacturing: Operational issues and mathematical models. *International Industrial Engineering Conference Proceedings*, strony 379–382, 1988.

[27] C. L. Li. Scheduling to minimize the total resource consumption with a constraint on the sum of completion times. *European Journal of Operational Research*, 80:381–388, 1995.

[28] C. L. Li, E. C. Sewell, T. C. E. Cheng. Scheduling to minimize release-time resource consumption and tardiness penalties. *Naval Research Logistics*, 42:949–966, 1995.

[29] J. Mao, C. G. Cassandras. Optimal admission control of discrete event systems with real-time constraints. *Proceedings of 46th IEEE Conference on Decision and Control*, strony 3005–3010, 2007.

[30] J. Mao, C. G. Cassandras. Optimal control of two-stage discrete event systems with real-time constraints. *Discrete Event Dynamic Systems*, 17:505–529, 2007.

[31] J. Mao, C. G. Cassandras. Optimal control of multi-stage discrete event systems with real-time constraints. *IEEE Transactions on Automatic Control*, to appear.

[32] J. Mao, C. G. Cassandras, Q. Zhao. Optimal dynamic voltage scaling in energy-limited nonpreemptive systems with real-time constraints. *IEEE Transactions on Mobile Computing*, 6(6):678–688, June 2007.

[33] C. L. Monma, A. Schrijver, M. J. Todd, V. K. Wei. Convex resource allocation problems on directed acyclic graphs: Duality, complexity, special cases, and extensions. *Mathematics of Operations Research*, 15(4):736–748, 1990.

[34] J. Moon, Y. Wardi. Optimal control of processing times in single-stage discrete event dynamic systems with blocking. *IEEE Transactions on Automatic Control*, 50(6):880– 884, 2005.

[35] J. Moon, Y. Wardi. Optimal release times in a single-stage manufacturing system with blocking: Optimal control perspective. *Journal of Optimization Theory and Applications*, 125(3):653–672, 2005.

[36] E. Nowicki, S. Zdrzalka. A survey of results for sequencing problems with controllable processing times. *Discrete Applied Mathematics*, 26:271 – 287, 1990.

[37] D. L. Pepyne, C. G. Cassandras. Modeling, analysis, and optimal control of a class of hybrid systems. *Journal of Discrete Event Dynamic Systems: Theory and Applications*, 8(2):175–201, 1998.

[38] D. L. Pepyne, C. G. Cassandras. Optimal control of hybrid systems in manufacturing. *Proceedings of the IEEE*, 88(7):1108–1123, 2000.

[39] M. Pinedo. *Scheduling: Theory, Algorithms, and Systems*. Prentice Hall, 2002.

[40] M. J. D. Powell. A fortran subroutine for solving systems of nonlinear algebraic equations. *Numerical Methods for Nonlinear Algebraic Equations*, 1970.

[41] Ö. Selvi, K. Gökbayrak. A search method for optimal control of a flow shop system of traditional machines. *European Journal of Operational Research*, submitted.

[42] Ö. Selvi, K. Gökbayrak. Service time optimization of mixed line flow shop systems. *IEEE Transactions on Automatic Control*, submitted.

[43] D. Shabtay, M. Kaspi, G. Steiner. The no-wait two-machine flow shop scheduling problem with convex resource-dependent processing times. *IIE Transactions*, 39:539–557, 2007.

[44] D. Shabtay, G. Steiner. A survey of scheduling with controllable processing times. *Discrete Applied Mathematics*, 155:1643–1666, 2007.

[45] N. V. Shakhlevich, V. A. Strusevich. Single machine scheduling with controllable release and processing times. *Discrete Applied Mathematics*, 154(15):2178–2199, 2006.

[46] P. Tomek. Tooling strategies related to FMS management. *The FMS Magazine*, 5:102–107, 1986.

[47] Y. Wardi, C. G. Cassandras, D. L. Pepyne. A backward algorithm for computing optimal controls for single-stage hybrid manufacturing systems. *International Journal of Production Research*, 39(2):369–393, 2001.

[48] P. Zhang, C. G. Cassandras. An improved forward algorithm for optimal control of a class of hybrid systems. *IEEE Transactions on Automatic Control*, 47(10):1735–1739, 2002.